AUTOMATIC SELECTIVE DISASSEMBLY TIME COMPUTATION AND PRODUCT ARCHITECTURE REDESIGN SUGGESTION

By

YANG HU

A dissertation submitted in partial fulfillment of
the requirements of the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

WASHINGTON STATE UNIVERSITY
School of Mechanical and Materials Engineering

JULY 2013

To the Faculty of Washington State University:

The member of the Committee appointed to examine the dissertation of YANG HU, find it satisfactory and recommend that it be accepted.

Gaurav Ameta, Ph D., Chair

Uma Jayaram, Ph D.

Anantharaman Kalyanaraman. Ph D.

ACKNOWLEDGEMENT

I would like to express the deepest gratitude to my advisor, Dr. Gaurav Ameta, for his full support on my study and research, for his continually of and convincingly conveyed a spirit of adventure in regard to research, for his enlightening teaching and for his patience and encouragement when I met difficulties and felt frustration in research. Without his guidance and persistent help this work would not have been possible.

I would also like to thank my committee members, Dr. Uma Jayaram and Dr. Anantharaman Kalyanaraman for sparing their valuable time to interact with me and for sharing their inputs and feedback. I appreciate to them for accommodating my requests.

Thanks also go to Raghunathan Srinivasan, for his self-giving sharing of experience and material regarding to my research. Thanks my fellow graduate students, Annie Chawla, Amanider Singh Gill, Tesfay Niguse Tesfamicael and staff at the Mechanical and Material Engineering Department -- it was an enriching learning and wonderful working with you.

I would like to thank my friend Yalan Liu, Britney Bergmans and Nurdan Yurt for their concerns in my study and life.

I would like to thank my husband Yudi Zhu and my parents who support me completely on my graduate studies and abroad living.

AUTOMATIC SELECTIVE DISASSEMBLY TIME COMPUTATION

AND PRODUCT ARCHITECTURE REDESIGN SUGGESTION

Abstract

by Yang Hu, M.S.
Washington State University
July 2013

Chair: Gaurav Ameta

The goal of this research is to develop automated tools to estimate disassembly time and suggest changes in the product in order to reduce the disassembly time. Disassembly is a critical process in the end-of-life stage of a product. Disassembly is usually followed by sorting and then material recovery for recycle or part recovery for remanufacturing. Manual estimation of disassembly time based on how the components of a product are assembled is time consuming. To the best of our knowledge, no automatic disassembly time estimation method and tool currently exists.

The methodology utilized to estimate disassembly time is based on metrics from assembly graph and is also based on the assumption that disassembly is inverse process of assembly. Solid Works Application Programming Interface (API) and C# is chosen as the language to demonstrate the tool and algorithms. The assembly graph consists of components as nodes and edges as relationships/mates between components. The assembly graph is extracted from Solid Works mate list of a Product. Selective disassembly is introduced as a time saving method to recovery particular components or material from a product. Material selective disassembly time estimation is based on modifying the assembly graph by merging the nodes that have same material and are neighbors (nodes with direct edges connecting one another) in the assembly graph. The merge nodes (representing group of components) can be disassembled as one group. The principle for providing guidelines to designer is based on reducing the disassembly time. These guidelines are generated by traversing the assembly mate list and finding neighbors and non-neighbor components of different material and same material, respectively. In order to achieve component selective disassembly time estimation, ray tracing is used to identify accessibility to the component. Destructive disassembly based guidelines are provided for selectively disassembling the components that limited or no accessibility. Four case studies are presented to demonstrate the tool developed for complete disassembly, material selective disassembly and component selective disassembly time estimation.

TABLE OF CONTENT

ACKNOWLEDGEMENT iii
Abstract iv
LIST OF TABLES viii
LIST OF FIGURES ix
Chapter 1 Introduction 1
1.1 Motivation 1
1.2 Design Stage 1
1.3 End-of-life (EOL) Stage 2
1.4 Problem Statement 4
1.5 Outline 5
Chapter 2 Literature Review 6
2.1 End-Of-Life Operations 6
2.2 Product complexity assessment methods 7
2.3 Selective Disassembly 9
2.4 Graph Theories in Disassembly 11
2.5 Redesign for ease of disassembly 16
2.6 Graph Algorithm 17
2.6.1 Graph Building 17
2.6.2 Graph Implementation 18
2.6.3 Graph Traversing and Breadth-First Search (BFS) 19
Chapter 3 Automatic Disassembly Time Estimation and Redesign Suggestions Generation 20
3.1 Mathematical Model of Assembly Time Estimation 20
3.2 Overview of the Implementation 21
3.3 Assembly Model Graph 26
3.3.1 Assembly Graph Generation 27
3.3.2 Assembly Graph Re-generation 30
3.4 Disassembly Time Estimation 37
3.4.1 Total Number of Components 37
3.4.2 Number of Connection Relationship 38
3.4.3 Calculate Total Path Length (TPL) 41
3.5 Generate Redesign Suggestion 42
Chapter 4 Part Position Location and Disassembly Suggestion Generation 47
4.1 Introduction 47
4.2 Determining Start and End Point of Ray 47

4.3 Part Position 50
4.4 Disassemble advice generation 54
Chapter 5 Case Studies 55
5.1 Introduction 55
5.2 Interface 55
5.3 Case Study 1 – Standard Toaster 59
5.3.1 Disassembly Time Estimation 61
5.3.2 Destructive disassembly 62
5.4 Case Study 2 – Eco-Toaster 63
5.4.1 Disassembly Time Estimation 65
5.4.2 Destructive Disassembly 66
5.5 Case Study 3 – Drill 67
5.5.1 Disassembly Time Estimation 68
5.5.2 Destructive Disassembly 70
5.6 Case Study 4 – Kid Bike 71
5.6.1 Disassembly Time Estimation 72
Chapter 6 Conclusion and Future Work 76
6.1 Contributions 76
6.2 Limitations 76
6.3 Future Work 77
Reference 79

LIST OF TABLES

Table 2.1 Table of disassembly reference 9
Table 2.2 Table of graph theory reference 14
Table 3.1 Example of complexity metrics and complete disassembly time estimation process 24
Table 3.2 Example of complexity metrics and material selective disassembly time estimation process ... 26
Table 4.1 Ease level of grabbing cuboids with face exposure 52
Table 5.1 Complete Disassembly Time and Material Selective Disassembly Time 61
Table 5.2 Complete Disassembly Time and Material Selective Disassembly Time 65
Table 5.3 Complete Disassembly Time and Material Selective Disassembly Time for Drill 69
Table 5.4 Complete Disassembly Time and Material Selective Disassembly Time for Kid Bike 73

LIST OF FIGURES

Figure 1.1 Product life cycle with end-of-life options 3
Figure 1.2 End-of-life product processes for recycle and remanufacture 3
Figure 3.1 General algorithms for automatic disassembly time estimation and redesign suggestions. 23
Figure 3.2 Example of Assembly Graph for Complete Disassembly 23
Figure 3.3 Example of Bi-Partial graph for Complete Disassembly 24
Figure 3.4 Example of Assembly Graph for Material Selective Disassembly 25
Figure 3.5 Example of Bi-Partial graph for Material Selective Disassembly 25
Figure 3.6 Node class 28
Figure 3.7 Algorithm for obtaining mate graph from an assembly model 29
Figure 3.8 Algorithm for building an assembly graph from an existing mate graph 30
Figure 3.9 Nodes of same material are directly connected together 32
Figure 3.10 Selective Material Parts Distribute in Different Places 32
Figure 3.11 Re-generated Graph for Selective Material Parts Connect Together 33
Figure 3.12 Re-generated Graph for Selective Material Parts Distribute in Different Places 33
Figure 3.13 Building Raw Material List 35
Figure 3.14 Grouping Parts Based on Selected Raw-Material 36
Figure 3.15 Obtaining total part number of in an assembly 38
Figure 3.16 Bolting Diagram [23] 39
Figure 3.17 Connection Relationships in Solid Works Assembly 39
Figure 3.18 Algorithm flows for counting connection relationships 41
Figure 3.19 Algorithm flows for calculating Total Path Length (TPL) 44
Figure 3.20 Algorithm flows for giving re-design suggestion 46
Figure 4.1 Bounding Box for a Part [60] 48
Figure 4.2 Bounding Box for a Toaster Bottom sub-assembly 49
Figure 4.3 Algorithm flows for getting ray start and end points 50
Figure 4.4 Algorithm flows for determining selected part position 52
Figure 4.5 Algorithm flows for generating re-design advices 54
Figure 5.1 Dialog box screen shot for toaster model 56
Figure 5.2 Complete disassembly time estimation results 56
Figure 5.3 Highlighted part and notification dialog box for the standard toaster CAD assembly model. .. 57
Figure 5.4 Screen shot of Destructive Disassembly interface dialog 58

Figure 5.5 Shot screen of highlighting selected part 58
Figure 5.6 Screen shot of highlighting part that suggested broke 59
Figure 5.7 Graphic image of standard toaster assembly model 60
Figure 5.8 Explosion view of standard toaster assembly model 60
Figure 5.9 Screen shot of suggestions of changing highlight parts material 62
Figure 5.10 Screen shot of selected side wall belongs to standard toaster 63
Figure 5.11 Screen shot of parts impede disassembling side-wall 63
Figure 5.12 Graphic image of Eco-toaster assembly model 64
Figure 5.13 Explosion view of Eco- toaster assembly model 64
Figure 5.14 Screen shot of suggestions of changing highlight parts material 66
Figure 5.15 Screen shot of selected side wall belongs to eco-toaster 67
Figure 5.16 Screen shot of parts impede disassembling side-wall 67
Figure 5.17 Graphic image of drill assembly model 68
Figure 5.18 Explosion view of drill assembly model 68
Figure 5.19 Screen shot of suggestions of changing highlight parts material 70
Figure 5.20 Screen shot of selected gear inside the drill 71
Figure 5.21 Screen shot of parts impede disassembling side-wall 71
Figure 5.22 Graphic image of kid bike assembly model 72
Figure 5.23 Explosion view of kid bike assembly model 72
Figure 5.24 Screen shot of suggestions of changing highlight parts material 73
Figure 5.25 Screen shot of highlighted sprocket 74
Figure 5.26 Screen shot of parts impede disassembling neck-fork 75

Chapter 1 Introduction

1.1 Motivation

Based on a report from U.S. Environmental Protection Agency (EPA), in 2010, Americans generated about 250 million tons of waste. Out of the total waste, durable goods, such as furniture, home appliances, consumer electronics, etc., make up the third largest segment, accounting for about 49 million tons. The material recovery rate of durable goods is only 18.5% [1] because most of the generated solid waste is disposed of by landfill or combustion. A proper method recommended by EPA to increase the recovery rate is for manufacturers to take back products in their End-Of-Life (EOL) stage [2]. However, take back of retired products at their EOL stage implies that the manufacturers have to pay the expenses for EOL operations, such as recollection cost, recycling cost, remanufacturing cost, etc. According to the report in [3], only 10% to 20% of the recycling cost depends on recycling process optimization while the rest is already determined at the product design stage. In other words a product that can be easily disassembled can (a) save on recycling cost and (b) encourage manufacturers to plan for recycling and reusing instead of ignoring the retired products for land filling and combustion.

1.2 Design Stage

Every single commercial product has its own life cycle. Should it be a plastic bottle or an automobile, from modern sustainable perspective [4], manufacturers have to consider five phases across each product life cycle. They are: design phase, raw material phase, manufacturing phase, use phase and end-of-life (EOL) phase. Product design and EOL management are two ends of a commercial product.

The design and planning stage is where the detailed design and development, based on the customers/market/strategic requirements, of the product is performed. The design team usually considers multiple criteria like cost, quality, ease-assembly, product system functionality etc.

This phase also includes redesign and ramp up for improvement to existing products. Redesign is often based on the feedback and suggestions from the customer and/or manufacturing team regarding the product functionality and manufacturability. From sustainable perspective, product design determines the costs associated with the EOL stage [3]. Therefore, the design stage is crucial for the entire product life cycle.

1.3 End-of-life (EOL) Stage

End-of-life (EOL) is a term used with respect to a product supplied to customers, indicating that the product is in the end of its useful lifetime. Furthermore, EOL may imply that the vendor will no longer be marketing, selling, or sustaining the product and may also be limiting or ending support for the product [6]. In this stage, retired products are usually recollected by their manufacturers or recycling companies. These companies will determine proper operations for those retired products according to the products state. There are four major processing options to treat out of work products: reuse, recycle, remanufacture and disposal (land filling, incineration, etc). Recovered substances flow back to a new product’s life-cycle after going through the operations shown in Figure 1.1.

Two main EOL processes that manufacturers or recycling companies can select are shown in Figure 1.2. One of them is named as remanufacturing while the other is recycling. The differences between these are shown in Figure 1.1. Substances in recycling are converted to raw material first and then used to produce new product. On the contrary, substances in remanufacturing skip the conversion step to raw materials and are processed directly to be used in similar products.

For both recycling and remanufacturing, it is necessary to perform certain primary treatments for the retired products. There are two primary treatments shown in Figure 1.2. The first treatment is disassembly, which is essential for the following steps. The second one is sorting, which is subjected to results of the disassembly process. Therefore, disassembly is a critical bottleneck for most of the EOL operations. If the cost and time of disassembly is too high, then the manufacturer might not view recycling or remanufacturing as a viable option. In order for disassembly to be efficient, products have to be designed for disassembly.

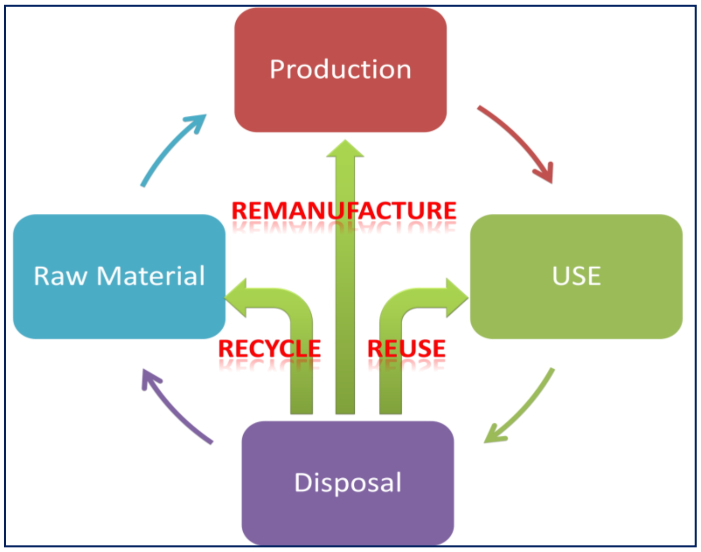


Figure 1.1 Product life cycle with end-of-life options

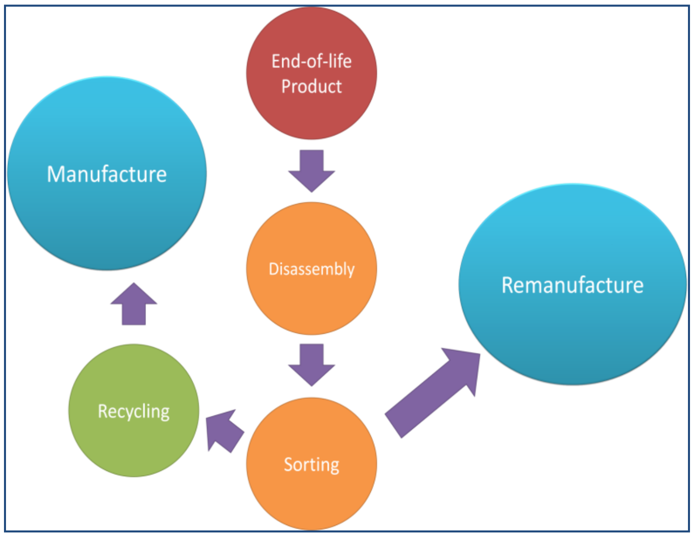


Figure 1.2 End-of-life processes of a product related to recycle and remanufacture

1.4 Problem Statement

As stated above that 80% to 90% of recycling operation cost is determined at design stage [3] which indicates that the product architecture determined at design stage influences recycling operations at EOL stage. Disassembly is the first operation at EOL stage as shown in Fig 1.2, either remanufacture or recycling is operated after sorting. As is evident from the discussion in the previous section, disassembly is the critical process for recycle, remanufacturing to be viable. Products have to be designed for disassembly.

Furthermore, many products have only a few components that can create cost recovery for manufacturers. The cost recovery may be due recovery of recyclable material from the product or due to recovery of a particular costly/hazardous component from the product. When a group of components are disassembled because of their material, in this thesis, the disassembly process will be called material selective disassembly. When only a particular component is to be disassembled, in this thesis, the disassembly process will be called component selective disassembly. In these cases, manufacturers only desire to disassemble those particular components only rather than completely disassembling the product. Such partial/selective disassembly requires fewer operations than complete disassembly [5] and can save manual labor cost in sorting. Therefore, products need to design for complete disassembly and/or partial/selective disassembly as the case may be. In order to design products for disassembly, designers can follow prescriptive guidelines [7]. Such prescriptive guidelines cannot assist a designer to choose between alternative product designs. In such a case, alternative product designs need to be compared based on ease of disassembly and/or disassembly time. Manual estimation of disassembly time based on how the components of a product are assembled is time consuming. To the best of our knowledge, no automatic disassembly time estimation method and tool currently exists.

Therefore, the aim of this research is

a) To develop a software tool to automatically estimate

a. Disassembly time based on the given CAD assembly model of a product.

b. Selective disassembly time based on the given CAD assembly model of a product including the criteria for selective disassembly

b) To provide guidelines to designer regarding the changes in the product that can lead to reduction in complete/selective disassembly time.

1.5 Outline

Chapter 2 includes the literature review. Chapter 3 provides the algorithms for completely disassembly time estimation and material selective disassembly time estimation. Chapter 4 gives the algorithms for component selective disassembly. Chapter 5 presents five case studies to test the complete disassembly time estimation program, material selective disassembly time and component selective disassembly. Chapter 6 provides the conclusion and future work of this research.

Chapter 2 Literature Review

In order to explore feasible ways to increase manufacturer's profit at end-of-life stage, it is necessary to understand the operations in this stage; therefore in section 2.1 a brief review of end-of-life operations is presented, followed by literature review for generating disassembly/assembly sequence and product complexity assessment in section 2.2. Finally a brief review of graph algorithms is presented in section 2.3.

2.1 End-Of-Life Operations

The four major processing options at the end-of-life of products are reuse, recycle, remanufacture and disposal. Only by minimizing disposal can reuse, recycle and remanufacture of products or components be really beneficial to the manufacturer, user, society and the environment [5].

Disassembly is the first step when the manufacturer decides to reuse, recycle or remanufacture. Disassembly is [9] defined as a systematic approach for separating a product into its constituent parts, components, subassemblies, or other groupings. For a given purpose (such as recycling of raw materials, reuse of electric components and refurbishing a recalled product) manufacturers can partially disassemble a component or a part, or a group of parts or a subassembly or completely disassemble a product into all of its parts [8]. The disassembly and assembly time is important in order for the recycle and recovery to be cost effective [10-12]. Gupta stated in [13] that a design's manufacturability is a measure of the effort required to manufacture the part according to the design specifications, since all manufacturing operations have measurable time and cost. Similarly, disassembly time and/or cost can be used as an underlying basis to form a proper evaluation of ease of disassembly. Furthermore, it presents a realistic view of the difficulty in disassembling a product and can be used to aid designers in making redesign decisions.

Reuse of products and materials is not new to manufacturing enterprises; they have many economic and ecological motivations to take this action. They can reuse products or materials by themselves or transfer products or materials to other companies, such as their suppliers to reuse them, or to other companies outside their business chain. Normally, products that can be reused directly without prior repair operations (though possibly after cleaning and minor maintenance) are reusable packages such as bottles, pallets or containers. Consumer goods are mostly only returned at the end of their life cycle, which implies that the products are already outdated, retired or out of work [14-15]. Under such situation, performing repair or changing broken parts is necessary to make the product usable and continue making profit. Recycling of raw materials from retired products is a process that converts retired product into smaller size [14] that can be used as material. The related processes include breaking product into scraps and pelleting. Operations for material recovery include performing the necessary disassembly, sorting and chemical operations [16]. In the case of remanufacturing, the product's identity is preserved and the required disassembly, sorting, refurbishing and assembly operations are performed to bring the product back to a desired level of quality [16-18]. However, not all products are suitable for remanufacture, only durable products which are able to withstand multiple lifecycles and contain high value parts. Furthermore there should be potential market demand for the remanufactured products [19]. Examples of remanufactured products [20] include auto parts, electric home appliances, personal computers, cellular phones, photocopiers, single-use cameras, ink and toner cartridges for printers, etc. As stated above, disassembly is the first step in reusing, recycling and remanufacturing. Therefore disassembling efficiency relates directly to these end-of-life processing.

2.2 Product complexity assessment methods

Disassembly time if estimated in the design stage will help designers to optimize their product model with respect to disassembly. Assembly and disassembly time of a product is highly influenced by the arrangement, organization and assembly constraints of each component within the product [7, 22]. Complexity is most often defined in the terms of systems [23]. Product complexity can be defined as a set of interrelated parts which through their interrelations to display functionalities that the individual part cannot do independently. Mathieson and Summers developed a method to determine the complexity of a product utilizing its architecture graph [24], which is an abstract description to represent the assembly relationship between two parts in a product. They also developed an approximate equation to calculate product assembly time [24].

In order to increase the feasibility of evaluating ease of disassembly, Desai et al. [25] generated an assessment method by assigning weighting various factors, such as size and shape of parts being disassembled, weight, frequency of disassembly tasks, requirement of manpower, postural requirements and material handling requirements. Beardsley et al. [26] developed a methodology to assess disassembly performance through data management and certain metrics which are developed using the data to evaluate ease of disassembly. In this assessment methodology, they evaluate the disassembly efficiency of a design by calculating the percentage rating of how far the current design is from a reference design of the same product. They developed the equation by obtaining values such as theoretical minimum number of parts, number of repetitions in each disassembly task, required tools and difficulty rating and so on. Kriwet et al. [27] developed a methodology to advise manufactures on optimal disassembly plan based on company objectives and future economic uncertainty regarding its product at end-of-life. In their method they identify the optimal operations at end-of-life by comparing each operations utility, computed through the developed equations.

To meet different demands, manufacturers can select either complete disassembly or selective disassembly. Complete disassembly serves the demand for recycling all parts and selective disassembly serves the demand for recycling only one or multiple parts that special for manufacturers [28].

2.3 Selective Disassembly

Selective disassembly requires the disassembly of selected parts with reuse/recycling/remanufacturing potential [29]. The purpose of selective disassembly is to determine a disassembly sequence for these selected parts with minimal removal of other parts. Selective disassembly proves to be helpful in saving cost and time when manufacturers want to only remove some specific parts from retired products [30-32].

Behdad et al. [33] presents a method for determining disassembly extent and EOL operations that can be shared for multiple products. They use a transition matrix to represent parts and sub-assemblies that can be disassembled by same operation based on the joint type in the assembly. Another metric they used to making decision is to estimate the EOL value, which is defined as the income or loss generated from a particular EOL decision for a specific module.

Kara et al. [34-35] developed a methodology to generate optimal disassembly sequences for selected parts, this method was based on the connection networks of each part, the network is called liaison. The optimal sequence is estimated by a precedence rule established by user's answer towards each liaison. Smith et al. [36-37] developed a program to generate selective disassembly sequence based on the 2D geometric relationship between every two parts and the corresponding position of each part; they also improved the program by using union-find algorithm [38].

Generally speaking, partial disassembly costs less time than completely disassembly. The following section 2.4 will state current development in graph theory applications that can be used in complete disassembly and selective disassembly.

Below is a brief summary of literatures discussed above:

Table 2.1 Table of disassembly reference

Related Literatures	**Conducted Researches**	**References**
	• Minimizing disposal can really benificial to the manufacturer, user, society and the environment	Srinivasan, 2011

End-Of-Life operation	• Disassembly definition • Partial disassembly	Gungor et al.,1988 Moore et al.,1998
	• Disassembly and assembly time is important for recycle and recovery cost effective	NAVTN-CHANDRA, 1994 DeRon et al.,1995 Penev et al.,1996
	• Design's manufacturability can be measured by the efforts required in its manufacturing • Outdated products are mostly returned by customers • Recycling definition • Material recovery operations	Gupta et al.,1997 Thierry et al.,1995 Fleischmann et al.,1997 Gungor et al.,1999
	• Remanufacturing definition	Guide,2000 Lund,1996
	• Potential market demand for remanufactured products • Remanufactured product examples	Hatcher et al.,2013 Matsumoto et al.,2011
Product complexity assessment methods	• Assembly and disassembly time of a product is highly influence by the assembly contrains of each part	Boothroyd et al.,1992
	• Product complexity definition	Mathieson et al.,2010
	• Method to determine product complexity by its architecture graph	Mathieson et al.,2012
	• Utilizing weighting various factors, such as size and shape of parts to evaluse ease of disassembly • Developed a methodology to assess disassembly performance by comparing current design disassembly efficiency to a reference design of the same product • Developed a method to advise manufacturers on optimal disassembly plan based on company objectives and future retired product economic uncertainty • Define complete disassembly and selective disassembly	Desai et al.,2003 Kroll et al., 1996 Zussman et al.,1994 Güngör et al.,2002
Selective Disassembly	• Purpose of selective disassembly	Kwak et al.,2009
	• Proofs of selective disassembly helps to save cost and time	Kara et al.,2005 Gerner et al.,2005 Yi et al.,2008

	• Present a method for determining disassembly extent and EOL operations that can be shared for multiple products.	Behdad et al.,2010
	• Developed a methodology to generate optimal disassembly sequences for selected parts	Kara et al.,2006
	• Developed a program to generate selective disassembly sequence based on the 2D geometric relationship among parts	Smith et al.,2011

2.4 Graph Theories in Disassembly

As stated before, manufacturers or recycling companies can take every distinct part or component back for complete disassembly. Early studies on complete disassembly target on searching for all possible disassembling sequences [39], and a number of modeling strategies were proposed, i.e., Component-Fastener Graph, directed graph, AND/OR graph, and Disassembly Petri Net (DPN) [40].

Component-fastener Graph is proposed in 1997 by Zhang et al. [41-42], it is an undirected graph that can represent the component-fastener relationship among components. A component is a constituent part of a product and a fastener is used to attach one component to another, such as screws, rivets, inserts, etc. To simplify disassembly they divided the entire graph into cut-vertices and sub-graphs. The cut-vertex is defined as a vertex whose removal disconnects the graph and sub-graph is used to represent a group of vertices that is fastened to same vertex. They utilized Depth-First-Search (DFS) algorithm to determine cut-vertices and sub-graphs as well as utilized three dimensions coordinates to determine the removability of each vertex. The movable vertices could be disassembled first. Those blocked ones will be computed again as a new graph to determine their removability in iteration. The disassembly sequence of entire product is determined by this manner.

The component-fastener graph is good at representing product assembly complexity. However, the method developed by Zhang cannot identify fasteners automatically, the sub-graph is determined as group of parts that could be disassembled when the fasteners are removed. But the model proposed by Zhang does not provide a method to identify fasteners in a model instead they prescribed specific assembly method. They further present a method to generate a disassembly sequence using parts coordinates but do not determine part position relative to other parts in the model. Hence it is necessary to generate method that identifies fasteners and part position automatically.

Unlike component-fastener graph, AND/OR graph is a directed graph [43]. Directed graph implies it is a hierarchical graph, the beginning could be part and points to assembly and also could be assembly points to simple part. Therefore AND/OR graph could be utilized to plan either assembly sequence or disassembly sequence. AND/OR graph provides an equivalent representation of all plans, eliminates duplicate nodes when representing all feasible plans and provides the basis for planning by tree search methods [36]. Homem de Mello et al. developed an algorithm to generate all possible assembly sequences using AND/OR graph [43-44].

In Homen de Mello et al. [43-44] algorithm any non-empty set of parts that are joined to form a stable unit is called an assembly. An assembly made up of certain number of parts can be characterized by the set of parts plus a four-dimensional homogeneous transformation matrix to distinct different assemblies but made up of the same set of parts. Subassembly of an assembly can be marked by the same coordinate transformation matrix with the assembly. The configuration of a given assembly is a set of sub-assemblies. Under such principal, every configuration member characterizes a connected sub-assembly, every part of the assembly is one of the elements of the configuration and not part of the assembly belongs to two configuration elements. The space of all possible configurations of parts is a subset of the assembly, which is the set of all partitions of the assembly. And the assembly plan can be seen as a sequence of configurations. In [43-44] they also proposed that disassembly sequences could be generated by their algorithm based on the assumption that disassembly tasks are the inverse of feasible assembly tasks.

Another type of graph that has been used in assembly/disassembly planning is Petri Net [45]. Petri net (PN) is a graph formed by two kinds of nodes, namely places (p_j) and transitions (t_j), these two kinds of nodes are connected by directed edges, called arcs. A non-negative integer is assigned to each place as number of tokens, and it can vary based on the state of the Petri net. Each arc has a weight, which is assigned as a positive non-zero integer. Suzuki et al. [46] proposed a methodology which was based on PN, namely Assembly marked Petri Net (APN). Suzuki defined APN as a four-tuple and utilized linear programming to find the optimal assembly or disassembly sequence. Zussman et al. [47] extended APN into Disassembly Petri net (DPN), which was defined as a nine-tuple, for planning disassembly processes with a guarantee to maximize the parts obtained for repair and reuse and to minimize the disposal quantity when benefit of a subassembly or part and cost of a disassembly operation is fixed. Their framework enables to derive optimal disassembly process plan when even the desired disassembled parts are different within same discarded product. Tiwari et al. [48] proposed a cost-based index methodology to generate disassembly decision according to manufacturers' selection, utilizing Disassembly System Decision Petri Net (DSDPN) graph.

Cao et al. [49] extended previous work and developed AND/OR graph into a net structure by mapping into a PN. This modified PN presents a framework for task sequence planning for a general robotic either assembly or disassembly work by using AND/OR net. They changed the original AND/OR graph from a directed graph into an undirected graph, the AND-arc which connects nodes in the graph are defined as bidirectional which means both directions may be feasible. In assembly task implementation, the AND-arc represents the feasible decomposition from a sub-assembly to a corresponding set of sub-assemblies. They also add a new arc, namely IST-arc, which represents nodes internal state transition, and in the assembly task implementation, it represents the feasible internal state transition from a subassembly to another subassembly. In PN graph, they utilized a five-tuple, which is constructed by places, transitions, arcs directed from places to transition, arcs directed from transition to places and marks. After building these two graphs, they defined a function to connect the undirected AND/OR graph to directed PN graph to construct a directed AND/OR net. In generating assembly work sequence, they utilized Breadth-First Search (BFS) algorithm first to find all possible operation sequences and then utilized certain evaluation criteria, such as cost, number of steps to find the optimal operation sequences.

AND/OR graph and PN graph are good at determining sequences. AND/OR graph can determine the proper disassembly/assembly sequence based on the assumption that each step has to be achieved during the disassembly/assembly process. However, in practical disassembly, removing parts step by step is time consuming. PN graph is good at generating operation sequence. However in disassembly process, operations are not prescribed as assembly, such as the operation of locating parts to the specific place is required in assembly but is not necessary in disassembly.

Below is a brief summary of literatures discussed above:

Table 2.2 Table of graph theory reference

Type of graph	**Node**	**Edge**	**Reference**
Component-fasteners graph	• Nodes represent components	• Edges represent a group of assembly relationships among components.	Zhang et al.,1996-1997
Assembly graph	• Nodes represent components	• Edges represent direct assembly relationship	Mathieson et al.,2010
Bi-partite graph	• Two kinds of nodes, one represents system element, such as contact, or fasteners, the other one represents components	• Edges represent connections between contact or fasteners and components	Mathieson et al.,2010
AND/OR graph	• Nodes represent possible states of assembly process	• Edges represent possible transitions between states of the assembly processes	Homen de Mello et al.,1990-1991
	• Two kinds of nodes. Node T represents events, Node P represent conditions	• Edges, called arcs, connect P to T and T to P	Reddy et al.,1993

Petri Net (PN)	• Assembly Petri Net (APN): Node P regards to subassembly, node T regards to transition,	• Edges represented as an adjacency matrix D to connection between P and T	Suzuki et al.,1993
	• Disassembly Petri Net (DPN): two types of node P, P1 regards to product/root without input, P' regards to part without output. Defined three values (cost, decision and probability) for node T to generate disassembly plan	• Edges, called arcs, connect P to T and T to P	Zussman et al.,1999
	• Disassembly System Decision Petri Net (DSDPN): developed P' in DPN into a set of final parts which treated by disassembly systems. T regards to decision step and has five output edges	• Edges, called arcs, connect P to T and T to P	Tiwari et al.,2002

Graph combined AND/OR graph and PN	• Node in AND/OR graph represent object states; • Two kinds of nodes in PN graph P for place and T for transition	• Two kinds of edges in AND/OR graph, AND-arc regards fesible decomposition from last state (node) to next state (node); IST-arc regards to feasible internal transition from last state(node) to next state(node); • Edge in PN graph regards to connection from P to T and T to P	Cao et al.,1998

2.5 Redesign for ease of disassembly

Redesign suggestions for ease of disassembly at design stage will help in design optimization. These suggestions should point out the weakness and give design suggestions towards the purpose of making disassembly easy.

As for assemblies are constructed by connected parts, easily detachable fasteners makes disassembly work easily. As K. Saitou concludes two design principals in [50] that aids in design which can be easily disassembled. One method is using those heat-reversible locator-snap systems. These snaps are easily detached when heat is added to expend the locators, thereby releasing the snaps. This kind of disassembly is easy, non-destructive, and clean detaching. The other method is using self-disintegrating assembly. Parts within this assembly are constrained by the locators (tabs, slots, lips, rests, etc) integral to the parts, therefore when one or few fasteners are removed, it would cause the assembly to self-disintegrate in a desired sequence. If one part has all the fasteners then all the other parts can be removed by just removing all the fasteners on that specific part. When remove the fastener that holding the specific part, all the other parts are removable. In this way labor cost for removing fasteners will be saved as well as disassembly time will be reduced.

2.6 Graph Algorithm

As stated above, relationships among parts in an assembly can be regarded as a set of connections between part pairs. In many computational applications, people use abstract objects called graphs to model such connections [38].

A graph is defined as a set of vertices and a set of edges that connect pairs of distinct vertices (with at most one edge connecting any pair of vertices), depending on different connection patterns, graph types can be classified as connected graph and connected components. Connected graph refers to the graph that contains a path from every vertex to every other vertex, and connected components refers to the graph that contains some sub-graphs that there is no path from this sub-graph vertex to any vertex in the graph that is not in this sub-graph.

For the purpose of solving graph problems using computational programming, two basic aspects are important: one is how to manage random vertices and edges, the other one is by what way is proper to manage these data. The following section 2.6.1 and 2.6.2 are literatures related to each question, respectively.

2.6.1 Graph Building

According to [38], the first step to develop an efficient algorithm in solving a given problem is to implement a simple algorithm that solves the problem. This is the basic but important step to guarantee correctness of data. All data from a sauce could be either orderly or randomly, if a more complicate program was called for, then the simple implementation can provide with a correctness check for small cases.

The first step is to save the entire inputs data and then to write a function to pass through them to try to discover whether the current object is connected to the next object. Because there is no simple method can suggest itself for determining whether two objects are connected from the set of the entire vertex array, [38] advises to use an array to hold the requires information to be able to implement union and find. The basis of this algorithm is an array of integers with the property that object a and object b are connected if and only if the specific ordered array entries each two object are equal.

2.6.2 Graph Implementation

Graph could be represented on computer as an adjacency list or an adjacency matrix. In adjacency list, vertices are numbered from zero to the sum of vertices minus one, each vertex stores a linked lists consisting of all of his successors. In adjacency matrix, vertices are in same way with adjacency list but in each vertex where stores a matrix (two-dimensional array) with size N×N, where N is the number of vertices. This means that for each edge between the vertices i and j, there will have the value of 1 ($A_{[i][j]} = 1$), and 0 otherwise[51].

G. Rossum suggests Python Patterns to implement graph in [52] by utilizing a hash table, which is a data structure in computer science to implement an associative array [53], to associate each vertex in a graph with an array of adjacent vertices. In this algorithm, vertex is represented as an object that containing the nodes that are connected by a direct arc from the vertex. Edges are not represented as objects.

Goodrich and Tamassia suggest an object oriented list structure in [54] that both vertex objects and edge objects have special class. Each vertex object has an instance variable pointing to a collection object that lists the neighboring edge object. In turn, each edge objects points to the two vertex objects at its endpoints. This version of the adjacency list uses more memory than the version in which adjacent vertices are listed directly, but the existence of explicit edge objects allows it extra flexibility in storing additional information about edges [55].

2.6.3 Graph Traversing and Breadth-First Search (BFS)

Visiting each vertex and edge in a graph through a systematic way is one of the most fundamental graph problems [56]. A useful traversing algorithm must be powerful enough to get out of an arbitrary graph and this algorithm should satisfy both efficiency and correctness. Every vertex in a graph can be represented as a node in programming. To satisfy efficiency the algorithm should prevent repeat visiting node and to satisfy correctness the traversal should go in a systematic way to guarantee the algorithm get out of the graph. The search should pass through every node in the graph.

Breadth-first search (BFS) algorithm is usually used in undirected graph. BFS is limited to essentially two operations: (a) visit and inspect a node in a graph; (b) gain access to visit the nodes that neighbor the currently visited vertex [57].It begins at a "root" node and inspects all the neighboring nodes. Then for each of those neighbor nodes in turn, it inspects their neighbor nodes which were unvisited, and so on. These traversal properties reveal that each path in the graph must be the shortest path in the graph [56].

Chapter 3 Automatic Disassembly Time Estimation and Redesign Suggestions Generation

3.1 Mathematical Model of Assembly Time Estimation

Summers et al. [23] developed an equation to estimate assembly time from the representation of product architecture by modeling complexity of assembly. Based on the hypothesis that disassembly can be considered as inverse process of assembly process [43] this equation could be used to estimate disassembly time for a product. The constant n in equation (1) is defined as under no constraint situation each part removal takes one second. The equation is shown below:

$t_d = (APL \times n^{1.188+PLD})$ ....(1)

where, $APL = \frac{TPL}{n\times(n-1)}$ ....(2)

$PLD = \frac{APL}{N}$ ....(3)

where , *n*: Total Number of parts

TPL: Total Path Length

N: Number of Relationships

APL: Average Path Length

PLD: Path Length Density

During manufacturing of parts, assembly can be seen as building connections among these parts. Therefore their connective relationships construct the product assembly graph. There are four parameters required to calculate the assembly time: Total Path Length (TPL), Number of Relationships (N), Average Path Length (APL), and Shortest Path Length Density (PLD). All these parameters are utilized in measuring graph complexity [24] by translating connection relationships into rational design structure matrix (rDSM) which is an array based hyper-graph representation capturing relationships between multiple elements through a single instance and element pairs that are related through multiple relationship instances [24].

Matrix is a square whose size can be measured by square the number of one dimension element, TPL is the summation of path length of elements. Each single element represents a part, which has its own connection pattern so that its path length is the summation of all the shortest path length that the others parts connect to it. Shortest path length is defined as number of counting connections from the rooted part to all the other parts in an assembly, the minimum amount of connection of each pair is the shortest path length. Number of relationships (N) is defined by Summers et al. [23] as total number of connection types of parts in an assembly. Single part may have many various connection types with other parts. APL is measured by dividing the TPL by the size of the matrix minus the unused diagonal, which is equal to total number of elements in one dimension. PLD serves as a test for the level of interconnection created on average by each connection type so that it is measured by dividing the APL by the number of relationships [24].

Raghunathan S. [5] utilized the above equation to estimate the assembly time of a practical standard toaster and a practical eco-toaster. He manually disassembled these two toasters and created their assembly graphs. These assembly graphs were then utilized to compute the metrics needed to estimate disassembly time from equation (1).

3.2 Overview of the Implementation

The focus in this chapter is to accomplish automatic disassembly time estimation and generate redesign suggestions for any given Solid Works 3D assembly model containing mates. This purpose is achieved through a program, which was developed using Solid Works Application Programming Interface (API) in Visual C#. Although, Solid Works is chosen to demonstrate the algorithms, any other CAD system with API access can be used to implement the algorithms.

The high-level algorithm is shown in Figure 3.1. It includes both complete disassembly time estimation and material selective disassembly time estimation. The complete disassembly time estimation could be done by counting part number and the connection path length for each part in an assembly graph as Figure 3.2 shows, and calculate all parameters, such as TPL and APL and that required estimating assembly time in [23]. The connection relationships between each two parts are assumed and represented by bi-partial in Figure 3.3. The rDSM for this assembly graph example is shown in Table 3.1 and the processes of estimating disassembly time are shown in this table. When set M1 be the selective material to classify the assembly graph example in Figure 3.2, a simplified assembly graph is obtained and shown in Figure 3.3. Its relative connection relationships are represented in bi-partial and shown in Figure 3.4. The rDSM for material classified assembly graph is shown in Table 3.2 and the material selective disassembly time estimation processes are shown in the table.

As a case study for selective disassembly, material is chosen as one category based on which components are grouped for selective disassembly. The high-level algorithm begins by building graphs, for a given assembly model, and then utilizes disassembly time estimation equations [23] to estimate time. The values of the variables in the equation are computed form the graph of the assembly model. User can select either complete disassembly or material selective disassembly through a user-interface (UI).

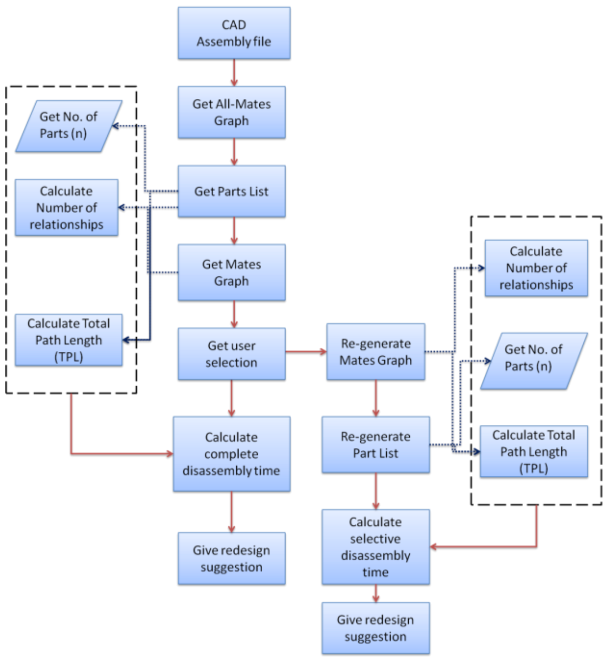


Figure 3.1 General algorithms for automatic disassembly time estimation and redesign suggestions.

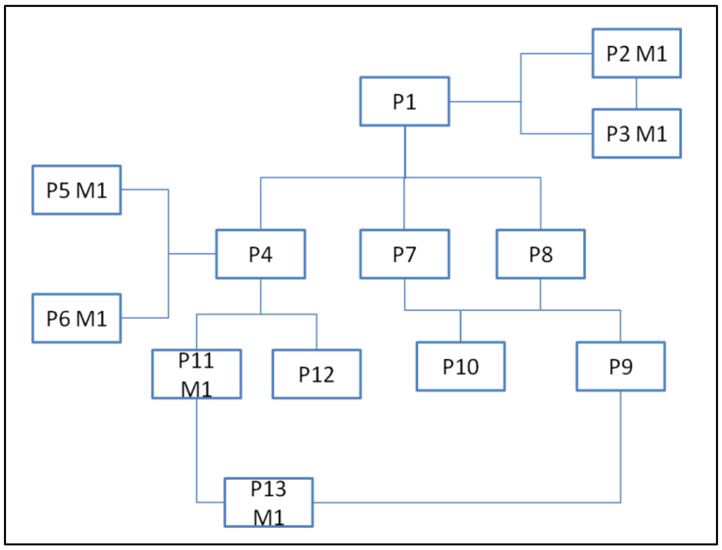


Figure 3.2 Example of Assembly Graph for Complete Disassembly

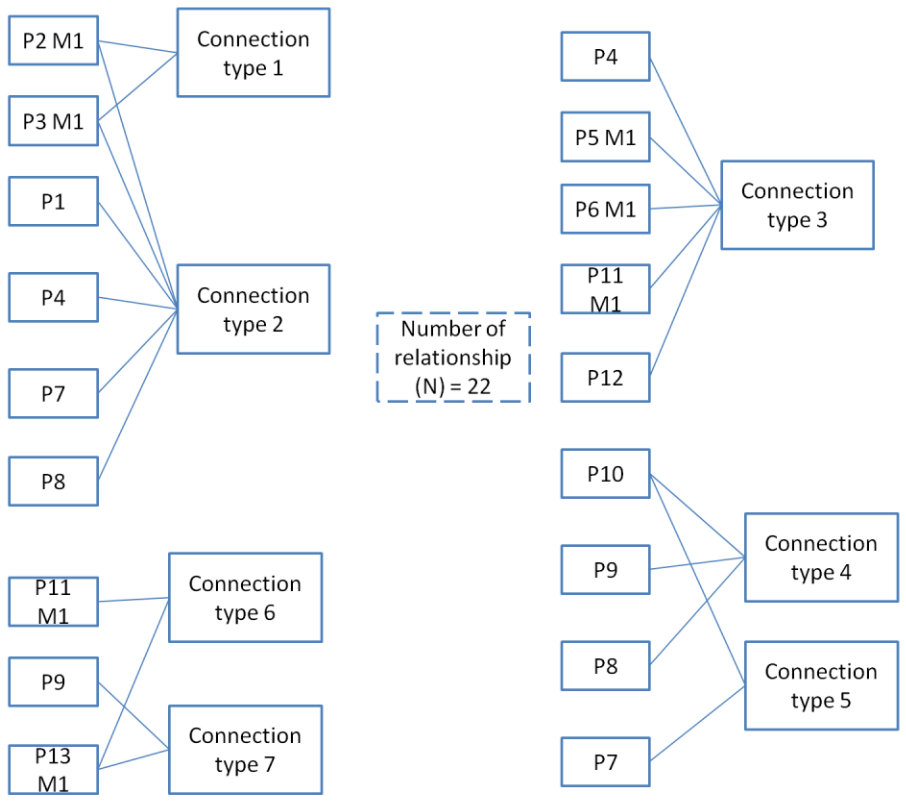


Figure 3.3 Example of Bi-Partial graph for Complete Disassembly

Table 3.1 Example of complexity metrics and complete disassembly time estimation process

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	Path Length(PL=$\sum_1^{13} P_n$)
P1	0	1	1	1	2	2	1	1	2	2	2	2	3	20
P2	1	0	1	2	3	3	2	2	3	3	3	3	4	30
P3	1	1	0	2	3	3	2	2	3	3	3	3	4	30
P4	1	2	2	0	1	1	2	2	3	3	1	1	2	21
P5	2	3	3	1	0	2	3	3	4	4	2	2	3	32
P6	2	3	3	1	2	0	3	3	4	4	2	2	3	32
P7	1	2	2	2	3	3	0	2	3	1	3	3	4	29
P8	1	2	2	2	3	3	2	0	1	1	3	3	2	25
P9	2	3	3	3	4	4	3	1	0	2	2	4	1	32
P10	2	3	3	3	4	4	1	1	2	0	4	4	3	34
P11	2	3	3	1	2	2	3	3	2	4	0	2	1	28
P12	2	3	3	1	2	2	3	3	4	4	2	0	3	32
P13	3	4	4	2	3	3	4	2	1	3	1	3	0	33
Total Path Length (TPL)=$\sum_1^{13} PL_n$ = 378														
Average Path Length(APL)= TPL/(n×(n-1))=378/(13×12)= 2.423														

Path Length Density (PLD)= APL/N=2.423/22=0.110
Disassembly Time $t_d = (APL \times n^{1.188+PLD}) = 2.423 \times 13^{1.188+0.110} = 67.64\ sec$

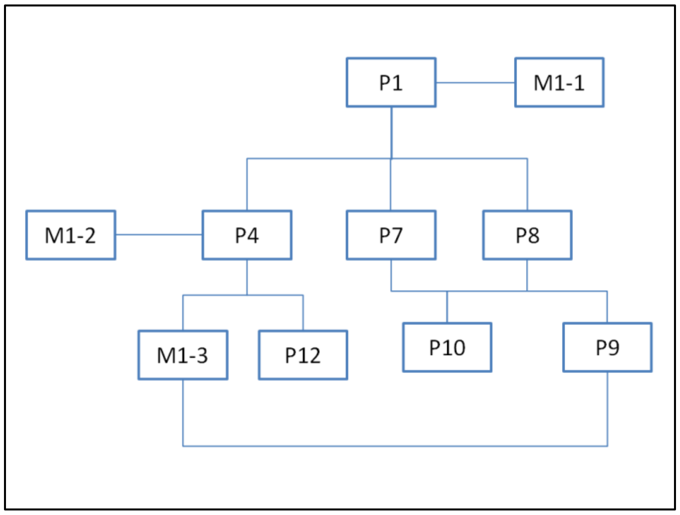


Figure 3.4 Example of Assembly Graph for Material Selective Disassembly

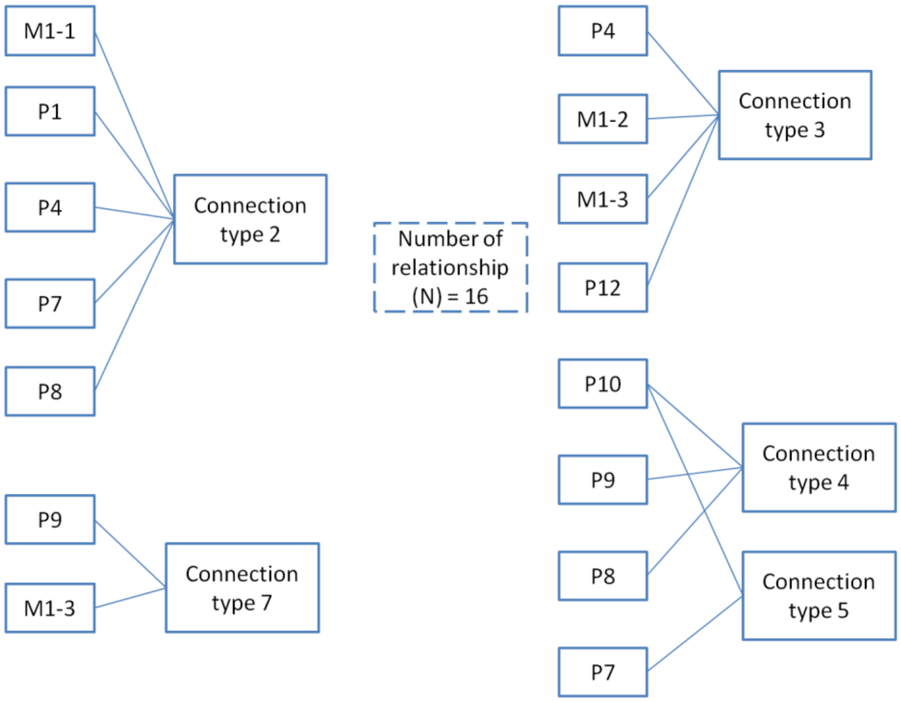


Figure 3.5 Example of Bi-Partial graph for Material Selective Disassembly

Table 3.2 Example of complexity metrics and material selective disassembly time estimation process

	P1	M1-1	P4	M1-2	P7	P8	P9	P10	P12	M1-3	Path Length(PL=$\sum_1^{10} P_n$)
P1	0	1	1	2	1	1	2	2	2	2	14
M1-1	1	0	2	3	2	2	3	3	3	3	22
P4	1	2	0	1	2	2	3	3	1	1	16
M1-2	2	3	1	0	3	3	3	4	2	2	23
P7	1	2	2	3	0	2	3	1	3	3	20
P8	1	2	2	3	2	0	1	1	3	2	17
P9	2	3	3	3	3	1	0	2	3	1	21
P10	2	3	3	4	1	1	2	0	4	3	23
P12	2	3	1	2	3	3	3	4	0	2	23
M1-3	2	3	1	2	3	2	1	3	2	0	19
Total Path Length (TPL)=$\sum_1^{10} PL_n = 198$											
Average Path Length(APL)= TPL/(n×(n-1))=198/(10×9)= 2.2											
Path Length Density (PLD)= APL/N=2.2/16=0.138											
Disassembly Time $t_d = (APL \times n^{1.188+PLD}) = 2.2 \times 10^{1.188+0.138} = 46.60\ sec$											

From the disassembly time results in Table 3.1 and Table 3.2, it can be seen that when using material selective method to simplify the assembly graph, the disassembly time reduces from 67.64 seconds to 46.60 seconds, the reduction percentage is approximate 31.1%.

Section 3.3 details the algorithm for building and re-building graph from a given Solid Works assembly model; section 3.4 presents the algorithm for automatic estimation of disassembly time; section 3.5 details the algorithm for generate re-design advices to the designer.

3.3 Assembly Model Graph

Assembly architecture can be complex consisting of hundreds components or even subassemblies or be simple that consists of just ten parts. In Solid Works, an assembly is built by adding a part, bottom-up approach, or creating the parts in-place in the assembly document, top down approach. Creating assembly generates connection relationships between the assembly and the part, such as making align relationship between the part and the assembly when one of the part sides aligns to one of the assembly sides. Once the relationship between two objects (could be either component or subassembly) is defined, mates are then created to represent the corresponding geometric relationship between parts (or sub-assemblies or coordinates systems) within the assembly. Each mate consists of two objects. These objects could be parts, sub-assemblies or coordinate systems.

According to these properties, assembly architecture in Solid Works shows similar properties with component-fastener graph [41-42] which discussed in Chapter 2. Solid Works provides a list of assembly mates. Each mate, as state earlier, is between two parts (or subassemblies or coordinate systems). This property helps building the connection relationship graph when utilizing algorithm is in [38]. Because selective disassembly time estimation is one of the purposes of this chapter, it is necessary to re-construct assembly graph after material selection is made.

Section 3.3.1 is details the construction of the assembly graph from Solid Works assembly model and section 3.3.2 discusses reconstruct graph from a built graph.

3.3.1 Assembly Graph Generation

The first step is to extract all mates from the Solid Works assembly file. Solid Works API support functions to access mate information like component name and mate type, etc. Solid Works defines nine kinds of mate [58], they are: coincident, parallel, perpendicular, tangent, concentric, distance, angle, align and anti-align.

Because list is a proper data structure to store information for graph, linked-list feature is utilized to store accessible mates in assembly file. A linked-list class is created to store nodes and to do searching and computation. A node class is created to store sufficient component information and its members include component name, raw materials, mate type etc. Figure 3.6 shows a diagram for node class.

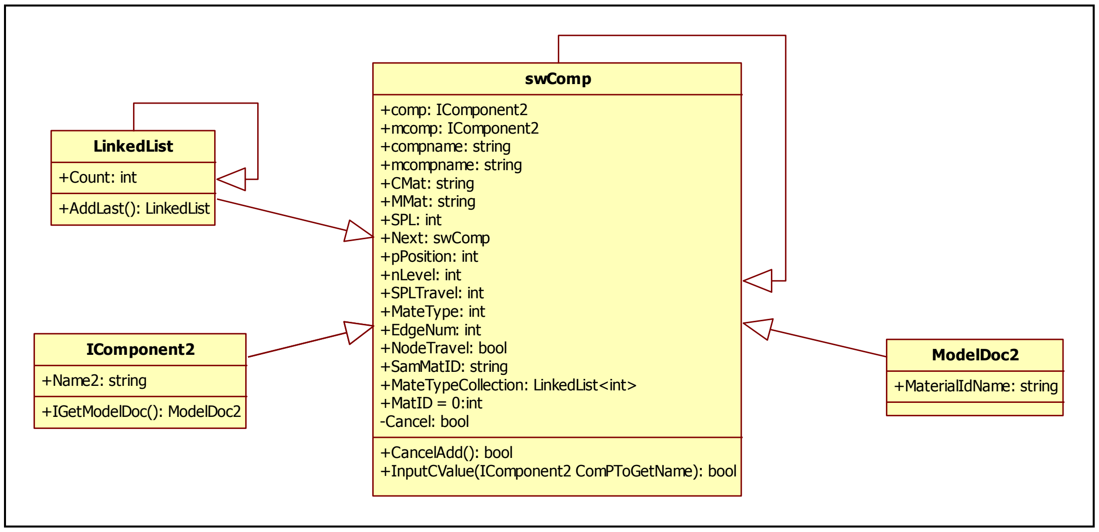


Figure 3.6 Node class

The algorithm begins by passing through the feature manager tree, whose access is provided through the API, to identify the mate feature. Within the mate feature, the algorithm identifies the information regarding mate type, mate parts, their part ID's and raw material and copies the information into a new node representing the particular mate feature. This new node is then added to a liked list called all-mate-list.

The above process ends when all mates are added to the all-mates-list. In each node of the all-mates-list, there are two components - part A and part B. By traversing through each node of all-mates-list, each component that is traversed for the first time is added to a new list of components, called component-list. The visualized process flows are shown in figure 3.7.

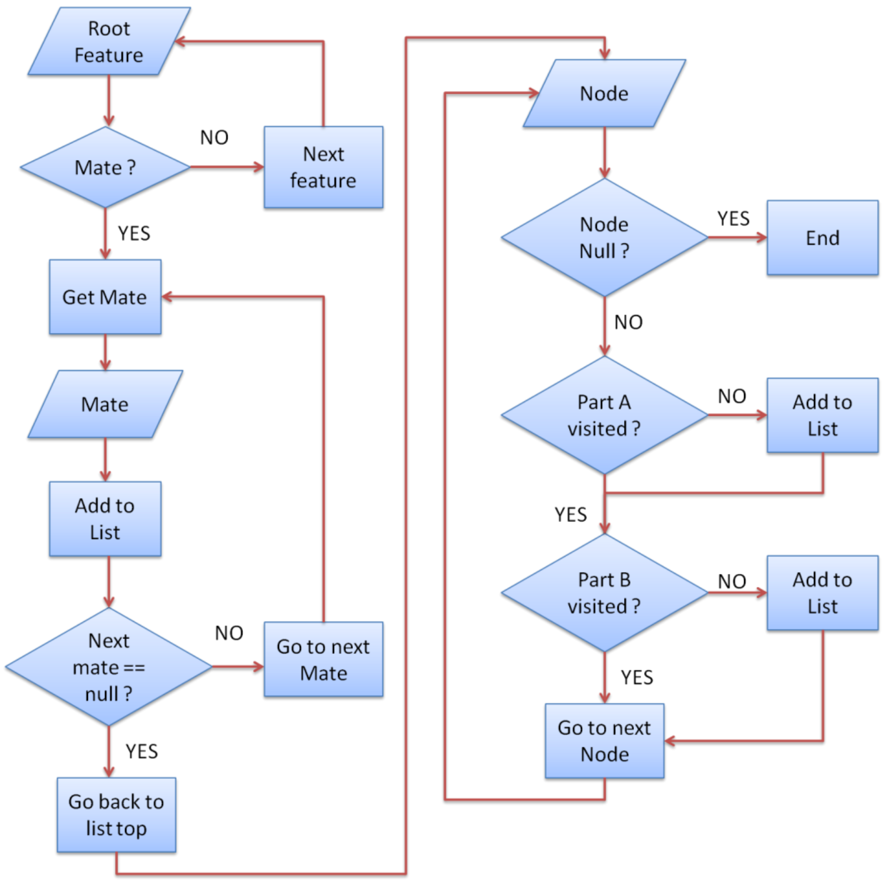


Figure 3.7 Algorithm for obtaining mate graph from an assembly model

The All-Mate-List is further cleaned up to avoid any duplicate mates. The cleaned up list is called Mates-list. The visualized process flows are shown in Figure 3.8.

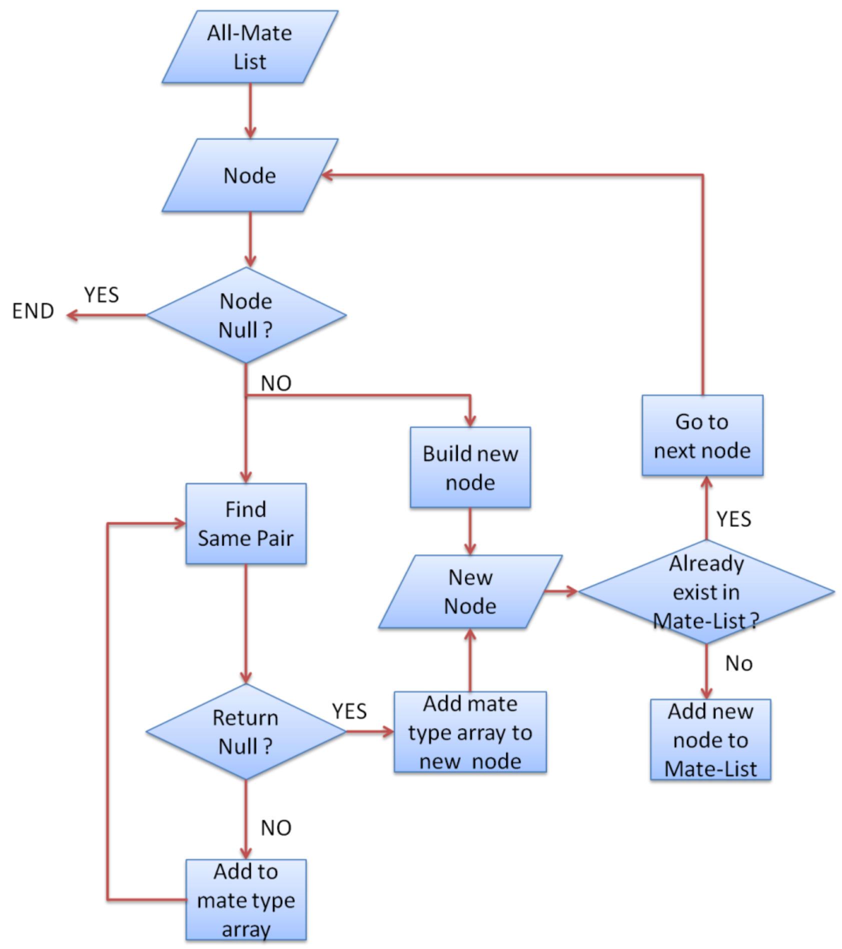


Figure 3.8 Algorithm for building an assembly graph from an existing mate graph

3.3.2 Assembly Graph Re-generation

Assembly graph re-generation is required to support selective disassembly time estimation. The main idea as explained in section 3.3.1 is that in selective disassembly only particular parts or groups of parts are required to be disassembly. As an example, the purpose, for disassembling only a few components or groups of components, could be to remove hazardous components before land filling the rest of the assembly or recycling a group of components with the same material.

We have chosen material as a basis to group components together – which implies merging the component nodes, into one node, in the assembly graph. Therefore, assembly graph re-generation is required. Solid Works allows designers to define component raw materials. In graph building process (described in section 3.3.1) each node stores raw material information for the corresponding components and this information will now be utilized to mark and group components together. In addition to raw materials, another condition required to merge components nodes is that the nodes with the same raw material should be direct neighbors in the assembly graph (have a mate between one another). When the grouping components and merging of nodes is accomplished for the whole assembly, usually a simpler graph is now generated. This new graph is used to compute the variables in the equation [23]. The estimated disassembly time with the new graph is the selective disassembly time based on material recovery.

To implement above graph algorithm, mate-list that is generated in last section is utilized as original graph. Before regenerating the graph according to material recovery it is necessary to know number of different raw materials in the assembly. If all components share same raw material then the selective disassembly time is zero because the entire retired product could be put into material recovery. An array is built to store each type of raw material information including the part ID of the components that have the particular raw material. Two possibilities for the assembly graph arise based on the required re-grouping for material recovery:

1. The nodes for same type of materials are all neighbors with each other. This situation is illustrated in Figure 3.9. Nodes 1,2,3,5 and 7 are of material M1. Nodes 12 and 13 are of material M2. Nodes 8,9 and 10 are of Material M3.

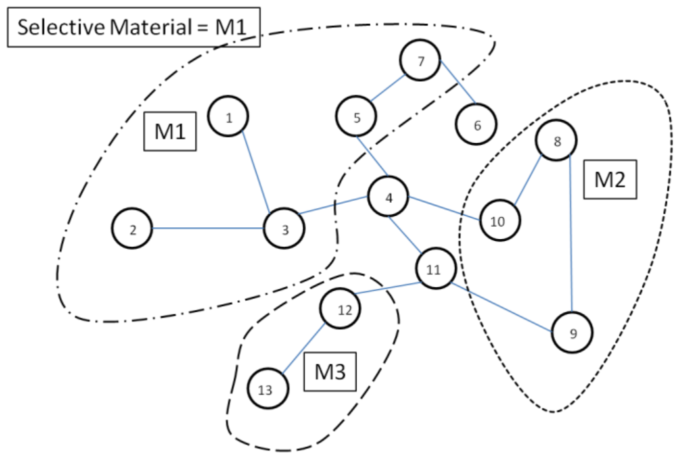


Figure 3.9 Nodes of same material are directly connected together

2. The nodes of same material are distributed in different places, some of them are neighbors (directly) while others are not. This situation is illustrated in Figure 3.5.

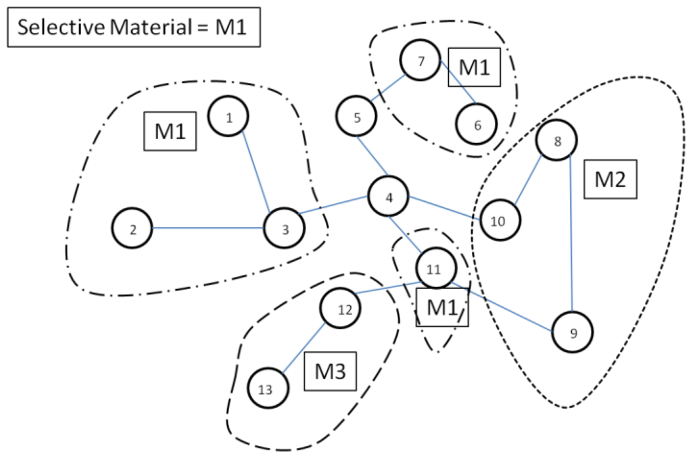


Figure 3.10 Selective Material Parts Distribute in Different Places

In situation 1, all M1 raw material parts can be grouped together, and the assembly graph change as shown in Figure 3.11. On the other situation, M1 raw material parts are distributed in three different places so that they are grouped into three groups. This is shown in Figure 3.12.

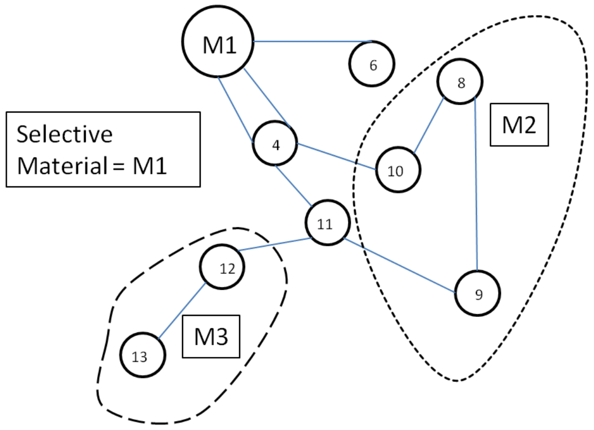


Figure 3.11 Re-generated Graph for Selective Material Parts Connect Together

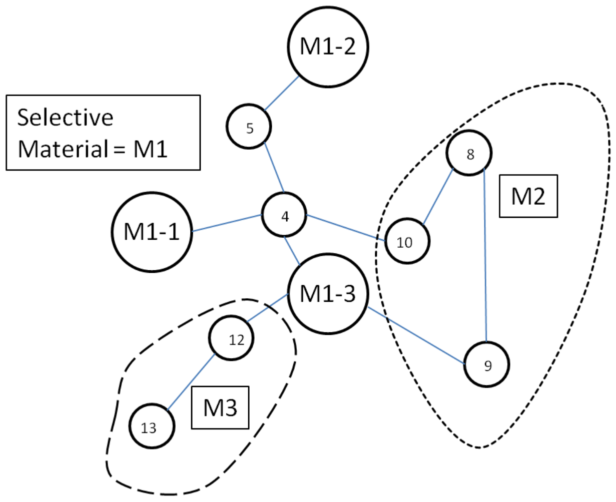


Figure 3.12 Re-generated Graph for Selective Material Parts Distribute in Different Places

The illustrations shown above indicate through grouping parts in an assembly graph can reduce graph complexity in both connection edges and vertices. In both situations the summation of all parts is 13. But after grouping in different ways, in situation 1, the parts summation reduces to 9; in situation 2, the parts summation reduces to 10.

In another hand, the label work for new group in situation 1 is simpler than the work in situation 2. But in most cases raw material distributions are more likely to situation 2 therefore the regeneration algorithm considers situation 2 primarily. There are two main steps to attain the purpose.

The first step is to group the selective material parts and the following step is to re-name parts involving in one group.

In first step the selective material is obtained from user's selection so that a user interface is required to present raw material list for user and receive the raw material that user wants to recover. Solid Works API environment only allows classes included in windows form designer namespace so that the user interface is built in this namespace class with Visual C#. Combo box function is utilized in the user interface to present raw materials for user and to send the selected raw material back to mate-list. If the user keeps selection empty the program would take it as a completely disassembly. The raw material list building algorithm is shown in Figure 3.13. Component raw materials are stored as string type in material list. As shown in Figure 3.13 this algorithm gives top priority to same material mate because such consideration can improve the algorithm efficiency, or it has to visit each part in every node. Another purpose to build the same material mate list is based on the reason that if a part's material same with user's selection but does not connect with any same material part then this part will be treated as single part in grouping process, just like part 11 shown in Figure 3.10 after grouping component it still be counted as a part M1-3 as shown in Figure 3.11.

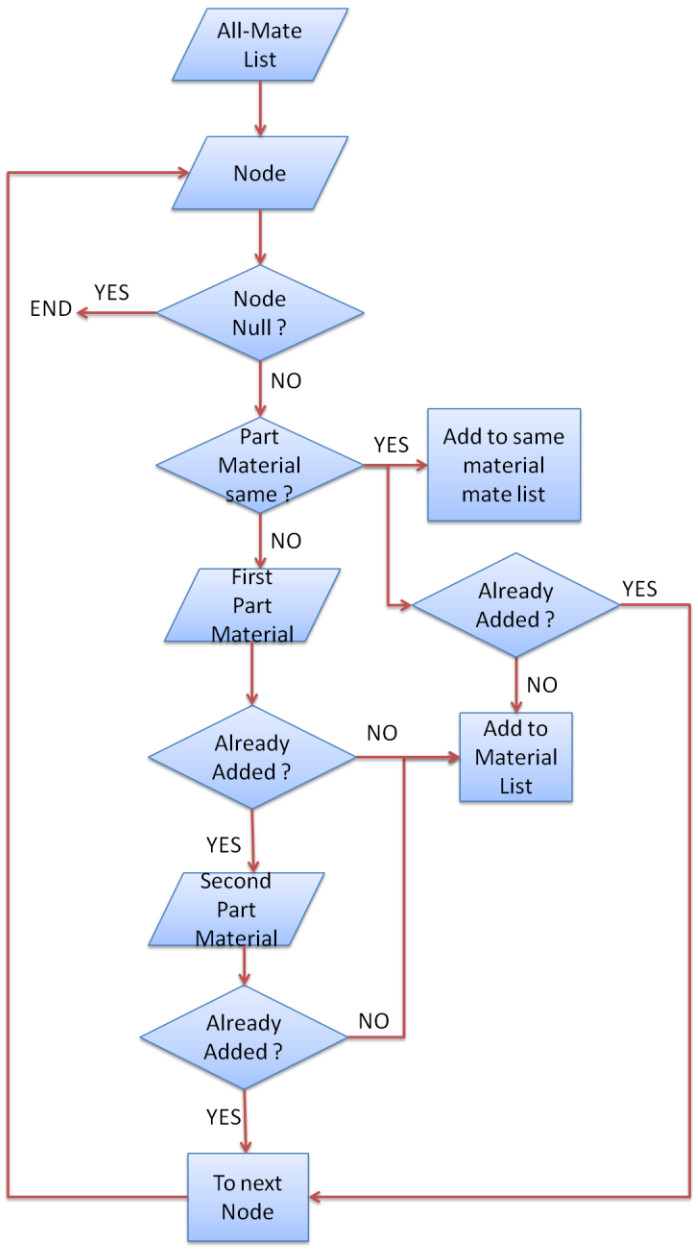


Figure 3.13 Building Raw Material List

When the user selects material the user interface sends this string type information to the same material mate list. The algorithm goes through the same-material mate-list when the material is same with user's selection. It checks whether these two parts visited before if not the algorithm adds these two parts to a temporary list. An integer type augment is used to label components that connected together as a group.

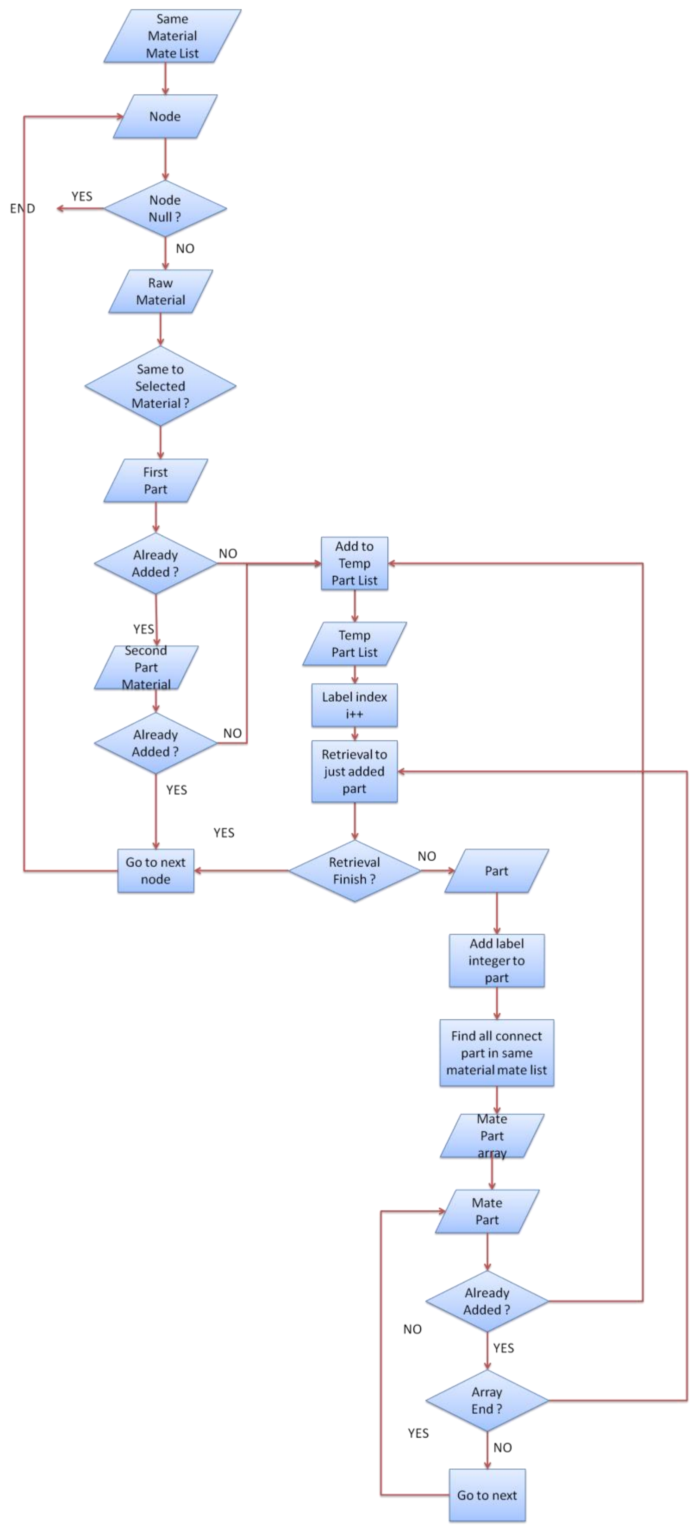


Figure 3.14 Grouping Parts Based on Selected Raw-Material

As shown in Figure 3.14 after labeling, the program retrievals to new-added parts, adds the label integer to the part node and finds all its connected parts from mate-list. If the program finds some new connect part it would add the new one to temp list. This retrieval process goes in a loop until all new-added parts visited. The label use integer i adds one automatically when new same material node is found from mate-list. When the process finishes the final numeric value of label use integer represents total number of groups.

After grouping components the algorithm continues to re-name them. This process changes part's original into the combination of material name and label integer just as M1-1, M1-2 and M1-3 shown in Figure 3.12. The process re-sets connection edges between same group parts as zero but keeps the connection edges with all the other parts.

3.4 Disassembly Time Estimation

This section describes the algorithm to compute the variables in equation (1) from the assembly graph's build in section 3.3 the algorithms described work for both completely disassembly time estimation and material selective disassembly time estimation. This is because the algorithm starts from mate-list and part-list and focuses only on the information in the nodes. .

3.4.1 Total Number of Components

The total number of parts in an assembly is counted of each single part, excludes sub-assembly. This value is obtained by courting total number of nodes in the component-list.
The total number of nodes in this list is the total number of parts in an assembly. Another usage of this component list is to calculate total path length.

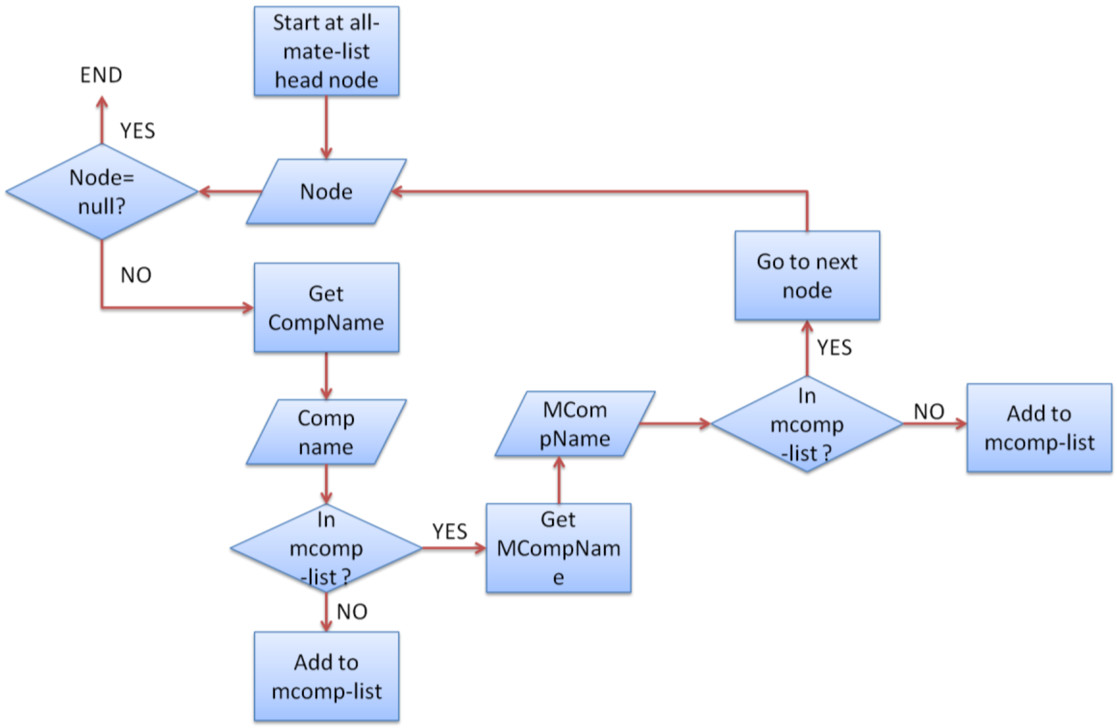


Figure 3.15 Obtaining total part number of in an assembly

3.4.2 Number of Connection Relationship

Connection relationships among parts are represented by bi-partial graphs and are classified into four types: surface contact, fasteners, snap and interference fit and others.

These relationships are observed through the product physical assembly structure and are different in CAD assembly model. Take two parts that are assembled by a screw and a nut as an example, such as shown in Figure 3.16. In [23] part 1 and part 2 have a surface contact relationship and all four parts have another bolting relationship. Surface contact relationships between part 1 and part 2 counts to two and bolting relationships between these four counts to four. The total connection relationship number of these four parts is six.

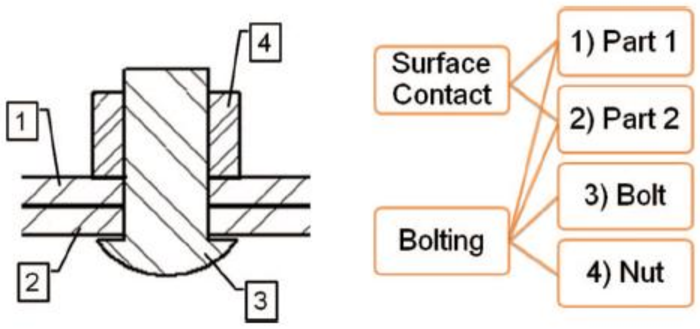


Figure 3.16 Bolting Diagram [23]

However such relationships cannot be defined directly in Solid Works assembly. Part 1and part 2 should be aligned, made their screw holes concentric and their surface coincident. And the screw should be concentric to part 2's screw holes center and its surface should be coincident to part 2. The nut should be concentric to the screw center and coincident to part 1. Their connection relationships in Solid Works can be represented as Figure 3.17. And their total connection relationships are six as same in physical observation shown in Figure 3.16.

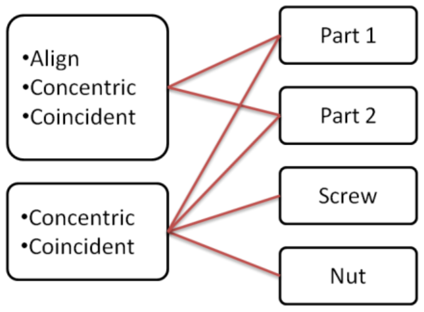


Figure 3.17 Connection Relationships in Solid Works Assembly

In this research the connection relationship type definition is according to the combination of Solid Works mate types. As shown in Figure 3.17 every pair has a mate type list which is consisted by mate types. The algorithm of connection relationship calculation is shown in Figure 3.18 This algorithm starts from mate-list counting total mate types for each pair and concludes to a total-mate-types array regardless same mate type summation. The total-mate-type is defined as total number of mate types in every pair. As shown in Figure 3.17in left part the first mate type list has three mate types and the other has two. These two values are their total mate type. The purpose for building this array is to improve counting efficiency. Because in assembly graph all pairs are listed randomly and every pair's total mate type list is possible different to its neighbor. Moreover even though the number of total-mate-types may be the same, the actual mate types inside the mate-type-list may be different.

Hence the following steps are utilized starts to filter mate type list of Node A in Figure 3.18. Two conditions have to be satisfied at same time they are: 1) same total-mate-type with which in total-mate-type array at current traversal position; 2) parts in the nodes are different. The second condition is considered for re-generated graph. Because in selective material grouped graph, it is possible the two parts in a node are same with the selected material. After this first filtering the algorithm keeps Node A and continues to find another node that has the same total-mate-type value which called Node B. If Node A and Node B had totally different mate types the connection relationship will adds two and labels Node B. (Because Node A and Node B are in same mate list which means when traversal to labeled Node A the algorithm continues.) If they have totally same mate types the algorithm goes and checks whether same part exist, if so the connection relationship adds one if not the connection relationship adds two. When all qualified Node B visited the algorithm adds two to connection relationship and label current Node A as shown in Figure 3. 18.

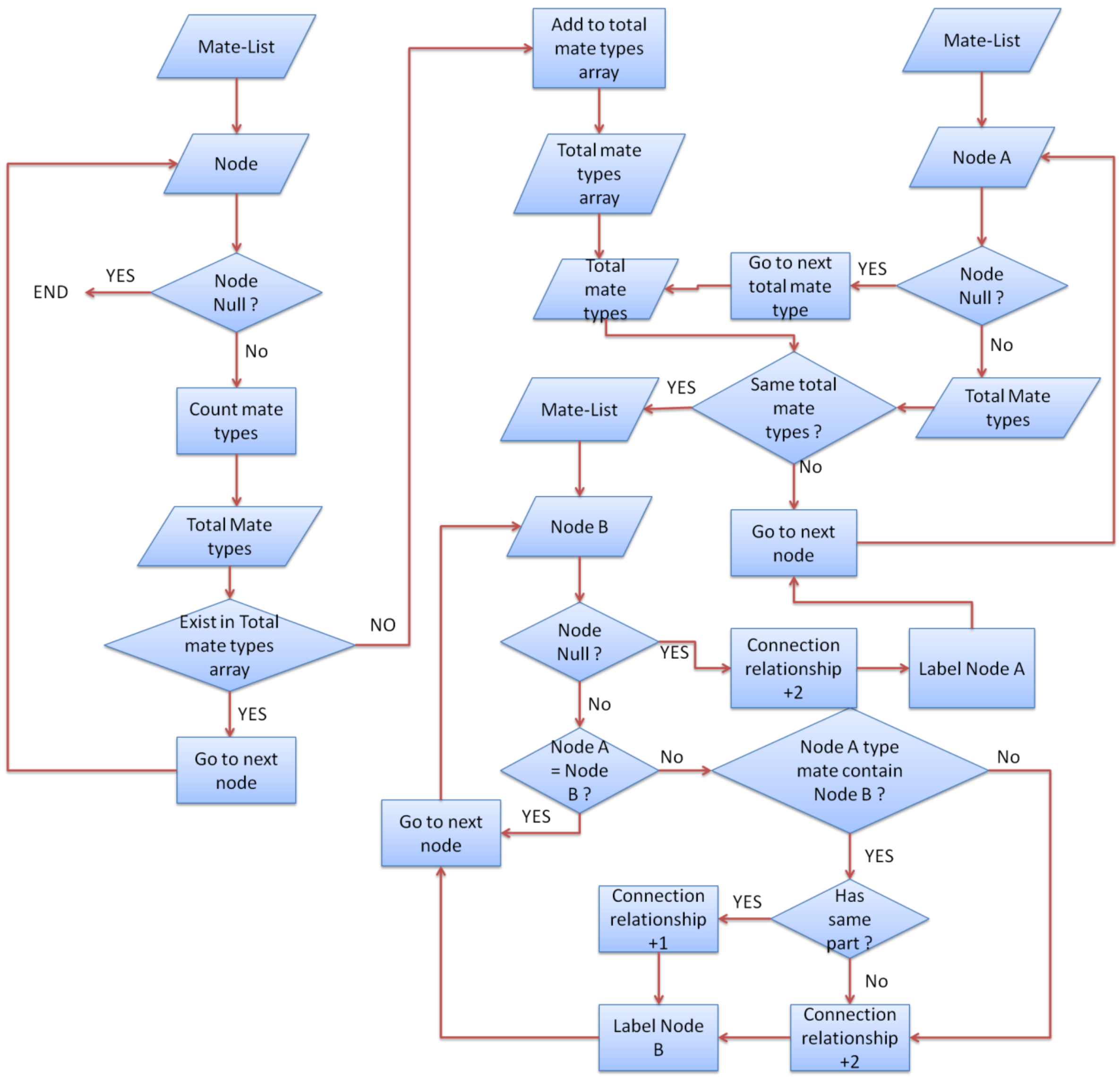


Figure 3.18 Algorithm flows for counting connection relationships

3.4.3 Calculate Total Path Length (TPL)

When all variables required in disassembly time equation are obtained, the program continues to compute the Total Path Length (*TPL*). TPL is sum of all the shortest path length. As Rene et al. discussed in [58], the shortest path means following paths from one component to another. The method utilized in this section is called Breadth-first search (BFS) algorithm, as stated in 2.4.3, its traversal properties reveal each path in the graph and is the shortest path in the graph [56].

Both component-list and mate-list are needed in this algorithm. The algorithm starts from component-list, uses part in current node as “root part” and builds a new empty temporary list to store all the other parts in the assembly. In each node of this temporary list there is a feature called “nlevel” and is used to represent the hierarchical relationship of current part to the “root part”. The directly connected parts (neighbors) “nlevel” is assumed as one. The algorithm finds all directly connected parts from mate-list and adds them to node in temporary list one by one. When all directly connected parts are traversed, the algorithm goes to first node in the temporary list and sets it as “root part”. Then the algorithms finds those directly connected parts (neighbors) to this new “root part”, but the “nlevel” of directly connected parts (neighbors for the current “root part”) is set as summation of current “root part” plus one, these direct connected parts are also added to the same temporary list. The algorithm continues till all parts in the assembly are visited and sum of all “nlevel” is added into the temporary list. After that the temporary list is emptied and the algorithm moves on to second node of component-list and continue the same process for part in this node. When all parts in component list are visited, the sum total value of all “nlevels” is the total path length of given assembly graph. The algorithm is shown in Figure 3.19.

3.5 Generate Redesign Suggestion

The focus on redesign suggestions is to reduce disassembly time. Based on this purpose there are two situations considered in giving redesign suggestions.

1. Giving redesign suggestions for users who choose to completely disassembly of the product.
2. Giving redesign suggestions for users who choose to selective disassemble the product.

In first situation, redesign suggestion advises users to try to disassemble product by selective disassembly. In second situation, redesign suggestion advises users to change particular highlighted parts’ raw materials into the material selected for material recovery. The basic idea is to have as much grouping of parts as possible in the assembly graph. This further implies that material should be assigned to the components in order to create a simpler regenerated assembly graph. Based on the considerations stated above mate-list is utilized in this algorithm in searching those qualified parts.

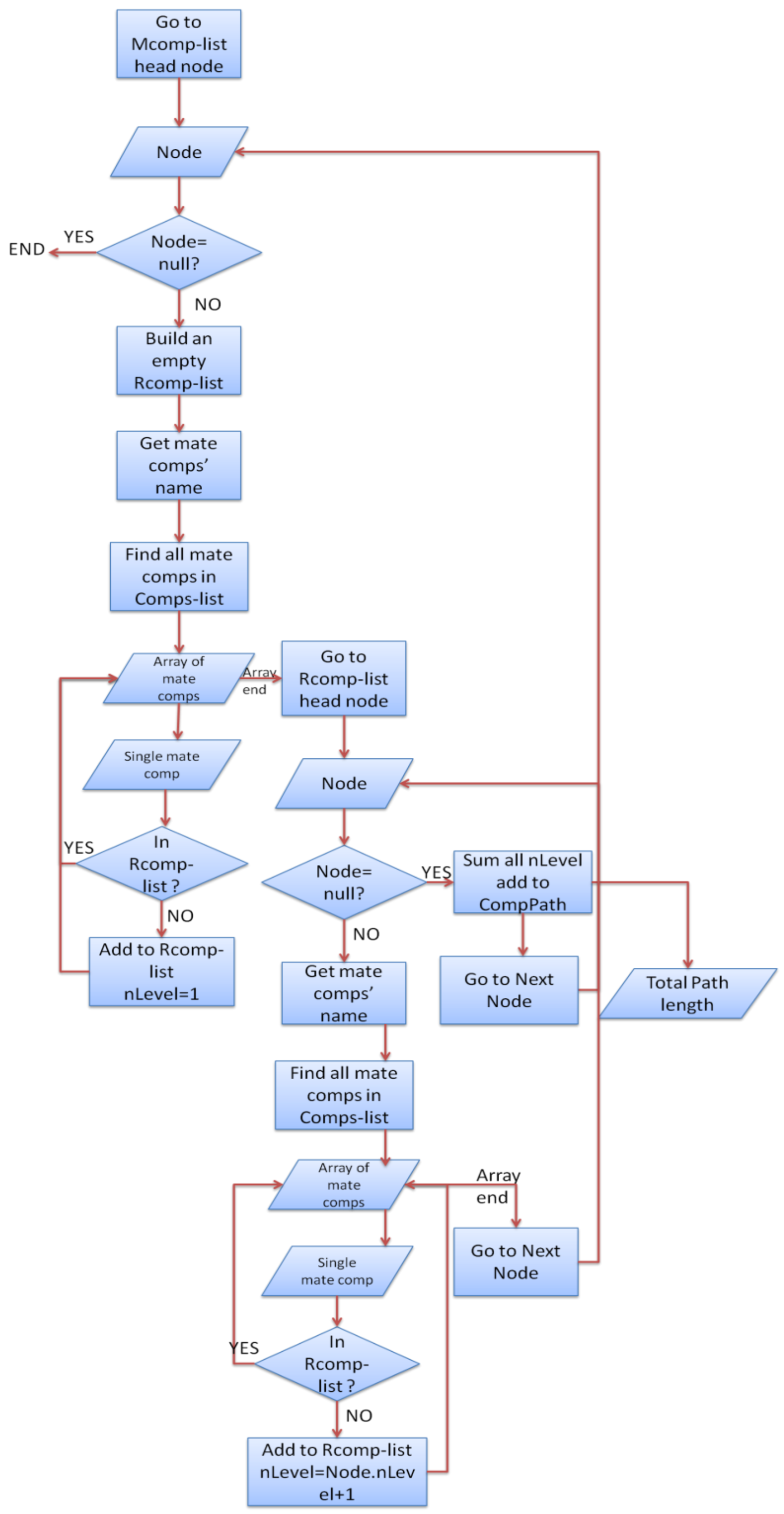


Figure 3.19 Algorithm flows for calculating Total Path Length (TPL)

The algorithm starts with the judgment of what kind of disassembly the user selected. If it was completely disassembly a diagram will show up with string type sentence to advice user try to use selective disassembly. If the selection was selective disassembly the algorithm goes to mate list to find those parts that connects to selective material parts and to add them to a temporary array, which shown in Figure 3.20 named as temporary array 1. When all these qualified parts added to the array a diagram shows up with instructions to users pay attention to following highlight parts because if change these parts' raw material then the selective disassembly time reduce. After this diagram the program visits each part in the temporary array utilized a selection function supplied by Solid Works to highlight current part in the graphic area. When all parts in the temporary array visited the program goes back to mate-list to find those scatter selective material parts. In this process all parts whose raw material same with selected material be added to a temporary array 2. Since in grouping component process part's name changed into the combination of material and an integer index therefore if all parts in temporary array share same name then there is only one group and it is no need to suggest users to do such re-design. On the other hand if parts name were different then a diagram shows up with an instruction to users to pay attention to following highlight parts because connects these parts can reduce selective disassembly time. Here considered those single parts whose raw material is same with selective one but are not belong to same material groups. The algorithm flows is shown in Figure 3.20.

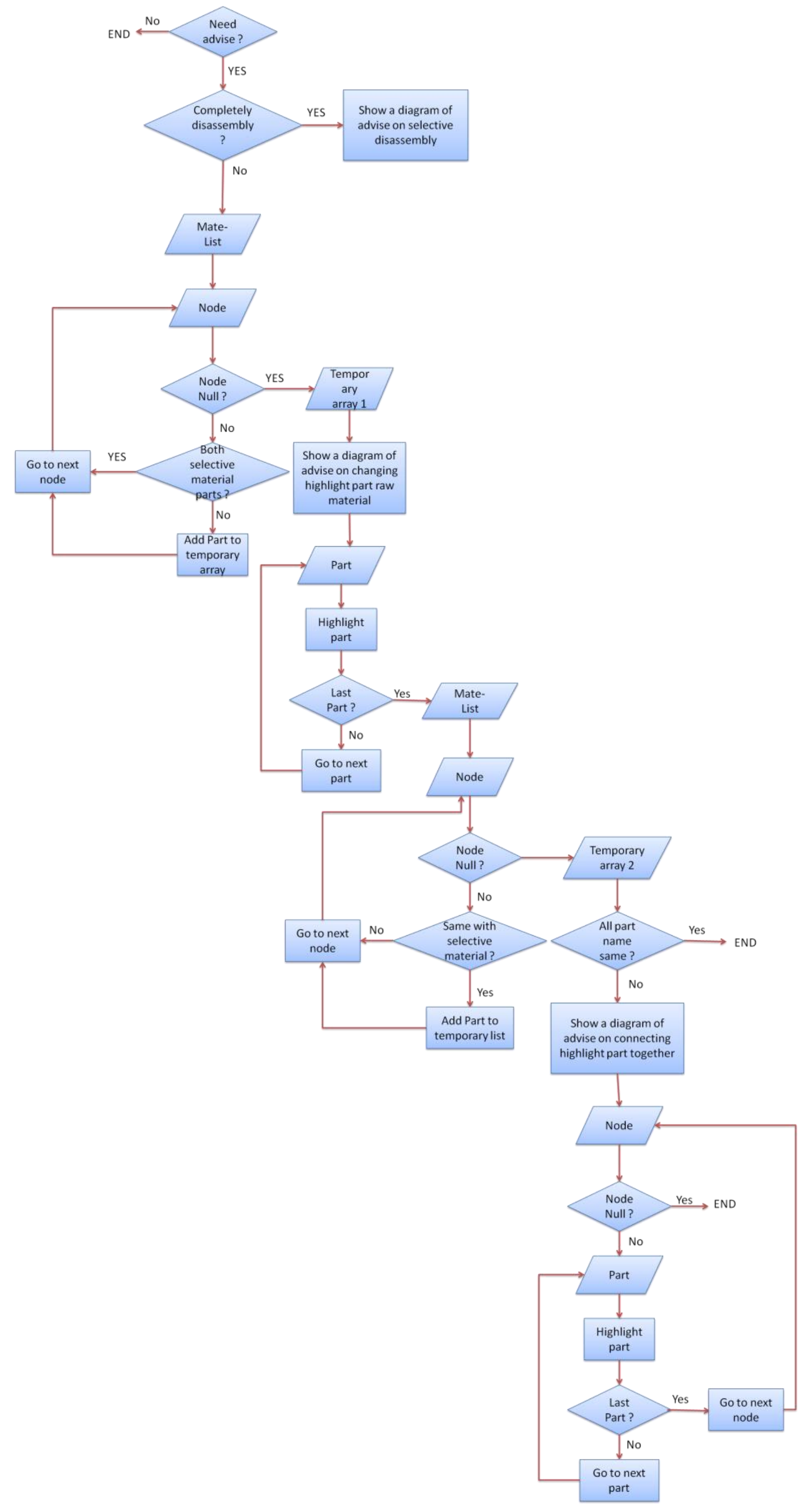


Figure 3.20 Algorithm flows for giving re-design suggestion

Chapter 4 Part Position Location and Disassembly Suggestion Generation

4.1 Introduction

This chapter will present algorithms for an API computer program that can automatically generate disassembly suggestions for destructively disassembling a particular part. The purpose could be that the part is hazardous and the rest of the assembly can be safely land filled or recycled. The algorithms are implemented in Visual C# using Solid Works Application Programming Interface (API).

The overall processes described in this chapter uses ray tracing to identify the accessibility to the part. Ray tracing is a process of following the light ray path from lighting source, then passing through lens and slightly beyond [59]. This idea comes from the physics light properties. It is an image synthesis method in computer graphics.

According to [59] the heart for ray tracing is ray intersection routines. No matter what lighting models, there is always the need to find the intersection point of a ray and an object. When a ray sent towards an object, any obstacle between ray source and the object will return an intersection point. This property will help in determining part relative position (inside or outside) in an assembly by tracing the ray from outside source.

Main algorithm can be divided into three steps: a) determining the start and end points of rays, b) determining the chosen part position in the assembly and c) generating disassembly suggestions according to part position.

4.2 Determining Start and End Point of Ray

Solid Works API supplies a method that can provide a list of object of the specified type that are intersected by a ray. This ray is defined with a given ray radius, a start point and a direction vector. In order to make sure that the start point of the ray is outside the box and the direction vector is towards the assembly model, “bounding box” is utilized. A Bounding box is a smallest size cuboid that can fit a given 3D Object [60]. As an example, Figure 4.1 shows the bounding box of the part. This bounding box is aligned with the XYZ coordinate system.

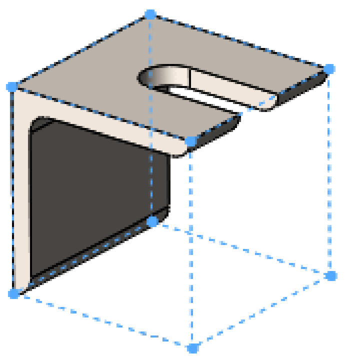


Figure 4.1 Bounding Box for a Part [60]

Solid Works API for bounding box method returns an array with six different double type values. They are XYZ points of the lower- and upper-diagonal corners that bound the object with the box sides parallel to the X, Y and Z axes. Box dimensions that enclose the object are typically close to the minimum possible size [61]. Figure 4.2 shows an example of a bounding box of toaster bottom sub-assembly, it can be seen that each box has six rectangle faces and two opposing faces are parallel to each other. Therefore the center of each face can be determined using the average of the four corner coordinates.

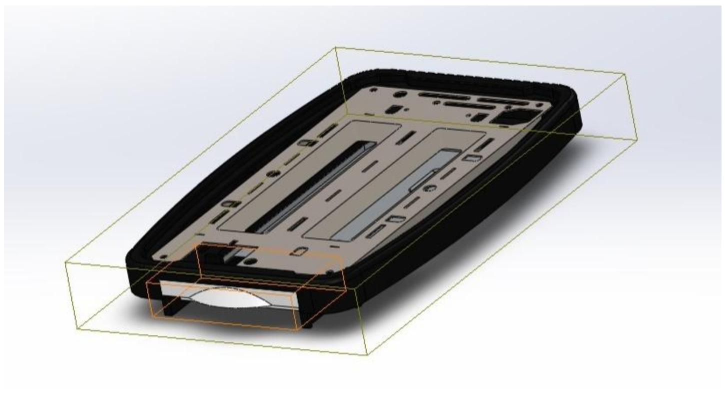


Figure 4.2 Bounding Box for a Toaster Bottom sub-assembly

Since, each bounding box of each component will have six side faces; the component has accessibility from six directions. Partial accessibility from any one of the six directions will not lead to the ability of the component to be disassembled. Therefore, start point of each ray is defined as face center of the assembly bounding box and the end point is defined as center of its further parallel face in the chosen part bounding box. The ray distance radius is the distance between each start point and its corresponding end point.

Results returned by Solid Works ray selection are in the form of array. This array is used to locate the part identified for destructive disassembly. Because all intersected objects are stored in the array it is necessary to eliminate duplicated objects. The algorithm flow for this section is shown in Figure 4.3.

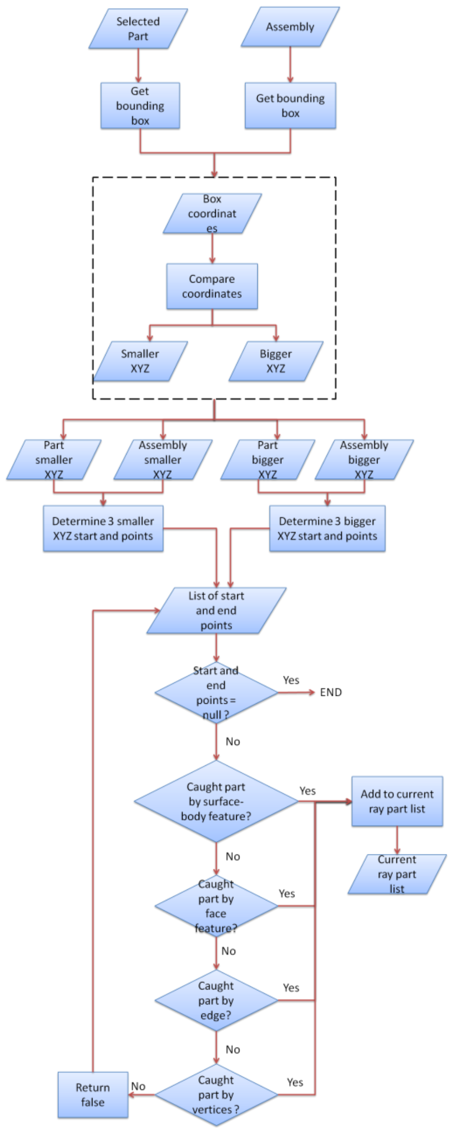


Figure 4.3 Algorithm flows for getting ray start and end points

The sequence of saved objects is identical to the order of intersection of the ray. The sequence is essential in determining part position, which will be discussed in next section.

4.3 Part Position

As stated in section 4.2 the array of all the parts t provided by Solid Works ray API are stored in the order of intersection of the ray from the start position to the end position. This implies that the first and the last parts in this array are exposed\accessible and the parts ranked between these two ends are covered by the parts in that sequence. Thus, the part position and accessibility evaluation is performed according to the part's ranking in all six arrays which represent intersections in six surfaces for the corresponding rays. Part that exposed completely is those whose ranking in the six arrays is either first one or the last one. On the contrary, part that is completely covered by all the other parts will be at the middle position in all the six arrays.

Partial exposure/accessibility could be only one face, two faces, three faces, four faces and five faces of the part. The effect of partial exposure/accessibility to the ease of disassembly even in the case of destructive disassembly is demonstrated by a simple example of a cube. If five of its faces are covered (only one face is accessible) then it is impossible to grab the part. When two faces of the cube are exposed, two cases arise (see table 4.1). In both the situations disassembly is not easy. But situation changes when three faces of the cube are exposed. As more and more faces are exposed, it becomes easy to grab and/or disassemble the part destructively. Examples of partially exposed face on the cube are shown in Table 4.1. After analyzing these three situations, additional criteria are added to evaluate part position and accessibility. Algorithm flows for this step is shown in Figure 4.4.

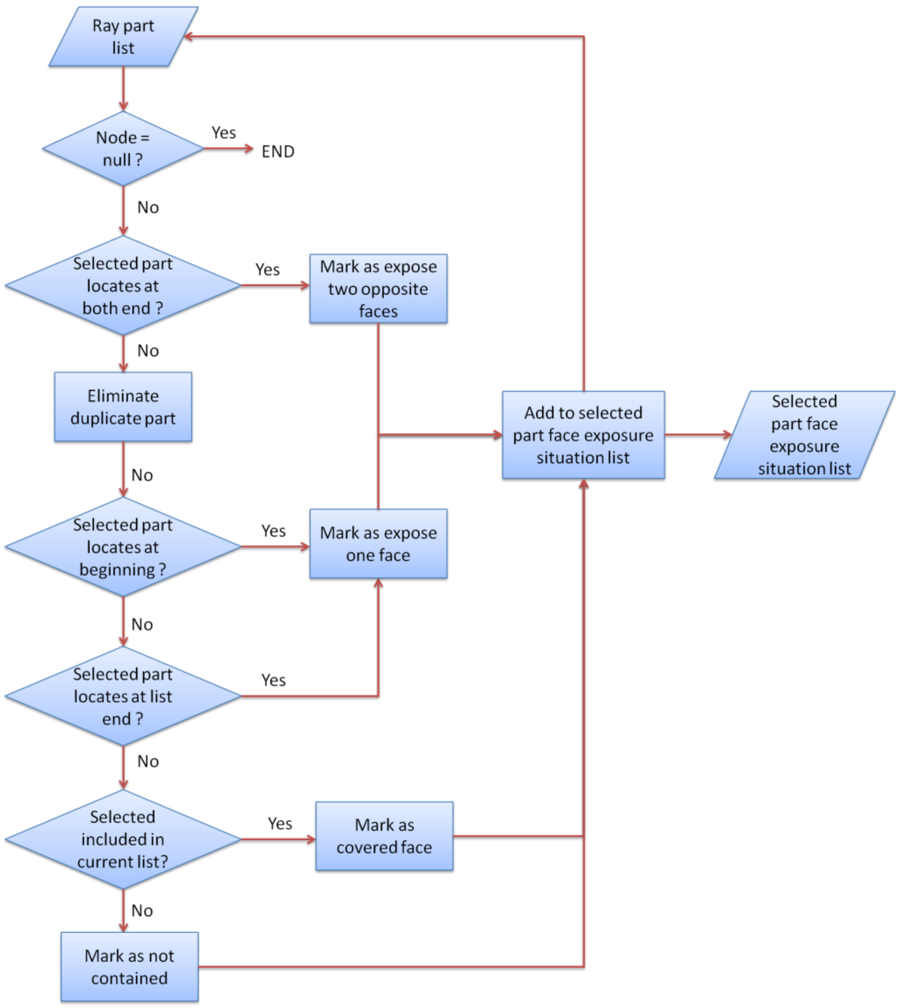
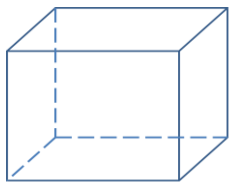
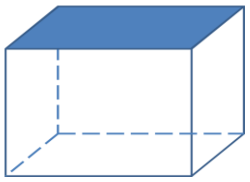
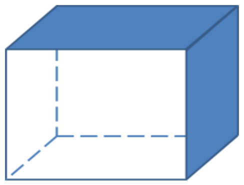
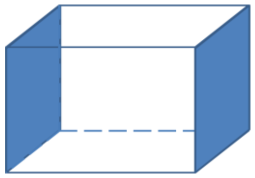
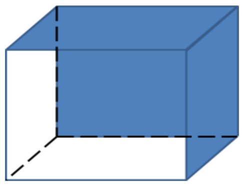
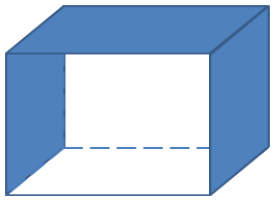

Ray part list
Node = null ?
Yes
END
No
Selected part locates at both end ?
Yes
Mark as expose two opposite faces
No
Eliminate duplicate part
No
Selected part locates at beginning ?
Yes
Mark as expose one face
Add to selected part face exposure situation list
Selected part face exposure situation list
No
Selected part locates at list end ?
Yes
No
Selected included in current list?
Yes
Mark as covered face
No
Mark as not contained

Figure 4.4 Algorithm flows for determining selected part position

Table 4.1 Ease level of grabbing cuboids with face exposure

i. Six faces completely covered, hard to contact.	

ii. One face exposed, hard to grab by hand.	
iii. Two faces exposed but share same corner, hard to grab by hand.	
iv. Two faces exposed oppositely can be hold by hand but inconvenience to break assembly relationships.	
v. Three faces exposed but share same corner, hard to grab by hand.	
vi. Three faces exposed can be grabbed and break assembly relationship easily.	

4.4 Disassemble advice generation

As stated in section 4.3 there are five part position situations but considering destructive disassembly there are just two choices there. If at least three faces of the part bounding box are exposed it is possible to grab the part for disassembly. But it is still necessary to break the assembly constraints (rivet, screws, glue, weld, snap-fit, etc.) with the connected parts. Therefore, the part position judgments are determined by whether the part has at least three accessible faces. If not, then the program will advise designers to modify the part positions so as to have three accessible faces. The algorithm flow of this section is shown in Figure 4.5.

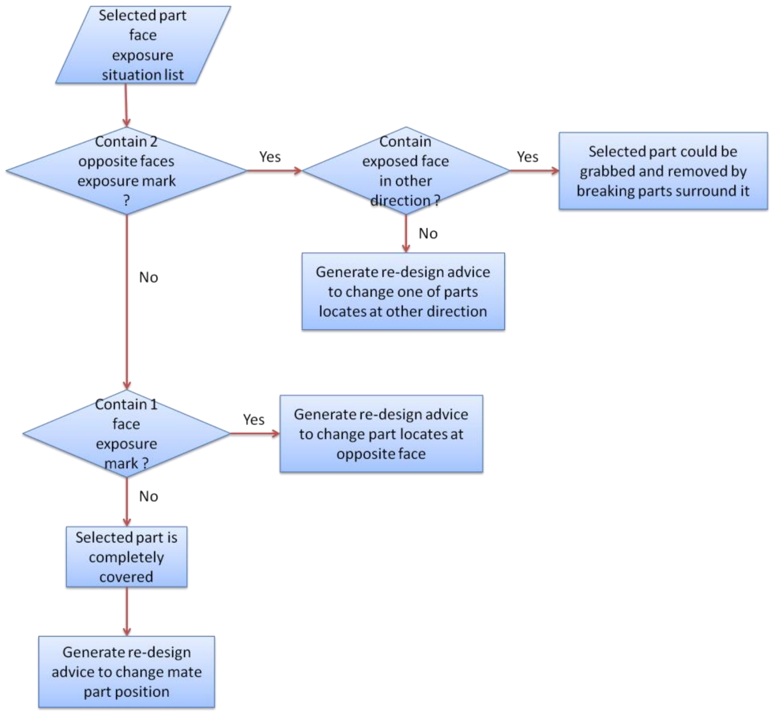

Selected part face exposure situation list
Contain 2 opposite faces exposure mark ?
Yes
Contain exposed face in other direction ?
Yes
Selected part could be grabbed and removed by breaking parts surround it
No
Generate re-design advice to change one of parts locates at other direction
No
Contain 1 face exposure mark ?
Yes
Generate re-design advice to change part locates at opposite face
No
Selected part is completely covered
Generate re-design advice to change mate part position

Figure 4.5 Algorithm flows for generating re-design advices

Chapter 5 Case Studies

5.1 Introduction

This chapter presents four case studies in order to demonstrate the functionality and use of the developed algorithms that were presented in Chapter 2 and 3. The case studies will be for four products: standard toaster (36 parts), Eco-toaster (42 parts), Drill (24 parts) and Kid Bike (31 parts). Both the toaster CAD assembly models were created from physical products[1] while the other two models are obtained from Grab CAD [62].

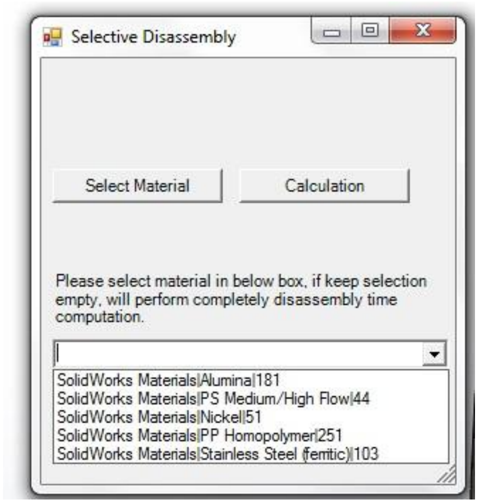
5.2 Interface

In disassembly time estimation part, complete disassembly time and material selective disassembly time will be estimated by the program as well as results of design optimization. In destructive disassembly portion the disassembly suggestions will be tested and presented virtually.

The interface for disassembly time estimation is a dialog box with a "Select material" button, a "Calculation" button and a pull-down list. To estimate complete disassembly time, the user should keep the selection empty and click on the calculation button. If the user desires to estimate material selective disassembly time, he/she should select a material from the pull-down list. Then, click "Select material" button followed by the "Calculation" button. Figure 5.1 shows a screen shot of a dialog interface. Regardless of the user's selection for complete disassembly or material selective disassembly, the program will show the results using a pop-up dialog box as shown in Figure 5.2. The elements in Figure 5.2 are; "SUM of components" is total number of parts; "SUM of components' TPL" is total path length counted under current assembly architecture; "SUM of connection relationships" is total number of mate types of current assembly. APL and PLD are average path

[1] Joshua Meyer, while at Washington State University, helped create the CAD model for Eco-Toaster

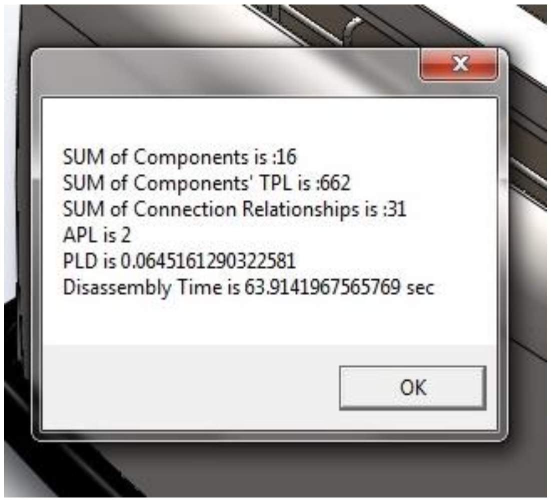
length and path length density, respectively. APL, PLD and disassembly time estimation is computed using equations discussed in Chapter 3 and presented in [30].



Please select material in below box, if keep selection empty, will perform completely disassembly time computation.

```
SolidWorks Materials|Alumina|181
SolidWorks Materials|PS Medium/High Flow|44
SolidWorks Materials|Nickel|51
SolidWorks Materials|PP Homopolymer|251
SolidWorks Materials|Stainless Steel (ferritic)|103
```

Figure 5.1 Dialog box screen shot for toaster model



```
SUM of Components is :16
SUM of Components' TPL is :662
SUM of Connection Relationships is :31
APL is 2
PLD is 0.0645161290322581
Disassembly Time is 63.9141967565769 sec
```

Figure 5.2 Complete disassembly time estimation results

After computing the disassembly time, the program will continue to provide suggestions in order to reduce disassembly time. The suggestions are presented through a dialog box. The suggestions are shown from two view points. According to one of the view points, the user is suggested to change material of particular parts. In the second view point the user is suggested to change the mate relationships between particular parts. Suggested part's name will be shown using a pop-up dialog box. Suggested part will be highlighted in the Solid Works graphics area. Figure 5.3 is shows the notification and highlighted components for the standard toaster CAD assembly model.

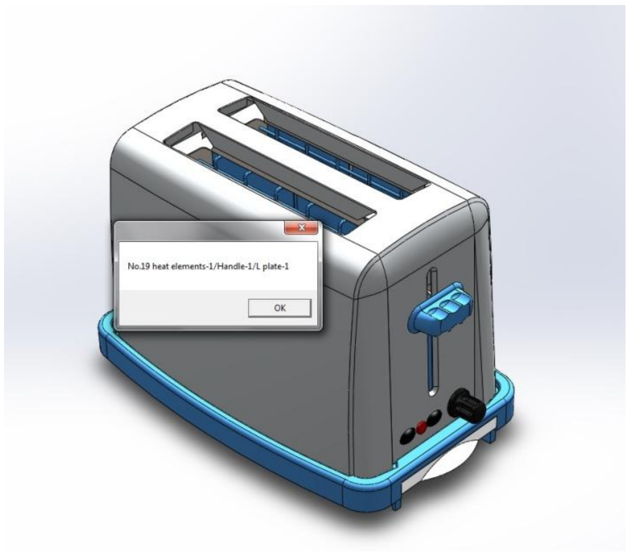
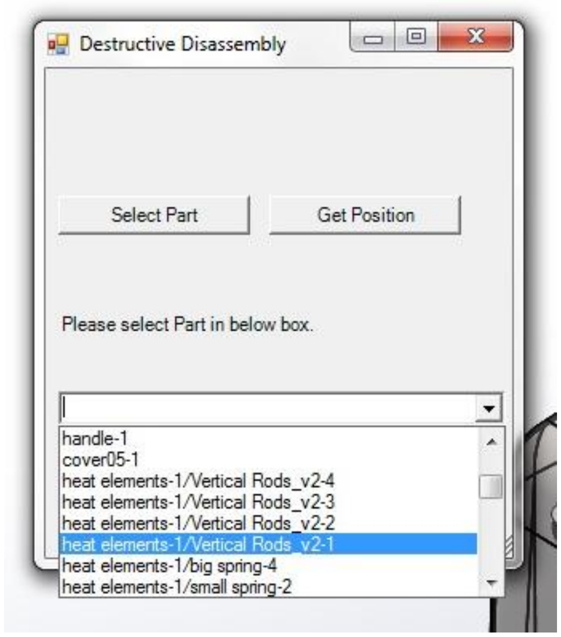

No.19 heat elements-1/Handle-1/L plate-1

Figure 5.3 Highlighted part and notification dialog box for the standard toaster CAD assembly model.

The interface of destructive disassembly consists of a dialog box with a "Select Part" button, a "Get Position" button and a pull-down list. Through the pull-down list users can select the part that they would like to selectively disassemble. The selective disassembly computation in this case assumes that the disassembly allows for destruction (breaking) of other parts. After selecting the part (example Figure 5.4), the user should click "Get Position" button. The selected part will be highlighted in Solid Works graphic area (example Figure 5.5) and the program will examine the selected part position with respect to the entire assembly architecture. Furthermore, the program will highlight parts (example Figure 5.6) that should be broken if they restrict in any manner the disassembly of the selected part alone. Figure 5.5 shows the selected top plate of the toaster model. Figure 5.6 shows the highlighted cover part to tell the user that only after breaking or removing the cover part, the top-plate could be disassembled.



Please select Part in below box.

handle-1
cover05-1
heat elements-1/Vertical Rods_v2-4
heat elements-1/Vertical Rods_v2-3
heat elements-1/Vertical Rods_v2-2
heat elements-1/Vertical Rods_v2-1
heat elements-1/big spring-4
heat elements-1/small spring-2

Figure 5.4 Screen shot of Destructive Disassembly interface dialog

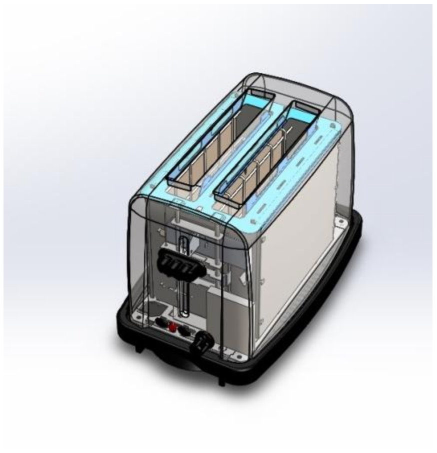


Figure 5.5 Shot screen of highlighting selected part

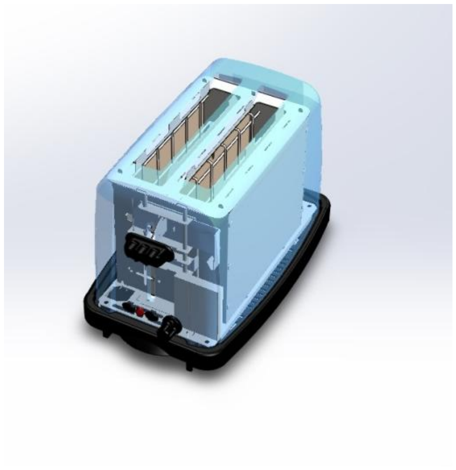


Figure 5.6 Screen shot of highlighting part that suggested broke

The following sections will present four case studies, based on the discussion of the software interface presented in this section, for disassembly time estimation and part selective destructive disassembly suggestion.

5.3 Case Study 1 – Standard Toaster

The standard toaster CAD model is estimated from a physical toaster (Oster Toaster Model# 6325). Raw material setting is referred from Srinivasan [5]. The standard toaster is manufactured using five kinds of raw materials: steel, Polypropylene, Polystyrene, Nickel and Aluminum. Steel is assumed as the material that the manufacture desires to recycle in the EOL stage. Figure 5.7 shows an image of the standard toaster assembly model in Solid Works. Figure 5.8 shows partially exploded view of the standard toaster model.

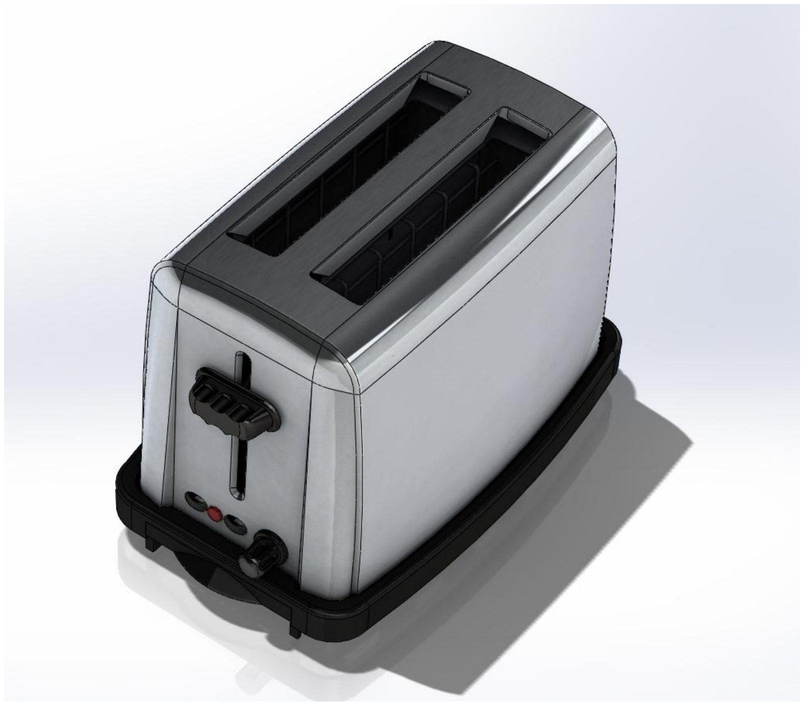


Figure 5.7 Graphic image of standard toaster assembly model

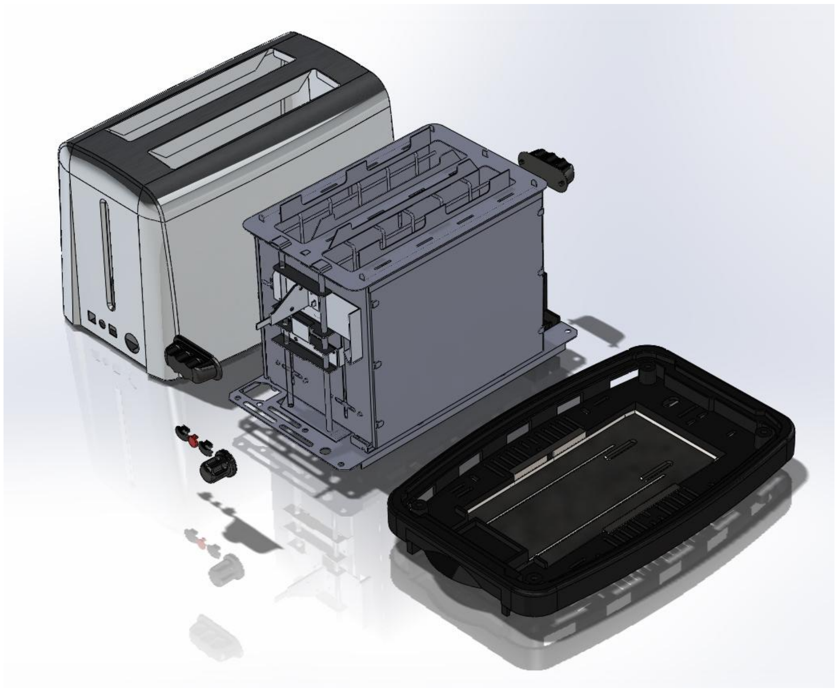


Figure 5.8 Explosion view of standard toaster assembly model

5.3.1 Disassembly Time Estimation

After running the program following the process stated in section 5.2, the results for complete disassembly time estimation and selective disassembly time estimation are shown in Table 5.1. It can be seen that when steel is selected as the material to recycle in EOL stage, the selective disassembly time reduces by 73% from the complete disassembly time. If polypropylene is selected as material for recycling, the selective disassembly time reduces by 3%. All the other materials do not affect disassembly time when they are selected for recycling and selective disassembly.

Table 5.1 Complete Disassembly Time and Material Selective Disassembly Time

	No. Component	TPL	No. Relationship	APL	PLD	t_d	t_d reduction %
Complete Disassembly	36	4222	96	3	0.031	234	base line
Steel	16	662	31	2	0.065	64	73%
Polypropylene	35	3996	94	3	0.032	227	3%
Polystyrene	36	4222	96	3	0.031	234	0%
Nickel	36	4222	96	3	0.031	234	0%
Aluminum	36	4222	96	3	0.031	234	0%

For re-design suggestion based on steel as the recycling material, the software suggests to change raw materials of 13 parts to steel in order to further reduce the disassembly time. The highlighted parts are shown in Figure 5.9.

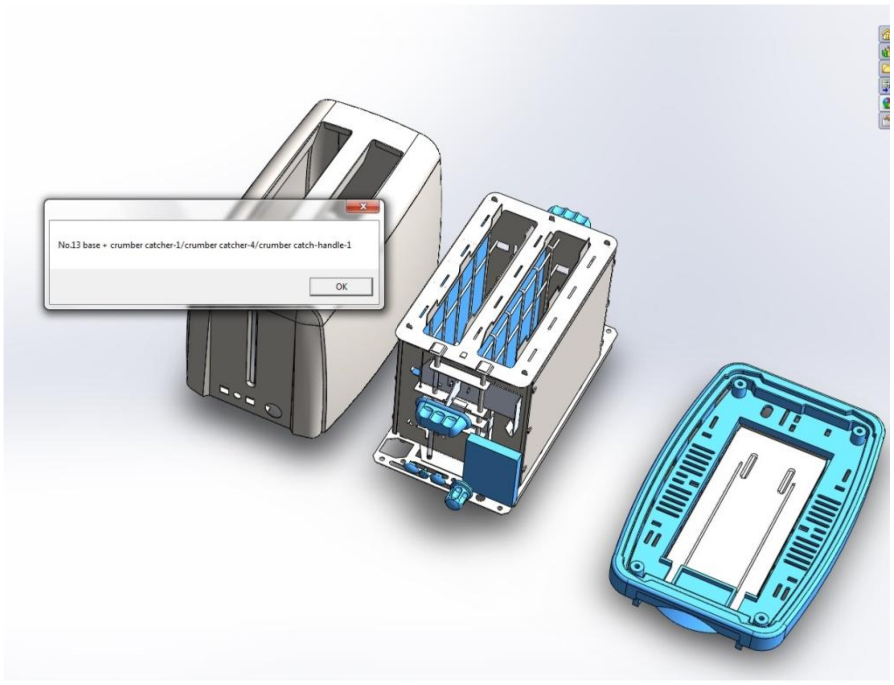

No.13 base + crumber catcher-1/crumber catcher-4/crumber catch-handle-1

Figure 5.9 Screen shot of suggestions of changing highlight parts material

Following the suggestions, if the program is re-run on the modified CAD assembly model, the total number of effective parts/nodes reduces to 1. Total path length reduces to 0, total mate types reduce to 0 and disassembly time is zero since all the parts are made of same material and the product can be completely recycled.

5.3.2 Destructive disassembly

In this case, one of the inside sidewall of the standard toaster is selected to test the program. In order to present the effects clearly, outside cover is set to transparent. Figure 5.10 shows the highlighted components when the sidewall is selected. As the program runs, it checks accessibility from all six directions of the sidewall and highlights those parts that block the accessibility. Figure 5.11 shows four parts that hinder the removal of the sidewall, they are: toaster cover, top-plate, the other side-wall part and one of heating parts.

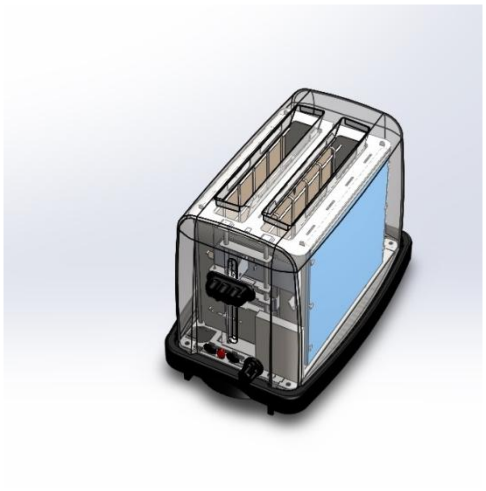


Figure 5.10 Screen shot of selected side wall belongs to standard toaster

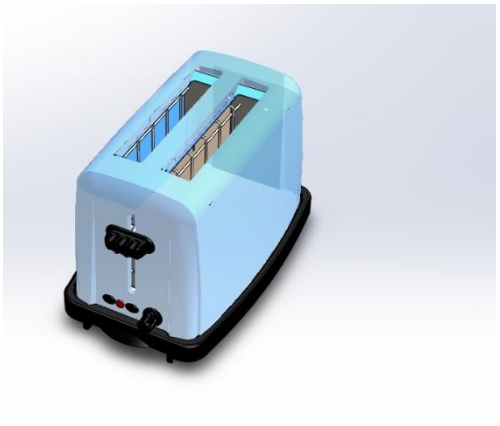


Figure 5.11 Screen shot of parts impede disassembling side-wall

5.4 Case Study 2 – Eco-Toaster

The physical eco-toaster (TE-249) was disassembled by Srinivasan [5]. Raw material is referred from the study described in Srinivasan [5]. The eco-toaster is manufactured using four kinds of raw materials: Steel, Polypropylene, Polystyrene and Nickel. Steel is assumed as the material that manufacture desires to recycle in the EOL stage. Figure 5.12shows the CAD assembly model of the eco-toaster. Figure 5.13 shows partially exploded view of the eco-toaster model.

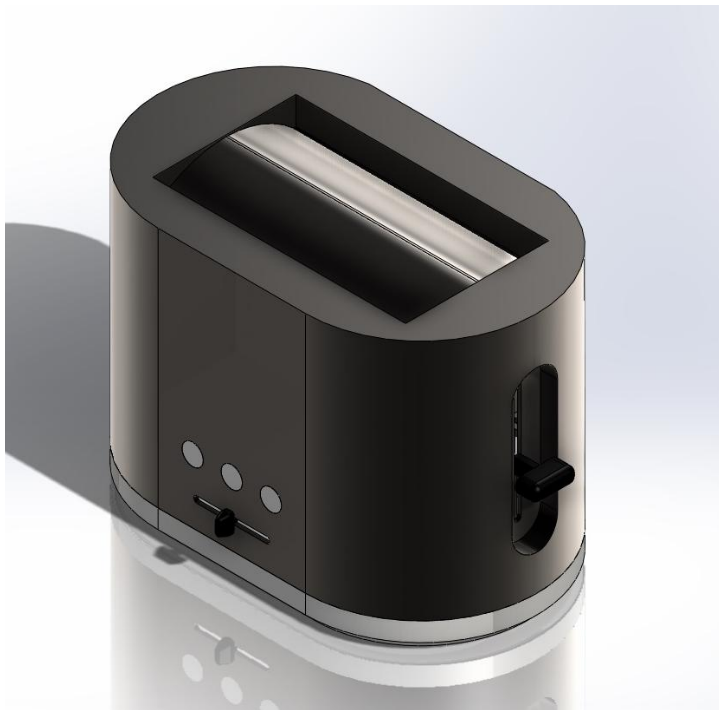


Figure 5.12 Graphic image of Eco-toaster assembly model

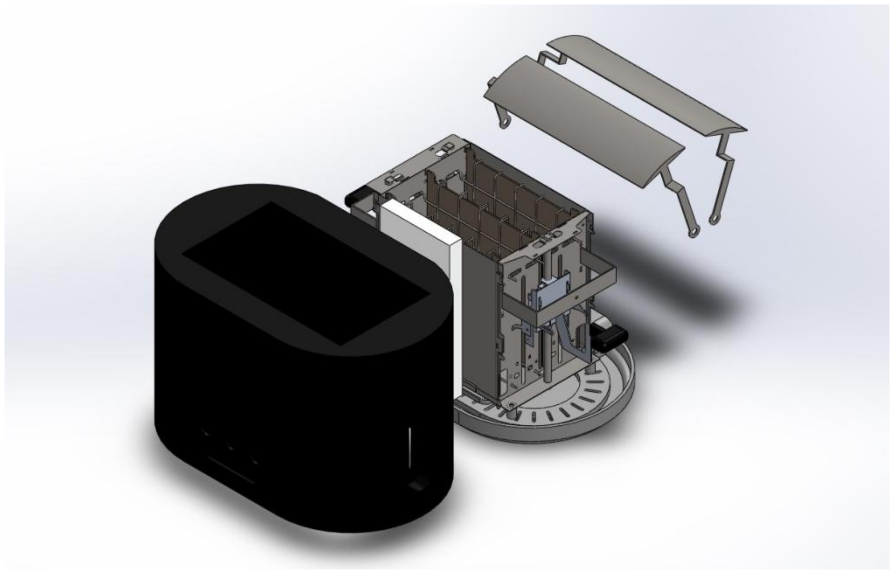


Figure 5.13 Explosion view of Eco- toaster assembly model

5.4.1 Disassembly Time Estimation

Following the program execution and the process described in section 5.2, the results for complete disassembly time estimation and selective disassembly time estimation are shown in Table 5.2. It can be seen that when steel is selected as the material to recycle, the selective disassembly time reduces by 87% from complete disassembly time. Selecting polystyrene as selective material helps reduce the disassembly time by 3%. All the other materials do not affect disassembly time when they are selected.

Table 5.2 Complete Disassembly Time and Material Selective Disassembly Time

	No. Component	TPL	No. Relationship	APL	PLD	t_d	t_d reduction %
Complete Disassembly	42	6514	98	3	0.031	282	base line
steel	10	232	24	2	0.083	37	87%
Polypropylene	42	6514	98	3	0.031	282	0%
Polystyrene	41	6150	96	3	0.031	274	3%
Nickel	42	6514	98	3	0.031	282	0%

For re-design suggestion based on steel as the recycling material, the software suggests to change raw materials of 6 parts to steel in order to further reduce the disassembly time. The highlighted parts are shown in Figure 5.14.

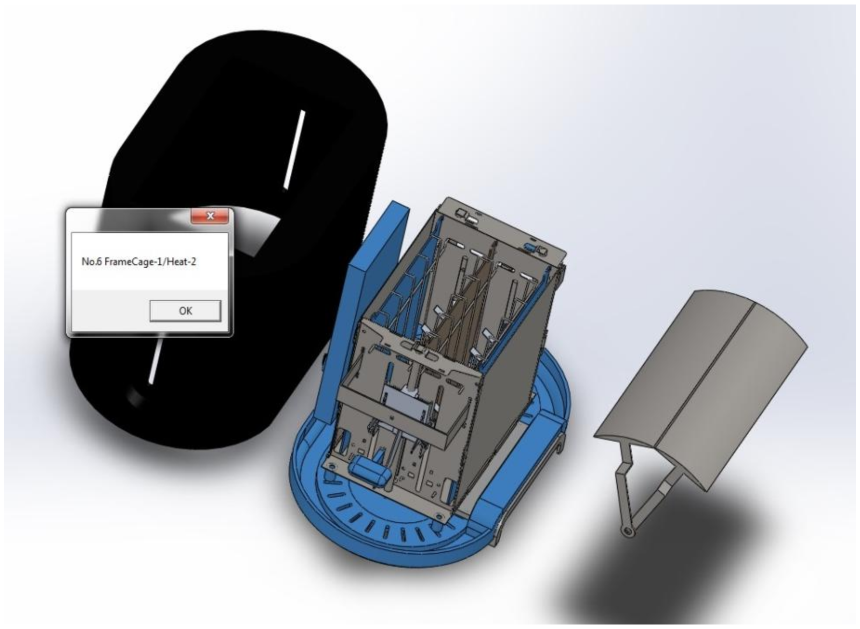

No.6 FrameCage-1/Heat-2

Figure 5.14 Screen shot of suggestions of changing highlight parts material

Following the suggestions, if the program is re-run on the modified CAD assembly model, the total number of effective parts/nodes reduces to 2. Total path length reduces to 2, total mate types reduce to 2 and disassembly time is reduces to 3.

5.4.2 Destructive Disassembly

In this case, one of the sidewalls of Eco-toaster is selected to test the program. In order to present the effects clearly, outside cover is set to transparent. Figure 5.15 shows the highlighted components when the sidewall is selected. As the program runs, it checks accessibility from all six directions of the sidewall and highlights those parts that block the accessibility. Figure 5.16 shows five that hinder the removal of the sidewall, they are: outside cover, one of top lid, lights plate, the another side-wall part and one of heating parts.

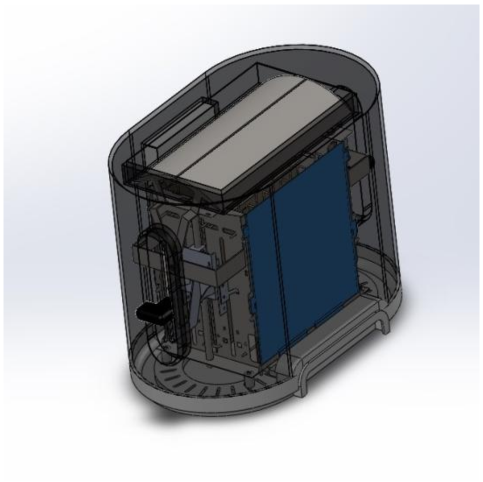


Figure 5.15 Screen shot of selected side wall belongs to eco-toaster

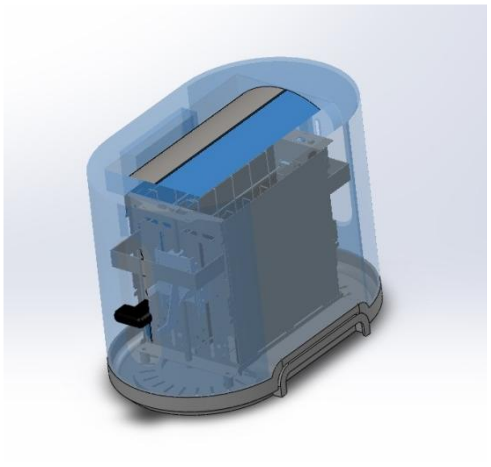


Figure 5.16 Screen shot of parts impede disassembling side-wall

5.5 Case Study 3 – Drill

The drill Solid Works assembly model is obtained from Grab CAD [62]. The drill is manufactured using seven kinds of raw materials: steel, chrome stainless steel, alloy steel, AISI Type A2 Tool Steel, Polyester resin, 1060 alloy and ABS. Figure 5.17 shows the CAD assembly model of the drill. Figure 5.18 shows partially exploded view of the drill model.

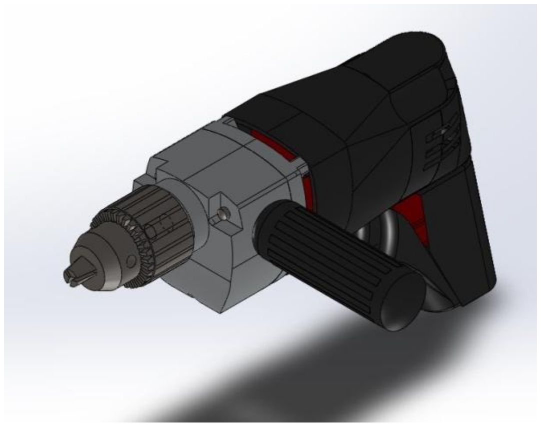


Figure 5.17 Graphic image of drill assembly model

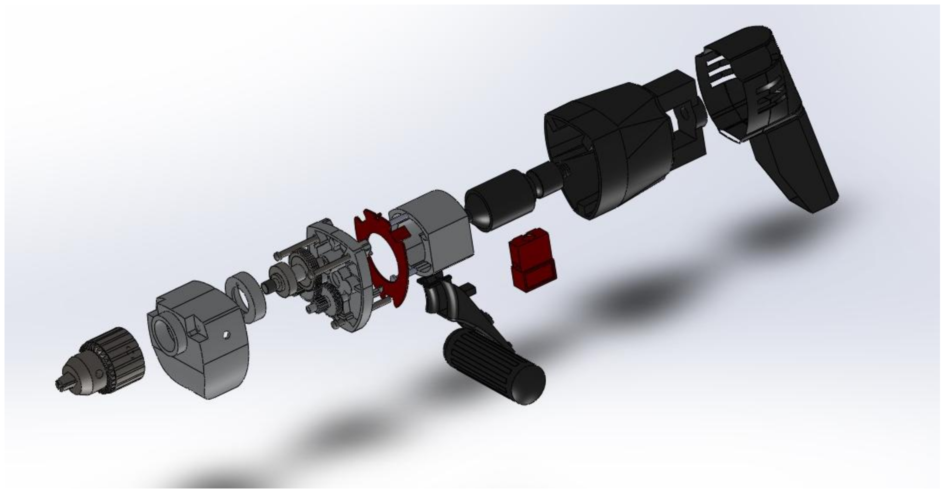


Figure 5.18 Explosion view of drill assembly model

5.5.1 Disassembly Time Estimation

Following the program execution and process described in section 5.2, the results for complete disassembly time estimation and selective disassembly time estimation are shown in Table 5.3. It can be seen that when ABS is selected as the material to recycle, the selective disassembly time reduces by 43% from complete disassembly time. Selecting 1060 Alloy as selective material helps reduce the disassembly time by 26%. Selecting both alloy steel and polyester resin as selective material help reduces by 5% from complete disassembly time. All the other materials do not affect disassembly time when they are selected.

Table 5.3 Complete Disassembly Time and Material Selective Disassembly Time for Drill

	No. Component	**TPL**	**No. Relationship**	**APL**	**PLD**	**t_d**	**t_d Reduction %**
Complete Disassembly	24	1768	63	3	0.048	150	base line
steel	24	1768	63	3	0.048	150	0%
chrome stainless steel	24	1768	63	3	0.048	150	0%
alloy steel	23	1685	61	3	0.049	143	5%
AISI	24	1768	63	3	0.048	150	0%
Polyester resin	23	1638	61	3	0.049	143	5%
1060 alloy	18	918	46	3	0.065	111	26%
ABS	22	1294	60	2	0.033	86	43%

Re-design suggestion based on ABS as the recycling material, the software suggests to change raw materials of 7 parts to ABS in order to further reduce the disassembly time. The highlighted parts are shown in Figure 5.19.

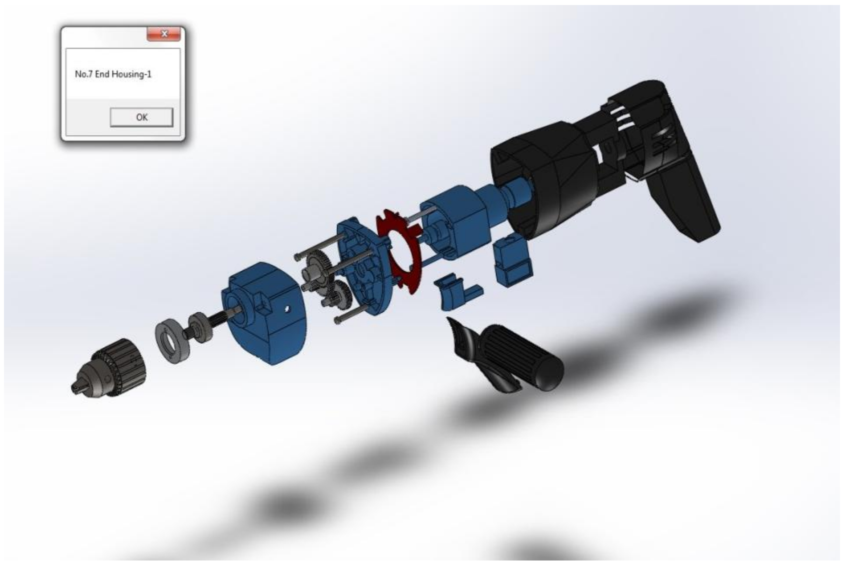

No.7 End Housing-1

Figure 5.19 Screen shot of suggestions of changing highlight parts material

Following the suggestions, if the program is re-run on the modified CAD assembly model, the total number of effective parts/nodes reduces to 13. Total path length reduces to 335, total mate types reduce to 36 and disassembly time reduce to 48.

5.5.2 Destructive Disassembly

In this case the inside gear is selected to test the program. In order to present the effects clearly, outside parts are set to transparent. Figure 5.20 shows the highlighted components when the gear is selected. As the program runs, it checks accessibility from all six directions of the gear and highlights those parts that block the accessibility. Figure 5.21 shows three parts that hinder the removal of the gear, they are: motor housing, the inside aluminum piece and the handle main.

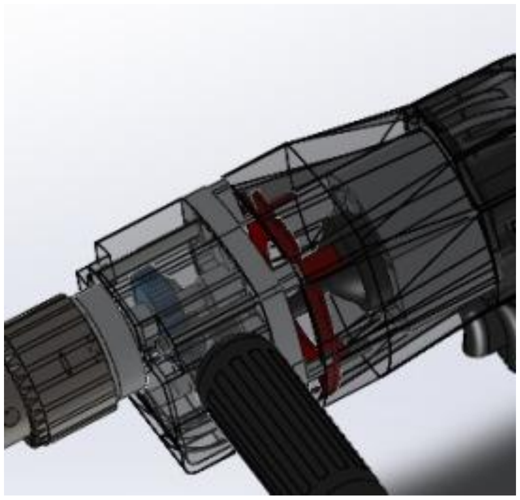


Figure 5.20 Screen shot of selected gear inside the drill

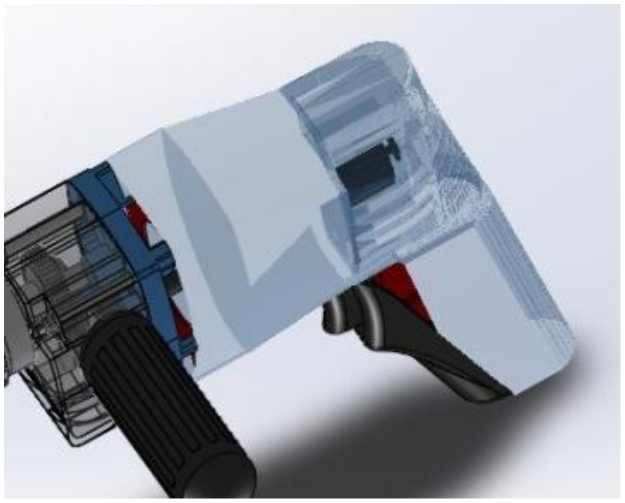


Figure 5.21 Screen shot of parts impede disassembling side-wall

5.6 Case Study 4 – Kid Bike

The kid bike Solid Works assembly model is obtained from Grab CAD [62]. The kid bike is manufactured using five kinds of raw materials they are: polypropylene, rubber, stainless steel, alumina and SBR. Figure 5.22 shows a graphic image of the kid bike assembly model, and its explosion view is shown in Figure 5.23.

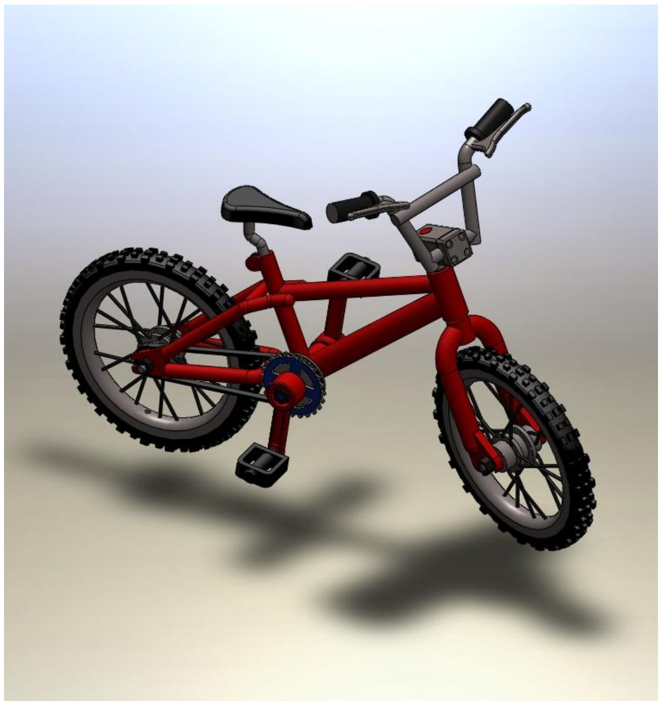


Figure 5.22 Graphic image of kid bike assembly model

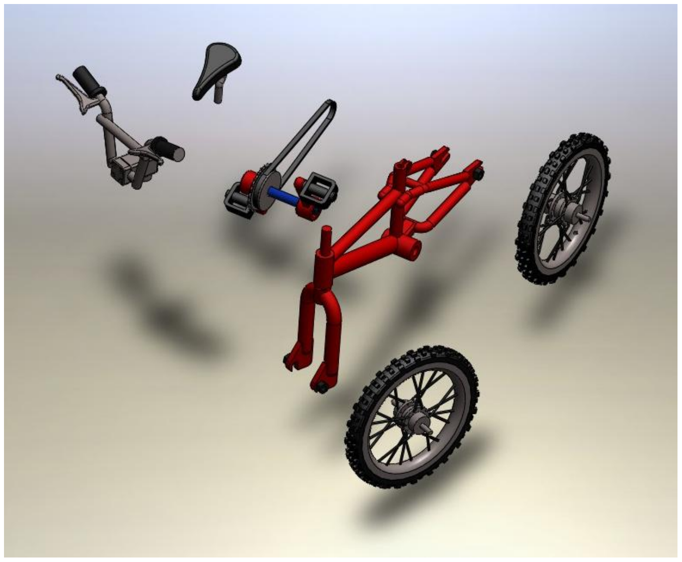


Figure 5.23 Explosion view of kid bike assembly model

5.6.1 Disassembly Time Estimation

Following the program execution and process described in section 5.2, the results for complete disassembly time estimation and selective disassembly time estimation are shown in Table 5.4. It can be seen that when steel is selected as the material to recycle, the selective disassembly time reduces by 92% from complete disassembly time. Selecting SBR as selective material helps reduce the disassembly time by 4%. All the other materials do not affect disassembly time when they are selected.

Table 5.4 Complete Disassembly Time and Material Selective Disassembly Time for Kid Bike

	No. Component	TPL	No. Relationship	APL	PLD	t_d	t_d reduce %
complete disassembly	33	3609	80	3	0.0375	216	Base line
PP	33	3609	80	3	0.0375	216	0%
rubber	33	3609	80	3	0.0375	216	0%
stainless steel	10	170	24	1	0.0416	17	92%
alumina	33	3609	80	3	0.0375	216	0%
SBR	32	3268	78	3	0.0385	208	4%

Re-design suggestion based on steel as the recycling material, the software suggests to change raw materials of 8 parts to steel in order to further reduce the disassembly time. The highlighted parts are shown in Figure 5.24.

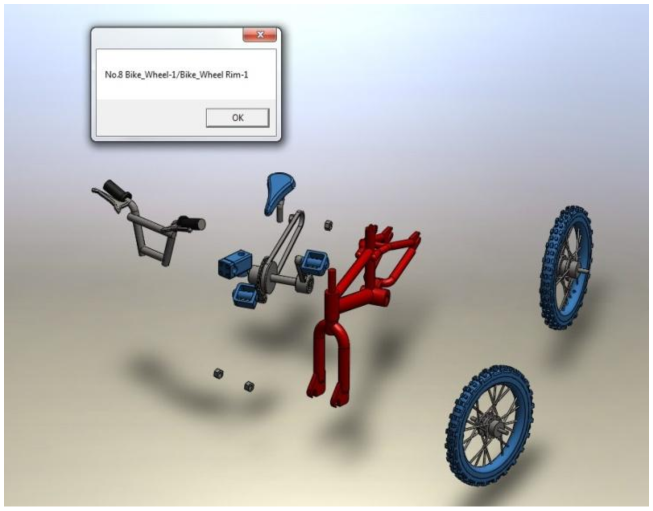

No.8 Bike_Wheel-1/Bike_Wheel Rim-1

Figure 5.24 Screen shot of suggestions of changing highlight parts material

Following the suggestions, if the program is re-run on the modified CAD assembly model, the total number of effective parts/nodes reduces to 8. Total path length reduces to 106, total mate types reduce to 17 and disassembly time reduce to 13.

5.6.2 Destructive Disassembly

In this case the sprocket of the standard toaster is selected to test the program. Figure 5.25 shows the highlighted components when the sprocket is selected. As the program runs, it checks accessibility from all six directions of the sprocket and highlights those parts that block the accessibility. Figure 5.26 shows three parts that hinder the removal of the sprocket, they are: crankshaft, crank and a small nut that firm crankshaft and crank to sprocket.

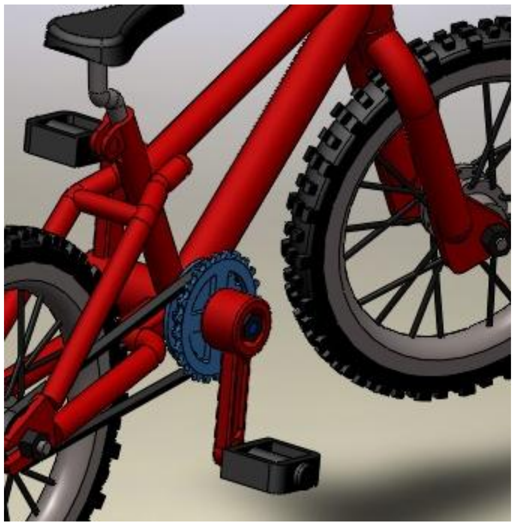


Figure 5.25 Screen shot of highlighted sprocket

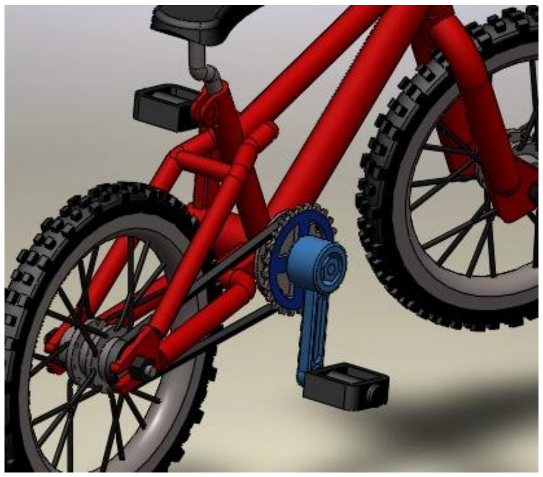


Figure 5.26 Screen shot of parts impede disassembling neck-fork

Chapter 6 Conclusion and Future Work

This chapter and following sections will summarize the contributions, limitations and future work of the research presented in this thesis.

6.1 Contributions

The research work in this thesis

- Presents algorithms to implement complete disassembly time estimation and material selective disassembly time estimation by utilizing graph traversal algorithms.
- Presents algorithms to generate re-design suggestions on the purpose of reducing disassembly time.
- Presents algorithms to determine the selected part position with respect to the entire assembly architecture by utilizing ray intersection.
- Presents algorithms to generate destructive disassembly suggestions according to the estimated part position.
- Demonstrates application of the software tool through four case studies each with complete disassembly, material selective disassembly and component selective disassembly time estimation and redesign suggestions.

6.2 Limitations

The research presented in this thesis has the following limitations;

- The methods for disassembly time estimation assume disassembly as inverse of assembly. This is not true for all assembly operations, for example welding and gluing takes more time in disassembly then assembly.

- The method lacks justification through comparisons with manual estimation of disassembly time. Although comparisons with manually estimated assembly time are presented for complete disassembly, assembly time is not the same disassembly in most cases.
- The metric, path length density (PLD), of selective disassembly method is higher than that the one for complete disassembly method. This is contrary to the purpose of reducing graph complexity by grouping nodes in selective disassembly.
- The disassembly time estimation program cannot estimate multi-material selection.
- The destructive disassembly program cannot give disassemble suggestions with multi-parts selection or sub-assembly selection.
- Ray intersection application in destructive disassembly is limited to solid objects. This method currently does not work with thin features, such as sheet objects.

6.3 Future Work

Future work relates to addressing the limitations of current work. The future work can be summarized as;

- Develop charts for manual estimation of complete disassembly and selective disassembly time estimation.
- Develop algorithms for destructive disassembly time estimation with multi-part selection or sub-assembly selection.
- Develop better equation that can estimate complete, selective and destructive disassembly time.

Multi-material and multi-parts that can be selected and disassembled in one step can assist in reducing the disassembly cost at the EOL stage.

On the other hand, the actual disassembling work cannot be simulated as the inverse of assembly process. Commonly because of the limitation of product situation, difficulty of disassembly and so on, disassembling work is performed regardless to the type assembly relationships. Therefore a disassembly time estimation method that can approximate actual disassembly work is more suited for the purpose.

Reference

[1] U.S. Environmental Protection Agency (EPA), Municipal Solid Waste Generation, Recycling, and Disposal in the United States: Facts and Figures for 2010.

[2] U.S. Environmental Protection Agency (EPA), 2008a. eCycling, http://www.epa.gov/epawaste/conserve/materials/ecycling/basic.htm

[3] Pnueli, Y., & Zussman, E. (1997). Evaluating the end-of-life value of a product andimproving it by redesign. International Journal of Production Research,35(4), 921-942.

[4] U.S. Environmental Protection Agency (EPA), Risk Management Sustainable Technology, Life Cycle Perspective, http://www.epa.gov/nrmrl/std/lifecycle.html

[5] Srinivasan, R., (2011)Sustainability Analysis and Connective Complexity Method for Selective Disassembly Time Prediction, M.S. Thesis, Washington State University, Pullman, WA, USA.

[6] Wikipedia, End-of-life (Product), http://en.wikipedia.org/wiki/End-of-life_(product)

[7] Boothroyd, G., Dewhurst, P., & Knight, W. A. (2010). Product design for manufacture and assembly (Vol. 74). CRC Press.

[8] Moore, K. E., Gungor, A., & Gupta, S. M. (1998). A Petri net approach to disassembly process planning. Computers & Industrial Engineering, 35(1), 165-168.

[9] Gungor, A., & Gupta, S. M. (1998). Disassembly sequence planning for products with defective parts in product recovery. Computers & Industrial Engineering, 35(1), 161-164.

[10] NAVTN-CHANDRA, D. U. N. D. E. E. (1994). The recovery problem in product design. Journal of Engeering Design, 5(1), 65-86.

[11] DeRon, A. and Penev,K., (1995), Disassembly and recycling of electronic consumer products: an overview. Technovation, 15, 363-374.

[12] Penev, K.D. and deRon, A. J., (1996), Determination of a disassembly strategy. International Journal of Production Research, 34, 495-506.

[13] Gupta, S. K., Regli, W. C., Das, D., & Nau, D. S. (1997). Automated manufacturability analysis: a survey. Research in Engineering Design, 9(3), 168-190.

[14] Fleischmann, M., Bloemhof-Ruwaard, J. M., Dekker, R., Van Der Laan, E., Van Nunen, J. A., & Van Wassenhove, L. N. (1997). Quantitative models for reverse logistics: a review. European journal of operational research, 103(1), 1-17.

[15] Thierry, M. C., Salomon, M., Nunen, J. V., & Wassenhove, L. V. (1995). Strategic issues in product recovery management. California management review, 37(2), 114-135.

[16] Gungor, A., & Gupta, S. M. (1999). Issues in environmentally conscious manufacturing and product recovery: a survey. Computers & Industrial Engineering, 36(4), 811-853.

[17] Guide, V. D. R. (2000). Production planning and control for remanufacturing: industry practice and research needs. Journal of Operations Management,18(4), 467-483.

[18] Lund, R. (1996) The Remanufacturing Industry: Hidden Giant, Boston, Massachusetts: Boston University.

[19] Hatcher, G. D., Ijomah, W. L., & Windmill, J. F. (2013). Design for remanufacturing in China: a case study of electrical and electronic equipment. Journal of Remanufacturing, 3(1), 1-11.

[20] Matsumoto, M., & Umeda, Y. (2011). An analysis of remanufacturing practices in Japan. Journal of Remanufacturing, 1(1), 1-11.

[21] Behdad, S., Kwak, M., Kim, H., & Thurston, D. (2010). Simultaneous selective disassembly and end-of-life decision making for multiple products that share disassembly operations. Journal of Mechanical Design, 132, 041002.

[22] Boothroyd, G., & Alting, L. (1992). Design for assembly and disassembly.CIRP Annals-Manufacturing Technology, 41(2), 625-636.

[23] Mathieson, J. L., Wallace, B. A., & Summers, J. D. (2012). Assembly time modelling through connective complexity metrics. International Journal of Computer Integrated Manufacturing, (ahead-of-print), 1-13.

[24] Mathieson, J. L., & Summers, J. D. (2010). Complexity metrics for directional node-link system representations: theory and applications.Proceedings of the ASME IDETC/CIE 2010, 15-18.

[25] Desai, A., & Mital, A. (2003). Evaluation of disassemblability to enable design for disassembly in mass production. International Journal of Industrial Ergonomics, 32(4), 265-281.

[26] Kroll, E., Beardsley, B., & Parulian, A. (1996). A methodology to evaluate ease of disassembly for product recycling. IIE transactions, 28(10), 837-845.

[27] Zussman, E., Kriwet, A., & Seliger, G. (1994). Disassembly-oriented assessment methodology to support design for recycling. CIRP Annals-Manufacturing Technology, 43(1), 9-14.

[28] Güngör, A., & Gupta, S. M. (2002). Disassembly line in product recovery.International Journal of Production Research, 40(11), 2569-2589.

[29] Kwak, M. J., Hong, Y. S., & Cho, N. W. (2009). Eco-architecture analysis for end-of-life decision making. International Journal of Production Research,47(22), 6233-6259.

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

[31] Gerner*, S., Kobeissi, A., David, B., Binder, Z., & Descotes-Genon, B. (2005). Integrated approach for disassembly processes generation and recycling evaluation of an end-of-life product. International Journal of Production Research, 43(1), 195-222.

[32] Yi, J., Yu, B., Du, L., Li, C., & Hu, D. (2008). Research on the selectable disassembly strategy of mechanical parts based on the generalized CAD model. The International Journal of Advanced Manufacturing Technology,37(5-6), 599-604.

[33] Behdad, S., Kwak, M., Kim, H., & Thurston, D. (2010). Simultaneous selective disassembly and end-of-life decision making for multiple products that share disassembly operations. Journal of Mechanical Design, 132, 041002.

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

[35] Kara, S., Pornprasitpol, P., & Kaebernick, H. (2006). Selective disassembly sequencing: a methodology for the disassembly of end-of-life products. CIRP Annals-Manufacturing Technology, 55(1), 37-40.

[36] Smith, S. S., & Chen, W. H. (2011). Rule-based recursive selective disassembly sequence planning for green design. Advanced Engineering Informatics, 25(1), 77-87.

[37] Smith, S., Smith, G., & Chen, W. H. (2012). Disassembly sequence structure graphs: An optimal approach for multiple-target selective disassembly sequence planning. Advanced Engineering Informatics, 26(2), 306-316.

[38] Sedgewick, R. "Algorithms in C++, Part 1-4 Fundamentals Data Structures Sorting Searching." (2002). [51] http://www.stoimen.com/blog/2012/08/31/computer-algorithms-graphs-and-their-representation/

[39] Lambert, A. J. (2003). Disassembly sequencing: a survey. International Journal of Production Research, 41(16), 3721-3759.

[40] Tang, Y., Zhou, M., Zussman, E., & Caudill, R. (2000). Disassembly modeling, planning and application: a review. In Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on (Vol. 3, pp. 2197-2202). IEEE.

[41] Zhang, H. C., & Kuo, T. C. (1997, October). A graph-based disassembly sequence planning for EOL product recycling. In Electronics Manufacturing Technology Symposium, 1997., Twenty-First IEEE/CPMT International (pp. 140-151). IEEE.

[42] Zhang, H. C., & Kuo, T. C. (1996, October). A graph-based approach to disassembly model for end-of-life product recycling. In Electronics Manufacturing Technology Symposium, 1996., Nineteenth IEEE/CPMT (pp. 247-254). IEEE.

[43] Homem de Mello, L. S., & Sanderson, A. C. (1990). AND/OR graph representation of assembly plans. Robotics and Automation, IEEE Transactions on, 6(2), 188-199.

[44] Homem de Mello, L. S., & Sanderson, A. C. (1991). A correct and complete algorithm for the generation of mechanical assembly sequences. Robotics and Automation, IEEE Transactions on, 7(2), 228-240.

[45] Reddy, V. N., Mavrovouniotis, M. L., & Liebman, M. N. (1993, July). Petri net representations in metabolic pathways. In ISMB (Vol. 93, pp. 328-336).

[46] Suzuki, T., Kanehara, T., Inaba, A., & Okuma, S. (1993, May). On algebraic and graph structural properties of assembly Petri net. In Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on(pp. 507-514). IEEE.

[47] Zussman, E., & Zhou, M. (1999). A methodology for modeling and adaptive planning of disassembly processes. Robotics and Automation, IEEE Transactions on, 15(1), 190-194.

[48] Tiwari, M. K., Sinha, N., Kumar, S., Rai, R., & Mukhopadhyay, S. K. (2002). A Petri net based approach to determine the disassembly strategy of a product. International journal of production research, 40(5), 1113-1129.

[49] Cao, T., & Sanderson, A. C. (1998). AND/OR net representation for robotic task sequence planning. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 28(2), 204-218.

[50] Saitou, K. (2011).Built to be reclaimed,Mechanical Engineering, V. 133(9),52-54

[51] Adjacency list, http://en.wikipedia.org/wiki/Adjacency_list#cite_note-A-3

[52] Rossum, G., Python Patterns - Implementing Graphs, 1998

[53] Hash table, http://en.wikipedia.org/wiki/Hash_table

[54] Goodrich, M. T., & Tamassia, R. (2006). Algorithm Design: Foundation, Analysis and Internet Examples. Wiley. com.

[55] Skiena, S. S. The Algorithm Design Manual. 2008.

[56] Wikipedia, http://en.wikipedia.org/wiki/Breadth-first_search

[57] Keller, R., Eckert, C. M., & Clarkson, P. J. (2006). Matrices or node-link diagrams: which visual representation is better for visualising connectivity models?. Information Visualization, 5(1), 62-76.

[58] Mate PropertyManager - Mates Tab, http://help.solidworks.com/2013/English/SolidWorks/sldworks/HIDD_NEW_ADD_MATE.htm?id=4e042839087a40b5a731948d394e6530#Pg0

[59] Glassner, A. (Ed.). (1989). An introduction to ray tracing. Morgan Kaufmann.

[60] Bounding Box for Machined parts, http://www.ddicad.com/blogs/techcenter/2012/11/29/bounding-box-for-machined-parts/

[61] Bounding box http://help.solidworks.com/2013/English/solidworks/sldworks/c_wn2013_3d_bounding_boxes.htm

[62] Grab CAD http://grabcad.com/