5-2012

Automated Assembly Time Prediction Tool Using Predefined Mates From CAD Assemblies

Joseph Owensby
Clemson University, jowensb@g.clemson.edu

AUTOMATED ASSEMBLY TIME PREDICTION TOOL USING PREDEFINED MATES FROM CAD ASSEMBLIES

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Mechanical Engineering

---

by
Joseph Eric Owensby
May 2012

Accepted by:
Dr. Joshua D. Summers, Committee Chair
Dr. Gregory M. Mocko
Dr. Brian Malloy

ABSTRACT

Current Design for Assembly (DFA) methods and tools require extensive amounts and types of user inputs to complete the analysis. Since the methods require extensive amounts and types of inputs, certain issues arise: the analysis can become tedious, time consuming, error prone, and not repeatable. These issues eventually lead to the DFA methods being used as a redesign tool or not being implemented at all.

The research presented in this thesis addresses the current DFA limitations and issues by developing and implementing an automated assembly time prediction tool that: extracts explicitly defined connections from SolidWorks assembly models, determines the structural complexity vector of the connections, and inputs the complexity vector into trained artificial neural networks (ANNs) to predict an assembly time. The automated assembly time prediction tool does not require any user inputs other than a mated assembly model. To complete the analysis with the automated tool, the user has to open up the assembly model and click on the developed SW add-in button. Since no additional inputs are required to complete the analysis, the results are completely repeatable when given the same SolidWorks assembly model to evaluate.

The results in this thesis show that the developed tool can predict a product's assembly time with as little as 4% error or with as much as +68% error depending on the ANN training set used. Eight different ANN training sets are tested in this thesis, the results show that larger more variable ANN training sets typically predict assembly times with less percent error than smaller less variable ANN training sets. Since the tool extracts mates from assembly models, the sensitivity of the method with respect to

different mating styles is also investigated. It is determined that the mating style does have an effect on the predicted assembly time, but this effect is typically within the normal variation ranges of existing DFA methods.

DEDICATION

I would like to dedicate this thesis to my friends and family, there enduring support enabled me to complete this work.

ACKNOWLEDGMENTS

I would like to thank my advisory committee: Dr. Joshua Summers (chair), Dr. Gregory Mocko, and Dr. Brian Malloy for their support and guidance throughout the course of this work.

I would also like to thank the members of the CEDAR lab who supported me throughout my two years in the Masters program here at Clemson. Without such an amazing research lab the quality of my graduate experienced would have been significantly diminished. I extend a special thanks to Aravind Shanthakumar, Essam Namouz, and Vikrant Rayate for their specific contributions that helped me complete and publish parts of this work. I also extend a special thanks to the CEDAR lab mates that provided valuable feedback and much needed encouragement at our weekly Wednesday meetings.

Finally, I would also like to thank the Department of Mechanical engineering for providing me with a Teaching Assistantship which provided the funding that allowed me to complete my Masters degree.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1. DESIGN FOR ASSEMBLY: MOTIVATION AND CHALLENGES TO AUTOMATION

This thesis presents a design tool to automatically predict a product's assembly time by extracting defined connections from assembly models from a commercial solid modeling system (SolidWorks). The tool is defined by three elemental steps: (1) extract the explicitly defined mating connections from SolidWorks assembly models, (2) determine the structural complexity vector of the connection graphs, and (3) input the complexity vector into a trained artificial neural network to predict the assembly time.

The initial motivation for this work originated from the author's personal experience applying the original table based Boothroyd Dewhurst Design for Assembly (DFA) method to the re-design of a Black and Decker One Touch Chopper. The results of the analysis identified the initial assembly time as 228.5 seconds and a redesign assembly time of 201 seconds reducing the assembly time by 12%. The ability of the method to improve the design with respect to assembly was recognized, but completing the analysis was tedious, time consuming, and largely subjective. The author, and others in literature, determined that if the benefits do not significantly outweigh these issues then designers will be reluctant to implement DFA methods resulting in poorly designed products [1,2,3,4]. To mitigate these issues, automated assembly time prediction is recommended.

To ensure clarity of discussion, the difference between tools and methods as used in this research must be defined. A method is a process that ends with defined results and a tool is a specific way to achieve the defined results. Therefore, a DFA method is a

process applied to a product to improve it with respect to assembly. An example of a DFA method is assembly time prediction which can be used to determine and reduce a products assembly time. A product's assembly time can be predicted through a variety of approaches, these different approaches can become different DFA tools. DFA tools are specific ways to improve a product with respect to assembly. Based on their success at providing measurable criteria that can be used to analyze and improve designs, assembly time prediction tools are a critical part of effective DFA methods [5]. An example of a DFA tool used within the assembly time prediction method is the connectivity complexity assembly time prediction tool [6]. Assembly time prediction and the development of an automated tool are the focus of the research presented in this thesis. To understand the limitations and issues of current methods, the rest of this chapter presents a variety of DFA methods and tools with a specific focus on attempts at automating them.

An overview of basic DFA methods along with their benefits and issues is covered in the remainder of Chapter 1. . This review is continued into Chapter 2. , where the focus shifts towards specific research efforts that attempt to automate existing DFA methods and how these might be exploited in new automation efforts.

1.1 Overview of Design for Assembly and Assembly Time Prediction

Design for Assembly (DFA) methods have been extensively researched since the 1960's, progressing from basic rules/guidelines to the development of fully automated analysis tools [7,8]. The progression of DFA research is illustrated in Figure 1.1. Integration of DFA methods into software focuses on the development of software

versions of existing DFA methods that still require user inputs to complete the analysis, Figure 1.1. Automation of DFA methods in Figure 1.1 is defined as systems that extract some or all of the inputs required for assembly time analysis, requiring minimal user inputs.

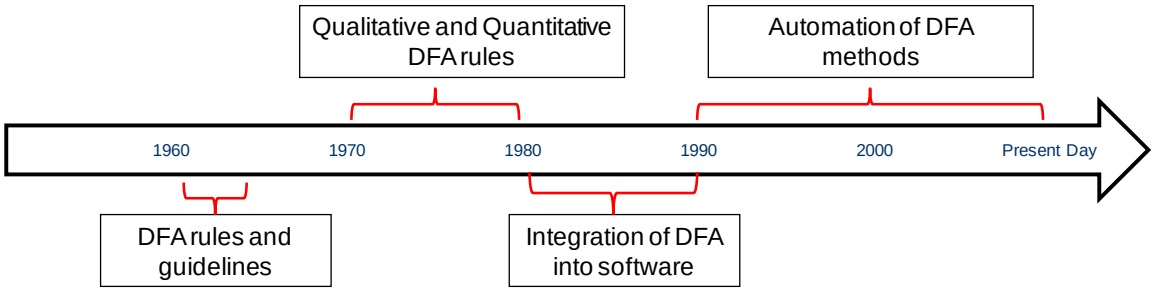


Figure 1.1: DFA research timeline

Design for Assembly (DFA) methods originated in the 1960's when companies first started publishing manuals to aid designers during the design process [7,8]. These manuals gave the designers basic guidelines to improve their products with regards to manufacturing and assembly [7]. In the 1980's, these guidelines were integrated into systematic qualitative/quantitative DFA analysis tools that would help the designer predict the products assembly time [7,9]. These systematic DFA analysis tools used extensive time studies to develop tables where users assigned assembly penalties based on individual part features to predict a product's assembly time [7,9]. These tools help the designer identify the products assembly cost and measure design improvements with respect to the assembly times [7]. After the development of these table based methods (approximately 1970-1980 in Figure 1.1) researchers began to realize the advantages of implementing DFA through computer software to improve the speed and ease of the analysis [8,10,2]. This research is shown from the early 1980's until the 1990's in Figure

1.1 where integration of DFA into software focuses on taking existing methods and making software versions of them. These software implementations of DFA methods improved the issues of analysis setup time, but they still required the user to go through the systematic process and provide the software with the required inputs. This directed the focus of DFA software implementation towards automating DFA methods shown from 1990 to the present day in Figure 1.1.

The development of automatic DFA methods focuses on implementing methods through software that gather required inputs from an external source, typically two-dimensional or three dimensional modeling software, rather than relying on the user's input [3]. Modeling systems store geometric information about the product that can be extracted and used to provide some of the inputs to the methods. This type of DFA method would limit the amount of input information required from the user, thereby improving the speed and consistency of the analysis [10].

Specific DFA methods that represent the different eras shown in Figure 1.1 are listed in Table 1.1. Table 1.1 contains the name of the DFA method, a description of the method, the developer of the method, and the date the method was created. Some of the frequently used or researched DFA methods shown in Table 1.1 are the Boothroyd Dewhurst method [11], the Methods-time Measurement (MTM) method [12], the Assembly Evaluation Method of Hitachi [13], and the Lucas Method [14]. The Boothroyd Dewhurst method, highlighted in Table 1.1, is evaluated and used to complete the research presented in this thesis.

Table 1.1: Existing DFA Methods

DFA Method	Description	Developer	Date	Ref.
Methods-Time Measurement (MTM)	Assign operations with pre defined assembly times to parts	Harold Maynard	1948	[12,15]
Manufacturing Producibility Handbook	Reference manual of manufacturing and assembly guidelines	Corporation (GE)	1960	[7]
Boothroyd and Dewhurst DFA	DFA based on minimum part criteria and handling and insertion difficulties	Academic & Consulting (Boothroyd and Dewhurst)	1977	[11,7]
Assembly Evaluation Method (AEM)	DFA based on one motion for one part	Corporation (Hitachi)	1980	[7,13, 16]
Design for Assembly and Cost Effectiveness (DAC)	Uses 30 key words to evaluate design	Corporation (Sony)	1988	[7,17]
Assembly Oriented Product Design	Accesses a parts functional value	Warnecke & Bassler	1988	[7]
Lucas DFA Method	Set of questions to determine assembly time	Academic & Consulting (Miles & Swift)	~1986	[2,7]
MOSIM	Focus of implementing DFA through CAD software	Corporation (Angermuller & Moritzen of Siemens)	1990	[7]
DFA Sandpit	Proactive DFA software based on original Lucas method	Academic (Swift & Jared)	2000	[18,3]

The different DFA methods in Table 1.1 highlight some of the prevalent DFA methods developed in both academia and industry. From the descriptions of the methods in Table 1.1 the variety of approaches that researchers have applied to DFA can be seen. Some methods like the MTM method conduct the analysis by evaluating the assembly motions and others like the Lucas method focus on indentifying a part's features that make it difficult to assemble. The variety of developers in Table 1.1 show that the push

to develop more effective methods is not driven by one group or type of researcher, but instead by a wide range of researches including both corporations and academia.

The Boothroyd Dewhurst DFA method, the Lucas DFA method, and the DFA Sandpit shown in Table 1.1 relate to the research presented in this thesis and are discussed in Chapter 2. . Details on other methods can be found by following the respective reference. Many of these methods have been implemented in industry and shown to provide benefits improving the design with respect to assembly but they still have issues, for example the subjectivity of the user inputs. The benefits of DFA are explored in Section 1.2 while some issues with DFA methods are considered in Section 1.3.

1.2 Identified Benefits of Existing DFA Methods

Since up to seventy percent of a product's life cycle cost is determined early in the design process, it is important to conduct DFA analyses early to improve the design before the majority of its cost has been determined [19,20,18,2,21]. Further, nearly forty percent of manufacturing cost can be related directly to assembly costs [22]. Incorporating DFA methods early into the design process provides advantages such as shortened development time, assembly time reduction, and manufacturing cost savings [8]. DFA methods have also been industry tested and proven to be advantageous by reducing a product's total part count, manufacturing cost, production lead time, inventory, assembly time, and assembly cost [8,20,23]. Table 1.2 summarizes some of the recorded DFA benefits, the effect the benefit has, and the references that identified

these benefits. The benefits listed were identified by applying or observing a DFA method.

Table 1.2: Identified DFA benefits

Reference	Benefit	Effect
[24,25,26,8]	Reduced product cost	Increase profit, reduce consumer expense
[8,20,23,7,16,8,27,28,3,26]	Reduced assembly time and cost	Increase production volume
[24,8,20,23,8,28]	Reduces manufacturing time and cost	Increase production volume
[8,20,23,25]	Reduced part count	Reduces mass, assembly time, and cost
[7,16]	Reduced design time	Improves use of resources
[24]	Reduces repair costs	Cost savings
[28,7,16,8,28,25]	Improved quality and reliability	Happy consumer
[8]	Fewer suppliers	Improves use of resources
[24,8,20,23,8]	Reduced inventory	Improves use of resources
[27,8,20,23,8,3,25]	Reduced product development time and time to market	Improves use of resources
[25]	Minimize assembly problems	Efficient assembly process

Table 1.2 lists eleven different benefits identified by eleven different researchers which can be achieved by applying DFA methods in the design process. Every benefit has a resulting effect which is what the companies ultimately want to achieve by implementing DFA methods. By implementing DFA methods the benefits listed are achieved which in turns effectives the company positively by reducing the cost to produce the product and increasing the company's profit. One case study presents Motorola's application of DFMA methods to a vehicular adaptor [8]. The results of the

DFMA study improved the assembly time by 87%, decreased the part count by 78%, and eliminated all of the fasteners [8]. This is only one case study out of many that prove that companies are interested in applying DFA methods to achieve their benefits and the resulting positive effects. Even with these identified benefits, DFA methods are often not implemented in industry because of their associated issues. These hindering issues and their effects are identified and discussed in Section 1.3.

1.3 Identified Issues with Existing DFA Methods

Even with the proven benefits achieved by applying DFA methods, they still have associated limitations and issues that hinder their full industrial acceptance and implementation. One issue is that the development of DFA methods often focuses on generating stand alone tools that are intended to improve designs with respect to assembly [10]. Stand alone systems require the user to balance their mental resources as they switch back and forth from designing to analysis instead of focusing on one specific aspect at a time [10]. The ideal analysis tool would be integrated into the current computer aided design and modeling tools thus reducing the burden on the designer [10].

Another issue is that these current DFA tools require inputs and calculations from the user to complete the analysis. These inputs may range from envelope dimensions of parts to specific motions required by an operator to assemble the part. Calculations required by the user may be summing the number of parts in an assembly or calculating the products design efficiency. If the inputs required by the method are subjective and require user interpretation, the result will vary within the analysis [10,4,1]. The variability of the results will be present between different users conducting the analysis

on the same product, and the variability can even be present between the same user conducting the analysis on the same product at a different time [29]. For some methods even a small user interpretation could result in +/- 50% error depending on how often that part is being used [30]. Also, if the user is required to enter extensive inputs to complete the analysis the method can become tedious and time consuming [31]. Some identified DFA issues and their resulting effect on the analysis are summarized in Table 1.3.

Table 1.3: Identified DFA issues

Reference	Issues	Effect
[7,1,10,26]	Requires subjective or implicit user inputs	Varying results, user interpretation
[31,1]	Tedious	Reluctance to use, accidental errors
[31,1,10,3,25]	Time consuming	Reluctance to use, accidental errors
[31,1,10,26]	Extensive user inputs	Reluctance to use, accidental errors, distraction from design
[27,18]	Require design details (geometry, etc.)	Used late in design process
[29,27,3,25,18,26]	Reactive or redesign tools	Used late in design process, less cost impact, lost benefits
[10]	Stand alone systems	Increases design difficulty
[17]	Implicitly identified design improvements	Varying results, user interpretation
[26]	Lack foundation to relate DFA time and cost to part geometry	Difficult to automate

Several of the issues in Table 1.3 ultimately lead to reluctance in industry to implement the DFA methods [10,2] which, in turn, prevents the DFA benefits from being achieved. The issues of requiring design detail or the use of the methods as redesign tools force the methods to be used iteratively possibly increasing the cost of design changes [27,18]. Examples of the design details required by some methods include:

geometric information, securing methods, or assembly motions which are not generally known until the detailed design stage of the design process. Section 1.4 explores where existing DFA methods are used in the design process based on the information required by the method, where the greatest benefits are achieved by implementing the method, and where issues with the methods are encountered.

1.4 DFA in the Design Process

DFA methods were originally intended to be applied throughout the design process to maximize cost savings [10,7]. These cost savings can be maximized since 60% to 80% of a products cost is determined during the early phases of the design process [20,2,21]. By applying DFA methods early in the process, the design can be changed before it is finalized, which maximizes the resulting DFA benefits [32,2,8]. Figure 1.2 shows a simplified version of a systematic design process along with where DFA methods are currently used and where they would ideally be used [33].

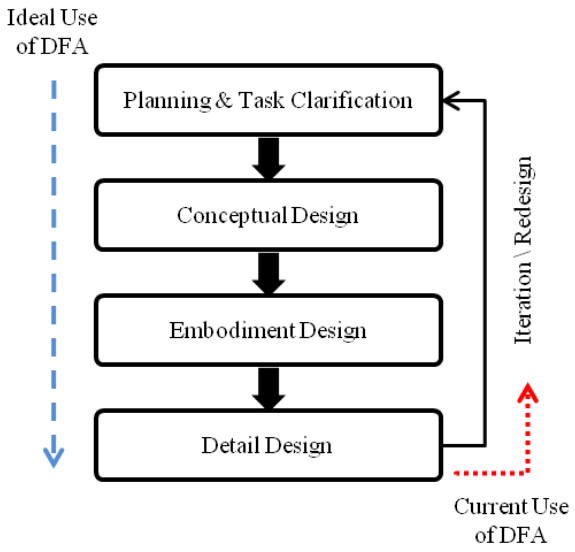


Figure 1.2: DFA in the Design Process (Adapted from [33])

Due to the issues with DFA methods and the lack of design details early in the design process, DFA is typically used as a redesign tool instead of a forward engineering tool so the full benefits during the initial product design are seldom achieved [34,25]. Also, many of the methods developed in the last fifteen years require information that is only available during or after the embodiment design stage which is late in the design process [27]. If DFA methods are applied late in the design process and design improvements are identified, the product will have to be redesigned resulting in an iterative redesign phase which increases development cost [35,27]. Designers are more likely to welcome design suggestions if they are made concurrently and early throughout the process and less likely to welcome suggestions if they are made based on identified weaknesses of their final product [3].

The frequent use of DFA methods as redesign tools is demonstrated in the following example. The Boothroyd Dewhurst DFMA software is one of the most widely published and used methods in industry today [28,27,26]. The Boothroyd Dewhurst DFMA website posts eighteen case studies that all boast a variety of benefits that different companies achieved by implementing DFA on their products [36]. All eighteen of these case studies proved to be beneficial but all of them are with regards to the redesign of an existing product, not the design of a completely new product.

To reduce or eliminate DFA issues and to improve the methods so that they can be more effectively used and applied earlier in the design process, research has shifted its focus towards integrating methods into computer aided design and solid modeling systems. Chapter 2. focuses on the DFA methods that are partly automated or implemented through Computer Aided Design (CAD) systems.

CHAPTER 2. DESIGN FOR ASSEMBLY METHODS

For DFA methods to be truly effective the current issues that they posses have to be eliminated. This goal can theoretically be achieved by implementing the methods through computer software. If the issues with DFA can be eliminated, or at least mitigated, through computer based implementation, this approach essentially becomes a requirement for all DFA methods [25]. Attempts at meeting this requirement have been ongoing since the early 1980s [7]. Before discussing the progression of DFA methods, these different attempts have to be classified into one of the following categories:

- Manual Methods: the user conducting the analysis provides all of the information required by the method to complete the analysis
- Semi-Automated Methods: a portion of the information required by the method can be extracted from an external source other than the user
- Automated Methods: this method requires no information from the user, all information required by the method is extracted from an external source

The typical development progression of a DFA method is from completely manual to computer implementation to automate the method as much as possible. This typically results in a semi-automated DFA method since some of the required information cannot be extracted from external sources. The primary focus of this chapter is on DFA methods that are implemented through computer software and whether those methods are manual, semi-automated, or fully automated. The rest of this section presents the

movement towards automated DFA followed by sections detailing the progression or automation of specific methods.

The early attempts of software based DFA focused on developing stand alone programs that were essentially computer based versions of the original methods [14,32]. The user must answer the same questions required by the original method to complete the analysis, but the computer software would accept the answers as inputs, and compute the outputs. These computer based DFA methods improved some of the DFA issues by hiding some of the information processing from the user [10]. While these software based DFA methods improved the issues, they did not eliminate them. One study showed that both experienced and novice DFA users conducting manual DFA with or without the computer software based version would complete the analysis with approximately the same number of mistakes [10]. This study shows that by converting manual DFA methods to manual input computer software versions of the method, the fundamental problems with the methods are still not solved. To reduce the number of mistakes and improve the methods, the methods should be partially or fully automated requiring little analysis input from the user [26].

The natural progression of implementing a software based DFA method was to program the original methods, then shift the focus towards automating the methods. The ideal DFA method would be fully automated so that it could give the designer repeatable feedback to improve the design with respect to assembly in real time as they go through the design process [14]. This would eliminate the tediousness, subjectivity, time consuming issues that reduce current DFA implementation. A semi-automatic or fully

automatic DFA method would also allow the designer to focus primarily on the functionality of the product instead trying to consider functionality and assemblability at the same time [26]. Attempts at automating current DFA methods have been inhibited since they often use a variety of subjective information which is difficult to program [1,37]. Even though the automation of methods has been inhibited, extensive work has been completed to automate parts of these methods (semi-automated) and to reduce the effort required to complete them.

All DFA methods require inputs to complete the analysis and so the first step in automation is to determine what sources can provide these inputs if they are not received from the designer. The answer that most researchers have identified is that some of the required information to complete the given analysis can be extracted from solid models of parts or assemblies [3]. This requires geometric reasoning algorithms to evaluate, interpret, and extract the information from the solid models [25]. In most cases, only the objective inputs like part symmetry could be extracted from these models. Subjective information like difficulty to handle would be difficult to extract because the information about the part would have to be interpreted and analyzed to come up with the subjective information. Another issue with automating these methods is that they require geometric information from the models which may not be known until late in the design process directing their use as a redesign tool instead of a concurrent design tool [18]. The result of this previous research has been methods that are partially automated by extracting specific parts of the analysis from solid modeling software. These partially automated methods improve the issues but do not eliminate them [18].

The issues that prevent a fully automated DFA method go back to the original development of software based DFA. Since the progression of software DFA methods started with the programming of manual DFA methods, any flaws or requirements that the original methods had would also be integrated into software. Some of these flaws are the types of information required to complete the analysis. If the method requires subjective information then a fully automated version of that method would be difficult to achieve. Typical computer algorithms solve step by step calculations. A computer algorithm to solve for subjective information would require reasoning and interpretation which varies based on the given perspective. A program to solve for subjective information would be difficult to develop without a detailed knowledge base and complicated algorithms. To fully or partially automate DFA, methods that are based on objective low level or geometric information about the product have to be developed [26]. The next several sections focus on the development and extension of prevalent DFA methods specifically with regards to implementing them into software or attempts at automating them.

2.1 Development and Extension of the Boothroyd Dewhurst DFA Method

The Boothroyd Dewhurst DFA method was one of the first systematic approaches applied to DFA allowing designers to quantitatively compare different designs with respect to assembly [7]. The method was developed by conducting extensive time studies and relating different design features to certain assembly time penalties. To complete the analysis the user has to answer a series of questions relating to: minimum part criteria, envelope dimensions, securing method, handling difficulties and insertion difficulties [8]

[11]. The minimum part criteria questions are used to identify the theoretical minimum number of parts that the product can have [11]. The designer then evaluates the parts that are identified and eliminates them or re-designs them if the design can be improved. The questions relating to envelope dimensions, handling difficulties, and insertion difficulties are used to predict the products assembly time and cost providing the designer with a quantitative way to evaluate the designs [11]. The handling and insertion difficulty questions are typically subjective where the answer to the questions can vary based on user interpretation [1]. One example of a subjective handling difficulty is "does the part severely nest or tangle" and an example of an insertion difficulty is "is the part easy to align."

The questions required to complete the analysis are presented to the user through a set of paper based tables. The user has to choose the row and column that best describe the given part and the time penalty will be at the intersection. There are four handling difficulty tables and three insertion difficulty tables that have to be considered while determining the handling and insertion times. [11]

The original Boothroyd Dewhurst table based method provides a systematic way to improve designs with respect to assembly but to complete the analysis the user has to manage all of the information required by the analysis [11]. The amount of information and time required to complete the analysis grows with the number of components. The subjectivity of many of the required inputs results in variability between analyses. These and other issues are the driving factors that push research focused on the original Boothroyd Dewhurst DFA method towards automation of the method. The rest of this

section presents the continual development and extension of the Boothroyd Dewhurst DFA method. Section 2.1.1 discusses the Boothroyd Dewhurst DFMA software (a computer based version of the original method), Section 2.1.2 discusses a Product Architecture based method that allows the Boothroyd method to be applied earlier in the design process, and Section 2.1.3 discusses Fuzzy DFA which attempts to automate parts of the Boothroyd Dewhurst method.

2.1.1 Boothroyd Dewhurst DFMA Software

The Boothroyd Dewhurst DFMA software is discussed in detail in Chapter 4. but a brief overview of its development and usage is presented in this section.

After the original development of the Boothroyd Dewhurst table based DFA method, focus shifted towards implementing this method into a computer version to improve the issues of the original method [32,38]. Early versions of the software presented the same number and types of questions to the user but improved the analysis since the user no longer had to manage the information. Once the answer to a question was specified the software would make the required calculations, display the results to the user, and store the information as needed [32].

The Boothroyd Dewhurst DFA software has been continuously developed and improved from a basic computer version of the original methods in the early 1980's to a method that now presents the questions to the user through a user friendly GUI. The user friendly GUI reduces the DFA issues by hiding information from the user and providing the user with hints to help reduce the subjectivity of the inputs. Extensive industry case studies conducted throughout the development of the Boothroyd Dewhurst software have

continued to show its benefits [20,8] but evaluations of the method continue to show that it still has issues [1]. The implementation of the original Boothroyd Dewhurst tables into the DFMA software improve the method but until more information can be extracted from 3D modeling programs, the DFA analysis will most likely be conducted after the parts are designed, minimizing the benefits that the method would provide to the user[7]. The following sub sections discuss several research efforts outside of the original developers which have explored a variety of other ways to use or implement the original Boothroyd method in hopes to eliminate the exposed issues.

2.1.2 Product Architecture Based Conceptual DFA

The Product Architecture Based Conceptual DFA method was developed so that DFA could be applied during the conceptual design phase using function models of the given product [27]. The steps to complete the method are as follows:

1. Generating a function structure of the desired product
2. Identify product modularity by apply heuristic methods to developed function structures
3. Conceptual design is applied to each module trying to solve the functional requirement of each module with as few components as possible (goal is one component per module)
4. Optional: Apply Boothroyd Dewhurst method to predict and reduce product's assembly time based on handling characteristics

The product function structures are generated using the functional basis which is a standardized vocabulary used to specify the verb-object pairs required by the structures [27]. Using function structures as inputs allows this conceptual method to be applied

with just the functional requirements of the product, not geometric information which is required by other DFA methods and often not known early in the design process.

One issue with this conceptual DFA method is that it requires function structures as inputs which can be difficult to generate and vary between design concepts [39]. This method does not appear to be automated in any way so all of the required inputs must be generated and provided by the designer. The only advantage that this method may provide is that it would force the design to consider reducing part count during conceptual design as opposed to thinking it about it post design.

2.1.3 Fuzzy DFA

The Fuzzy DFA method automates part of the Boothroyd Dewhurst DFA method using geometric reasoning, artificial intelligence, and fuzzy logic resulting in a semi-automated DFA method [28]. Fuzzy logic is the attempt to simulate a human being's reasoning and approximation capabilities which allow for imprecision in the final result [37]. Fuzzy DFA uses fuzzy logic to computationally interpret the subjective information inputs required by DFA methods. Fuzzy DFA automates part of the Boothroyd Dewhurst method by combining the original method with fuzzy logic which can then be used to provide inputs to the analysis using feature based codes. Fuzzy DFA breaks down the Boothroyd Dewhurst DFA method into technological inputs (handling/insertion difficulties) and geometric inputs (envelope dimensions, orientation, etc.) [28]. By applying the Fuzzy DFA method the following geometric aspects of the Boothroyd method were automated by extracting geometric information from two-dimensional and three-dimensional part models:

- Identification of rotational or non rotational parts
- Identification of minimum part bounding box
- Identifying alpha symmetry, symmetry about an axis perpendicular to the insertion axis
- Identifying beta symmetry, symmetry about the insertion axis
- Orientation considerations for automatic bowl feeding

As shown, the Fuzzy DFA only automates the geometric aspects of the Boothroyd Dewhurst DFA analysis, the user is still required to manually provide the technological inputs to complete the analysis [28]. This will successfully reduce the analysis effort required by the designer but it only provides a semi-automated method, not a fully automated DFA method. Also, applying this semi-automated Fuzzy DFA method early in the design process may be difficult since well defined geometric data may not be known.

2.2 Development and Extension of the Lucas DFA Method and the DFA Sandpit

The implementation of the Lucas DFA method into software systems and solid modeling systems has been research extensively [2,10,25,18,40,3]. This research eventually leads to the development of a proactive DFA Sandpit which attempts to eliminate the issues that the Lucas method has. The Lucas method, the DFA Sandpit, and a brief evaluation of the two are presented in the following subsections.

2.2.1 Lucas DFA Method

The Lucas DFA method requires five different types of inputs to complete the analysis: a functional, a manufacturing, a feeding, a fitting, and a gripping analysis [10]. The early paper based versions of this method received the above inputs as the user answered a variety of questions about the product and its parts. Each input serves a different purpose and provides different results that help improve the design with respect to assembly. The five analysis requirements, the goals of the analysis, and the basis of the analysis are shown in Table 2.1.

Table 2.1: Lucas Method analysis requirements and goals

Analysis Type	Goal	Evaluation Based On
Functional	Eliminate redundant components while accomplishing desired functionality	Component functionality
Manufacturing	Estimation of part manufacturing costs use to improve design	Material, manufacturing process, and geometric based complexity
Feeding	Selection of feeding tools and methods	Ease of orientation
Fitting	Stability in assembly operations	Insertion considerations
Grip	Focus on automatic assembly operations	Accessibility a part's locating features

The issues with the original paper based version of this method were the time to conduct the analysis, the tediousness of the analysis, and the subjectivity of the analysis [2]. The research focused on the Lucas method shifted towards eliminating these issues [3].

Early on the ability to implement part of or all of this method through computer software was realized. In 1989 Lucas Engineering launched the first commercially

available software based version of this method which decreased the analysis time and made the method easier to implement improving some of the previous issues [2]. The first software version of the Lucas method was a computer based implementation of the previous paper based method so it still required the user to answer the same input questions. This new Lucas DFA software was an improvement over the paper based method because it reduced the amount of information and computational requirements presented to the user but the user still had to answer the same number of questions to complete the analysis.

Continuation of this research identified that the Lucas DFA method could be improved by implementing it through solid modeling systems [41,10]. Solid modeling systems are used to generate virtual representations of products and parts while storing information about size, location, material, and other aspects. It was determined that this stored information could be used to help complete the DFA analysis benefiting DFA by requiring less user inputs. Early attempts focused on extracting this information from 2D solid modeling drawings but issues arose as limited amounts of information required by the method were stored in these models [10,41]. Most of the information required by the Lucas method are based on geometric features which are not present in 2D solid modeling drawings, but this information is included in 3D solid modeling systems [41].

A detailed study was completed to determine how much of the Lucas method could be automated by extracting the inputs required for the analysis from 3D solid models [10]. The study breaks down each area of the user requirements to determine what aspects can be automated and to what extent. The study found that 72% of the

method could be accomplished by extracting information stored within solid models (in 1994) and that with moderate amounts of research even more information could be extracted [10]. This study showed that by extracting geometric information from solid models that a large percent of the Lucas DFA method could be automated or semi-automated but that the subjective issues like feed ability or handling aspects would be difficult to automate [10].

Around the time this study was completed (1994), the Lucas Method developed by Lucas Engineering & Systems was purchased by Computer Sciences Corporation (CSC) and integrated into their TeamSET software. The TeamSET software incorporates a variety of design tools (DFA, DFM, FMEA, QFD, and Design Target Cost (DTC)) into one encompassing tool [3]. After this integration, the Lucas Method and the TeamSET software are seldom mentioned in DFA literature. Even though the specific Lucas Method is not mentioned in literature, the fundamental aspects and the idea of incorporating them into solid modeling software was extensively researched. This research led to the development of the DFA Sandpit.

2.2.2 DFA Sandpit

From 1994 to 2004 the researchers who originally developed the Lucas method continued to focus on achieving the possible benefits identified in the Lucas DFA automation study. This research was focused on extracting the information required to complete the analysis from solid models. While conducting this research a new proactive DFA tool called the Design for Assembly Sandpit was developed [25]. The research

progression from the Lucas Method to the DFA Sandpit and the continuing research focused around improving the DFA Sandpit is summarized in Table 2.2.

Table 2.2: DFA Sandpit research and development timeline

Research Concept	Description	Year
Lucas Computer Based Method	Implementation of paper based Lucas Method into a computer software [14]	1989
Extraction of DFA information from CAD	Detailed study of what amounts and types of information required by the Lucas Method can be extracted from CAD Models [10]	1994
Prototype Assembly Oriented CAD Environment	Development and presentation of a proactive DFA Software, explanation of information required along with the GUI [25]	1999
Introduction of DFA Sandpit Software	Implementation of proactive DFA prototype system into useable software (DFA Sandpit) that also considers product functionality [18]	2000
Implementation of DFA Sandpit Software	Presents the progress of the DFA Sandpit software and describes how it can be used with a test case [3]	2000
Utilization of complexity metrics within the DFA Sandpit	A study of where and how complexity metrics and aspects can be used with proactive DFA [40]	2004

The basic research progression shown in Table 2.2 is as follows: implementation of paper based Lucas method into basic computer software, evaluation study to determine what information is required that can be extracted from solid modeling tools, development of a proactive DFA tool with a general focus on assembly sequence, prototype implementation of proactive DFA tool (assembly planning, CAD, DFA, geometric reasoning) [25], first generation of proactive DFA Sandpit which incorporates functional analysis [18], and continued advancements of the DFA Sandpit. Based on this research, an overview of the DFA Sandpit software is described below.

The goal of the DFA Sandpit was to develop a proactive DFA tool that could be used throughout the design process, not a standalone tool that is only used in the redesign process [3]. The DFA Sandpit incorporates the basic aspects of the Lucas method into three separate focus levels: the Product Group, the Product Structure, and the Component Design [3]. The Product Group helps to identify reuse of products and existing knowledge. This could include current products, the designs ability to become a platform or family based product, and reuse of existing information. The Product Structure identifies aspects about: part count, product structure, and the assembly sequence. The Component Design identifies processes related to: manufacturing, assembly, joining, insertion, and fastening. The method uses a combination of workspaces that contain knowledge and data stored in different expert systems. The method is implemented through a computer program that allows the designer to consider all aspects throughout the design process. The program has windows for the assembly sequence, structure builder, and a CAD Solid Modeler. [3] The DFA Sandpit GUI can be seen in Figure 2.1.

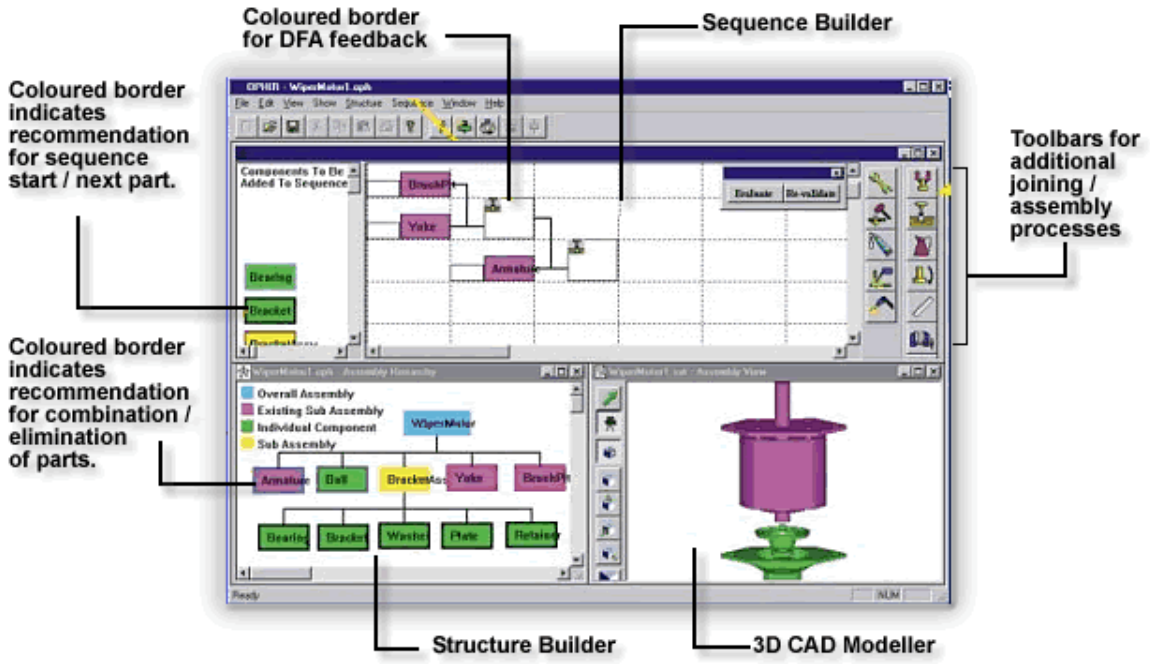


Figure 2.1: DFA Sandpit GUI [42]

Overall the DFA Sandpit improves the Lucas method by modifying it so that it can be applied early in the design process. The DFA Sandpit is also effectively implemented through a solid modeling system which: accepts the user inputs, manages the required inputs, provides guidance as needed, and provides comparison tools to evaluate the methods based on DFA. It guides the designer towards a more effective design with respect to assembly by asking the designer questions and providing windows with design suggestions as needed. The DFA Sandpit reduces the amount of information inputs required from the user which makes it more user friendly.

Even with all of the improvements, the DFA Sandpit still has its issues. The original focus was to reduce the amount and types of inputs required by the user by implementing the Lucas method through solid modeling systems. As research continued the goal shifted towards developing a proactive DFA method which requires a broader

range of user inputs to complete the analysis. Even if the DFA Sandpit does not require more information from the user to complete a more detailed analysis, the analysis covers a broader range of design issues which could distract the designer from the true goal of designing a working product. For example, while using the DFA Sandpit on one part or assembly, the user may have four windows open on the GUI at once: Product Structure/Function, Assembly Sequence, 3D CAD Modeler, and Cost Analysis windows. Using the DFA Sandpit may hide some of the volume of information required to complete the analysis, but the user will still have to interpret the volume of results required to use the analysis.

2.3 Development and Extension of the VIRAD Method

To evaluate a product early in the design process with regards to assembly and disassembly, a Virtual Assembly and Disassembly (VIRAD) system was developed [43]. This system can be integrated into CAD/CAM systems to help designers consider manufacturing and de-manufacturing issues. The method is implemented using a hierarchical work cell model called the Generic Assembly and Disassembly (GENAD) work cell that represent the production system. The GENAD model represents the variety of assembly operations using a Structured Assembly Coding System (SACS). The SACS code assigns a cost to each assembly operation so that once completed the method can predict the assembly/disassembly cost. The GENAD model based on the SACS code forms the base of the VIRAD model.[43]

The implementation of this method requires three steps to be completed for a given product: the binary part-merging tree must be extracted from solid-modeling data,

the handlers required for mating the parts together must be assigned to each part (based on SACS code), and finally the assembly instructions for the different configurations must be generated [43]. It is unclear as to how much of this method is automated based on the publication but it appears that only the first step (extraction of binary part-merging tree) is automated [43]. This means that the user still has to complete at least part of the analysis manually by performing the second step (assigning handler based SACS code) and third step (generation of assembly instructions).

2.4 Development and Extension of the Expert System for Automatic DFA

Expert or knowledge based systems are programs that use databases of stored human knowledge in the form of rules to solve problems that traditionally require human reasoning [44]. The Expert System for Automatic DFA is an expert based tool that requires limited user inputs to complete the analysis in an attempt to address the issues with existing DFA methods [31]. Existing DFA methods improve the product development time but they are static, which means that modifying them to consider other DFA aspects can be difficult. To solve this issue, the Expert DFA System does not use one large expert system but instead uses four separate expert sub systems that use expandable knowledge bases. These expandable knowledge bases can be changed so that improvements or modifications to the method can be implemented without developing an entire new system [31]. The four expert sub systems and their functionality are listed in Table 2.3.

Table 2.3: Four expert sub systems that make up the Expert DFA System [31]

Expert Sub System	Functionality
CAD expert	Extracts geometric and assembly information from CAD drawings
Automated assembly expert	Determines if automatic assembly is feasible using knowledge base system populated based on literature reviews, handbooks, and assembly experts
Manual assembly expert	Simplifies automated assembly expert results and interprets them to represent manual assembly operations
Design analysis expert	Uses knowledge base to determines assembly: cost, time, problems, suitable assembly techniques, and makes design recommendations

The Expert DFA tool uses a separate GUI for each of the expert sub systems so that the user can view one, or up to all four GUIs as needed. Determining the exact number of user inputs required by this method is difficult, but some form of user interaction and inputs are required with all four expert systems [31]. Two of the user inputs required by this method are: product specifications (production quantity / volume) and the user must specify which handling device should be used on a given part.

The Expert System for Automatic DFA solves many DFA issues by reducing the amount of information required by the user to complete the analysis. The presentation of the method seems to be nearly automated, extracting most of the required information from solid modeling drawings. This method is said to be a concurrent DFA tool which can be applied early in the design process and used throughout to improve the designs with respect to assembly.

The presented Expert DFA System uses key technologies (expandable knowledge base, artificial intelligence, etc.) to provide a robust method that solves many of the current DFA issues [31]. Even though it solves many of the issues, some user inputs are

required which means that it is only a semi-automated method. It is also an expert based system that uses some rule based suggestions so its overall effectiveness will be determined by the size of the knowledge base. Upon contacting the corresponding author, no commercially available version of the Expert DFA System is available and the research on the system ended with the graduation of one of the primary authors.

2.5 Summary of DFA Automation Attempts

The methods presented in the previous sections all result in a computer based version of an original or modified DFA. These methods, their level of automation, and what prevented their automation are summarized in Table 2.4.

Table 2.4: Success of previous DFA automation attempts

DFA Method	Level of Automation	Automation Prevented By:
Boothroyd DFMA Software	Manual	Subjective User Inputs
Product Architecture Based Conceptual DFA	Manual	Subjective User Inputs
Fuzzy DFA	Semi-Automated	Subjective User Inputs
Lucas DFA Method	Manual	Subjective User Inputs
DFA Sandpit	Semi-Automated	Subjective User Inputs
VIRAD Method	Semi-Automated	Subjective User Inputs
Expert System for Automatic DFA	Semi-Automated	Subjective User Inputs

All of these methods improve DFA analyses by reducing the input information required by the user, but all of them still require some inputs from the user. Since the methods still require some user inputs they are at best semi-automated DFA methods so they still have room for improvements.

The results of this literature review identify the need for a fully automated assembly time prediction method which is provided by the developed method covered in this thesis. To develop this new assembly time prediction method, several research questions are identified for investigation. These research questions and their hypotheses are explained in Chapter 3. and specifically addressed in Chapter 4. , Chapter 5. , and Chapter 6. .

CHAPTER 3. RESEARCH QUESTIONS

The goal of this research is to address the major issues of previous DFA methods, such as subjectivity, tediousness, and analysis time, by automating an assembly time prediction tool. The tediousness of the methods comes from the amount of time and the dullness of the tasks required by the analysis. As discussed in Chapter 2. , current DFA methods are still conducted manually or are at best semi-automated, requiring the user to provide at least some inputs to complete the analysis. The inputs required by these analyses are typically subjective information that requires interpretation by the user. Providing this subjective information automatically is difficult since it requires the development of complex algorithms which have to make assumptions or interpretations to complete the analysis [37]. To address this issue and fully automate an assembly time prediction tool, a method must be identified or developed that does not require subjective information to complete the analysis.

By automating an assembly time prediction tool to be used within solid modeling systems (SolidWorks is used in this research), the user can receive feedback about their design quickly and make changes accordingly. The feedback will be the predicted assembly times so that after making changes or modifications to the model, the user can determine if the design was improved based on an increase or decrease in the assembly time. To successfully automate this assembly time prediction method, several research questions must be addressed. The rest of this chapter presents these research questions and respective hypotheses.

The first step in automating an assembly time prediction method is to identify if an existing method can be automated. This method should require limited amounts and types of objective information that can be extracted from three-dimensional solid modeling software. By identifying a method that requires no subjective information the complex algorithms required to interpret this information can be eliminated. The benefits of automating DFA provide the motivation for the first research question.

RQ1:	Which existing assembly time prediction method should be selected for automation based on the amounts and types of information it requires?

Not all DFA methods can be partially or fully automated so in order for this research to be successful, an evaluation of existing methods with respect to their automation capabilities has to be conducted. The automation capabilities are determined by how many different types of inputs the method requires and how many of these types are subjective. The amount of information is measured by the number of different user inputs required to complete the analysis. To automate a method, a separate algorithm will have to be developed for each type of information. If the information type is subjective the complexity of the algorithm increases dramatically. The evaluation conducted will also investigate the general effectiveness of the DFA method which can be used to bench mark or identify weaknesses with existing methods. The results of this evaluation should identify an assembly time prediction method that can be easily automated; this forms the hypothesis to the first research question.

RQ1 Hypothesis:	An existing assembly time prediction method that requires limited amounts of objective user inputs can be identified for automation.

The hypothesis to the first research question is based on the review of existing DFA methods presented in the first two chapters. This review identified that no method had been currently automated but that the progression of DFA development has been towards automation. Since current DFA development focuses on automation, the amounts and types of information required by existing methods will have to be reduced or modified so that it can be interpreted using computer algorithms. By evaluating new DFA methods or updated versions of existing methods it is expected that with the recent developments one method will stand out for automation. If an existing method does not present itself as being easily automatable then a new method may have to be developed. Based on the results from the first research question, if the hypothesis is correct the next step would be to automate the identified method. If the hypothesis is not correct then a new method will have to be developed that only requires objective information before it can be automated.

Once a method that requires only objective information inputs to complete the analysis has been identified, the review in Chapter 2. identifies that to automate a method, the information required to complete the analysis must be extracted from an external source. Before the selected method can be automated, an external source that can provide the required inputs must be identified. If an external source cannot be identified then another method will have to be selected.

The most common external source identified for extracting the information required to complete an analysis is two-dimensional or three-dimensional solid modeling software [3]. If the analysis inputs can be extracted from solid modeling software then an

external information source has been identified. If the information cannot be extracted then another source must be identified, the required information has to be modified so that it can be extracted from solid modeling software, or another DFA method must be selected for automation.

This research intends to automate the identified method by providing the required analysis inputs using information extracted from three-dimensional solid modeling software. If the information cannot be extracted or interpreted to provide the required inputs, an alternate method will be selected. The extraction of information inputs from solid modeling software to complete the analysis forms the basis of the second research question.

RQ2:	Can the identified assembly time prediction method be automated so that it predicts an assembly time using information extracted from 3D solid modeling software?

The results from the first research question identified a method that only requires objective information but just because the method requires objective information does not mean that it can be automated. The second research question has to determine if this information is or is not contained within solid modeling software. If the information is not contained within solid modeling software, then other types of information that are contained within the software could possibly be used to replace the original required information.

The focus of research question two is on identifying the types of information required by the method and then determining how that information can be extracted or interpreted from information within the 3D solid modeling software. The difficulty of

this research question is based on the amounts and types of information required by the method identified by research question one. As the types of information required by the analysis increases, so does the amount and types of information that have to be extracted from a solid modeling software.

RQ2 Hypothesis:	The identified assembly time prediction method can be automated so that it predicts an assembly time using only information extracted from 3D solid modeling software.

The hypothesis for the second research question is based on the large quantity and variety of information that is currently stored within solid modeling software. If the information from the identified method is objective, it should be explicitly or implicitly available within solid modeling software. The challenge comes from identifying what information contained within the software should be used, and how to use it.

If the hypothesis to the second research question is correct then an assembly time prediction method would have been successfully automated. To determine if the automated assembly time prediction method solves the current DFA issues then it must be evaluated based on each issue which motivates the third research question.

RQ3:	Does the automated method address the issues that the previous methods have: time consuming, repeatability, ease of use?

The automated assembly time prediction tool will be analyzed with respect to each identified DFA issue listed in Table 1.3 to determine if it addresses the issue or not. The automated tool will address the issue, partially address the issue, or not address the issue and a justification for each answer will be provided. Along with specifically

addressing each DFA issue, the third research question will also be evaluated based on the DFA evaluation presented for the first research question.

RQ3 Hypothesis:	The automated method addresses the issues that current DFA methods have.

The hypothesis to the third research question is based on the existing DFA issues identified in Table 1.3. The majority of the issues that existing DFA methods have would be addressed if any of the methods were automated. For example if a method is tedious and requires extensive user inputs to complete the analysis. If the method is automated then the user no longer has to provide inputs to the method to complete the analysis completely addressing the issue. If the hypothesis to the third research question is correct then an automated assembly time prediction method will successfully address the issues of current DFA methods. These three research questions are specifically addressed in the following chapters.

CHAPTER 4. CAN A DFA METHOD BE IDENTIFIED FOR AUTOMATION?

The literature reviewed in Chapter 1. and Chapter 2. cover a variety of different DFA methods, many of which are semi-automated. These research efforts focused on developing automated DFA methods, but full automation was prevented since the methods require at least some subjective information. To develop a truly automated assembly time prediction method, a method that requires minimal amounts and types of subjective information must be identified. This can be accomplished by answering the first research question addressed by this thesis; which existing assembly time prediction method should be selected for automation based on the amounts and types of information it requires?

To answer this research question, the rest of this chapter focuses on evaluating two DFA methods, Boothroyd and Dewhurst's Design for Manufacturing and Assembly (DFMA) software and the Mathieson-Summers connective-complexity algorithm [1]. A brief overview of the Boothroyd DFMA software was presented in Section 2.1.1, but before it can be evaluated, both the DFMA software and the connective-complexity method are presented in detail within this chapter. The DFA evaluation discussed below completely analyses the methods including aspects that do and do not directly relate to DFA automation. This complete evaluation is presented to understand the strengths and weakness of the methods, not just the aspects of the method that relate to DFA automation. Even though it is a complete evaluation, it does focus on the amount of information required from the designer to complete the analysis and the subjectivity of

this information. These aspects of the evaluation can then be used to determine which method should be selected for automation.

4.1 Overview of DFA Evaluation

To effectively benchmark and improve DFA methods they need to be evaluated to identify their strengths and weaknesses so that future research and development can focus on improving their critical needs. By conducting a DFA evaluation, the general effectiveness of a method and its automatibility can be determined. Based on the variety of methods reviewed in Chapter 1. and Chapter 2. , two methods were chosen for this evaluation. The Boothroyd Dewhurst Design for manufacture and assembly (DFMA) method was chosen for evaluation based on its extensive use in industry. The second method that was chosen for the evaluation was the Mathieson-Summers connective-complexity metric DFA method since the original publication discusses its limited amounts and types of required user inputs.

The DFMA software developed by Boothroyd Dewhurst Inc. requires the user to provide specific information about the product as an assembly, the sub-assemblies of the product, and the individual parts of the product. The user specifies information used to apply part count minimization rules and different information used to determine the assembly time of each part. To determine the assembly time of the part, questions regarding the size, assembly orientation, handling difficulties, and insertion difficulties are answered [8].

The Mathieson-Summers connective-complexity metric method predicts assembly time using only the topological connections between parts within assemblies. To do this

each part is evaluated by determining what other parts it is connected to and how they are connected. The specified architecture is then represented in bi-partite graphs and the connective complexity of the architecture is calculated. The complexity information is then used to predict the assembly time of the product [6].

Both the Boothroyd Dewhurst and the Mathieson-Summers connective-complexity metric methods require different amounts and different types of information to be specified by the user to complete the DFA analysis. Three different consumer products are analyzed with each method and the information requirements and results are evaluated. The results from this evaluation and comparison can be used to benchmark the two methods and to identify areas for potential improvement. The results will also determine which method should be automated by comparing the amounts and types of user inputs required by each method.

4.2 Boothroyd and Dewhurst method

The Boothroyd Dewhurst DFA method has two main sections of the analysis: determining the theoretical minimum number of parts and determining assembly times and costs. The theoretical minimum number of parts is used to identify parts that can be eliminated from the assembly. These are often fasteners, fittings, or parts that have multiple instances. The theoretical minimum number of parts is determined first by answering three questions:

1. Does the part move relative to the other parts during the operation of the product?

2. Does the material of the part have to be different from the other parts within the assembly?
3. Does the part have to be separated so that other parts can be assembled or disassembled?

If the answer to any of these questions is yes, then the part is not a candidate for elimination and the minimum number of this part has already been achieved. If the answers to all three questions is "no" then the part could theoretically be eliminated [8]. This is the section of the analysis that suggests design improvements to the user focusing primarily on eliminating or reducing the number of excessive parts. One of the results presented to the user during this section of the analysis is the design efficiency which shows the user how efficient the product is with respect to design for assembly. This design efficiency is determined by comparing the number of parts included in the original design and the theoretical minimum number of parts. This gives the designer one way of documenting the improvements that a product undergoes from pre to post DFA analysis.

The second part of the Boothroyd Dewhurst design for assembly analysis focuses on estimating an assembly time and assembly cost. This is achieved by determining: the size, orientation/symmetry, the handling difficulties, and the insertion difficulties of the part. Each area requires the designer to choose from several options to determine the correct assembly time of the part. The estimated assembly time can be used to compare the assembly time of a suggested redesign to the current design.

The original table based design for assembly method is implemented through a software package that guides designers through the analysis [8]. The software makes the

analysis less demanding by eliminating the need for the user to manually collect and perform calculations. The software has been effectively used to analyze products for assembly improvements as well as estimating assembly times [8].

4.3 DFA Connectivity Complexity Metrics Method

The connective-complexity metrics method calculates the complexity of the part connections within an assembly, mapping the results to previously predicted assembly times based on the Boothroyd Dewhurst DFA tables [6]. Thus, the Mathieson-Summers connective-complexity tool is based on the same empirical data on which the Boothroyd Dewhurst method is based. The key difference is not the source of historical trends, but the usability of the method from the perspective of the engineer that is running the design for assembly analysis.

Complexity metrics can be used to create surrogate models of engineering design representations that capture knowledge not explicitly encoded in the models [6,45,46]. These graphs are used to track similarities so that relationships or trends between properties can be developed [47,48]. The connective-complexity tool is used to map structural graph properties of the assembly architectures to established assembly times. A historical regression model is then created to predict future assembly times on different architectures. It should be noted that use of the historical regression model will be limited by the types of products used for the regression training. Using this method on products whose connective-complexity does not fall within the regression training set may yield varying results since a product's complexity is partially determined by its type.

The previously established assembly times that were used for this model are derived from DFA analysis on ten products using Boothroyd Dewhurst's DFA manual tables [6].

The system architecture used to identify a trend between it and assembly time is developed by identifying connections between system elements and representing them in a bi-partite graph. The bi-partite graph is defined by two independent sets, the elements (components or parts) within the system and the relationships (connections or contact) between the elements. This graph is then used to determine three system properties that were found to be predictors for assembly time: path length, entity count, and path length density [6,45]. A function of these three measures is used to create the surrogate connective-complexity model for assembly time. The results were within 20% of the original assembly times predicted by the Boothroyd Dewhurst tables, which is considered acceptable for use in early stages of engineering design if the cost of estimation is reduced. More information on the development of this method can be found in [6].

To use the Mathieson-Summers connective-complexity method the first step is to build the assembly bi-partite graph. Every part in the assembly is captured, even if the parts are repeated within the assembly. The type of connection between each part set is defined using one of four general types of connections: surface contact, fasteners, snap/press/interference fits, and other connections. For example, a fastening relationship is defined when a part is used to hold/secure other parts (a nut and bolt used to hold two plates together). Details and examples of the other types of contacts or connections can be found in [6].

4.4 Evaluation of Methods

To evaluate the two different DFA methods a full design for assembly analysis of three consumer products is done. A Black & Decker One Touch Chopper, a Black and Decker cordless drill, and a RIVAL can opener were chosen for the analysis because they are all similar in product type. These three products are commercially available, have part counts less than fifty, are low cost, and are mature products, Figure 4.1.

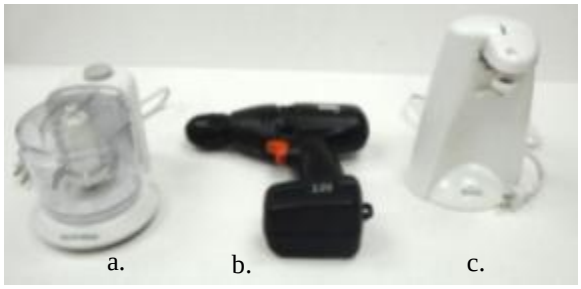


Figure 4.1: (a) One Touch Copper, (b) Black & Decker Cordless Drill, (c) RIVAL Can Opener

These products were disassembled and the DFA analysis was conducted during the reassembly. It should be noted that the analysis done in this exercise is for reverse engineering instead of forward design. The conclusions on effectiveness should be tempered when considering the use of the DFA methods to assist designers in generative forward design problem scenarios. As the analysis was being conducted the following information was recorded to evaluate each method:

- The approximate time required to complete the analysis
- The predicted assembly times for each product
- The amounts and types of information required by the user to complete the analysis
- The method's repeatability/subjectivity

- The method's features for redesign support

Since the time to complete the analysis is approximate it has a general scale that determines a designer's level of satisfaction with the amount of time required to complete the analysis. A high level of satisfaction would have an analysis time in measured in minutes because it would give the user quick results, a medium level of satisfaction in hours, and a low level of satisfaction in days. The comparison between the predicted assembly times is a relative one since the connective-complexity DFA times are based on a regression analysis using assembly times from the Boothroyd Dewhurst original manual tables. This method has been extensively used in industry, so the assembly times it predicts are assumed to be close to the true values and are used as the baseline datum. The different amounts and types of information required will focus on identifying the total number of possible questions per part and whether these questions are subjective or objective. The repeatability of each method is then determined by the percentage of subjective questions to the total questions required. Finally, the features that each method provides to support redesigns to improve assembly are identified. The evaluation criterion results for each method are discussed in their individual sections and they are summarized again in the comparison section.

4.4.1 Evaluation of Boothroyd & Dewhurst Software

Conducting the DFA analysis using Boothroyd Dewhurst DFMA software requires the user to develop the product structure of a desired assembly by answering a series of questions. The software uses this information, a mix of objective and subjective

inputs, to automatically estimate the assembly time for the specified product structure. The typical DFMA graphical user interface (GUI) for a subassembly of the drill is shown in Figure 4.2. The DFA analysis is performed with Boothroyd Dewhurst Inc.'s DFMA software version 9.4.

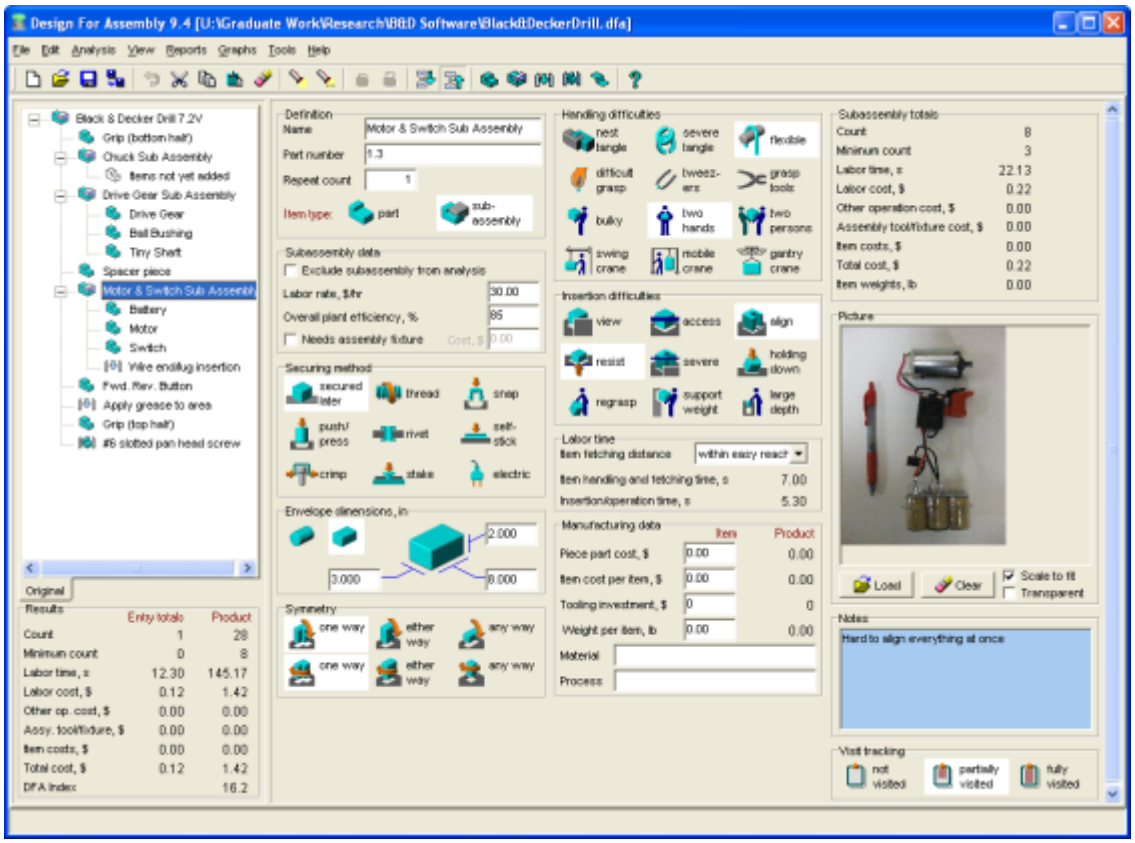


Figure 4.2: DFMA Software Graphical User Interface

The information input by the user as answers to DFA questions include a broad spectrum of data related to symmetry, minimum part criteria, handling difficulties, operation characteristics, operations (e.g. apply grease or not, soldering, and adhesive operations), labor rate, and envelope size. To build the product structure in the software the user needs to: have a thorough knowledge about the product, operations required

during assembly, and have sufficient expertise to use the software. If the user is new to the software, the user manual and built in help file can be used for navigation and clarification. This help file is useful for obtaining clarifications on many of the DFA questions but it does leave some ambiguous instances where the user has to make a decision. For example, the four bushings from the Black and Decker chopper assembly which are inserted into the product's base structure are semi flexible parts. According to the DFMA software help file, these parts can be "flexible" because they deform when pressed, but the help file does not tell the user how much force should be applied to see if it deforms. Another issue was that the bushing's flexibility offered no difficulty for assembly which was a mild press fit; therefore it may or may not be considered rigid.

Conducting the DFA analysis using the DFMA software requires many information inputs from the user. To conduct the analysis on one part using the software eight different areas are evaluated by the user. The user determines if these areas are applicable to the part, specifically the handling and insertion difficulties. The eight areas, the number of questions per area, the number of subjective questions from each area, and the percentage of subjectivity in each area are found in Table 4.1.

Table 4.1: DFMA Software Required User Inputs

	Inputs required from the user	Total # of Questions	# Subjective Questions	% Subjective
1	Product definition	2	0	0.00
2	Securing method	9	1	11
3	Minimum part criteria	7	3	43
4	Envelope dimensions	3	0	0.00
5	Insertion & Orientation Symmetry	6	0	0.00
6	Handling difficulties	12	6	50.00
7	Insertion difficulties	9	6	67
8	Fetching distance	1	0	0.00
	Total	49	16	33

During the assembly analysis, the user answers 49 or more questions to complete the analysis for one part. The cognitive workload on answering these questions is reduced through the software interface and the use of icons and keywords. This allows the user to quickly skim the questions and determine which ones apply to the part being analyzed. This is the number of possible questions that the user has to evaluate per part, not per assembly so the amount of information required by the user grows quickly with the complexity of the product.

Answering these questions can be tedious and time consuming while still yielding inconsistent results because sixteen of the forty nine queries are based on subjective information or the designer's opinion. This means that one third (33%) of the total analysis is based on subjective information. Different designers, when answering the subjective questions, may answer in different ways, resulting in different time estimates, thereby reducing the repeatability and confidence of the method.

4.4.2 DFMA software subjective information

This section focuses on identifying the subjective information required by the user to conduct the DFA analysis using the DFMA software. As each area of subjective information is identified examples of this information are given.

4.4.2.1 Handling difficulties

When determining the handling difficulties, the designer is asked to assign "penalties". This subjectivity is mitigated through the use of example parts for different scenarios, as presented through the software. This is limited to a small set of general, non-specific examples. An example of the subjectivity of the handling difficulties can be seen in the drive gear sub assembly shown in Figure 4.3. The handling difficulties for this sub assembly were specified as "flexible" and "two hands." This sub assembly has several small parts and once they are assembled they have to be held together using two hands. The other handling difficulties of the sub assembly could be "difficult to grasp" because the parts in the assembly are small. Alternatively, the sub assembly could be considered "flexible" because the sub assembly is not fully constrained. The user then has to choose which one is more appropriate and "flexible" was eventually chosen.

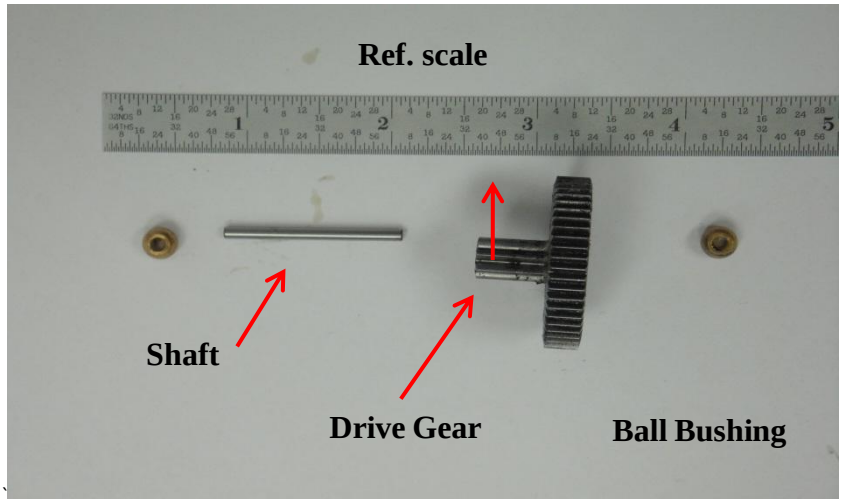


Figure 4.3: Drive gear sub assembly

An example of the subjective handling difficulties tangling, severe tangling, and flexible can be found in the switch pin sub assembly of the Rival Can Opener shown in Figure 4.4. The handling difficulties chosen for this sub assembly were "severe tangle" and "flexible". One of these parts is a spring which makes handling difficult due to tangling. If the user has to remove one spring from a box of springs then it may require them to use two hands to separate the springs giving it the "tangling" penalty. In some cases designers may not consider tangling as a handling difficulty if it is easy for them to hold the spring or remove the spring from a box. The presence of the spring also allows the sub assembly parts to move relative to one another making it "flexible."

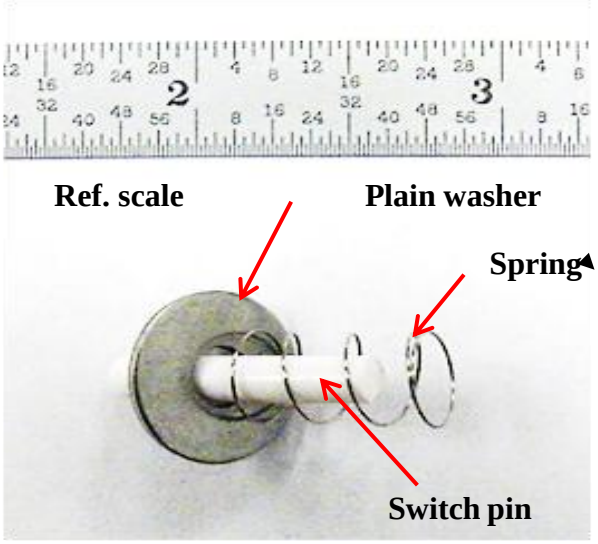


Figure 4.4: Switch pin sub assembly

The assessment of the sub assembly being flexible is subjective because flexibility cannot be measured. It is left up to the user's judgment to decide if the movement of the assembly justifies a penalty of "flexible" or not. Some users may neglect relative motion of the parts since it is a relatively small amount of movement.

Designers experiencing easy assembly and little assembly time may not consider the selection of certain handling difficulties while other designers experiencing difficulties may consider multiple handling difficulties. These types of decisions depend on their perception of the handling difficulties that they experienced during assembly of the product.

4.4.2.2 Insertion difficulties

Another aspect of the DFMA software that can be subjective is determining the insertion difficulties of parts and assemblies. The subjectivity of the insertion difficulties

comes from determining when and to what extent these difficulties apply. If the answer is not clear the user does not decide what insertion difficulty is correct but instead which one they think is more appropriate.

An example of the subjectivity of choosing insertion difficulties is found in the drill's motor and switch sub assembly shown in Figure 4.5. This sub assembly was given insertion difficulties of "align" and "resist." The alignment difficulties came from trying to locate several parts at once that were flexible connected to each other by wires. At one end, the battery pack has to be located and at the other end the motor has to be located. These alignment issues make selecting "align" as an insertion difficulty less subjective since they are easily identified. One issue with these alignment issues is that they can cause insertion resistance if every part is not exactly aligned. This resistance becomes subjective because it may only be present one out of five times meaning that one designer may include it in the analysis and another may not.

Figure 4.5: Motor and switch sub assembly

An example of subjective insertion difficulties “access” and “resistance” can be found where the switch pin sub assembly from Figure 4.4 is inserted into the housing shown in Figure 4.6. This “access” difficulty is present because the designer has to hold the spring down, and then insert the assembly at an angle so it goes through a hole in the housing. The “resist” difficulty comes from the designer having to push the spring against the housing before the pin can be pushed into place. The subjectivity of these difficulties in this example comes from the ease at which the designer can insert the assembly. A designer with small fingers experienced little insertion difficulties where a designer with larger fingers experienced significant insertion difficulties. These two different points of views will result in different insertion difficulties being specified in the analysis.

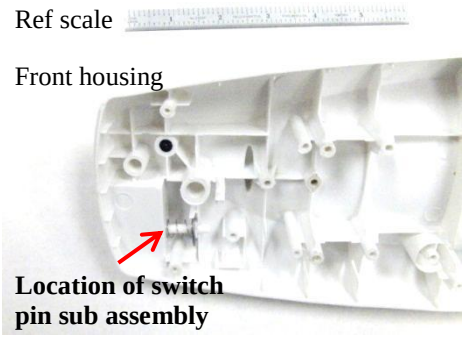


Figure 4.6: Switch pin sub assembly inserted into housing

During the assembly of the can opener top assembly shown in Figure 4.7 an insertion difficulty of resist was specified. While tightening the screw, a spring on the other side caused insertion resistance. Designers may or may not specify resistance depending on their perception of the difficulty. The switch pin sub assembly in Figure

4.7 is flexible, inserting it from the top and tightening the screw through the metal-plastic sub assembly. This is difficult if the bottom part is not aligned with the top sub assembly. Since the top sub assembly is flexible it is difficult to keep it in the same position because it needs continuous pressing from above. The small screw size and the varying resistance experienced also add to the insertion difficulties experienced by the designer. If one designer is able to tighten the screw easily they will not face any alignment or resistance issues whereas, for those who experience difficulties, they will consider selecting these as insertion penalties.

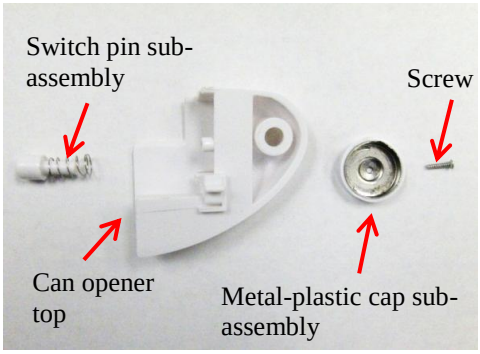


Figure 4.7: Can opener top assembly

4.4.2.3 Wiring Harness Operation

Another type of subjective information included in the DFMA software comes from the wire harness specifications. The DFMA software includes methods that can be used to conduct DFA on wires, wire connectors, and other aspects involved with wire harness assemblies. This information allows the assembly labor time to be accurately estimated but it also adds another area of subjective information. Several different features have been included in the software to accommodate assembly issues regarding

wiring. The two main areas are specifying electrical securing methods or specifying an assortment of wiring operations that can be chosen. The securing method determines that the part is going to be secured immediately by that method. It gives the designer options of choosing from thirteen specific electrical operation characteristics like a standard electrical plug to secure the part. The wiring operations list lets the designer choose operations like wire preparation, wire assembly, wire installation, and more that can be applied to parts and assemblies.

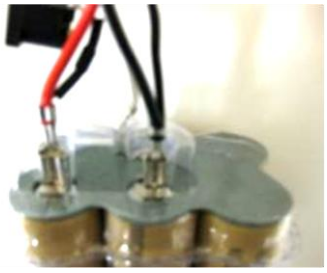


Figure 4.8: Quick wire connections from switch to battery pack within motor & switch sub assembly

An example of subjective wire information can be found in the drill's motor and switch sub-assembly and the wire connections within it shown in Figure 4.8. The issue with the wiring assembly information comes from the fact that it is hard to determine if the switch's securing method should be secured later or if it should be documented as electrical securing. If it is secured later then wiring operations could be specified separately to connect it to the battery pack and the motor. If it is secured immediately using the electrical securing method, operation characteristics can be selected to account for the assembly operations. Since the switch has five quick wire connections the user has to be delicate in how the operations are specified because if the chosen penalty is

incorrect the error will compound. One of the wiring harness operations that can be chosen under wire assembly is "wire end/lug insertion." This lets the designer choose from three connector pin rows, specify the repeat count, specify lug orientation requirement, and ease of insertion. Determining if the connector is easy or difficult to insert is subjective information that affects the assembly time and must be determined by the designer.

4.4.2.4 Minimum part criterion

The minimum part criterion does not directly affect the predicted assembly time but it is the primary method used to identify design improvements within the product. The information required to identify the minimum part criterion is subjective and requires the designer to answer multiple questions to determine it. The subjectivity of this information will not affect the overall initial assembly time but it will affect the re-design's predicted assembly time. A more important issue that occurs since this information is subjective is that the designer has to determine the most appropriate answer for it to be effective. This will increases the amount of time the DFA analysis takes to conduct.

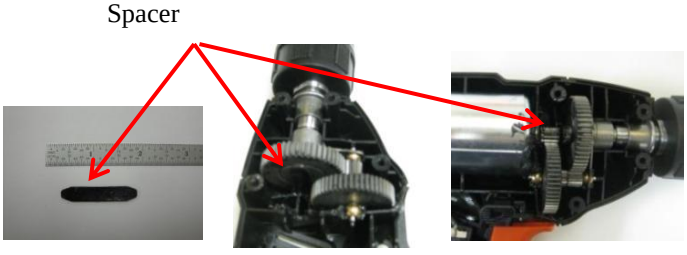


Figure 4.9: Spacer as a minimum part criterion

An example of minimum part criterion subjectivity is shown in the assembly analysis on the spacer piece shown in Figure 4.9. This part is located between the motor and the gear on the chuck assembly. The piece appears to be a spacer to prevent the gear on the motor from touching the gear on the chuck so the minimum part criteria could be based on "material" where the part must theoretically separate from the others. Another way of looking at this part is that it is just a spacer not serving a special task and that "other" could be chosen for its minimum part criterion which would make it a candidate for elimination. If the person conducting the assembly analysis is not the designer they will have to find the designer to determine if that part could be eliminated or not and why. This is the case with many of the parts that the minimum part criterion may identify as possible candidates for elimination.

4.4.3 DFMA evaluation criterion summary

The results from the DFMA evaluation based on the five criteria are summarized in Table 4.2. The DFMA requires extensive amounts and types of user imputed information which slows down the analysis time and reduces its repeatability, consistency, and accuracy. Even though the extensive amounts of information required inhibit the analysis process as seen by the designer, it also provides critical information about the product that would otherwise be overlooked. This information provides the user with validated assembly times and eleven areas to focus redesign efforts both of which are critical for a DFA method to be effective.

Table 4.2: DFA evaluation criterion summary

Evaluation Criteria	Evaluation Results	Justification
Satisfaction with analysis time	Medium	Not minutes (High) but not days (Low)
Predicted assembly times	Baseline	Previously validated results
Amounts/types of information	8 types, 49 questions, 16 subjective	Requires extensive amounts & types of user inputs
Repeatability/ subjectivity	33% Subjective	Reduces repeatability and accuracy
# of Features for redesigns	11	Identifies eleven types of issues to focus on

4.5 Evaluation of Connectivity Complexity Metric DFA Method

Two types of information are required from the user to complete the analysis using the connective-complexity DFA method. The user must evaluate each part based on which parts it is connected to and the type of connections between those parts. These two types of inputs are listed in Table 4.3 along with the number of questions that have to be answered per type and how many of those questions are subjective.

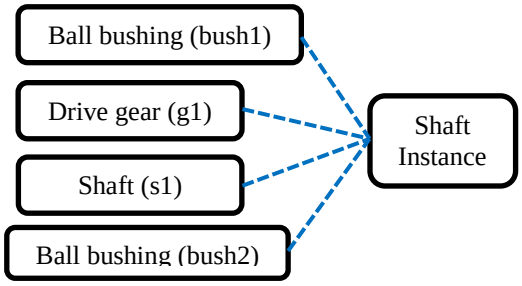
Table 4.3: Connectivity required user inputs

	Inputs required from the user	Total # of Questions	# Subjective Questions	% Subjective
1	What parts is it connected to	1	0	0
2	What type of connection	4	0	0
	Total	5	0	0

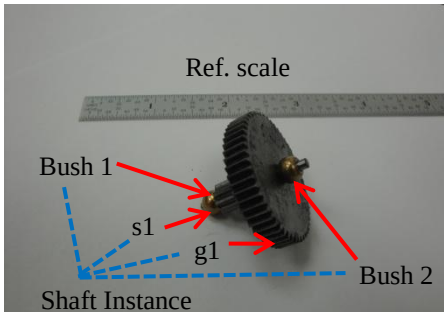
The number of basic questions required by this method is five and none of them are subjective, (Table 4.3). Determining which parts a part is connected to can be determined quickly and objectively. All the user has to ask themselves is “Does the part

touch the part next to it within the assembly?" The answer to this question is "yes" or "no" which minimizes user miss-interpretation. Once a connection between parts has been identified the user has to specify the type of connection. To do this the user determines if the connection is: a fastening instance, a snap/interference/press fit instance, a shaft instance, a surface instance, or another type of connection instance. In most cases determining the connection instance is obvious since they are separated into distinct types of connections. For example shafts are easy to identify so if a part connects to it then it is part of the shaft instance. If the part is used to fasten or secure another part then a fastening instance is chosen as the connection. In some cases the user may not be able to distinguish which type of connection instance is most appropriate but as long as the user chooses a similar connection type that will have the same path length the results will not be affected.

This method requires the user to identify that a connection instance between parts exists and does not typically distinguish between the types of connection instances. This is because the number of parts connected by that one instance increases the path length in the bi-partite graph. Two parts connected by a snap fit instance and two parts connected by a surface instance will have the same path length so there is no distinction between these instances within the algorithm. In the case of a shaft instance or a bolting instance where more than two parts are connected through one instance there is distinction between these types but only from instances with different path lengths. An example of a shaft instance and its bi-partite graph can be seen in Figure 4.10.



(a)Bi-partite graph for a shaft instance within the drill



(b) shaft and the parts connected within the sub-assembly

Figure 4.10: Shaft Connectedness

The shaft instance in Figure 4.10 is from the drive gear sub assembly of the drill. This sub assembly connects the gear on the motor to the gear that drives the chuck assembly. Looking at the parts of the sub assembly it is easy for the user to identify that a shaft is the common part that all of the other parts are connected to. This signifies that a shaft instance is the main connection unifying all of these parts. All of the connections that exist for the parts of this sub assembly are shown in Table 4.4.

Table 4.4: Drive gear sub assembly connections

Parts				Instance	Description
bush1	g1	s1	bush2	Shaft Instance	Drive gear assembly shaft connections
bush1		h1		Surface Instance	Bushing 1 to Bottom Grip
bush1		h2		Surface Instance	Bushing 1 to Top Grip
bush2		h1		Surface Instance	Bushing 2 to Bottom Grip
bush2		h2		Surface Instance	Bushing 2 to Top Grip
g1		m1		Surface Instance	Drive gear to motor gear
g1		cs		Surface Instance	Drive gear to chuck gear
s1		h1		Surface Instance	Shaft to bottom grip
s1		h2		Surface Instance	Shaft to top grip

The shaft instance in Figure 4.10 is shown in the first row of Table 4.4. The other rows show the other connections that exist between the parts of this sub assembly. The first four columns, highlighted in red, of this table are the only items that are put into the bi-partite excel table that is processed by the Matlab algorithm. The algorithm does not need column five or column six to determine the assembly time. These extra two columns shown in Table 4.4 are included for documentation purposes and user readability. The fifth column shows the instance between the parts and the sixth column describes which parts are being connected by that instance.

The results from the connective-complexity DFA method evaluation based on the five criteria are summarized in Table 4.5. The connective-complexity method requires moderate amounts of time to complete the analysis and only requires the user to provide

input based on a few different types of objective questions. This should make the analysis repeatable and consistent between users. The analysis would not be repeatable or consistent if the designer overlooked a connection within the product, or specified the wrong type of connection which would create a different bi-partite graph resulting in a different assembly time. The predicted assembly times that the method provides have not been fully validated so they cannot be accepted as correct. This method currently does not provide the user with features to aid in redesigning the part to improve assembly.

Table 4.5: Connectivity evaluation criterion summary

Evaluation Criteria	Evaluation Results	Justification
Satisfaction with analysis time	Medium	Not minutes (High Satisfaction) but not days (Low satisfaction)
Predicted assembly times	Not accurate	Validation needed
Amounts/types of information	5 types, 0 subjective	Requires few types of objective user inputs
Repeatability/ subjectivity	0% Subjective	Repeatable, and consistent
# of Features for redesigns	0	Currently provides no redesign features

4.6 Comparison of Methods

The results from the evaluations of each DFA method based on the specified criteria are discussed and compared in the following sub sections. These results from these criteria ultimately determine how effective each method is and which one should be selected for automation.

4.6.1 Comparison of approximate time to use each method

The approximate time to conduct the DFA analysis using each method was evaluated to determine which method could be implemented the fastest. Note that an approximate time was used since the exact time required to conduct each analysis was not recorded due to frequent interruptions. Care should be taken during future studies to ensure accurate analysis times are recorded. Without the exact analysis time, only an approximate time to conduct the analysis could be determined and used for comparison. After the analyses were conducted on each product using both methods, it was determined that the connectivity method could be implemented approximately 25% faster than the DFMA software. This is based off of approximate times since the analyses did not always take place in one sitting. Both methods required between 1.5 to 2.5 hours to complete the analysis depending on the complexity of the products. A high level of satisfaction would have an analysis time in minutes because it would give the user quick results, a medium level in hours, and a low level in days. Both methods had analysis times within hours so a medium level of satisfaction was chosen (Table 4.6).

Table 4.6: Satisfaction with approximate analysis time

Evaluation Criteria	DFMA Software	Connective-Complexity Method
Satisfaction with analysis time	Medium	Medium

Reducing the analysis time for both methods will make them more appealing to designers because they will be faster and easier to implement.

4.6.2 Comparison of predicted assembly times

The two DFA methods were compared based on their predicted assembly times to determine how close the connective-complexity method's times were to the DFMA times. This data was gathered from three designers (D1, D2, and D3) who were trained on both methods before conducting the assembly analyses on the three products. This comparison includes the designer that conducted the analysis, their respective predicted assembly times per product, and the differences between the times (Table 4.7). The DFMA software has been in use since the early 1980's [32] so its predicted assembly times are considered to be accurate and therefore they are the baseline for this comparison.

Table 4.7: DFA comparisons of method effectiveness

Measures of Effectiveness	Designer	DFMA Software Assembly Time	Connectivity Assembly Time	Time Difference	% Difference between methods
B&D Drill	D1 D2	2.42 2.16	1.22 -	1.20	50 44
B&D Drill with chuck assembly	D1	2.89	1.69	1.21	42
RIVAL Can Opener	D2	5.49	4.77	0.72	13
B&D Chopper	D1 D2 D3	6.40 5.52 6.36	4.18 4.61 -	2.21 1.34 2.18	35 24 34
**All times are in minutes*					

For all of the DFA analyses on the different products the connective-complexity DFA times were substantially lower than the DFMA predicted times. These times varied

considerably where the smallest difference was 13% lower and the largest difference was 50% lower. The average of the % differences of the six analyses was 35% lower than the DFMA times. This is substantially higher than the +/- 16% difference originally found in the complexity connectivity DFA paper [6]. These significant differences were unexpected, so some possible causes are investigated.

Since the drill has the largest percent difference of 50%, it is the primary area of investigation. The original assembly analysis of the drill assumed the chuck assembly to be one pre-assembled part so it was treated as a part during the analysis. This assumption was re-evaluated and both analyses were preformed again separating the chuck assembly into individual parts to be assembled as a sub assembly. This resulted in an even twenty eight second predicted assembly time increase with both methods reducing the percent difference by 8%. This shows that the two methods predict similar assembly times for certain parts of the drill but there are still significant differences between the two methods.

Another possible source of the discrepancies between the predicted assembly times could be because the connective-complexity metric is based off a regression model that uses assembly times determined by the original Boothroyd Dewhurst DFA tables. The DFMA software has been improved over the years incorporating more features, like wiring harness analysis features, to improve the DFA method which were not included in the original tables. Future research could be to identify the cause of the discrepancies found in this part of the study.

4.6.3 Comparing amounts of required user information

Both methods require the user to disassemble a product, and then reassemble it to conduct the DFA analysis. Both methods also require the user to go through a set of procedures or questions to conduct the DFA analysis but they require different types and amounts of information. The specifics about the types and amounts of information that each method requires have been discussed in the previous sections. The total number of questions and the total number of subjective questions from each method are summarized in Table 4.8.

Table 4.8: DFA methods required information summary

	Method	Total # of Questions	# Subjective Questions	% Subjective
1	DFMA Software	49	16	33
2	Connectivity DFA method	5	0	0.00

The DFMA software requires the user to answer a total of forty nine questions per part where sixteen of them are subjective. The extensive amounts of information required by the DFMA software does slow down the analysis time and increase the overall subjectivity but they allow for the product to be analyzed in great detail. The connectivity DFA method requires the user to evaluate a total of five questions per part, none of which are subjective. The limited amounts of objective information are advantageous with regards to automation and conducting the analysis but it does not gather as much detail about the product possibly limiting its overall applications. Since the connective-complexity method requires only objective information it should be

repeatable between users. Also, since the connective-complexity method only requires a few types of objective information it should theoretically be completely automatable.

4.6.4 Comparing repeatability of methods

The repeatability of each method is measured by comparing the output predicted assembly times when the same analysis is conducted by different designers. The analyses of the drill and chopper were conducted by two and three designers respectively using the DFMA software. The designers along with respective assembly times for each product can be seen in Table 4.7. The analysis of the chopper was conducted by two designers using the connectivity method, the designers and the respective assembly times can also be seen in Table 4.7. The maximum percent internal differences of the method's assembly time on the respective product are shown in Table 4.9.

Table 4.9: Repeatability of methods

Measures of Repeatability	DFMA Internal % Difference	Connectivity Internal % Difference
B&D Drill	11	-
B&D Chopper	14	9

Based on the comparison of the amounts and types of information required by the user to complete each analysis, it was expected that the connectivity would have no internal difference. The connective-complexity method and DFMA software had internal differences of 9% and 14% respectively for the chopper analyses. This shows that the connectivity method has a lower percent difference but it doesn't appear to be significant. One possible reason that the connective-complexity method showed repeatability issues could be due to the lack of formalized rules.

4.6.5 Comparison of methods redesign features

The two methods were compared based on their redesign features to aid the designer in improving their assembly. This is important because for a DFA method to be effective they need to provide the designer with suggestions on how to redesign their product to improve its assembly characteristics [21]. The DFMA software has eleven redesign features and the connective-complexity DFA method currently provides the user with no redesign features, Table 4.10.

Table 4.10: Comparison of redesign features

Evaluation Criteria	DFMA Software	Connective-Complexity Method
Features for redesigns	11	0

The DFMA software is effective at providing eleven different areas to focus designers redesign efforts. The software identifies the area, the parts that are relative to that area, and the amount of assembly time or cost that could be improved by focusing their efforts accordingly. This feature does not always help the designer redesign the part but it will identify and prioritize areas for the designer to focus on to improve assembly. Currently the connective-complexity DFA method provides no aids to help the designer redesign the product to improve assembly.

4.7 Summary of Evaluation Results

This chapter evaluated Boothroyd Dewhurst's DFMA software and a connective-complexity DFA method based on five criteria. The results from the evaluations of the

two methods are summarized in Table 4.11. These results can be used to bench mark DFA methods and they can be used to identify a method that should be selected for automation. The two criteria that relate specifically to selecting a method for automation are the amounts and types of information required to complete the analysis and the subjectivity of these inputs. Since the connective-complexity method satisfies these two criteria more effectively it is identified for automation. The rest of the evaluation comparison results are analyzed in the following paragraphs.

Table 4.11: Comparison summary of two DFA methods

Evaluation Criteria	DFMA Results	Connectivity DFA results
Approximate analysis time	Medium	Medium
Predicted assembly times	Baseline	Not accurate
Amounts/types of information	8 types, 49 questions, 16 subjective	5 types, 0 subjective
Repeatability/ subjectivity	33% Subjective	0% Subjective
# of Features for redesigns	11	0

The DFMA software satisfies all five criteria but does not perform well with the required amounts and types of information required by the user and its repeatability. The connective-complexity method does not provide the user with accurate results and does not provide the user with features to aid in the redesign to improve assembly.

The amount and type of information required by the user to conduct the DFA analysis using the connectivity method was substantially less in quantity and in subjectivity compared to that of the DFMA software. This suggests that the connective-complexity method would be more repeatable and consistent than the DFMA software. Even if this is the case, until the connective-complexity method can provide the user with

accurate results and provide the user with suggestions for redesign it will not be a truly effective design for assembly method.

The results from this evaluation and comparison can be used to identify weaknesses in existing DFA methods. This will allow researchers to focus their efforts so that the method in question can reach its full potential. If this study is going to be repeated or used to compare other DFA methods some possible improvements could be made. This research did not implement a full user study to obtain the results which limits the effectiveness of the study. The results from this study indicate that differences between these two DFA methods does exist and that a full user study would effectively document all benefits and drawbacks of each method including the time to conduct the analysis.

4.8 Identified DFA Method for Automation

As previously stated, to automate a DFA method one must be identified that does not have fundamental flaws like requiring many different types of subjective information which makes automation difficult. To identify this method, research question one was addressed:

RQ1:	Which existing assembly time prediction method should be selected for automation based on the amounts and types of information it requires?

Based on the evaluation of the Boothroyd Dewhurst DFMA software and the Connectivity Complexity method presented in the previous sections, it was determined that the hypothesis to the first research question was correct.

RQ1 Hypothesis:	An existing assembly time prediction method that requires limited amounts of objective user inputs can be identified for automation.

The results of the DFA evaluation identify that the connective-connectivity DFA method only requires five types of information, all of which are objective. The objective information required to complete the analysis using this method are the physical connections between parts. The identification of the physical connections between parts should theoretically be extractable from solid modeling software allowing the method to be automated. The DFA evaluation presented in this chapter successfully answers the first research question by comparing two methods and identifying the Connectivity Complexity method to be automatable. To determine if the connective-complexity DFA method can be automated the second research question is addressed in the next chapter.

CHAPTER 5. CAN THE IDENTIFIED DFA METHOD BE AUTOMATED?

Based on the DFA evaluation presented in Chapter 4. , the connective-complexity DFA method was identified for possible automation since it only requires five types of information, all of which are objective. The rest of this chapter is focused on answering the second research question; can the identified assembly time prediction method (the connective-complexity method) be automated so that it predicts an assembly time using information extracted from 3D solid modeling software? The first step is to determine if the information required by the method is stored in solid modeling software explicitly or implicitly, Section 5.1 and 5.2. The second step is to determine how to extract and process the explicit or implicit information to complete the analysis, Section 5.3. The third step is to use the extracted information to predict the assembly time, Section 5.4 and 5.5, and the final step is to evaluate the automation attempt and its effectiveness, Section 5.6 and 5.7.

5.1 Automation of Connective-Complexity DFA Method

Recent work on complexity based assembly time prediction methods has shown that assembly times can be predicted using complexity metrics and different types of relationship found within products [6,5]. The original work on the connective-complexity method presented in Chapter 4. used a regression analysis to relate a products physical connection complexity to assembly times. The advantage of this method over existing DFA methods is that the physical connections between parts in an assembly can be identified objectively. The initial results predicted assembly times

within +/- 15% of the training times used. This proved that a products connection complexity could be used to determine a products assembly time [6].

To improve the accuracy of the Connectivity Complexity method, continuation of the original work replaced the linear regression training with Artificial Neural Network (ANN) training and applied it to an Automotive OEM assembly instead of consumer products [5]. Figure 5.1 shows a flow chart of the continued development of the Connectivity Complexity DFA method. The original work, shown by the top row in Figure 5.1, acted as a proof of concept to show that physical connections between parts could be used to determine a products assembly time. The continuation of the work, shown in the middle row of Figure 5.1, implemented the ANN training to improve the accuracy of the predicted assembly times [5]. The issue with the original regression based connectivity method and the neural network based connectivity method is that the inputs to complete the analysis, which are the product connection graphs, have to be manually generated which is time consuming and not completely repeatable.

The work presented in this chapter relates to the third version of the connective-complexity method, which focuses on developing an objective and automated assembly time estimation tool.

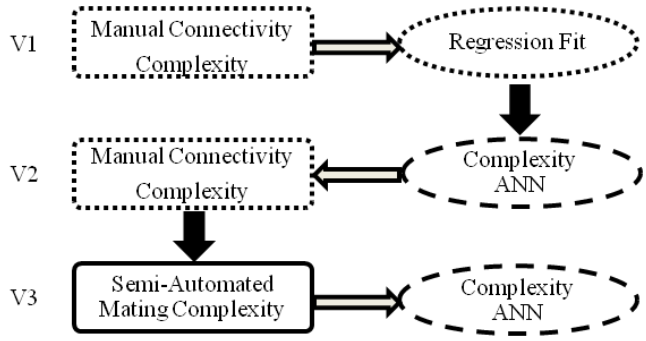


Figure 5.1: Connectivity Complexity DFA development flow chart

The focus of this chapter is shown in the third step, V3, of Figure 5.1 where the ability to automate this method is improved. During the early development of the Connectivity Complexity method it became apparent that part connections within a product can be identified early in the design process. The inter part connections that this method requires could be extracted from sketches and 3D solid models, which are generated as early as the conceptual design phase giving it the potential to be applied throughout the design process [49]. Extracting the connections from 3D assembly models would also enable a program to be developed to completely automate this method. The rest of this chapter presents the work towards developing an automated Connectivity Complexity assembly time prediction tool.

5.2 Assembly Model Connection Extraction Tool

To automate the connectivity complexity method, the creation of the connectivity bi-partite tables has to be automated. The steps to do this are: identify the types of information required by the connectivity method, determine if that information is

included in SolidWorks Assembly Models, and extract the information to create the tables.

5.2.1 Information Required for Connectivity Method

The original Connectivity Complexity DFA method used the complexity of the physical connections between parts to determine a given products assembly time. This analysis was completed by identifying what parts a specific part is connected to, creating bi-partite tables to represent those connections within the product, applying a custom algorithm to determine the complexity of the connections, and then applying the complexity metrics to the regression equation to determine the assembly time [6]. This means that to automate the original connectivity complexity method, the physical part connections would have to be extracted from the assembly models.

Three-dimensional assembly models contain virtual parts that are arranged in a specified way to create a final product. If assembly models are created correctly they should form virtual representations of the actual physical product where the virtual connections shown between parts in the CAD software should match those on the physical products. The issue comes from that fact that the virtual connections contained within the assembly model may not be explicitly defined and even if they are they may not represent the variety of connections required by the connectivity complexity DFA method. The use of both implicit and explicit connections is explored.

5.2.2 Use of Implicit Connections for Connectivity Method

To use the connectivity complexity DFA method, the connections between parts in the assembly have to be identified. The types of part connections are: surface contact, fasteners, snap/press/interference fits, and other connections, such as shaft connections, electrical connections, or spring connections [6]. In many cases, these types of connections are implicit in an assembly model and could not be determined without evaluating the parts on a feature level.

Take three separate parts for example, Part A, Part B, and Part C, Figure 5.2. If Part A and Part B have through holes and Part C has a circular cross section of the same size then they may form a shaft connection as shown in Figure 5.2. To determine if these parts do form a shaft connection the parts location would have to be determined and the features compared. If the hole in Part A aligns with the hole in Part B and the surface area of Part C overlaps with the surface area of both holes, then a shaft connection as used by the Connectivity Complexity method would be present.

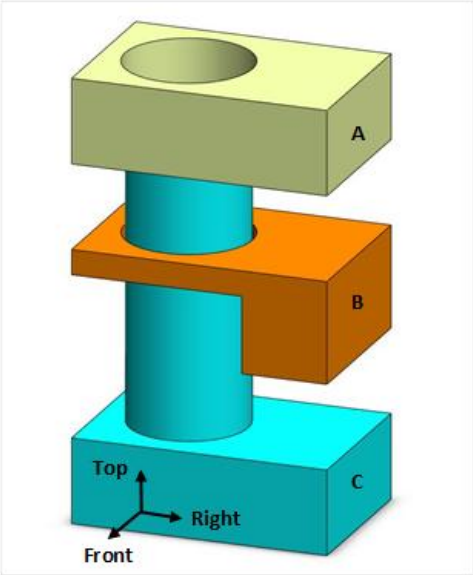


Figure 5.2: Part A, Part B, and Part C which could be mated or constrained in a variety of ways

To identify implicit connections, rules on how to identify the connections required using feature recognition would have to be developed. For instance, on a feature level, a rule could be developed to use the amount of surface area overlap between two parts to identify a surface contact connection. However, even with an effective set of rules to identify the connections it would be computationally expensive to identify the connections. A program would have to iterate through every feature on every part in the assembly and compare it to the features on other parts in the assembly.

Previous work in user defined feature recognition through the use of design exemplars can be used to support this activity. In this work, implicit relationships between geometric entities can be extracted, though this could result in redundant identification and over constrained entity-relation graphs [50,51]. Ultimately, the design

intent may be captured, but other relationships will also be captured, thereby hiding the underlying intent.

5.2.3 Use of Explicit Connections for Mate Based Connectivity Method

The alternative to using implicit connections is to use the explicit connections located in 3D modeling assemblies. In SolidWorks, mates are used to define the relationship between two components within the assembly [52]. These relationships determine the parts location and constrain the parts motion (translation and rotation) within the assembly. An effective designer will apply mates to simulate the actual constraints that the final product will have. Since mates define the location of the parts within an assembly, they can also be used to identify the connections between parts within an assembly.

Mates use different types of relationships to relate one part to another part based on position or orientation. Some common mates found in SolidWorks assembly models can be seen in Table 5.1 [52].

Table 5.1: SolidWorks mate types and descriptions

Mate Type	Description
Coincident	Connects two planar faces
Concentric	Aligns the axes of two circular parts
Distance	Specifies a distance between two planar faces
Parallel	Aligns two planar faces to be parallel
Perpendicular	Makes two planar faces perpendicular
Angle	Specifies an angle between two planar faces
Lock	Fixes the parts location
Advanced	5 advanced mates, ex: limit mates
Mechanical	6 mechanical mates, ex: gear mates

Once applied, these mates become explicit connections between the parts within an assembly. Ideally these explicit connections could be directly matched to the connections required by the Connectivity Complexity Method so that the method could be automated by extracting the mates. This is not the case since a variety of mating configurations could be used to constrain one part. For example, consider the three parts shown in Figure 5.2 described in Section 5.2.2: Part A and Part B which have holes extruded through them and Part C which has a shaft like feature. If Part C is a shaft that connects Parts A and B, then a shaft connection would be present. If Part C is constrained to Parts A and B using concentric mates, then the parts' locations could be analyzed to identify a possible shaft connection. If concentric mates were not used or if Part C was constrained to the assembly by other parts, it would required a different method to identify the shaft connection.

Since interpreting the connections required for the Connectivity Complexity Method from defined mates would be difficult, an alternative type of connection is considered. The mates themselves form a mate connection between the parts within an assembly. This forms a sub research questions for this chapter; can mate connections, as defined in assembly models, be used to predict a products assembly time?

5.2.4 Mate Based Connections

The inter part connections required to complete the original connective complexity method can be extracted from assembly models on an implicit level (feature based) or on an explicit level (mate based). Both of these methods would require new algorithms to relate the implicit or explicit information to the variety of connection types

required by the original method. Since the basic idea of the connectivity method is to relate a complexity vector to an assembly time, the inter part connection complexity vector could be replaced with another type of complexity vector. Since mate connections are defined within assembly models, this research uses the mate connections to determine the complexity vector and then uses artificial neural networks to relate the mate complexity to assembly times. This approach eliminates the need for extra algorithms or rules to relate the information within the assembly models to inter part connections, but the mating variability between designers may pose a new issue.

Assembly time estimation using mates may be effected by the designers' approach in creating the assembly model. The definition of mates may vary between designers based on the best practices followed, mates offered by software, expertise, and the part geometry itself. Variation in the use of different mates arises because parts in the assembly can be constrained using different combinations of available mates, such as using different surfaces for setting up a certain mate. An example of this variation in constraining parts can be seen by referring to Figure 5.2. Table 5.2 shows two different configurations that can be followed to fully constrain the parts in Figure 5.2 and achieve the same outcome.

Table 5.2: Mate configurations for Parts A, B, and C

Parts	Configuration 1	Configuration 2
C and B	C shaft concentric with B hole	C face right aligned with B face right
C and B	C face top coincident with B face bottom	C face top coincident with B face bottom
C and B	C face right parallel with B face right	C face front aligned with B face front
B and A	B hole concentric with A hole	B face right aligned with A face right
B and A	B face top coincident with A face bottom	B face top coincident with A face bottom
B and A	B face right parallel with A face right	B face front aligned with A face front

The two mating configurations in Table 5.2 use different approaches to accomplish the same goal. Configuration 1 takes an approach that captures more of the designer's intent by applying mates to similar features on the receiving part. Configuration 2 uses only planar and face mates which may not capture some of the design intent. These are only two of the possible mating configurations for a simple assembly with only three parts. As the size of the assembly grows, so will the variability of the different mating configurations, which may affect the predicted assembly time.

Based on the previous success of relating complexity vectors to assembly times, it is determined that using the defined mating information within the assembly models is the most direct method of predicting assembly times. This information is already stored in the assembly models and no extra interpretation or computation is required to use this information. It is speculated that as the size and variability of the training set grows, the mating variability will have less of an effect on the predicted assembly time. Before using mate connections to predict assembly times, the variability of different mating

configurations and their effect on the resulting assembly time must be investigated. Before these aspects can be considered a tool must be developed to extract the mates from solid modeling assemblies, this development is presented in the following section.

5.3 Mate Extraction SolidWorks Add-in

This section presents the development of a tool that automatically extracts mates from SolidWorks assembly models. The tool used for extracting mate information from assembly models was developed using SolidWorks 2010 API Software Development Kit (SDK). SW is a commercial three dimensional modeling software package which provides an intuitive Graphical User Interface (GUI). The software offers two options to develop the SolidWorks API application, macros and add-in programming [52]. Macros tend to be an easier way to develop API applications, since they typically depend on the users' actions with the interface. For example macros can developed to create a slot automatically since slots can be created using only GUI controls. If an API application requires information that cannot be extracted from user interface actions, then a separate add-in may be required. This is the case for extracting mate information from SolidWorks assembly models. Both options were considered, but the development of a separate add-in is chosen over the use of a basic macro.

Any programming language that supports Microsoft COM (Component Object Model) can be used to build add-ins in SolidWorks [52]. The C++ programming language is used in this research based on its easy implementation of COM objects [52], the author's proficiency of coding with this language, and for future extensibility. The

rest of this section briefly describes the algorithm developed to extract the mates from assembly models. The pseudo code for this algorithm is shown in Figure 5.3.

```
Get active assembly document
Get features list from feature manager tree
If feature = mate list
        Get Mate list from feature list
                For each mate in Mate list
                Get parts connected by mate
                Add parts to graph
        End
End if
```

Figure 5.3: Pseudo-code for Extracting Mate Information

To obtain the mate information from an assembly file, the program traverses through the types of features in the feature manager tree. A screen shot of the SolidWorks feature manager design tree for the Black & Decker Drill can be seen in Figure 5.4. This figure labels three main section of the feature manager design tree: reference features, parts and sub assemblies, and mates. Within the main assembly, everything in the feature manager design tree can be recognized as an assembly feature. Information stored within the sections of the feature manager design tree may include annotations, co-ordinate planes, part names, part features, and part constraints.

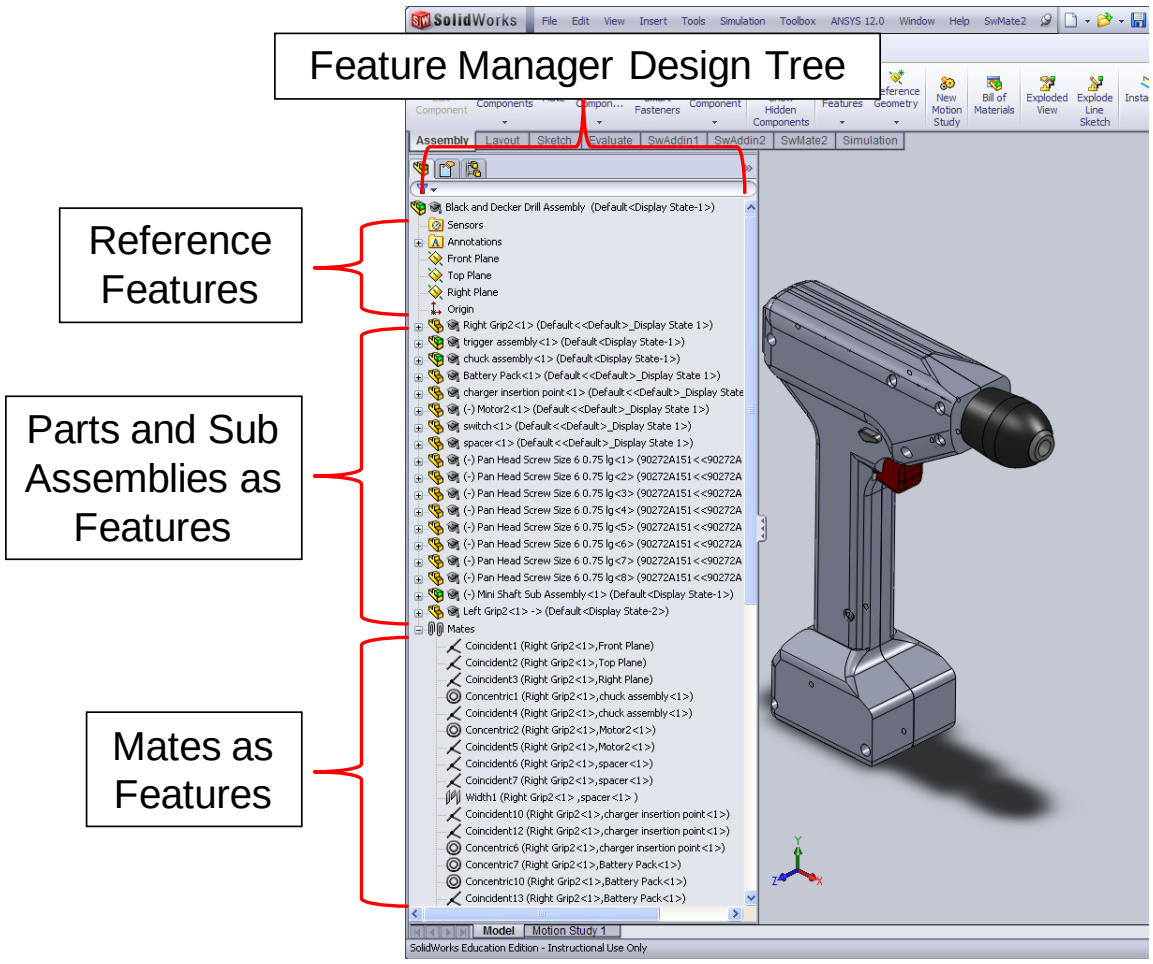


Figure 5.4: SolidWorks feature manager design tree

The program traverses through the feature manager tree until it reaches a container that has the mate information. Each mate consists of the name of the mate and the names of parts that are constrained by that mate. For each mate, the names of both parents (parts) are retrieved, which indicates the connection between the parts. The names of the connected parts are then stored in a bi-partite table which is currently saved as a *.csv file. This process is iterated until all connections between the parts are extracted from the feature manager tree.

Once the bi-partite table containing the mate connections found in the assembly file is generated, the complexity of the table based graph can be calculated using a custom Matlab algorithm. This complexity vector will be used along with Artificial Neural Networks (ANNs) to predict a products assembly time. Figure 5.5 shows a flow diagram of the SW mate extraction add-in, its required inputs, the information processing steps, and the assembly time output.

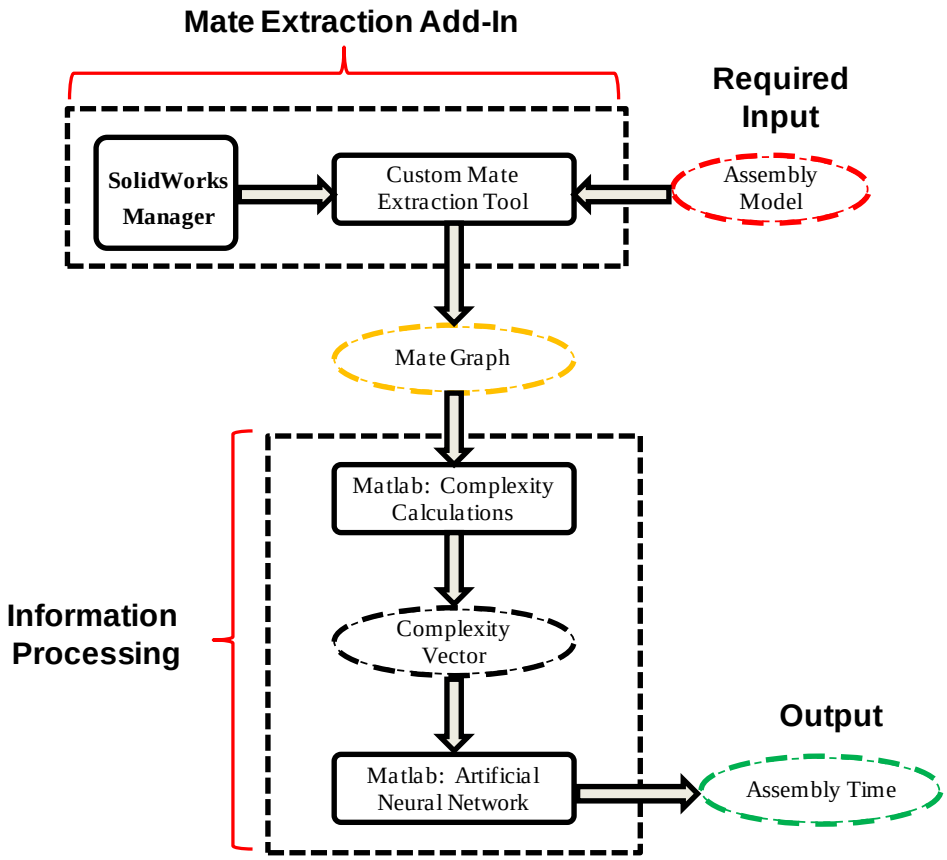


Figure 5.5: SW mate extraction add-in and information processing

The mate extraction add-in as shown in this figure generates a mate graph once it is given an SW assembly time. This mate graph that represents the product inter part

connections is processed externally from the mate extraction add in. The external processing is preformed using Matlab where custom algorithms are used to generate a complexity vector of the mate graph, and use this vector along with previously trained ANNs to predict an assembly time. Before the information processing can be accomplished, the ANNs have to be created and trained which is covered in the next section.

5.4 Creation of ANN Training Set

Before the information extracted from the mate extraction add-in can be used to predict an assembly time, the Artificial Neural Networks (ANNs) must be trained. Training an ANN requires a large set of inputs and respective target values to effectively identify relationships between them. Once an effective set of inputs and targets has been compiled it can be reused in future implementations. This eliminates the training process from the final tool implementation. Since the goal of this work is to identify the relationship between the mate complexity of three-dimensional assembly models and the assembly times, these items become the inputs and targets respectively. This means that a collection of three-dimensional assembly models with known assembly times has to be compiled. The following sub sections detail the process of collecting three-dimensional assembly models, determining their respective assembly times, and using this information to train ANNs.

5.4.1 Collecting Product 3D Assembly Models

To populate an effective ANN training set, a collection of 3D assembly models has to be collected. The original goal was to download 3D assembly models of consumer household products from an online 3D model database. The products would have moving components and total part counts ranging from ten to sixty. The actual consumer household product would then be purchased so that an assembly time for training and validation could be determined based on the Boothroyd Dewhurst DFA method described in Section 5.4.2. Conducting the Boothroyd Dewhurst DFA method on actual purchased products is desired so that all assembly aspects required by the method are accurately captured.

An extensive search of online solid modeling databases was conducted to identify one that would contain assembly models that met the desired criteria. Some of the online databases searched were: GrabCAD, SolidWorks 3D Content, the GICL Website, McMaster Carr, and TopFreeModel among others. A single online database was not identified due to a variety of issues including: compatibility issues with SW assembly files, single solid parts created to look like final products, assembly models of final products containing only a few parts, assembly model created but without a reference to an actual consumer product which could be purchased, or in many cases a combination of these issues.

From this attempt the next method to collect assembly models was to download any product assembly models from any online solid modeling database that met the specified criteria other than matching a physical product. Many of the assemblies

downloaded still had the same issues mentioned above but some were useable. To increase the number of assembly models to match actual products, several physical products were reverse engineered so that respective assembly models could be created. The complete list of product assembly models and how they were generated can be seen in Table 5.3.

Table 5.3: Collection of product assembly models

#	Product	Assembly Model Generation
1	G2 Pen	Reverse Engineered
2	Pencil Compass	Reverse Engineered
3	Solar Yard Light	Reverse Engineered
4	Pony Vise	Reverse Engineered
5	Black and Decker Drill	Reverse Engineered
6	Paper Pro Stapler	GICL Website [53]
7	6" MagLight	SW 3D Content [54]
8	Indoor Electric Grill	SW 3D Content [54]
9	Shift Frame LH	OEM
10	Wide Flag	OEM

Once ten different assembly models of consumer products were gathered, the Mate Extraction tool discussed in Section 5.3 is used to automatically generate the mate connection graphs. The complexity of the mate connection graphs will be determined using the complexity algorithm developed for the initial connectivity work and then the complexity vector will be used as inputs to train the ANNs to respective assembly times which are determined in the following section.

Should a company wish to deploy this system in their design group, company specific assembly models can be collected and used for training purposes with known product assembly times. These historical models should be ideally collected from different projects, have been authored by different designers, and have different levels of component count and mating resolution. Specific strategies for selecting and developing ANN training models are reserved for future work.

5.4.2 Calculating Product Assembly Times

To conduct the ANN training, an assembly time is needed so that the complexity metrics generated for each product can be given a respective assembly time to target [5]. With the implementation of the ANN training scheme, a total of twenty-nine complexity metrics can be used to form a relationship to the predicted assembly times as opposed to the original regression method which only used three complexity metrics. Since access to the actual assembly times for the products is unavailable, the Boothroyd Dewhurst Manual DFA tables were used to predict an assembly time for each product. The process of completing the Boothroyd Dewhurst DFA method consists of disassembling the product and analyzing each individual part while answering the questions from the handling and insertion tables [8]. This process is generally applied as a redesign method where the actual product can be disassembled so an attempt was made to obtain physical products of all of the items listed in Table 5.3. The physical products for items 1-6 in Table 5.3 were obtained but items 7-10 could not be located or did not have a specific consumer product to match the SolidWorks model.

Without the physical product, applying the Boothroyd DFA method would be difficult since the objective and subjective analysis questions typically require a true understanding of how the product is assembled. To solve this problem a combination of DFA analyses were conducted, evaluated, and used. First a "virtual" Boothroyd DFA analysis was conducted on the SolidWorks Assembly model. The challenge with this "virtual" method is that without disassembling and holding the actual parts, an understanding of the product structure, function, assembly sequence, handling difficulties, and insertion difficulties cannot be obtained which is essential when applying the Boothroyd DFA. Therefore, the first step before the "virtual" Boothroyd DFA was conducted was to generate an exploded view of the assembly. An example of an exploded view for one of the OEM components can be seen in Figure 5.6.

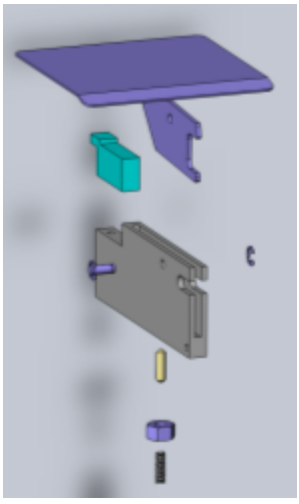


Figure 5.6: Exploded view of OEM Wide Flag Assembly

Generation of the exploded view makes the designer think about the assembly sequence and function of the given parts reducing some of the difficulty involved with a virtual DFA analysis. Even though generating the exploded view improves parts of the virtual DFA, other parts of the analysis like handling and insertion difficulties are still hard to identify.

The challenges of determining the handling and insertion difficulties come from the fact that these pieces of information require the designer to answer subjective questions about the product. For example deciding whether a part is difficult to grasp or if it has resistance to insertion is hard to do without actually picking up the part and inserting it. To reduce the impact of this issue, the designer was informed to not make assumptions about handling or insertion difficulties. If a difficulty is not obvious within the model then it is assumed to have no difficulty. Even though an attempt was made to not make assumptions about the assembly difficulties some of the answers may have been influenced by the fact that the product had previously been disassembled for the reverse engineered solid assembly models.

Once the "virtual" Boothroyd DFA was completed, if a physical product was present that matched the SolidWorks model it was disassembled and the DFA analysis was conducted on it as well. The "virtual" Boothroyd DFA method was always conducted first to reduce the chance that a handling or insertion difficulty experienced during the physical analysis would influence the designer during the "virtual" analysis. Between the Boothroyd DFA analyses on the physical products and the virtual products a

total of sixteen assembly times to match the respective CAD assembly models were determined.

Should a company wish to deploy this system in their engineering group, company specific known assembly time values can be used and matched to the assembly models selected for ANN training. These time values can be deterministic or probabilistic, depending on the type of analysis desired. A comparison of different training pair types, such as model + range of assembly times, is reserved for future investigation.

5.4.3 Training of Mate Complexity DFA Method

The research on the connectivity complexity method previously conducted used ANNs to increase the accuracy of the original connectivity complexity DFA method [5]. Artificial neural networks were selected to identify the relationship between the products connectivity complexity vector and respective assembly times because they are often used to complete nonlinear statistical analyses [55,5].

The basic overview of the previous research is that the physical connectivity complexity graphs of twenty four OEM assemblies were manually put together and related to a respective MTM DFA based assembly time using ANNs. Each of the connectivity graphs was generated by manually evaluating the connections within each assembly. The connectivity graphs were then analyzed using a custom Matlab algorithm which generates a complexity vector that contains twenty-nine different complexity metrics. Then, nineteen of the twenty-four product's complexity vectors along with their respective MTM assembly times were used as inputs and targets to train the ANNs. Five

of the twenty-four assemblies were left out of the ANN training to test the effectiveness of the trainings. [5]

The ANN training consisted of 189 neural network architectures which used up to three layers and fifteen neurons depending on the specific combination [5]. Once the architectures were created, each one was given the training set of complexity vectors as inputs and assembly times as targets to generate a unique mapping. Since each architecture may generate a different relationship every time it is given the same set of inputs and targets, each architecture was given the same training set 100 times and all 100 relationships were captured. Once the training was completed, it was tested using the five remaining product's connectivity complexity vectors.

The training was tested by inputting the withheld complexity vectors into the 189 types of trained networks 100 times, which then predicts 100 assembly times for each architecture. To evaluate and select the best architecture, the probability densities of the predicted times were used to determine if the times were within a given percent of the respective known time. The total probability that the predicted time would be within the given percent was then used to find the architectures that would be most likely to predict the correct assembly times. With the best architectures identified, these could be used to predict assembly times without having to train or test all 189 architectures. [5]

To determine if extracted mate connections from SW assemblies can be used to predict assembly times the ANN training method used from the motivation research was re-created. First the Automatic Mate Extraction Add-in discussed in Section 5.3 was used to extract the mate connections from the ten SW assembly models listed in Table

5.3. These mate connection graphs were then run through a Matlab Complexity Algorithm to generate respective complexity vectors that contained twenty-nine complexity metric values. The complexity vectors and assembly times of the Pencil Compass, the 6 Inch MagLight, and the Black and Decker Drill from Table 5.3 were held back to be used as test inputs once the ANN training was completed. These three products were chosen to be held back for testing because their part counts and assembly times form a good representation of the training set.

To train the ANNs for this research, 189 architectures were generated which consisted of one to three layers with up to fifteen neurons per layer depending on the configuration. Each architecture was given the training set 100 times so that the probability densities could be used to better approximate the relationship. The ANN training inputs for this research consisted of eleven complexity vectors for eleven of the sixteen assembly times. If a product had both a virtual and physical Boothroyd DFA predicted assembly time then the same complexity vector for that product would be trained towards the two different assembly times. Once the training inputs and targets were compiled the different ANN architectures were trained and the best ones were selected and evaluated for later use as described above.

Three separate Artificial Neural Networks training sets using different inputs and targets were evaluated to determine if the number of mates has an effect on the predicted results. The first training set called Case 1 was generated using complexity vectors that were based on all of the SW models being fully defined. This means that all parts in the assembly are constrained and cannot move. The second training set called Case 2 was

generated using complexity vectors that were based on the SW models being partially defined. Partially defined was achieved by having the designer mate the assembly model to the point where parts are constrained based on the design intentions. The third training set called Case 3 was generated using both the complexity vectors generated for the fully defined and partially defined SW assembly models. This means that Case 3 had twice as many training inputs and targets than Case 1 and Case 2.

5.5 Testing

Once the different training schemes for the given inputs and targets were generated they had to be tested. The complexity vectors from the three products held back for testing were given to the trained ANNs as inputs so that it could generate predicted assembly times. All of the 189 architectures for each ANN training case were evaluated to determine which ones were most effective. The effectiveness of the architecture was determined by evaluating the probability density that the 100 predicted assembly generated for each product would be within +/- 25% of the target assembly time.

Since each test input was given to each architecture 100 times the probability density of the predicted times can be generated as shown in Figure 5.7. These probability densities can then be compared to the assembly time predicted by the Boothroyd Dewhurst DFA method to see how effective the given architecture was. The Boothroyd Dewhurst DFA time is shown by the vertical red line on the plot and +/- 25% of this target time is shown by the vertical yellow lines. Figure 5.7 shows an example

probability density plot generated by one architecture for a given product. This figure shows that the majority of the predicted assembly times fall within the +/- 25% range.

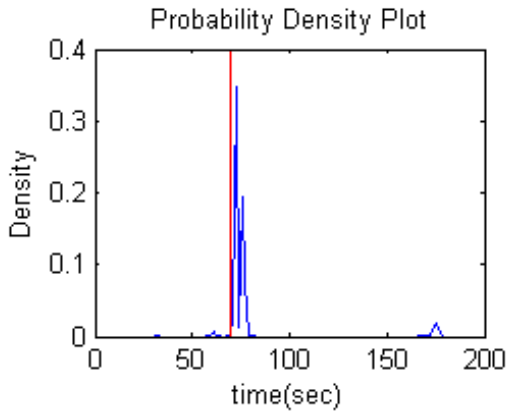


Figure 5.7: Example Probability Density Plot

Once all of the probability density values for all architectures had been evaluated, the overall probability that the architecture would predict a time within the given+/- 25% range was determined. The overall probability was calculated by finding the area under the probability density plot that was within the +/- 25% range of the target time for all three test products. The average probability for all 189 architectures was then found and compared to see which one would be most effective at predicting an assembly time within the specified target range. The five architectures with the highest average probabilities were selected for evaluation. Table 5.4 shows the top five architectures selected for the three training schemes: Case 1, Case 2, and Case 3.

Table 5.4: Selection of top 5 ANN architectures for each testing case

Case 1 (F. Def.)		Case 2 (P. Def.)		Case 3 (F&P Def.)	
Arch.	Avg. Prob.	Arch.	Avg. Prob.	Arch.	Avg. Prob.
95	0.601	56	0.999	109	0.992
173	0.541	64	0.963	45	0.736
79	0.537	174	0.789	154	0.699
90	0.500	147	0.753	30	0.639
99	0.500	52	0.737	133	0.625

Case 2, which was trained with the partially defined products, resulted in the overall best top five architectures based on the probability density curves. ANN training Case 3 which used fully and partially defined products was the second best, while training Case 1 which used only fully defined products was the least effective. The mates added to parts in an assembly define how that part is constrained within that assembly. If a designer is forced to add more mates than required, it is possible that the original constraint definition will be lost or negatively affected. This could be a possible cause for the fully defined assembly models predicting less accurate results, a detailed investigation into this issue is reserved for future work. For comparison purposes, the times for each of the top five architectures for each training case, were compared across the three test products.

To determine the effectiveness of each ANN training scheme, their predicted assembly times had to be compared. The average predicted assembly time generated for each product using the top five architectures for each ANN training scheme was computed and compared to the Boothroyd DFA target assembly time. Table 5.5 shows the average predicted assembly times from each training scheme and the respective target time which was determined using the Boothroyd Dewhurst DFA method. The cells in the

table are shaded to illustrate the level of accuracy for the different tests; green shading indicates that the values returned are within the +/- 25% tolerance range and the yellow shading indicates that the values are within the +/- 50% tolerance range.

Table 5.5: Comparison of predicted assembly times for each training case

Product Test Case	Level of Definition (Test)	Target Time (s)	Case 1 (Fully Defined Training) (s) (+/- % Error)	Case 2 (Partially Defined Training) (s) (+/- % Error)	Case 3 (Fully and Partially Defined Training) (s) (+/- % Error)
Pencil Compass	Fully	68.3	121.4 (+77.5)	NA	94.5 (+38.2)
	Partially		NA	96.6 (+41.2)	82.5 (+20.6)
MagLight	Fully	75.4	118.3 (+56.9)	NA	70.2 (-6.9)
	Partially		NA	65.1 (-13.7)	75.7 (+0.5)
Black & Decker Drill	Fully	189.6	226.3 (+19.3)	NA	319.3 (+68.4)
	Partially		NA	186.1 (-1.9)	202.3 (+6.7)

For training Case 1, the test cases as well as the training set were all fully defined models. For training Case 2, the test cases as well as the training set were all partially defined models. Training Case 3 used a combination of fully defined and partially defined models for training, and therefore both fully defined and partially defined models were used for testing.

The results of the testing indicate that using training Case 3 which had fully and partially defined models resulted in predicted assembly times that were closest to the Boothroyd target times. The percent error of the predicted assembly times for four of the

six inputs decreased by using the training Case 3 as opposed to the first two training cases. Out of the two percent errors that increased using the training Case 3, one of them was still within seven percent of the target time, which is still deemed acceptable. To determine what effect the level of product definition used in the training cases, fully or partially defined, has on the predicted assembly times these results are analyzed in the section below.

5.5.1 Effect of Training Assembly Definition

The three training cases presented were used to determine if the level of assembly definition had an effect on the predicted assembly times. Requiring a designer to add enough mates to fully defined every part would essentially fix the number of mates that a given part and product would have, which would theoretically provide a more repeatable result. If the designer is allowed to only partially define the assemblies then they only have to add the mates that they see fit which requires no extra work on their part. To compensate for both extremes a combination of fully and partially defined models was also included.

The training case results shown in Table 5.4 identified that training Case 2, which used a training set of only partially defined models, had the highest probability of predicting a product's assembly time within +/- 25% of the target time. When the products average predicted assembly times were compared, it was determined that training Case 3 generally resulted in a decrease in percent error over training Case 1 and Case 2, Table 5.5. Training Case 3's decrease in percent errors could be a result of its training set size which was twice the size of training Case 1 and Case 2. Looking in to

training Case 3 and comparing the partially and fully defined predicted times shown in Table 5.5, the partially defined models always had less percent error than the fully defined models. The results from these three training cases suggest that using partially defined assembly models generally provides better training results. It also suggests that an increase in training set size could also provide the accuracy of the predicted assembly times.

5.6 Summary of Initial Automation

In this preliminary investigation, with limited training sample sizes, it was found that an integrated training regime that includes both partially and fully defined assembly models performs better than the networks that were trained on only fully or only partially defined models. This suggests that there is, first a need for larger training sets and second that there is additional information captured within different assembly mating styles. The type of assembly models that were used for training did not necessarily fully span the types of mating options that are available. Therefore, a wider spanning set of training products is recommended.

One of the major difficulties with using mates to determine the assembly times of products is that different designers can and will mate the same assembly in different ways. To determine if different mating schemes have an effect on the results of the ANN predicted assembly times, two different mating schemes were tested. Two designers were asked to create an assembly model of the 6 Inch MagLight and mate the components as they normally would. Once they mated the product based on their style, complexity vectors were generated for each designer's partially defined assembly. The

designers were then asked to continue adding mates until their model was fully defined. A complexity vector of the fully defined assembly model was then generated. The fully and partially defined complexity vectors were then given to the third training set, Case 3, to evaluate and compare the predicted assembly times. The predicted assembly times and the percent error for each designer's mating schemes are shown in Table 5.6. A detailed study with a larger sample of mating configurations should be further investigated. The cells in the table are shaded to illustrate the level of accuracy for the different tests; green shading indicates that the values returned are within the +/- 25% tolerance range and the yellow shading indicates that the values are within the +/- 50% tolerance range.

Table 5.6: Mate configuration comparison

Product	Level of Definition	Target Time (sec)	Predicted Time (sec)	% Error (+/-)
MagLight Mates Designer 1.	FD	75.4	70.2	- 7
	PD		75.7	+ 1
MagLight Mates Designer 2	FD		95.8	+ 27
	PD		80.4	+ 7

While this chapter presents preliminary results, these results suggest that this method is feasible in creating a tool that can integrate into a commercial CAD system to provide automatic assembly time estimation. Should companies wish to integrate this tool in their product development process, strategies are needed for appropriate selection of company specific training sets and associated assembly time. It is recommended, preliminarily, that these training sets should vary in product type, author, complexity, and geometric classification. The development of these strategies is deemed out of scope for this paper, but is under current investigation.

5.7 Automated DFA Method

The goal of this chapter was to determine if the identified DFA method could be fully automated which addresses the second research question:

RQ2:	Can the identified assembly time prediction method be automated so that it predicts an assembly time using information extracted from 3D solid modeling software?

The method that was identified for automation in Chapter 4. was the Connectivity Complexity assembly time prediction tool. The results of the work presented in this chapter display a partially automated version of the Connectivity Complexity assembly time prediction tool that extracts mates from SolidWorks assembly models and uses them to predict an assembly time. The tool presented automates the most time consuming and subjective part of the original Connectivity Complexity method which is the identification of the inter part connections and the assembly of the bi-partite interconnection table. The steps to use the developed tool to predict an assembly time are as follows were the users actions are labeled "User", are green, and are bold while the programs actions are labeled "Program", are in red, and are in italics:

- **User: Open SolidWorks Assembly**
- **User: Click on SWMate2 Add-in**
- *Program: extracts mates and builds the bi-partite table*
- **User: Open Matlab and call custom complexity algorithm passing the generated file name as the input**

- *Program: Complexity algorithm reads mates from the bi-partite table and calculates a respective complexity vector*
- **User: Calls custom Matlab ANN function (accepts generated complexity vector as input)**
- *Program: Function uses complexity vector to predict and output the assembly time*

The developed tool is a C++ SolidWorks Add-in that appears as a button in the SolidWorks GUI. Once selected, the Add-in program extracts the mates from the assembly model and builds the bi-partite table required to identify the complexity of the assembly. With the automatically generated bi-partite table of mate connections, the only manual steps that the user has to complete is passing the bi-partite table to two custom Matlab functions which use the assembly's complexity and a trained ANN to predict an assembly time. These manual processes currently require opening programs or calling defined functions which can be easily automated to create a totally automated DFA tool. This chapter addressed the second research question and found the respective hypothesis to be correct:

RQ2 Hypothesis:	The identified assembly time prediction method can be automated so that it predicts an assembly time using only information extracted from 3D solid modeling software.

It was determined that the original Connectivity Complexity method could be modified to use mate connections instead of physical inter part connections. These mate connections are stored in solid modeling assembly models and can be extracted with appropriate tools. A custom SolidWorks Add-in was developed to automatically extract

the mate connections which are then used to predict assembly times using artificial neural networks. The effectiveness of the ANNs used to predict the assembly times are determined by the size and variability within the training set. To increase the effectiveness of this tool, larger training sets with more variability should be investigated. The goal of this research is to develop an automated DFA tool to reduce or eliminate the issues that current DFA methods and tools have. A semi-automated DFA tool was presented in this chapter. Chapter 6. will further investigate ANN training cases so that an encompassing set of architectures can be selected. Once a set of architectures are selected the method will be fully automated and evaluated to determine if it addresses existing DFA issues.

CHAPTER 6. DOES THE DEVELOPED AUTOMATED TOOL ADDRESS EXISTING DFA ISSUES

Chapter 5. presents a semi-automated DFA tool that only requires the user to click on a button to start the analysis and call a few Matlab functions to predict a product's assembly time from information it extracts from a respective SolidWorks assembly model. The tool could be fully automated by opening Matlab and calling the required functions using the SolidWorks add-in. Before a fully automated version of this tool is developed and implemented, the current semi-automated version should be evaluated to determine its overall effectiveness as a DFA tool. This will ensure that the current version is effective and should be continually developed into a fully automated version.

The evaluation of the current version is accomplished by answering the third and final research question; *does the new method solve the issues that current DFA methods have?* This research question is addressed by (1) exploring ANN training sets with regards to how they are affected by the training set size and types of inputs used, (2) studying the sensitivity of the predicted assembly time with respect to the types of mates used, and (3) comparing the new tool to the Boothroyd Dewhurst DFMA software.

6.1 Investigation of ANN Trainings

To understand how the training sets variability and size affects the predicted assembly times additional testing is done. To investigate this affect, the original three training sets from Section 5.5 along with five additional training sets are evaluated in the following sub sections. These eight training cases allow the effect of training case size

and the effect of uniqueness of training case inputs on the predicted assembly time to be evaluated. The eight training cases evaluated are summarized in Table 6.1 showing the name, the number of test inputs and targets, the level of assembly definition, the types of assembly time inputs, and whether the training used repeated inputs that were mapped to different targets. A full training case description including the products used for each training set can be found in the appendix.

Table 6.1: ANN training set descriptions

Training Set Name	# of Training Inputs & Targets	Assembly Definition: Full, Partial, or Both	Assembly Times: Virtual, Physical, Both	Repeated Training Inputs to Different Target
Case 1	11	Full	Both	Yes
Case 2	11	Partial	Both	Yes
Case 3	22	Both	Both	Yes
Case 4	11	Both	Both	Yes
Case 5	11	Both	Both	Yes
Case 6	12	Partial	Virtual	No
Case 7	12	Partial	Virtual	No
Case 8	12	Partial	Virtual	No

The specific details of the first three training cases were previously described in Section 5.5 along with their top five architectures and respective average probabilities (Table 5.4). Training cases four through eight were structured the same way as the first three cases, but using different test inputs and targets. By conducting different training cases, the effect that the training set size and types of inputs used can be explored. The top five architectures and respective average probabilities for training Case 4 through training Case 8 are shown in Table 6.2.

Table 6.2: Top five architectures with respective average probabilities for training cases 4 through 8

Case 4		Case 5		Case 6		Case 7		Case 8	
Arch	Avg. Prob.	Arch	Avg. Prob.	Arch	Avg. Prob.	Arch	Avg. Prob.	Arch	Avg. Prob.
143	0.796	1	0.759	89	0.989	110	0.999	43	0.823
161	0.763	120	0.675	31	0.966	124	0.996	22	0.735
153	0.627	169	0.531	91	0.782	4	0.982	69	0.732
49	0.627	188	0.513	9	0.772	113	0.982	24	0.694
34	0.599	166	0.484	112	0.743	18	0.970	78	0.653

The architectures and their average probabilities presented in Table 5.4 and in Table 6.2 can be used to determine a training Cases' ability to predict an assembly time within +/- 25% of the target time. These results can also be used to identify generalizations about which architecture structure tends to produce the highest probabilities.

6.1.1 Effect of Training Case Size

The initial ANN training investigation in Section 5.5 looked at three training cases:

- Case 1 that used partially defined models,
- Case 2 that used fully defined models, and
- Case 3 that used both fully and partially defined models.

The results identified that using only partially defined assembly models generally showed a decrease in percent error of the predicted assembly times. The results also identified that using training Case 3, which had a combination of fully and partially defined models and was twice as large as training Case 1 and Case 2, decreased the percent error of the predicted assembly times. Since the training size of Case 3 was

larger than those of Case 1 and Case 2, the decrease in percent error could be due to the increased training set size or to it using both fully and partially defined models as training inputs. The rest of this sub section evaluates training Case 3, training Case 4, and training Case 5 to determine specifically if an increase in training set size decreases the percent error in the predicted assembly times.

As shown in Table 6.1, training Cases 3 through 5 all use a combination of fully and partially defined assembly models where training Case 3 uses twenty-two inputs and targets where training Cases 4 and 5 only use eleven inputs and targets. Training Case 4 and training Case 5 are filtered versions of training Case 3. These were put together by selectively eliminating half of the inputs/targets from Case 3 while still using as much of the testing variety as possible. The predicted assembly times from Case 3, Case 4, and Case 5 can be seen in Table 6.3. The cells in the table are shaded to illustrate the level of accuracy for the different tests; green shading indicates that the values returned are within the +/- 25% tolerance range and the yellow shading indicates that the values are within the +/- 50% tolerance range.

Table 6.3: Comparison of predicted assembly times for training Case 3, Case 4, and Case 5

Product Test Inputs	**Level of Definition (Test)**	**Target Time (s)**	**Case 3** *Original Set* (s) (+/- % Error)	**Case 4** *Filtered Set_1* (s) (+/- % Error)	**Case 5** *Filtered Set_ 2* (s) (+/- % Error)
Pencil Compass	Fully	68.3	94.5 (+38.2)	137.9 (+101.6)	78.6 (+15.0)
	Partially		82.5 (+20.6)	84.5 (+23.6)	38.6 (-43.6)
MagLight	Fully	75.4	70.2 (-6.9)	56.2 (-25.5)	53.8 (-28.7)
	Partially		75.7 (+0.5)	42.1 (-44.2)	52.8 (-30.0)
Black & Decker Drill	Fully	189.6	319.3 (+68.4)	233.3 (+23.0)	383.5 (+102.2)
	Partially		202.3 (+6.7)	197.6 (+4.2)	258.3 (+36.2)

Comparing the percent errors from Case 3 to Case 5 in Table 6.3, Case 3 predicts assembly times with less percent error for all inputs except for one, the fully defined pencil compass. Comparing the percent errors from Case 3 to Case 5 in Table 6.3, Case 3 predicts assembly times with less percent error for four of the six test inputs. The percent error in predicted assembly time for the partially defined Black & Decker drill only increase by 2.5% from Case 3 to Case 4 which is not significant. These results signify that by increasing the size of the training case inputs and targets, there will be a general increase in accuracy with respect to the target. The general trends can be summarized as:

- Case 3 BETTER_THAN Case 4
- Case 3 BETTER_THAN Case 5
- Case 4 BETTER_THAN Case 5

6.1.2 Effect of Training Case Variability

The original training study used training sizes of eleven (Case 1), eleven (Case 2), and twenty-two (Case 3). These training sets, however, were not composed of unique inputs (the complexity vectors). Some of the product inputs were included twice in the training sets but mapped to different target assembly times, the virtual and physical Boothroyd DFA times for that specific product. Since the Boothroyd Dewhurst DFA method has variability in its predicted assembly times when conducted on the same product, this variability could be used to increase the effectiveness of a given training case. A training case could take the same complexity vector input (same assembly model / product) and map it to two different predicted assembly times (physical vs. virtual or from different designers. This tells the training case that with the same input, two possible outputs could occur, so as it develops a relationship it can compensate for some of the variability of the input target times. These types of training sets are describe as having non unique training inputs. Only seven of the eleven inputs for Case 1 and Case 2 were unique and only fourteen of the twenty-two inputs for Case 3 were unique. To develop a more effective training case the effect of training input variability is investigated in this section.

To investigate the effect of training input variability three different training cases were assembled (Case 6, Case 7, Case 8) by increasing the number of analyzed products. Based on the limited success of downloading product assembly models from online databases, the number of assembly models was increased by reverse engineering five additional consumer products. The updated list of product assembly models available for

training is presented in Table 6.4. Only certain combinations of the first ten assembly models shown were used to train Case 1, Case 2, and Case 3. The last five products were added to the training set to replace the repeated training inputs that were used in the first three test cases. The last three columns of Table 6.4 show Case 6, Case 7, and Case 8 where the products used to train each case are labeled "Training" and the products used as test inputs are labeled "Test".

Table 6.4: Increased product collection and training case products for training and testing

#	Product	Assembly Model Generation	Case 6	Case 7	Case 8
1	G2 Pen	Reverse Engineered	Training	Training	Training
2	Pencil Compass	Reverse Engineered	Training	Training	**Test**
3	Solar Yard Light	Reverse Engineered	Training	**Test**	Training
4	Pony Vise	Reverse Engineered	Training	Training	Training
5	Black and Decker Drill	Reverse Engineered	Training	**Test**	**Test**
6	Paper Pro Stapler	GICL [53]	**Test**	Training	Training
7	6" MagLight	SW 3D [54]	**Test**	Training	**Test**
8	Indoor Electric Grill	SW 3D [54]	Training	Training	Training
9	Shift Frame LH	OEM	Training	Training	Training
10	Wide Flag	OEM	Training	Training	Training
11	One Touch Chopper	Reverse Engineered	Training	**Test**	Training
12	Computer Mouse	Reverse Engineered	Training	Training	Training
13	Boothroyd Piston Assembly	Reverse Engineered	Training	Training	Training
14	3 Hole Punch	Reverse Engineered	Training	Training	Training
15	Durabrand Hand Mixer	Reverse Engineered	**Test**	Training	Training

Since all of the previous products had virtual Boothroyd Dewhurst DFA analyses already conducted on them, for consistency, the new ANN trainings, Case 6 through Case

8, only use virtual Boothroyd predicted assembly times as their targets which would be trained with unique complexity vector inputs for each product. The results of these ANN training cases can be seen in Table 6.5. Each test that yielded time estimations that were within the +/- 25% tolerance range is shaded.

Table 6.5: Comparison of predicted assembly times for the last three ANN training sets

Product Test Case	Level of Definition (Test)	Target Time (s)	Case 6 (s) (+/-% Error)	Case 7 (s) (+/-% Error)	Case 8 (s) (+/-% Error)
Pencil Compass	Partially	68.3	NA	NA	60.2 (-12.0)
MagLight	Partially	75.4	69.8 (-7.5)	NA	65.4 (-13.3)
Black & Decker Drill	Partially	189.6	NA	199.4 (+5.1)	233.8 (+23.3)
Paper Pro Stapler	Partially	123.5	118.3 (-4.2)	NA	NA
Durabrand Blender	Partially	263.2	271.8 (+3.3)	NA	NA
Solar Yard Light	Partially	128.8	NA	113.1 (-12.2)	NA
One Touch Chopper	Partially	316.6	NA	318.7 (+0.7)	NA

As shown in Table 6.5 the results for training Case 6, Case 7, and Case 8 have less than 14% error of the target time except one of the times generated by Case 8 which has 24% error. None of the first five training Cases investigated has percent errors this low for all test products. This signifies that providing a more diverse training set that does not reuse test inputs will increase the overall accuracy of the training set. Case 6 generally has the lowest overall percent error out of all eight training cases. The percent

errors for Case 6 range from -7.5% to +3.3% but it is closely followed by Case 7 which has percent errors ranging from -12.2% to +5.1%.

Training Case 8 was created such that it uses the same test inputs as the first five training cases which are: the pencil compass, the Mag light, and the Black and Decker drill. This was done so that the second through fifth training cases which did not have unique training inputs could be compared to training Case 8 which did have unique training inputs. Since Case 8 used partially defined assemblies it cannot be compared to Case 1 which used fully defined assemblies but Case 8 can be compared to Cases 2 through Case 5 for each product. A comparison between Case 8 and each of the other cases with respect to each product is as follows:

- For the pencil compass, Case 8 resulted in lower percent error than Case 2 through 5, the percent errors are as follows: Case 8 = -12%, Case 2 = +41%, Case 3 = +21%, Case 4 = +24%, and Case 5 = -44% error
- For the Mag Light, Case 8 showed a lower percent error for Case 2 trough Case 4 but an increase in % error with respect to Case 5, the percent errors are as follows: Case 8 = -13%, Case 2 = -14%, Case 3 = + 1%, Case 4 = -44%, and Case 5 = -30% error
- Out of the three products Case 8 preformed the worst on the Black & Decker drill, the percent errors are as follows: Case 8 = +23%, Case 2 = -2%, Case 3 = +7%, Case 4 = +4, and Case 5 = +36% error

6.1.3 ANN Training Recommendations

Based on the results from the investigation on ANN training case type, some recommendations to improve future training cases are provided in this section. Section 5.5.1 investigated how the level of assembly definition affected the results. It was determined that training cases that used partially defined models were more effective than those that only used fully defined models. Based on these results it is recommended that: *future training cases use only partially defined models*.

Section 6.1.1 investigated the effect of training case size on the predicted assembly times. Training cases that used eleven training inputs versus twenty two training inputs, both with fully defined and partially defined assembly models were evaluated and compared. The larger training case with more training inputs and targets resulted with better results where its average %error was at least 10% better than the closest training case of a smaller size. The training cases that used only eleven inputs resulted in average percent errors that ranged from 19% to 51% error depending on the case. Since the 51% error is at the limit of the +/- 50% tolerance range which means that the minimum training case size should be one that uses eleven inputs and targets. Based on these findings it is recommended that: *future training cases should use a minimum training case size of eleven inputs and targets, results will improved with larger training sets*.

Section 6.1.2 investigated the effect of training case variability on the predicted assembly times. Five of the eight training cases used non unique training inputs but mapped them to different targets. The other three of the eight training cases used unique

inputs mapped to unique targets. The results from this comparison determined that training cases that had more variability and unique training inputs were more effective than training cases that re-used training inputs. Based on these findings it is recommended that: *future training cases should use a set of unique training inputs*.

Based on the individual investigations and the recommendations listed above, the final recommendation is: *future training cases should use a set of at least eleven unique training inputs and targets that are made up of partially defined assembly models*.

The investigations into ANN training case types presented in this thesis are only initial studies used to make some initial recommendations. These studies should be continued using larger sample sizes to make more effective or specific recommendations but this is reserved for future work.

6.2 Overall Top ANN Architecture Selection

In order for the developed tool of Section 5.2 to be effective, ideally the user should be able to use the tool as is and not have to retrain the ANN's or go through an architecture selection process to use it. The architecture selection process has been used to evaluate the different training cases. This selection process involves running a complete training case on all 189 architectures and repeating each training 100 times. The probability that the architectures would predict the given test product target within a given target (+/- 25% of a manually estimated assembly time) is then evaluated. The average probability for the given architecture can then be determined and the top five architectures based on average probability are selected. Conducting this selection process will help improve the accuracy of the results but it is time consuming and requires

withholding inputs from the training case to evaluate the effectiveness of the architectures.

To eliminate the architecture selection process, this section presents five selected ANN architectures for the initial tool. The use of the suggested architectures may not provide the highest accuracy for one specific training case. Instead, architectures are sought that that perform well across multiple training cases. In this manner, the challenge of creating the "best" training case in terms of which cases to use for training, the size of the training set, and the diversity of the products can be avoided. Moreover, these suggested architectures allow for immediate tool use. If desired, the user can conduct new ANN trainings to customize the architecture selection for their specific product ranges as necessary. It is noted that the selection process used to suggest the five architectures presented is only an initial guideline. To identify an ideal set of architectures suitable for all possible training cases, extensive statistical analysis is needed, and as such is deemed to be out of scope for this research.

However, for a satisfying solution of selecting five "good enough" architectures, the average probabilities for the 189 architectures for all eight training cases were compiled and compared. Before comparing the values, the probabilities with values greater than one were set to one. The average probability for the given architecture was determined by identifying the area under the probability density plots discussed in Section 5.5. The area under the probability density plots within the given percent range was determined using the Matlab trapz function. This function often has rounding errors which could possibly result in an area and probability greater than one, which is not

possible. For the presented probabilities for each architecture, if the value is greater than one it is reduced to one. Then, the average of the probability for each architecture across all eight training cases was determined to represent the overall effectiveness of each architecture. The new average probability for each architecture was then sorted so that the architectures with the highest overall probabilities could be identified. The top thirty architectures based on the average probability of all eight training cases are shown in Table 6.6.

Instead of choosing the top five architectures based on overall average probability for all training cases, the selected architecture set should have emphases on the best training cases. Emphases are placed on the best training cases since these types of training cases are recommended for future use. Since the average probabilities for the top five architectures of training Case 6, Case 7, and Case 8 were higher than the other training cases, their top thirty highest probabilities are compared to the top thirty highest average probabilities of all training cases. The architectures that were present within the top thirty overall list and that were also present in the top thirty list of these three training cases were marked and counted to determine which ones showed up the most. The top five architectures that showed up the most were then selected and are highlighted in green in Table 6.6.

Table 6.6: Top thirty architectures based on average probability of all eight training cases

Rank	Architecture Number	Architecture Structure	Average Probability of all eight training cases
1	120	[3,2,1]	0.4317
2	68	[1,1,4]	0.3964
3	112	[2,5,3]	0.3668
4	9	9	0.3603
5	31	[3,2]	0.3438
6	45	[5,2]	0.3398
7	88	[1,5,4]	0.3352
8	49	[5,6]	0.3304
9	172	[5,2,3]	0.3243
10	95	[2,2,1]	0.3161
11	67	[1,1,3]	0.3149
12	32	[3,3]	0.3130
13	38	[4,2]	0.3110
14	100	[2,3,1]	0.3093
15	109	[2,4,5]	0.3065
16	37	[4,1]	0.3064
17	89	[1,5,5]	0.3044
18	159	[4,4,5]	0.2950
19	178	[5,3,4]	0.2948
20	113	[2,5,4]	0.2930
21	56	[6,6]	0.2891
22	143	[4,1,4]	0.2866
23	166	[5,1,2]	0.2833
24	90	[2,1,1]	0.2823
25	77	[1,3,3]	0.2813
26	141	[4,1,2]	0.2791
27	75	[1,3,1]	0.2785
28	64	[7,7]	0.2763
29	2	2	0.2763
30	11	11	0.2761

These top five architectures make up the suggested architecture set that best represents all eight training cases with an emphasis on the preferred training case type, Case 6, Case 7, and Case 8. Looking at the selected architecture structures shown in Table 6.6, it can be seen that they essentially span the entire range of possible structures.

The five selected architectures encompass structures consisting of one, two, and three layers with a wide variety of the number of neurons in each layer. This suggests that the structure of the architecture does not have a significant result on the generated outputs but instead results will be more affected by the use of appropriate training sets.

This section presented the selection of a representative architecture set that can be used when a custom training case and architecture selection process is not desired. Using a representative architecture set also eliminates many of the intermediate steps that were previously required to go from the SW Mate Extraction Add-in to a predicted assembly times. With a specified set of five architectures, only the desired architectures have to be trained instead of all 189 architectures. Training five selected architectures takes approximately 5 minutes where as training all 189 architectures takes approximately 120 minutes for the training set sizes used in this research. With a smaller set of trained architectures, they can be saved for later use, which, in turn, reduces the frequency of the time consuming training. A trained set of neural networks can then be loaded as needed and only the new test inputs have to be provided to generate a predicted assembly time. The benefits of selecting a smaller sample of architectures from the original 189 allow the tool to be more effectively automated.

6.3 Implementation of Selected Architectures

Based on the initial investigation into the effect of training case size (Section 6.1.1) and variability (Section 6.1.2) on the predicted assembly times, a set of five architectures were selected to be used when the implementation of a new training case or when conducting a specialized architecture selection process is not desired. The five

architectures selected become the default values for this assembly time prediction tool and they allow it to be fully automated. The fully automated version of this assembly time prediction tool and its effectiveness to predict an assembly time for a final test case are covered in sub section 6.3.1. The sensitivity of the tool to different mating configurations resulting from different engineers creating the assemblies is then investigated in sub section 6.3.2.

6.3.1 Automated Assembly Time Prediction Tool

Chapter 5. presented a semi-automated assembly time prediction tool that automated the tedious time consuming aspect of the original Connectivity Complexity method. However, it still required the user to complete part of the steps to conduct the analysis. Before these steps were automated the effectiveness of the tool and its ANN training sets were evaluated and it was suggested that larger training sets that use a larger variety of inputs and targets should be used. A general set of effective architectures were identified so that only five ANN architectures need to be trained and used to predict assembly times when needed. This small number of trained neural networks can be easily loaded, passed a given input, and be used to predict a products assembly time.

To predict an assembly time using the developed assembly time prediction tool the following steps must be completed (again, user actions are in bold green letters and program executions are in italics colored red):

- **User: Opens SolidWorks assembly model**
- **User: Click on SWMate2 Add-in**

- *Program: Extracts mates and builds the bi-partite table*
- *Program: Opens Matlab and calls custom complexity algorithm passing the generated file name as the input*
- *Program: Complexity algorithm reads mates from the bi-partite table and calculates a respective complexity vector*
- *Program: Calls custom Matlab ANN function (accepts generated complexity vector as input)*
- *Program: Loads previously determined ANN training case that uses top five selected architectures*
- *Program: Mate connection complexity vector is given to custom ANN assembly time prediction function as test input and the function outputs replicated results*
- *Program: Results are interpreted and a predicted assembly time is displayed*

To test the developed assembly time prediction tool, a product that has not been previously used for training or the interpretation of results is identified and used for testing. A Durabrand Electric Knife was selected for final testing because it is similar in size, part count, and product family to the products and assembly models used for training. The SolidWorks assembly model generated for the Electric Knife forms a rough representation of the actual product but it is not an exact replica. Once the Electric Knife assembly model was generated, a virtual Boothroyd Dewhurst DFA analysis was conducted, taking approximately thirty three minutes to complete, that predicted an

assembly time of 212.34 seconds. The new assembly time prediction tool is evaluated by opening the assembly model for the Electric Knife and clicking on the assembly time prediction SolidWorks Add-in.

The Electric Knife assembly model was tested using the top five selected architectures from Table 6.6. This testing was repeated for all eight training cases listed in Table 6.1. The predicted assembly times from the new tool are tabulated in Table 6.7. The cells in the table are shaded to illustrate the level of accuracy for the different tests; green shading indicates that the values returned are within the +/- 25% tolerance range and the yellow shading indicates that the values are within the +/- 50% tolerance range.

Table 6.7: Predicted assembly times for an electric knife using fully automated assembly time predication tool

Training Set Name	Electric Knife Target Time (s)	Predicted Time from Loaded Training Set (s)	% Error (+/-)	Analysis Time (s)
Case 1	212.34	457.83	+54	68
Case 2		665.87	+68	67
Case 3		315.23	+33	67
Case 4		434.02	+51	70
Case 5		407.4	+48	68
Case 6		251.7	+16	67
Case 7		204.59	-4	68
Case 8		225.34	+6	68

Table 6.7 shows that the percent error in the predicted time for the eight loaded training sets ranges from -4% to +68% error. If the cases are broken down into general categories, the same conclusions that were inferred in the previous training case investigation in Section 6.1 are made again in Table 6.7. Training Case 1, Case 2, Case 4, and Case 5 all had a training size of eleven inputs and targets but reused training inputs

which resulted with the highest percent errors ranging from 47% to 68% error. Training Case 3 which had twice the training size, twenty-two, but also reuses training inputs resulted in a percent error of 33%. Training Case 6, Case 7, and Case 8 had training sizes of twelve inputs and targets each of which are unique. . This resulted in the lowest percent error ranging from -4% to +16% error. These percent errors are well within the +/- 50% errors that are possible with the Boothroyd Dewhurst method [30].

Running the analysis on this test product while loading trained neural networks took less than 111 seconds once Matlab was opened. The total time to run the analysis including opening and initializing Matlab which takes approximately another 120 seconds making the total approximate analysis time 330 seconds. By fully integrating a trained ANN in C++ within the add-in the user of Matlab could be eliminated improving the run time efficiency.

6.3.2 Mate Sensitivity Analysis

During the initial automation of the method presented in Chapter 5. , the effect of different mating configurations was briefly explored. The Mag Light assembly model that was un-mated was given to two designers and they were each asked to add mates to the assembly as they saw fit creating a partially defined model. Once this was completed they were asked to add more mates until every part in the assembly was fully constrained resulting in a fully defined model. The predicted assembly times of all four assemblies were then analyzed to determine if two different mating styles would result in significantly different assembly times. The initial automation used the top five architectures for the specific training set in question and the results showed that there was

a change in the predicted assembly time depending on the mating configuration used, Table 5.6. This initial investigation only looked at two designer's mate configurations for one product which was enough to identify that the configuration does affect the assembly time but not to quantify by how much or what about the mate configuration caused the variations. A continuation of the initial mate sensitivity presented in Chapter 5. is presented in the rest of this sub section to quantify some of these aspects.

With a fully automated version of the assembly time prediction tool a more detailed mate sensitivity analysis can be conducted. Three separate products were chosen for this study: the Solar Yard Light, the Black & Decker Drill, and the One Touch Chopper. These three products and their respective part count, Boothroyd Dewhurst predicted assembly times, and their product structures are listed in Table 6.8.

Table 6.8: Selected products for mate sensitivity study

Product	Part Count	Boothroyd Predicted Assembly Time (s)	Product Structure
Solar Yard Light	15	128.79	Circular
Black & Decker Drill	26	186.65	Clam Shell
One Touch Chopper	43	316.67	Combo: Clam Shell & Stackable

Table 6.8 shows that these products represent the variety of products used in the different training sets. This variety includes assembly time, part count, and general product structure, all of which are different for all three products listed. Circular product structures are composed of products that generally have circular cross sections, have parts that can be located with two constraints, and where the majority of components are

inserted along the same axis. Clam shell product structures are products that sandwich the majority of parts between two halves. Stackable product structures generally have some type of base or foundation where other parts are stacked on top of one another to create the assembly. Products can also have structures that are based on any combination of these.

The assembly models for each product were prepared by creating an assembly file that contained all individual components for that product without any mates and by creating a separate reference assembly file that illustrated how the product is assembled. The reference assembly file allows the students to see how the product is put together. To prevent the students from being influenced by the reference assembly and the mates used to define it, the parts were fixed in place and all mates were deleted. An exploded view of the reference assembly was also created to aid them in determining the assembly sequence. The reference model for the Black & Decker drill is shown in Figure 6.1.

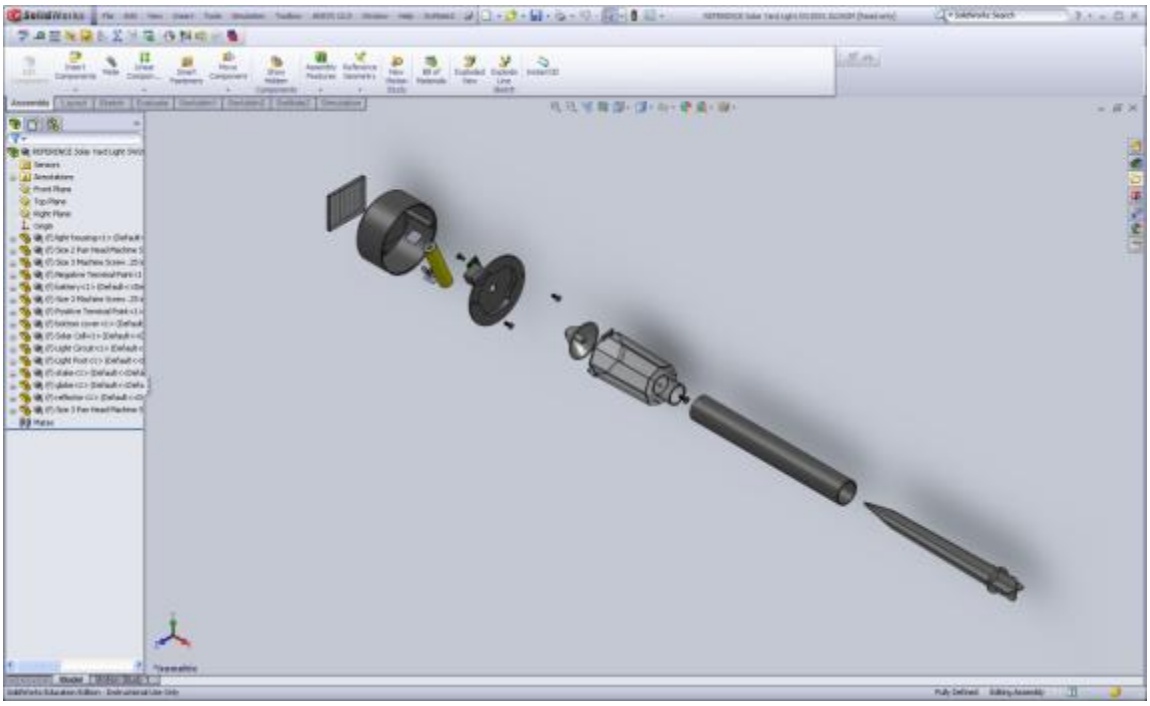


Figure 6.1: Exploded view of Solar Yard Light Reference Assembly

The exploded view of the reference assemblies could be collapsed so that the parts exact location within the assembly could be seen. The product assembly file provided to the students included all of the product parts in the general location with respect to the parts they will be mated to. The students will have to position the parts in the correct location and then add mates to the assembly as they see fit. Figure 6.2 shows the Solar Yard Light assembly model provided to the students so that they can add mates as they see fit.

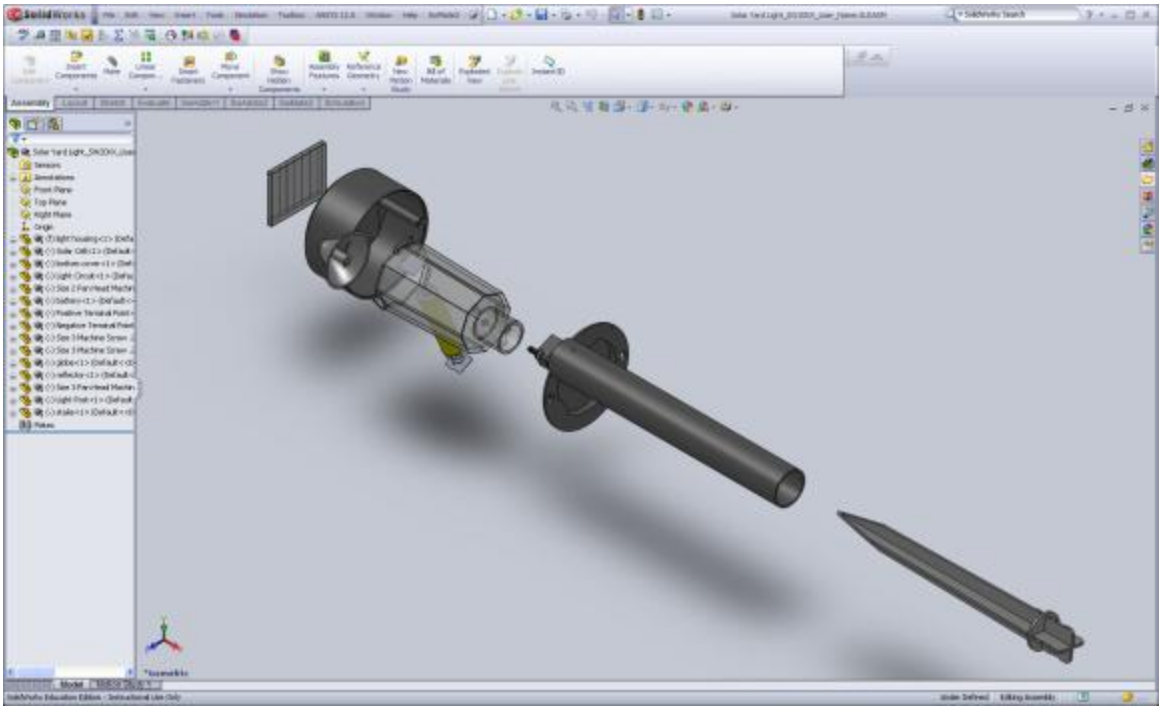


Figure 6.2: Solar Yard Light assembly model provided to students with no mates

The assembly models and reference assembly models for all three products were distributed to senior and graduate mechanical engineering students enrolled in a Design for Manufacturing course. The students were allowed to add mates to the unmated collection of parts as they felt appropriate. These final mated assemblies were then used for assembly estimation time analysis with the developed tool.

Upon completion of the mating activity, the students completed an on-line form asking: graduate or undergraduate student, SolidWorks Assembly experience, frequency of SolidWorks assembly creation, and approximate time to add mates to the assembly. The level of SolidWorks assembly experience is displayed in terms of low, medium, and high levels where low experience relates to mating less than ten assemblies, medium experience relates to mating between ten and fifty assemblies, and high experience is

mating more than fifty assemblies. The frequency of SolidWorks assembly usage is also displayed as low, medium, or high where high frequency relates to working with assembly models at least once a week, medium frequency relates to working with assembly models once a month, and low frequency relates to working with assembly models less than a year. It should be noted that some students choose not to add mates to all of the assemblies. The form results are summarized in Table 6.9.

Table 6.9: Form results from mate sensitivity study of assembly time prediction tool

Student	Under Grad. / Graduate	SW Assembly Experience	SW Assembly Usage Frequency	Mate Time Light (min)	Mate Time Drill (min)	Mate Time Chopper (min)
S1	UG	Low	Low	30 < t < 45	45 < t < 60	NA
S2	UG	Low	Low	60 < t < 90	NA	NA
S3	UG	Low	Med.	15 < t < 30	NA	NA
S4	Grad	Low	Med.	15 < t < 30	45 < t < 60	NA
S5	Grad	Med.	Med.	30 < t < 45	t < 15	60 < t < 90
S6	Grad	Med.	High	NA	30 < t < 45	30 < t < 45
S7	UG	Med.	Med.	15 < t < 30	45 < t < 60	30 < t < 45
S8	Grad	Low	Med.	45 < t < 60	t < 90	45 < t < 60
S9	Grad	Med.	Med.	30 < t < 45	45 < t < 60	45 < t < 60
S10	Grad	Low	High	45 < t < 60	t < 15	NA
S11	UG	Med.	Low	15 < t < 30	NA	NA

The results of Table 6.9 show that about half of the students had low assembly experience and the other half had medium assembly experience. These results also show that three students had low assembly usage frequency, six students had medium usage frequency, and two had high usage frequency. This shows that the majority of the students worked with assembly models with at least once a month on average.

Once all of the mated assemblies were compiled, the automated assembly time prediction tool developed was used to predict a respective assembly time. The number of mates the students added, the target time, the predicted assembly times for each student's assembly, the percent error in the predicted time, and the Matlab analysis time for the Solar Yard Light are shown in Table 6.10. The analysis time shown in Table 6.10 does not include the time to open up and initialize Matlab. The cells in the table are shaded to illustrate the level of accuracy for the different tests; green shading indicates that the values returned are within the +/- 25% tolerance range and the yellow shading indicates that the values are within the +/- 50% tolerance range.

Table 6.10: Mate sensitivity analysis for Solar Yard Light

Student	Solar Yard Light Target Time	# of Mates	Predicted Time from Loaded Training Set	% Error (+/-)	Analysis Time (s)
Student 1	128.79	33	129.56	+1	67
Student 2		32	110.99	-16	71
Student 3		25	88.71	-45	68
Student 4		36	121.08	-6	69
Student 5		38	115.95	-11	70
Student 7		36	145.95	+12	64
Student 8		35	131.32	+2	65
Student 9		41	107.08	-20	63
Student 10		36	125.39	-3	64
Student 11		36	111.3	-16	64

One of the students did not submit an assembled Solar Yard light resulting in only ten assembly models evaluated in Table 6.10. Of the ten different mated assembly configurations analyzed, the percent error in the predicted assembly time ranged from -45% to +12% error with the average of the absolute values being 13% error. Looking at

Table 6.10, the number of mates each student added does not appear to directly relate to the predicted assembly time and the percent error. Student one used thirty three mates and student two used thirty two mates but between but the predicted assembly times had +1% and -16% error respectively. Likewise, Students four, seven, ten, and eleven all used thirty six mates but the percent errors were -6%, +12%, -3%, and -16% respectively. Student three used the least number of mates, twenty five, but had the largest percent error, -45%. Since the number of mates does not appear to directly relate to the predicted assembly time, the significantly higher percent error for Student 3 could possibly be caused by different assembly definition, emphasis on one type of mate usage, or usage of reference geometry to mate parts. To fully understand the cause of this localized increase in percent error, a detailed study investigating the types of mates used and the respective complexity vectors created has to be conducted; this is reserved for future work.

It is important to note that all of these student mated assemblies were within +/- 50% of the target time and nine of the ten were within +/- 25% of the target. Excluding the predicted time from Student 3's model the percent error range changes from -20% to +12% error. The analysis time to predict these assembly times was less than seventy-two seconds for each model per model; this time to complete the analysis does not include the time it takes Matlab to open and initialize which is approximately 120 seconds. The original target assembly time for the Solar Yard Light was predicted using a Virtual Boothroyd Dewhurst DFA analysis, taking 3,300 seconds (55 minutes) to complete the analysis manually.

Eight of the eleven students added mates to the Black & Decker drill assembly. The number of mates they added to drill, their predicted assembly times and the percent error with respect to the target time are shown in Table 6.11. The cells in the table are shaded to illustrate the level of accuracy for the different tests; green shading indicates that the values returned are within the +/- 25% tolerance range and the yellow shading would indicate that the values are within the +/- 50% tolerance range.

Table 6.11: Mate sensitivity analysis for Black & Decker Drill

Student	Black & Decker Drill Target Time (s)	# of Mates	Predicted Time from Loaded Training Set	% Error (+/-)	Analysis Time (s)
Student 1	189.65	52	205.73	+8	68
Student 4		46	188.4	-1	67
Student 5		59	220.69	+14	68
Student 6		53	240.25	+21	64
Student 7		59	232.04	+18	65
Student 8		62	190.21	+0.3	64
Student 9		50	224.9	+16	63
Student 10		48	213.6	+11	65

Out of the eight different mating configurations analyzed for the Black & Decker drill, the percent errors ranged from -1% to 21% error with the average of the absolute values being 11% error. The analysis time required to complete the analysis for each assembly was less than sixty-eight seconds excluding the time it takes Matlab to open and initialize. The target assembly time for the Black & Decker drill was predicted using a Virtual Boothroyd Dewhurst DFA analysis which took 2,520 seconds (42 minutes) to complete.

Five of the eleven students added mates to the One Touch Chopper assembly, the predicted assembly times and the percent error with respect to the target time are shown in Table 6.12. The cells in the table are shaded to illustrate the level of accuracy for the different tests; green shading indicates that the values returned are within the +/- 25% tolerance range and the yellow shading would indicate that the values are within the +/- 50% tolerance range.

Table 6.12: Mate sensitivity analysis for One Touch Chopper

Student	One Touch Chopper Target Time (s)	# of Mates	Predicted Time from Loaded Training Set	% Error (+/-)	Analysis Time (s)
Student 2	316.62	89	336.91	+6	65
Student 6		90	357.1	+11	67
Student 7		91	322.17	+2	68
Student 8		104	325.07	+3	65
Student 9		86	352.57	+10	64

The percent errors of the five mating configurations for the One Touch Chopper ranged from +2% to +11% with the average of the absolute values being 6% error. The analysis time to predict the assembly times was less than sixty eight seconds excluding the time required to open up and initialize Matlab. The target assembly time for the One Touch Chopper was predicted using a Virtual Boothroyd Dewhurst DFA analysis which took 8,160 seconds (136 minutes) to complete.

A total summary of the products each student mated and the respective percent errors of the predicted assembly times are shown in Table 6.13.

Table 6.13: Summary of % errors for each student for each product

Student	Solar Yard Light % Error (+/-)	Black & Decker Drill % Error (+/-)	One Touch Chopper % Error (+/-)
Student 1	+1	+8	NA
Student 2	-16	NA	+6
Student 3	-45	NA	NA
Student 4	-6	-1	NA
Student 5	-11	+14	NA
Student 6	NA	+21	+11
Student 7	+12	+18	+2
Student 8	+2	+0.3	+3
Student 9	-20	+16	+10
Student 10	-3	+11	NA
Student 11	-16	NA	NA

All of the percent errors shown in Table 6.13 are within +/- 45% error of the target assembly times for the given product putting them within the +/-50% Boothroyd tolerance range. If you remove the predicted assembly time for Student 3's Solar Yard Light the range of percent errors drops to +/- 21% error. It should also be noted that the highest percent errors for the Black & Decker Drill and the One Touch Chopper were from both from Student 6 who had a medium level of SW assembly experience and a high SW assembly usage frequency. Since the percent errors shown in Table 6.13 do not significantly vary across the three products, this suggests that the automated tool performs well for the variety of test products used in this study which were summarized in Table 6.8. This preliminary study is, admittedly, not statistically significant, but it does illustrate the potential insensitivity of the tool to designer choice for mating approaches.

To effectively draw conclusions about a specific student's percent errors, all of the students should have added mates to all of the products. Every assembly model

would then have to be individually opened to investigate which types of mates were used. This information along with the mate connection complexity vector could be used to investigate what types of mates have the most significant effect on the predicted assembly time.

The results from this mate sensitivity study indicate that different mating configurations and the number of mates used to constrain a product do not significantly affect the predicted assembly times. With a small sample size of only eleven students across three products, all of the predicted assembly times were predicted with as little as 1% error but they were also predicted with as much as 45% error. Even though the mating configuration provided a range of results they were within the +/- 50% tolerance range. The results show that the predicted assembly time will vary between mating configurations but in all cases but one, the percent errors were within +/-21% which is an acceptable range. The range of percent errors within this mate sensitivity study is generally less than those found in the training case investigation, indicating that the training case will more generally govern the accuracy of the predicted assembly time over a variation in mating schemes. To understand what mating configurations increase or decrease the accuracy of the predicted assembly time a more detailed mate sensitivity study should be conducted; this is reserved for future work.

6.4 Evaluation and Comparison of Automated Assembly Time Prediction Tool

The purpose of this research is to develop an automatic assembly time prediction tool to address the issues with existing DFA methods summarized in Section 1.3. To determine if the automatic assembly time prediction tool addresses these issues it must be

evaluated. Chapter 4. discusses a DFA evaluation and comparison to identify the overall effectiveness and ability to automate two DFA methods, the original Connectivity Complexity method and the Boothroyd Dewhurst DFMA software [1]. This evaluation and comparison is used in this section to examine the effectiveness of the automated assembly time prediction tool.

The DFA evaluation presented in Chapter 4. consists of analyzing five aspects of the method in question:

- The approximate time required to complete the analysis
- The predicted assembly times for each product
- The amounts and types of information required by the user to complete the analysis
- The method's repeatability/subjectivity
- The method's features for redesign support

The automated assembly time prediction tool was evaluated based on each of these criteria to determine its overall effectiveness, to compare it to existing DFA methods, and to determine if it addresses the issues with existing DFA methods.

The results from the evaluation of the automated assembly time prediction tool are summarized in Table 6.14.

Table 6.14: Automated assembly time prediction tool evaluation criterion summary

Evaluation Criteria	Evaluation Results	Justification
Satisfaction with approximate analysis time	High Satisfaction	Analysis takes less than 5 minutes
Predicted assembly times	Varying accuracy (but generally within the B&D admitted range)	Depends on ANN training set: Best case 4% error Worst case 68% error
Amounts/types of information	0	Requires no additional inputs from user
Repeatability/ subjectivity	0% Subjective	Repeatable, and consistent
# of Features for redesigns	0	Currently provides no redesign features

Each evaluation criteria is addressed specifically in the following sub sections.

6.4.1 Approximate Analysis Time

Since the tool being evaluated is fully automated the time to complete the analysis is purely based on computation time. If a new ANN training scheme is developed based on the method presented in Section 5.4, the amount of training time could take several hours. If only a small set of selected ANN architectures are being trained, the training time can be significantly reduced. The fully automated assembly time prediction tool loads previously trained ANNs that only use the five architectures chosen in Section 6.2, greatly reducing the computation time.

By using these techniques the automated assembly time prediction tool can predict the assembly time of a SolidWorks assembly model within a few minutes. The majority of the time required to predict the assembly time is attributed to opening and initializing Matlab, which takes around two minutes. Once Matlab has been opened the

assembly time for all of the models tested in the previous sections took less than 72 seconds. The total analysis time is less than a few minutes, which equates to a high level analysis time satisfaction based on the original evaluation, Section 4.6.2. Analysis times measured in hours equate to medium levels of satisfaction and times measured in days equate to low levels of analysis time satisfaction.

6.4.2 Accuracy of Predicted Assembly Times

The accuracy of the assembly times predicted with the automated tool are defined with respect to the target times used to train the ANNs. Since the actual assembly times of the products used for training and testing were not known, the manual Boothroyd Dewhurst DFA tables were used to predict the assembly times of the physical products or the virtual products. The accuracy of the automated assembly time prediction tool is measured by its effectiveness to predict a time that would equal the Boothroyd Dewhurst predicted time.

The overall accuracy of the automated tool is undetermined since it ultimately depends on the training case used (Section 6.3.1) and the variability due to different mating schemes (Section 6.3.2). The variability due to training case usage was explored in the Electric Knife test (Section 6.3.1). The automated assembly time prediction tool was used to predict an assembly time for the Electric Knife assembly model using all eight training cases, Table 6.7. The percent error of the predicted assembly time varied from 68% to 4% depending on the training case. Based on this investigation it was determined that using larger sets with all unique training inputs, like training Case 7, would generally yield the best results.

6.4.3 Amounts and Types of User Inputs

The automated assembly time prediction tool only requires the user to open a SW assembly model that already contains specified mates and click the assembly time prediction tool add-in button. The tool then extracts the mates from the assembly and uses them to predict the assembly time of the SW assembly. Therefore, the tool does not require any inputs from the user that are not available directly from the assembly models.

6.4.4 Repeatability / Subjectivity and Features for Redesign

Section 4.6.4 evaluates the repeatability of a method by comparing the output predicted assembly times when the same analysis is conducted by different designers. The repeatability of the tool is defined by its ability to generate the same output when given the same input. Since the only information input required by the user to complete the analysis is an assembly model, the automated assembly time prediction tool is repeatable. If multiple designers open up the same assembly model and run the add-in then the same assembly time will be predicted. To illustrate the repeatability, the automated add-in tool was repeated five times on the Durabrand Hand Mixer and resulted with the exact same predicted assembly times but with different analysis times, Table 6.15.

Table 6.15: Repeatability of automated assembly time prediction tool

Durabrand Hand Mixer Target Time (s)	Training Case 6: Predicted Time (s)	% Error (+/-)	Analysis Time (s)
263.21	389.8946	32	85.5
	389.8946	32	73.4
	389.8946	32	73.9
	389.8946	32	76.0
	389.8946	32	76.3

This table shows that the analysis is different every time but that the results are not. For other DFA methods, since the analysis requires the user to answer subjective questions to complete the analysis, the results will vary between users.

Currently the automated assembly time prediction tool does not provide any features to aid the designer in redesigning to improve the product with regards to assembly.

6.4.5 Comparison of automated assembly time prediction tool

Table 6.16 summarizes the evaluation results for the automated assembly time prediction tool along with the original Connectivity Complexity and the Boothroyd Dewhurst DFMA software evaluation results from Section 4.7.

Table 6.16: Comparison summary of DFA methods

Evaluation Criteria	DFMA Results	Connectivity DFA results	Automated Assembly Time Prediction Tool
Approximate analysis time	Medium	Medium	High
Predicted assembly times	Baseline	Not accurate	Varying accuracy
Amounts/types of information	8 types, 49 questions, 16 subjective	5 types, 0 subjective	0
Repeatability/ subjectivity	33% Subjective	0% Subjective	0% Subjective
# of Features for redesigns	11	0	0

Based on the comparison in Table 6.16 the automated assembly time prediction tool has benefits over the Boothroyd Dewhurst DFMA software and the original Connectivity Complexity method. The automated tool takes less than five minutes to predict an assembly time where the other two methods require analyses times measured in hours. The accuracy of the predicted assembly time using the automated tool varies depending on the training case used with the program. For certain training schemes the predicted assembly time had as little as 4% error which is considered accurate, but for other training schemes the predicted time had as much as 68% error which is not accurate. In almost all cases tested, the predicted assembly time was within +/- 50% of the target value which is within the +/- 50% specified tolerance that could exist using the Boothroyd Dewhurst method [30]. The automated tool does show an improvement over the original Connectivity method which was identified as not being accurate in Section 4.7.

One of the major improvements that the automated tool has over the DFMA software and the original Connectivity method is that it requires no extra user inputs to complete the analysis. A SolidWorks assembly that has already been mated can be opened and the assembly time can be predicted by clicking on the developed assembly time prediction add-in. The Boothroyd DFMA software requires the user to answer extensive amounts and types of information to complete the analysis. This is where the Boothroyd method becomes tedious and time consuming which is not desired. Since the new tool is automated it requires no extra from the designer to complete the analysis. Since the automated method requires no extra user inputs it is completely repeatable between designers. The current version of the automated assembly time prediction tool does not offer any suggestions for redesign; this is reserved for future work. Overall Effectiveness of Developed DFA Tool

6.5 Overall Effectiveness of Developed DFA Tool

The literature review in Chapter 1. identifies a set of limitations that existing DFA methods have and it suggests that to eliminate these issues, automated methods or tools must be developed. This thesis has focused on developing an automated assembly time prediction tool to address these issues. Chapter 6. presents the developed fully automated assembly time prediction tool that extracts the required information from SolidWorks assembly models to complete the analysis, but does this tool address the third research question of this thesis:

RQ3:	Does the automated method solve the issues that the previous methods have: time consuming, repeatability, ease of use, etc?

The existing DFA issues identified in Chapter 1. are summarized again in Table 6.17 along with whether the automated assembly time prediction tool addresses the issue which is highlighted in green, does not address the issue which is highlighted in red, or partially addresses the issue which is highlighted in yellow.

Table 6.17: Does the automated tool address existing DFA issues

Issues	Issue Addressed?	Justification
Requires subjective or implicit user inputs	Yes	Requires no user inputs to complete the analysis
Tedious	Yes	Requires no extra effort from the user to complete the analysis
Time consuming	Yes	Predicts an assembly time within a few minutes
Extensive user inputs	Yes	Requires no user inputs to complete the analysis
Requires design details (geometry, etc.)	Yes & No	Addressed: finalized part features are not required to mate two parts Not Addressed: some sort of part representation must be created
Reactive or redesign tools	Yes & No	Addressed: if design process uses solid modeling assemblies concurrently Not Addressed: if design is finalized and then modeled
Stand alone systems	Yes	Automated tool is integrated into existing solid modeling software
Implicitly identified design improvements	No	Automated tool does not identify suggestions for redesign
Lack foundation to relate DFA time and cost to part geometry	Yes & No	Addressed: tool provides foundation to relate assembly mate connections to assembly time Not Addressed: Doesn't relate directly time specifically to geometry

The automated assembly time prediction tool specifically addressed five of the nine issues, partially addressed three of the nine issues, and did not answer one of the nine issues listed in Table 6.17. The first four issues in Table 6.17 are partially addressed

in Section 6.4.5 where the results from the DFA comparison identifies that the automated assembly time prediction tool only requires a few minutes to predict an assembly time and does not require additional inputs from the user which addresses the subjectivity and the tediousness of the analysis.

The three issues that the automated assembly time prediction tool partially addresses depend on how the analysis is approached. The automated tool does not require geometric details to complete the analysis but it does require mate connections to be specified to complete the analysis. These mate connections could be as simple as black box representations of how the model would be constrained and do not require exact geometric information to be specified.

The automated tool can be used as a concurrent or a redesign tool depending on the specific application. The presentation of the tool in this thesis only tested used the tool on fully assembled models, as a redesign tool, and did not investigate its use as a concurrent tool. Since the automated tool only requires mates to predict an assembly time, the tool could be used as designers start designing assemblies within solid modeling software. They can start evaluating the predicted assembly time as they go through the process. Future versions of this tool could display the predicted assembly time in real time, so as the designer adds parts and mates the assembly time would be updated on the screen. If a specific sub assembly is added that increases the assembly time significantly then it could be investigated for design improvements. Even though the tool can be used concurrently, its effectiveness will eventually depend on its ability to identify suggestions for redesign which is the DFA issue that this tool does not address. The automated tool

currently does not offer any suggestion for redesign but this will be investigated in future work.

The research hypothesis for the third research question was:

RQ3 Hypothesis:	The automated method addresses the issues that current DFA methods have.

Based on the DFA issues presented in Table 6.17, the automated assembly time prediction tool addresses all of the existing issues except for providing suggestions for redesign. By addressing these issues, the negative effects that each issue has on the implementation of DFA is reduced. With the automated assembly time prediction tool, designers will be more likely to implement the tool throughout the design process improving the design with respect to assembly.

CHAPTER 7. CONCLUSIONS AND FUTURE WORK

The development and implementation of an automated assembly time prediction tool that extracts mates from SolidWorks assemblies and uses them to predict a product assembly time was presented in this thesis. Chapter 1. surveyed current DFA methods and tools to identify the current limitations and issues that reduce their effectiveness. From this review, it was determined that a fully automated DFA method is required. Chapter 2. investigated previous research attempts that focused on automating existing methods. It was determined that most methods could never be fully automated because they require some type of subjective user inputs. Based on this information it was determined that a truly automated DFA method or tool should be developed so that it can be effectively implemented throughout the design process. To develop this tool three research questions were identified in Chapter 3. and successfully addressed in Chapter 4. , Chapter 5. , and in Chapter 6. , resulting in the automated assembly time prediction tool. These specific research questions and the respective research contributions are summarized in Section 7.1. The limitations and future work with the presented research are covered in Section 7.2.

7.1 Research Contributions

The first research question evaluated two assembly time prediction methods, the Boothroyd Dewhurst DFMA software and the Connectivity Complexity method, to determine which one should be automated based on the amounts and types of information required by the user to complete the analysis. The Connectivity Complexity method was

identified for automation since it only required five types of information inputs, none of which were subjective. The major research contribution from Chapter 4. was the DFA evaluation used to compare DFA methods. The evaluation identifies the important DFA aspects like number and types of user inputs or time to conduct the analysis. This evaluation allows DFA methods to be compared for bench marking purposes and it identifies their issues so that they can be improved. This evaluation can be applied to any method to determine its overall effectiveness and its ability to be automated.

The second research question which is addressed in Chapter 5. identifies that the Connectivity Complexity method can be automated by extracting the required user inputs from solid modeling software. The major research contribution from Chapter 5. was proving that the Connectivity Complexity method could be automated. The original Connectivity Complexity method used the physical inter part connections to predict an assembly time using a trained regression analysis.

The research in Chapter 5. modified the original method so that instead of using inter part connections it uses the mate connections from three dimensional assembly models. These mate connections are automatically extracted with the developed Mate Extraction Tool. This tool extracts the mates from SolidWorks assembly models and automatically assembles the bi-partite connection tables required to complete the analysis. The complexity of these mate connection graphs is identified using a custom Matlab algorithm. Trained artificial neural networks are then used to predict an assembly time based on the complexity vector. The tool developed and used in Chapter 5. was only partially automated, but it identified that defined mates in three-dimensional

assembly models could be automatically extracted and used to predict a products assembly time.

Chapter 6. addresses the third research question, which was to determine if the identified tool addresses the issues that the existing DFA methods have. Before this research question was specifically addressed, the tool from Chapter 5. was fully automated which required investigation and selection of ANN training schemes. The first research contribution of Chapter 6. was the ANN training case investigation, which determined which types of training cases are most effective. It was determined that larger training cases that used all unique training inputs were more effective than smaller training cases that reused training inputs but matched them to different targets. The results from the training case investigation were then used to select five ANN architectures out of the 189 to be used in the automated assembly time prediction tool.

The second research contribution presented in Chapter 6. was the development and evaluation of a fully automated assembly time prediction tool. The automated assembly time prediction tool was tested and evaluated to determine its effectiveness. The results of the evaluation determine that the automated tool addresses all of the DFA issues previously identified except for one, identify suggestions for redesign. Even though the current version of the automated assembly time prediction tool does not offer suggestions for redesign, it offers major improvements over existing DFA methods. The automated assembly time prediction tool requires no additional inputs from the user to complete the analysis. Since it does not require additional inputs it is completely repeatable. The total analysis time required to predict an assembly time of a SolidWorks

assembly takes less than five minutes. Traditional methods would require a designer to manually conduct the analysis which could take up to several hours depending on the product.

The automated assembly time prediction tool does address the majority of the issues with existing DFA methods but it still has limitations and requires future work, both of which are summarized in Section 7.2.

7.2 Limitations and Future Work

Even though the automated assembly time prediction tool does address all of the identified DFA issues except for one, it still has limitations that must be addressed with future work. The limitations with this research can be broken down into three categories related to: the ANN training cases used, the mating scheme sensitivity, and the robustness of the mate extraction add-in. Each of these limitations is addressed in the following sub sections.

7.2.1 Limitation with Regards to ANN Training Cases

The research presented in Chapter 6. identified that the training case used to train the ANNs can significantly affect the results of the predicted assembly times. For example the predicted times for the Electric Knife test case ranged from -4% to +68% depending on the training case used, Table 6.7. It was recommended that *future training cases should use a set of at least eleven unique training inputs and targets that are made up of partially defined assembly models* to improve the accuracy of the predicted assembly times. The investigations into ANN training case types used to make this

recommendation are only initial studies. These studies should be continued using larger sample sizes to make more effective or specific recommendations but this is reserved for future work. Future studies should also investigate whether the test inputs are internal or external to the training sets used. Internal test inputs would be products that have part counts, component counts, and complexities within the range of the training case and external inputs would have values outside of the range of the training case.

During the development of this tool eight different training cases were evaluated to determine their effect on the predicted assembly times and to select five ANN architectures to use with the automated tool. The selection process for choosing the five ANN architectures is a repeatable method, but it may not select the overall best architectures. A formalized architecture selection process that chooses the five most effective architecture structures should be investigated in future work.

7.2.2 Limitation with Regards to Mating Sensitivity

With the initial development of the automated assembly time prediction tool presented in Chapter 5. the tools sensitivity to different mating styles was identified early on. The variability in predicted assembly times was first identified in Section 5.6 which showed that between two designers that add mates to the same assembly model the predicted assembly times could vary from -7% to + 27% error. This motivated a mate sensitivity investigation (Section 6.3.2). The second study used the fully automated version of the assembly time prediction tool along with: a more effective ANN training case, three different test products, and up to eleven different test subjects. The results showed that for a given product the % errors are within +/- 20% error for all cases except

for one outlier that had a -45% error. The mate sensitivity study presented in this thesis only evaluated the variability between different test subjects' assembly times. This research did not explore the specific effect that different mating styles have on the predicted assembly times. Further investigation into mating variability and its effect on the predicted assembly time using this tool is reserved for future work.

7.2.3 Limitation with Regards to Program Robustness

The automated assembly time prediction tool is a SolidWorks custom add-in that extracts the defined mates from an assembly model and uses the complexity of the mate connection graphs to predict an assembly time based using trained ANNs. The automated tool has successfully predicted assembly times within 1% of the target values in less than five minutes. Even though this tool has proven to be effective it still has limitations that will need to be addressed in future versions of the tool. These limitations are summarized in the following list:

- Does not extract mates from subassemblies
- Does not handle part patterns within assemblies
- Extracts suppressed mates
- Requires Matlab to perform computations

The first three limitations listed can be addressed fairly easily in future versions of this tool. By calling a few more SolidWorks API functions and adding some if statements, these programming aspects can be addressed. The fourth limitation listed, requires Matlab for computations, can also be addressed in future versions but would

require more work. The current version of the automated tool predicts an assembly time within five minutes; half of this time is attributed to opening up and initializing Matlab. If a new version of the tool can be developed that does not use Matlab it will complete the analysis faster and it could be used on any computer instead of requiring a license of Matlab to use the too.

The current program uses Matlab for the computational aspects because it is designed for computational prototyping through its built-in toolboxes, including several ANN algorithms. Ideally the automated tool would not require a separate program to complete the analysis, it would have the computational aspects currently performed by Matlab integrated into it so that it becomes a standalone automated tool. Eliminating the use of Matlab from the automated assembly time prediction tool is reserved for Future work.

7.2.4 Extendibility of Current Tool

The automated assembly time prediction tool presented in this thesis uses mate based connections within SolidWorks to predict an assembly time using trained ANNs. The trained ANN's were given complexity vectors of mate connection graphs and mapped to Boothroyd Dewhurst predicted assembly times. The ANN's determine a relationship that relates the complexity of the mate graph to the predicted assembly time, essentially eliminating any specific information captured and used by the Boothroyd Dewhurst analysis to predict the time. This may not always be advantageous, there may be information stored in the Boothroyd Dewhurst assembly times that could be captured and used to improve the automated too.

The Boothroyd Dewhurst predicted assembly times use two times to specify the total assembly time for a part or product, the handling time and the insertion time. The mate connections stored in SolidWorks do not contain information that can be related to Boothroyd Handling times but it is possible that the part constraints defined by the mate do relate to insertion times. Part constraints as defined in SW are determined by their insertion axis and connections to other parts which can be related to insertion times as defined by Boothroyd Dewhurst. Using the complexity of an assemblies mate connection graph and training it to just Boothroyd Dewhurst predicted insertion times instead of the total predicted assembly time may provide better results. A detailed study into this question is reserved for future work.

CHAPTER 8. BIBLIOGRAPHY

[1] Eric Owensby, Aravind Shanthakumar, Vikrant Rayate, Essam Namouz, and Joshua D. Summers, "Evaluation and Comparison of Two Design for Assembly Methods: Subjectivity of Information Inputs," in *ASME 2011 International Design Engineering Technical Conferences (IDETC)*, Washington DC, 2011.

[2] Anonymous, "Expert system aids design for assembly," *Assembly Automation*, vol. 9, no. 3, pp. 132-136, 1993.

[3] G.G.F Dalgleish, "Desing for assembly: Influencing the design process," *Journal of Engineering Design*, vol. 11, no. 1, pp. 17-29, 2000.

[4] Essam Namouz, Joshua Summers, and Gregory Mocko, "Source of Variability In The Boothroyd and Dewhurst Assembly Time Estimation Method," in *IDETC/CIE*, Chicago, 2012, Submitted.

[5] Michael G. Miller, Product and Process Based Assembly Time Estimation Methods In Engineering Design, December 2011.

[6] James L Mathieson, Bradley A. Wallace, and Joshua D. Summers, "Assembly Time Modeling Through Connective Complexity Metrics," in *International Conference on Manufacturing Automation*, 2010.

[7] G. Boothroyd and L Alting, "Design for Assembly and Disassembly," in *CIRP Annals-Manufacturing Technology*, 1992, pp. 625-636.

[8] Geoffrey Boothroyd, "Product design for manufacture and assembly," *Computer-Aided Design*, vol. 26, no. 7, pp. 505-520, 1994.

[9] G. Boothroyd, "Design for Manual Assembly," in *Assembly Automation and Product Design*, 2nd ed. Boca Raton, FL: CRC Press, 2005, ch. 7, pp. 219-255.

[10] G.E.M. Jared, M.G. Limage, I.J. Sherrin, and K.G. Swift, "Geometric reasoning and design for manufacture," *Computer-Aided Design*, vol. 26, no. 7, pp. 528-536, 1994.

[11] Geoffrey Boothroyd, *Design for Assembly Handbook*. Amherst, Massachusetts: University of Massachusetts, 1982.

[12] J. Laring, M. Forsman, R. Kadefors, and R Ortengren, "MTM-based Ergonomic Workload Analysis," *International Journal of Industrial Engineering*, vol. 30, pp. 135-148, January 2002.

[13] Mohd Zakaria, Design for Assembly and Application Using Hitachi Assemblability Evalaution Method, November 2009.

[14] "Expert system aids design for assembly," *Assembly Automation*, vol. 9, no. 3, pp. 132-136, 1993.

[15] Harold Maynard, G.J. Stegemerten, and John Schwab, *Methods-Time Measurement*, 1st ed., McGraw Hill Book Company, Ed. New York: Toronto: London, 1948.

[16] S Miyakawa and T Ohashi, "The Hitachi Assemblability Evaluation Method (AEM)," in *Conference on Product Design for Assembly*, NewPort, RI, 1986, pp. 15-17.

[17] E Kroll, E Lenz, and J.R. Wolberg, "A Knowledge-based Solution to the Design for Assembly Problem," *Manufacturing Review*, vol. 1, no. 2, pp. 104-108, June 1988.

[18] Susan Tate, G. Jared, Nathaniel Brown, and K.G. Swift, "An Introduction to the Designer's Sandpit," in *ASME DETC/DFM*, Baltimore, 2000, pp. 84-87.

[19] Theory and Methodology Committee on Engineering Design, Commission on Engineering and Technical Systems, and National Research Council, "Executive Summary," in *Improving Engineering Design: Design for Competitive Advantage*. Washington, D.C.: National Academy Press, 1991, pp. 1, 23, http://www.nap.edu/catalog/1774.html.

[20] Boothroyd Dewhurst and Winston Knight, "Design for Assembly," *IEEE Spectrum*, vol. 30, no. 9, pp. 53-55, September 1993.

[21] L. T. Fazio, A.C. Edsall, R. E. Gustavson, J. A. Hernandez, P. M. Hutchins, H. W. Leung, "A Prototype of Feature-Based Design for Assembly," *Journal of Mechanical Design*, vol. 114, no. 5, pp. 723-734, 1993.

[22] X.F. Zha, H.J. Du, and J.H. Qiu, "Knowledge-based approach and system for assembly oriented design, Part I: the approach," *Engineering Applications of Artificial Intelligence*, pp. 61-75, 2001.

[23] Marcos Jr. Esterman and Krishna Kamath, "Design for Assembly Line Performance: The Link Between DFA Metrics and Assembly Line Performance Metrics," in *ASME IDETC*, Montreal, Quebec, 2010.

[24] S.Y. NOF and J. Chen, "Assembly and Disassembly: An Overview and Framework for Cooperatoin Requirement Planning with Conflict Resolution," *Journal of Intelligent and Robotic Systems*, vol. 37, pp. 307-320, February 2003.

[25] C.J. Barnes, G.F. Dalgleish, G. Jared, H. Mei, and K.G. Swift, "Assembly Oriented Design," *IEEE*, pp. 44-50, July 1999.

[26] Gerard J. Kim, George Bekey, and Kenneth Goldberg, "A Shape Metric for Design for Assembly," *IEEE*, vol. 2, pp. 968-973, May 1992.

[27] Robert Stone, Daniel McAdams, and Varghese Kayyalethekkel, "A Product Architecture-based Conceptual DFA Technique," *Design Studies*, vol. 25, no. 3, pp. 301-325, May 2003.

[28] O. Coma, C. Mascle, and P. Veron, "Geometric and form feature recognition tools applied to a design for assembly methodology," *Computer-Aided Design*, vol. 35, no. 13, pp. 1193-1210, February 2003.

[29] Eric Owensby, Essam Namouz, Aravind Shanthakumar, and Joshua Summers, "Representation: Extracting Mate Complexity From Assembly Models to Automatically Predict Assembly Times," in *IDETC/CIE*, Chicago, 2012, Submitted.

[30] Geoffrey Boothroyd, Peter Dewhurst, and Winston Knight, *Product Design for Manufacture and Assembly*, 3rd ed. Boca Raton, USA/FL.: CRC Press, 2007.

[31] D.D. Sanders, "An expert system for automatic design for assembly," *Assembly Automation*, vol. 29, no. 4, pp. 378-388, 2009.

[32] P. Dewhurst and G. Boothroyd, "Computer-Aided Design For Assembly," *Assembly Engineering*, pp. 18-22, 1983.

[33] G. Pahl, W. Beitz, J. Feldhusen, and K.H. Grote,.: Springer, 2007, ch. 4, pp. 128-134.

[34] Wynne Hsu, Jerry Fuh, and Yunfeng Zhang, "Synthesis of Design Concepts from a Design for Assembly Perspective," *Computer Integrated Manufacturing Systems*, vol. 11, pp. 1-13, 1998.

[35] DE Whitney, "Designing the Design Process," *Research in Engineering Design*, vol. 2, pp. 3-13, 1990.

[36] Boothroyd Dewhurst Inc. (2012, February) DFMA. [Online]. http://www.dfma.com/resources/studies.htm

[37] Lotfi A. Zadeh, "Fuzzy Logic, Neural Networks, and Soft Computing," *Communications of the ACM*, vol. 37, no. 3, pp. 77-84, March 1994.

[38] "Design for Assembly - the forgotten factor in automation," *Engineers' Digest*, vol. 45, no. 4, pp. 11-13, 1984.

[39] M. A. Kurfman, Michael Stock, and Robert Stone, "Experimental Studies Assessing the Repeatability of a Functional Modeling Derivation Method," *ASME Journal of Mechanical Design*, vol. 125, no. 4, pp. 682-693, December 2003.

[40] Carlos Rodriguez-Toro, Graham Jared, and Kenneth Swift, "Product-development complexity metrics: A framework for proactive-DFA implementation," in *International Design Conference-Design*, Dubrovnik, 2004, pp. 483-490.

[41] K. G. Swift, *Knowledge-based Design for Manufacture*. Englewood Cliffs, N.J.: Prentice-Hall, 1987.

[42] Ken Swift and Graham Jared. The University of HULL. [Online].

http://www.hullhistorycentre.org.uk/discover/mapp/sandpit.aspx

[43] Swee Mok, Chi-haur Wu, and D.T. Lee, "A System for Analyzing Automatic Assembly and Disassembly Operations," in *IEEE*, San Francisco, CA, 2000, pp. 3695-3700.

[44] Hayes-Roth, Frederick; Teknowledge, Inc., "The Knowledge-Based Expert System: A Tutorial," *IEEE*, vol. 17, no. 9, pp. 11-28, September 1984.

[45] J. Mathieson and J. Summers, "Complexity Metrics for Directional Node-Link System Representations: Theory and Applications," in *ASME International Design Engineering Technical Conferences*, vol. DTM, Montreal, Canada, 2010, pp. DETC2010-28561.

[46] J. Mathieson, C. Sen, and J. Summers, "Information Generation through the Design Process: Student Project Case Study," in *ASME Design Engineering Technical Conferences, , Aug. 30-Sep. 2, 2009, -.*, vol. CIE, San Diego, CA, 2009, pp. DETC2009-87359.

[47] J. Summers and J. Shah, "Mechanical Engineering Design Complexity Metrics: Size, Coupling, and Solvability," *Journal of Mechanical Design*, vol. 132, no. 2, p. 021004, February 2010.

[48] F. Ameri, J. Summers, G. Mocko, and M. Porter, "Engineering Design Complexity: An Experimental Study of Methods and Measures," *Research in Engineering Design*, vol. 19, no. 2-3, pp. 161-79, 2008.

[49] Holly K. Ault, "3-D Geometric Modeling for 21st Century," *Engineering Design Graphics Journal*, vol. 63, no. 2, pp. 33-42, 1999.

[50] J Summers, B Bettig, and J Shah, "The Design Exemplar: A New Data Structure for Embodiment Design Automation," *Journal of Mechanical Design*, vol. 126, no. 5, pp. 775-87, 2004.

[51] J Summers, A Divekar, and S Anandan, "Towards Establishing the Design Exemplar as a CAD Query Language," *Computer-Aided Design and Applications*, vol. 3, no. 1-4, pp. 523-532, 2006.

[52] Dassult Systems. (1995-2012) SolidWorks Help. [Online]. http://help.solidworks.com/2010/english/SolidWorks/SWHelp_List.html?id=44d1d06dc95b47c9944c5fa8544040fa

[53] William C. Regli. (2012, January) Geometric and Intelligent Computing Laborator. [Online]. http://gicl.cs.drexel.edu/wiki/Paper_Pro

[54] Dassault Systems. (2012) 3D Content Central. [Online]. http://www.3dcontentcentral.com/Default.aspx

[55] Jack V. Tu, "Advantages and Disadvantages of Using Artificial Neural Networks versus Logistic Regression for Predicting Medical Outcomes," *Journal of Clinical Epidemiology*, vol. 49, no. 11, p. 1225, November 1996.

[56] SoftScout. (2012, February) SoftScout: The Business Software Encyclopedia. [Online]. http://www.softscout.com/software/Engineering/Mechanical-Engineering/TeamSET.html

APPENDICES

Appendix A. ANN Training Test Cases

The specific test case used to train and test ANN training Case 1 is presented below. The first table shows the products withheld from training to test the trained ANN. The second table shows the ANN training inputs and respective targets. The inputs listed in both tables are the names of the products but the actual inputs used to train and test the ANNs are the complexity vector that matches that product. The complexity vector that matches the given input can be found by matching the product name and the product definition with the respective one listed in Appendix B.

ANN Training Test Case 1: Fully Defined Assembly Models

ANN Case 1 Test Inputs

Test Input	Target Time (s)	DFA Used for Target Time	Product Definition
Pencil Compass	68.38	Physical	Fully Defined
MagLight	75.4	Virtual	Fully Defined
Black & Decker Drill	189.65	Virtual	Fully Defined

ANN Training Case 1 Inputs and Respective Targets

Complexity Vector Training Input for:	Training Target Time (s)	DFA Used for Target Time	Product Definition
G2 Pen	36.4	Physical DFA	Fully Defined
G2 Pen	34.4	Virtual DFA	Fully Defined
Solar Yard Light	131.23	Physical DFA	Fully Defined
Solar Yard Light	128.79	Virtual DFA	Fully Defined
Paper Pro	135.06	Physical DFA	Fully Defined
Paper Pro	123.51	Virtual DFA	Fully Defined
InDoor Electric Grill	121.08	Virtual DFA	Fully Defined
Pony Vise	153.3	Physical DFA	Fully Defined
Pony Vise	143.69	Virtual DFA	Fully Defined
OEM 825 Shift Frame	313.7	Virtual DFA	Fully Defined
OEM 825 Wide Flag	58.33	Virtual DFA	Fully Defined

ANN Training Test Case 2: Partially Defined Assembly Models

ANN Case 2 Test Inputs

Test Input	Target Time (s)	DFA Used for Target Time	Product Definition
Pencil Compass	68.38	Physical	Partially Defined
MagLight	75.4	Virtual	Partially Defined
Black & Decker Drill	189.65	Virtual	Partially Defined

ANN Training Case 2 Inputs and Respective Targets

Complexity Vector Training Input for:	Training Target Time (s)	DFA Used for Target Time	Product Definition
G2 Pen	36.4	Physical DFA	Partially Defined
G2 Pen	34.4	Virtual DFA	Partially Defined
Solar Yard Light	131.23	Physical DFA	Partially Defined
Solar Yard Light	128.79	Virtual DFA	Partially Defined
Paper Pro	135.06	Physical DFA	Partially Defined
Paper Pro	123.51	Virtual DFA	Partially Defined
InDoor Electric Grill	121.08	Virtual DFA	Partially Defined
Pony Vise	153.3	Physical DFA	Partially Defined
Pony Vise	143.69	Virtual DFA	Partially Defined
OEM 825 Shift Frame	313.7	Virtual DFA	Partially Defined
OEM 825 Wide Flag	58.33	Virtual DFA	Partially Defined

ANN Training Case 3: Combination of Fully and Partially Defined Assembly Models

ANN Case 3 Test Inputs

Test Input	Target Time (s)	DFA Used for Target Time	Product Definition
Pencil Compass	68.38	Physical	Fully Defined
MagLight	75.4	Virtual	Fully Defined
Black & Decker Drill	189.65	Virtual	Fully Defined
Pencil Compass	68.38	Physical	Partially Defined
MagLight	75.4	Virtual	Partially Defined
Black & Decker Drill	189.65	Virtual	Partially Defined

ANN Training Case 3 Inputs and Respective Targets

Complexity Vector Training Input for:	Training Target Time (s)	DFA Used for Target Time	Product Definition
G2 Pen	36.4	Physical DFA	Fully Defined
G2 Pen	34.4	Virtual DFA	Fully Defined
Solar Yard Light	131.23	Physical DFA	Fully Defined
Solar Yard Light	128.79	Virtual DFA	Fully Defined
Paper Pro	135.06	Physical DFA	Fully Defined
Paper Pro	123.51	Virtual DFA	Fully Defined
InDoor Electric Grill	121.08	Virtual DFA	Fully Defined
Pony Vise	153.3	Physical DFA	Fully Defined
Pony Vise	143.69	Virtual DFA	Fully Defined
OEM 825 Shift Frame	313.7	Virtual DFA	Fully Defined
OEM 825 Wide Flag	58.33	Virtual DFA	Fully Defined
G2 Pen	36.4	Physical DFA	Partially Defined
G2 Pen	34.4	Virtual DFA	Partially Defined
Solar Yard Light	131.23	Physical DFA	Partially Defined
Solar Yard Light	128.79	Virtual DFA	Partially Defined
Paper Pro	135.06	Physical DFA	Partially Defined
Paper Pro	123.51	Virtual DFA	Partially Defined
InDoor Electric Grill	121.08	Virtual DFA	Partially Defined
Pony Vise	153.3	Physical DFA	Partially Defined
Pony Vise	143.69	Virtual DFA	Partially Defined
OEM 825 Shift Frame	313.7	Virtual DFA	Partially Defined
OEM 825 Wide Flag	58.33	Virtual DFA	Partially Defined

ANN Training Case 4: Fully and Partially Defined Assembly Models Filtered Set

Training Case 4 is designed to be compared with the first three training cases. It can be compared with Case 1 and Case 2 to determine if a combination of fully and partially defined models performs better than just one definition type. It can be compared to training Case 3 to determine if the size of a training case affects its results.

ANN Case 4 Test Inputs

Test Input	Target Time (s)	DFA Used for Target Time	Product Definition
Pencil Compass	68.38	Physical	Fully Defined
MagLight	75.4	Virtual	Fully Defined
Black & Decker Drill	189.65	Virtual	Fully Defined
Pencil Compass	68.38	Physical	Partially Defined
MagLight	75.4	Virtual	Partially Defined
Black & Decker Drill	189.65	Virtual	Partially Defined

ANN Training Case 4 Inputs and Respective Targets

Complexity Vector Training Input for:	Training Target Time (s)	DFA Used for Target Time	Product Definition
G2 Pen	36.4	Actual	Fully Defined
Solar Yard Light	128.79	Virtual	Fully Defined
Paper Pro	123.51	Virtual	Fully Defined
Pony Vise	153.3	Actual	Fully Defined
OEM 825 Shift Frame	313.7	Virtual	Fully Defined
G2 Pen	34.4	Virtual	Partially Defined
Solar Yard Light	131.23	Actual	Partially Defined
Paper Pro	135.06	Actual	Partially Defined
InDoor Electric Grill	121.08	Virtual	Partially Defined
Pony Vise	143.69	Virtual	Partially Defined
OEM 825 Wide Flag	58.33	Virtual	Partially Defined

ANN Training Case 5: Fully and Partially Defined Assembly Models Filtered Set Two

Training Case 5 is designed to be compared with the first three training cases. It can be compared with Case 1 and Case 2 to determine if a combination of fully and partially defined models performs better than just one definition type. It can be compared to training Case 3 to determine if the size of a training case affects its results.

ANN Case 5 Test Inputs

Test Input	Target Time (s)	DFA Used for Target Time	Product Definition
Pencil Compass	68.38	Physical	Fully Defined
MagLight	75.4	Virtual	Fully Defined
Black & Decker Drill	189.65	Virtual	Fully Defined
Pencil Compass	68.38	Physical	Partially Defined
MagLight	75.4	Virtual	Partially Defined
Black & Decker Drill	189.65	Virtual	Partially Defined

ANN Training Case 5 Inputs and Respective Targets

Complexity Vector Training Input for:	Training Target Time (s)	DFA Used for Target Time	Product Definition
G2 Pen	34.4	Virtual	Fully
Solar Yard Light	131.23	Actual	Fully
Paper Pro	135.06	Actual	Fully
InDoor Electric Grill	121.08	Virtual	Fully
Pony Vise	143.69	Virtual	Fully
OEM 825 Wide Flag	58.33	Virtual	Fully
G2 Pen	36.4	Actual	Partially
Solar Yard Light	131.23	Actual	Partially
Paper Pro	123.51	Virtual	Partially
Pony Vise	153.3	Actual	Partially
OEM 825 Shift Frame	313.7	Virtual	Partially

ANN Training Case 6: Partially Defined Assembly Models

Training Case 6 is designed to determine the effect of product variability within the training set on the results. Since a larger product sample size was available for this training case, none of the training inputs will be the same but the training case size will be approximately the same.

ANN Case 6 Test Inputs

Test Input	Target Time (s)	DFA Used for Target Time	Product Definition
Paper Pro Stapler	123.51	Virtual	Partially Defined
6" MagLight	75.4	Virtual	Partially Defined
Durabrand Blender	263.21	Virtual	Partially Defined

ANN Training Case 6 Inputs and Respective Targets

Complexity Vector Training Input for:	Training Target Time (s)	DFA Used for Target Time	Product Definition
G2 Pen	34.4	Virtual	Partially Defined
Pencil Compass	69.33	Virtual	Partially Defined
Indoor Electric Grill Model	121.08	Virtual	Partially Defined
Solar Yard Light	128.79	Virtual	Partially Defined
Pony Vise	143.69	Virtual	Partially Defined
Black and Decker Drill	189.65	Virtual	Partially Defined
OEM 825 Shift Frame LH	313.7	Virtual	Partially Defined
OEM 825 Wide Flag	58.33	Virtual	Partially Defined
One Touch Chopper	316.62	Virtual	Partially Defined
Mouse Model	81.25	Virtual	Partially Defined
Boothroyd Piston Assembly	48.01	Virtual	Partially Defined
Hole Punch	145.38	Virtual	Partially Defined

ANN Training Case 7: Partially Defined Assembly Models

Training Case 7 is designed to determine the effect of product variability within the training set on the results. Since a larger product sample size was available for this training case, none of the training inputs will be the same but the training case size will be approximately the same.

ANN Case 7 Test Inputs

Test Input	Target Time (s)	DFA Used for Target Time	Product Definition
Solar Yard Light	128.79	Virtual	Partially Defined
Black and Decker Drill	189.65	Virtual	Partially Defined
One Touch Chopper	316.62	Virtual	Partially Defined

ANN Training Case 7 Inputs and Respective Targets

Complexity Vector Training Input for:	Training Target Time (s)	DFA Used for Target Time	Product Definition
G2 Pen	34.4	Virtual	Partially Defined
Pencil Compass	69.33	Virtual	Partially Defined
Indoor Electric Grill Model	121.08	Virtual	Partially Defined
Paper Pro Stapler	123.51	Virtual	Partially Defined
Pony Vise	143.69	Virtual	Partially Defined
6" MagLight	75.4	Virtual	Partially Defined
OEM 825 Shift Frame LH	313.7	Virtual	Partially Defined
OEM 825 Wide Flag	58.33	Virtual	Partially Defined
Durabrand Blender	263.21	Virtual	Partially Defined
Mouse Model	81.25	Virtual	Partially Defined
Boothroyd Piston Assembly	48.01	Virtual	Partially Defined
Hole Punch	145.38	Virtual	Partially Defined

ANN Training Case 8: Partially Defined Assembly Models

Training Case 8 is designed to determine the effect of product variability within the training set on the results. Since a larger product sample size was available for this training case, none of the training inputs will be the same but the training case size will be approximately the same.

ANN Case 8 Test Inputs

Test Input	Target Time (s)	DFA Used for Target Time	Product Definition
Pencil Compass	68.38	Virtual	Partially Defined
MagLight Virtual	75.4	Virtual	Partially Defined
Black & Decker Drill Virtual	189.65	Virtual	Partially Defined

ANN Training Case 8 Inputs and Respective Targets

Complexity Vector Training Input for:	Training Target Time (s)	DFA Used for Target Time	Product Definition
G2 Pen	34.4	Virtual	Partially Defined
Solar Yard Light	128.79	Virtual	Partially Defined
Indoor Electric Grill Model (Grab CAD)	121.08	Virtual	Partially Defined
Paper Pro Stapler	123.51	Virtual	Partially Defined
Pony Vise	143.69	Virtual	Partially Defined
One Touch Chopper	316.62	Virtual	Partially Defined
OEM 825 Shift Frame LH	313.7	Virtual	Partially Defined
OEM 825 Wide Flag	58.33	Virtual	Partially Defined
Durabrand Blender	263.21	Virtual	Partially Defined
Mouse Model	81.25	Virtual	Partially Defined
Boothroyd Piston Assembly	48.01	Virtual	Partially Defined
Hole Punch	145.38	Virtual	Partially Defined

Appendix B. ANN Training Test Cases Products

This appendix includes all of the products used to train and test the artificial neural networks required to automate the assembly time prediction tool. The product details, a picture of the product, and the complexity vector of the product are listed in this appendix. This information can be matched to the different training cases above as needed.

G2 Pen

The G2 Pen details for the partially defined assembly model are below.

G2 Pen Product and DFA Specifications

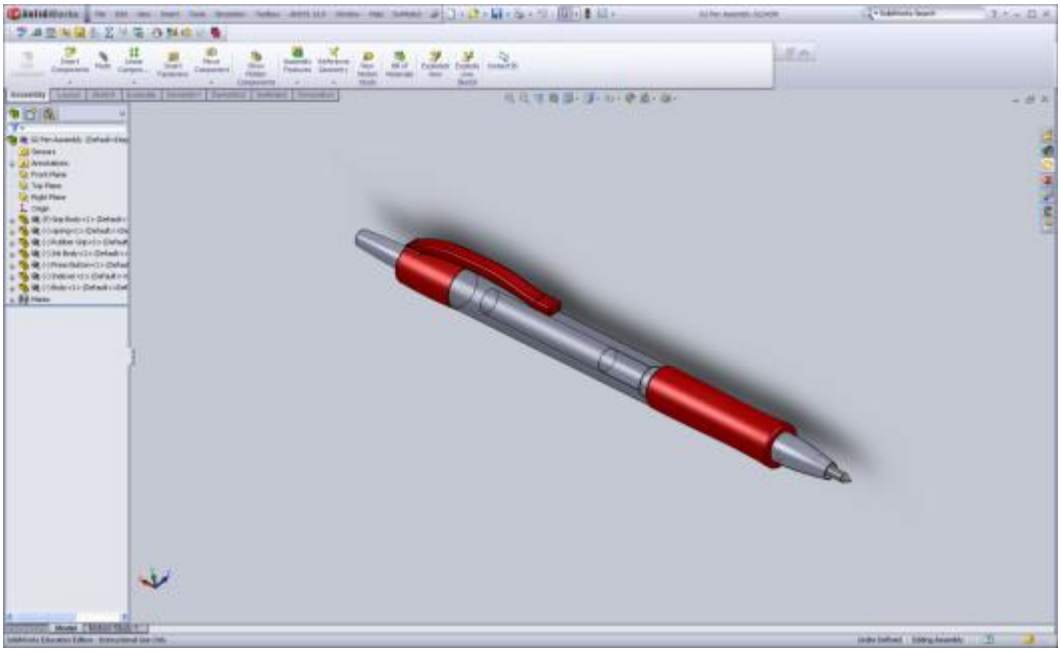
Product Name	G2 Pen
Constraint Definition	Partially Defined
# of Parts	7
# of Mates	12
Constraint Definition	Fully Defined
# of Parts	7
# of Mates	18
SW Assembly File Origin	Reverse Engineered by: Eric Owensby
Product Structure	Circular
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	36.4
Physical DFA Analysis Time (min.)	13
Boothroyd Dewhurst DFA on Virtual Product (s)	34.4
Virtual DFA Analysis Time (min.)	25

G2 Pen: Complexity Vector for Partially Defined Model

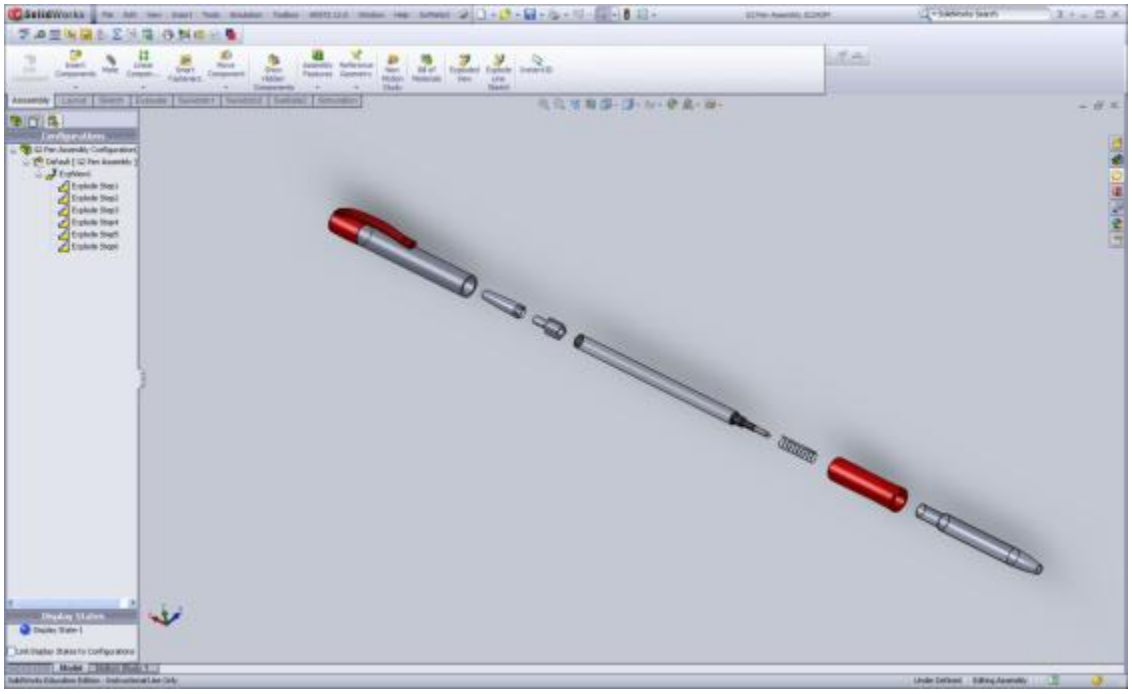
Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	7
			Relationships	12
	Connective		Degrees of Freedom	12
			Connectivity	24
Interconnection	Shortest Path		Total	72
			Maximum	3
			Average	1.714286
			Density	0.1429
	Flow Rate		Total	124
			Maximum	6
			Average	2.5306
			Density	0.2109
Centrality	Betweenness		Total	30
			Maximum	11
			Average	4.285714
			Density	0.3571
	Clustering Coefficient		Total	2.333333
			Maximum	1
			Average	0.3333
			Density	0.0278
Decomposition	Ameri-Summers			28
	Core Numbers	In	Total	14
			Maximum	2
			Average	2
			Density	0.1667
		Out	Total	14
			Maximum	2
			Average	2
			Density	0.1667

G2 Pen: Complexity Vector for Fully Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	8
			Relationships	18
	Connective		Degrees of Freedom	18
			Connectivity	36
Interconnection	Shortest Path		Total	82
			Maximum	2
			Average	1.464286
			Density	0.0813
	Flow Rate		Total	254
			Maximum	6
			Average	3.9688
			Density	0.2205
Centrality	Betweenness		Total	26
			Maximum	12.66667
			Average	3.25
			Density	0.1806
	Clustering Coefficient		Total	4.166667
			Maximum	0.666667
			Average	0.5208
			Density	0.0289
Decomposition	Ameri-Summers			45
	Core Numbers	In	Total	24
			Maximum	3
			Average	3
			Density	0.1667
		Out	Total	24
			Maximum	3
			Average	3
			Density	0.1667



G2 Pen Assembly Model



Exploded View of G2 Pen

Pencil Compass

The Pencil Compass details for the partially defined assembly model are below.

Pencil Compass Product and DFA Specifications

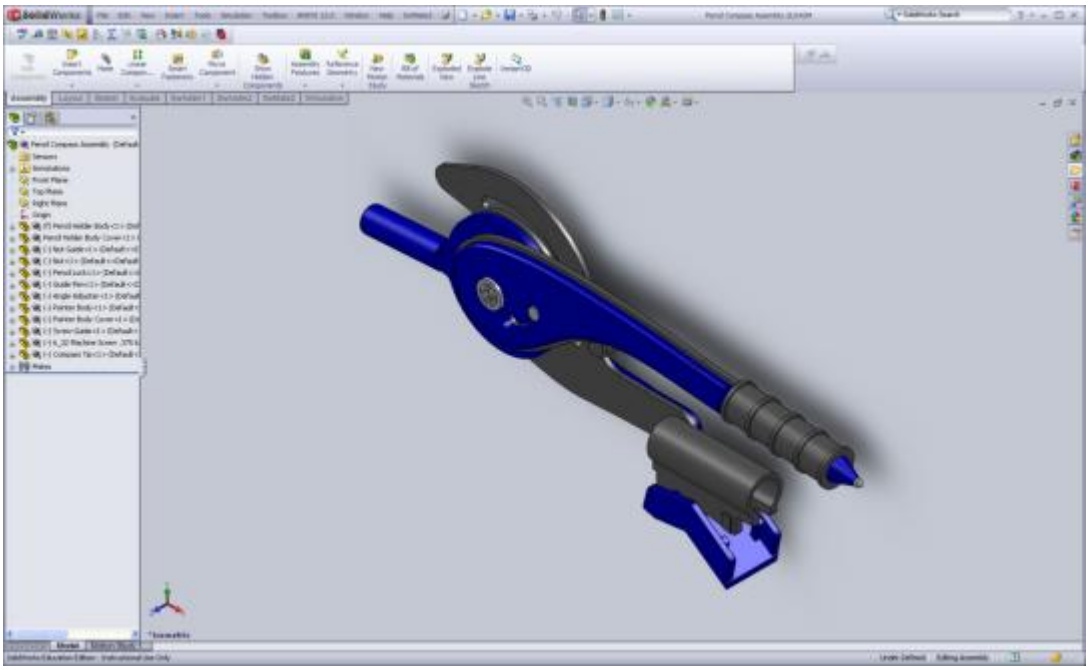
Product Name	Pencil Compass
Constraint Definition	Partially Defined
# of Parts	12
# of Mates	27
Constraint Definition	Fully Defined
# of Parts	12
# of Mates	34
SW Assembly File Origin	Reverse Engineered by: Eric Owensby
Product Structure	Stackable
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	68.38
Physical DFA Analysis Time (min.)	36
Boothroyd Dewhurst DFA on Virtual Product (s)	69.33
Virtual DFA Analysis Time (min.)	50

Pencil Compass: Complexity Vector for Partially Defined Model

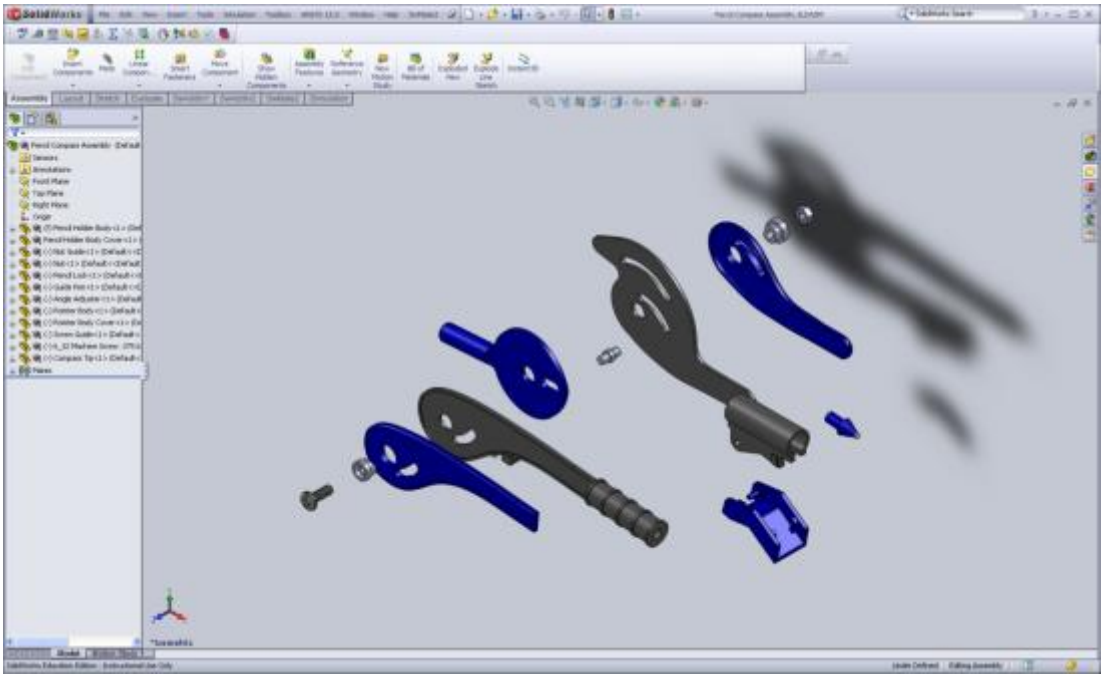
Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	13
			Relationships	27
	Connective		Degrees of Freedom	27
			Connectivity	54
Interconnection	Shortest Path		Total	390
			Maximum	5
			Average	2.5
			Density	0.0926
	Flow Rate		Total	394
			Maximum	9
			Average	2.3314
			Density	0.0863
Centrality	Betweenness		Total	234
			Maximum	60
			Average	18
			Density	0.6667
	Clustering Coefficient		Total	2.133333333
			Maximum	0.666666667
			Average	0.1641
			Density	0.0061
Decomposition	Ameri-Summers			116
	Core Numbers	In	Total	19
			Maximum	2
			Average	1.461538462
			Density	0.0541
		Out	Total	19
			Maximum	2
			Average	1.461538462
			Density	0.0541

Pencil Compass: Complexity Vector for Fully Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	15
			Relationships	34
	Connective		Degrees of Freedom	34
			Connectivity	68
Interconnection	Shortest Path		Total	486
			Maximum	4
			Average	2.314285714
			Density	0.0681
	Flow Rate		Total	688
			Maximum	10
			Average	3.0578
			Density	0.0899
Centrality	Betweenness		Total	276
			Maximum	63.83333333
			Average	18.4
			Density	0.5412
	Clustering Coefficient		Total	1.966666667
			Maximum	0.5
			Average	0.1311
			Density	0.0039
Decomposition	Ameri-Summers			134
	Core Numbers	In	Total	26
			Maximum	2
			Average	1.733333333
			Density	0.0510
		Out	Total	26
			Maximum	2
			Average	1.733333333
			Density	0.0510



Pencil Compass Assembly Model



Exploded View of Pencil Compass

Indoor Electric Grill

The Indoor Electric Grill details for the partially defined assembly model are below.

Indoor Electric Grill Product and DFA Specifications

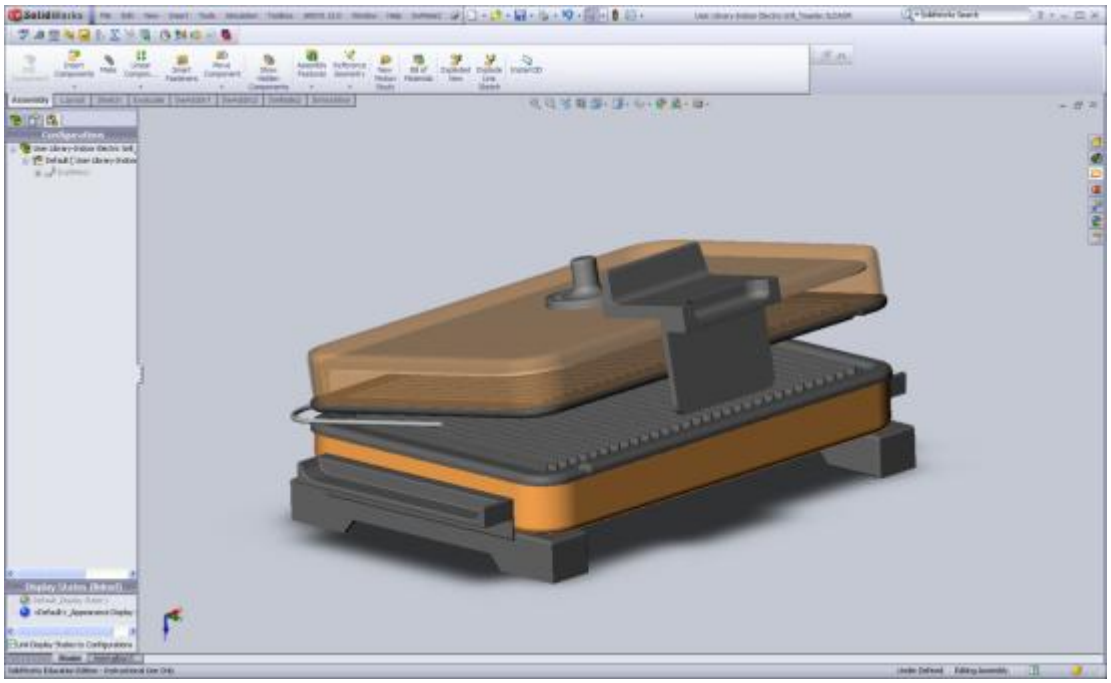
Product Name	Indoor Electric Grill
Constraint Definition	Partially Defined
# of Parts	17
# of Mates	29
Constraint Definition	Fully Defined
# of Parts	17
# of Mates	47
SW Assembly File Origin	Grab CAD
Product Structure	Combination
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	NA
Physical DFA Analysis Time (min.)	NA
Boothroyd Dewhurst DFA on Virtual Product (s)	121.08
Virtual DFA Analysis Time (min.)	85

Indoor Electric Grill: Complexity Vector for Partially Defined Model

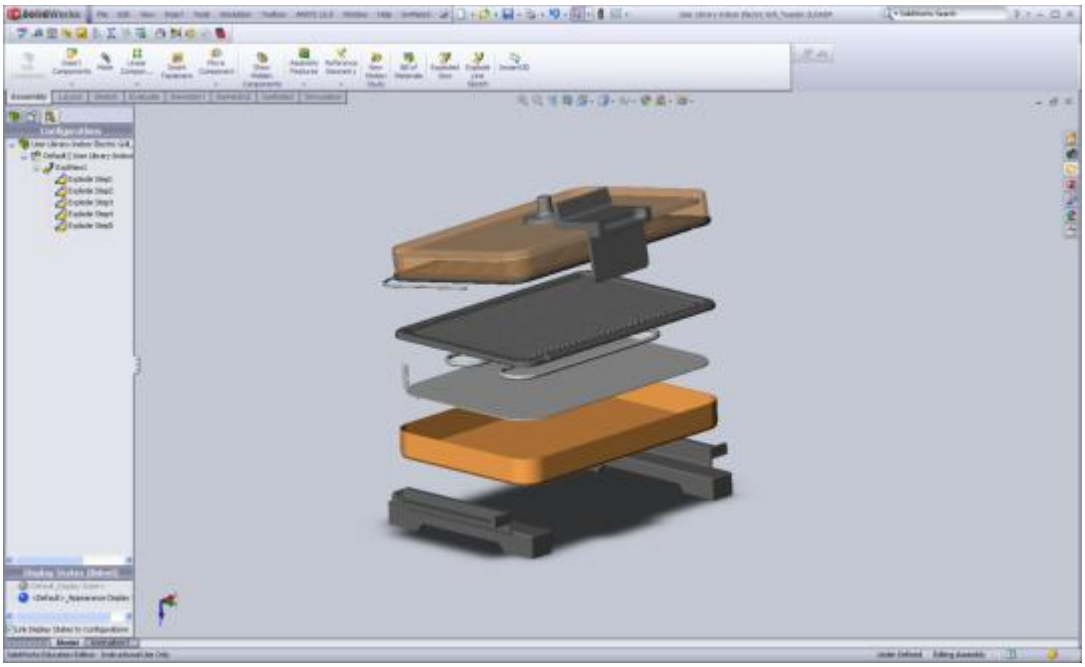
Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	11
			Relationships	29
	Connective		Degrees of Freedom	29
			Connectivity	58
Interconnection	Shortest Path		Total	196
			Maximum	2
			Average	1.781818182
			Density	0.0614
	Flow Rate		Total	324
			Maximum	27
			Average	2.6777
			Density	0.0923
Centrality	Betweenness		Total	86
			Maximum	86
			Average	7.818181818
			Density	0.2696
	Clustering Coefficient		Total	4.044444444
			Maximum	1
			Average	0.3677
			Density	0.0127
Decomposition	Ameri-Summers			123
	Core Numbers	In	Total	16
			Maximum	2
			Average	1.454545455
			Density	0.0502
		Out	Total	16
			Maximum	2
			Average	1.454545455
			Density	0.0502

Indoor Electric Grill: Complexity Vector for Fully Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	15
			Relationships	47
	Connective		Degrees of Freedom	47
			Connectivity	94
Interconnection	Shortest Path		Total	416
			Maximum	3
			Average	1.980952381
			Density	0.0421
	Flow Rate		Total	856
			Maximum	35
			Average	3.8044
			Density	0.0809
Centrality	Betweenness		Total	206
			Maximum	131.3666667
			Average	13.73333333
			Density	0.2922
	Clustering Coefficient		Total	2.703030303
			Maximum	1
			Average	0.1802
			Density	0.0038
Decomposition	Ameri-Summers			154
	Core Numbers	In	Total	27
			Maximum	2
			Average	1.8
			Density	0.0383
		Out	Total	27
			Maximum	2
			Average	1.8
			Density	0.0383



Indoor Electric Grill Assembly Model



Exploded View of Indoor Electric Grill

Paper Pro Stapler

The Paper Pro Stapler details for the partially defined assembly model are below.

Paper Pro Stapler Product and DFA Specifications

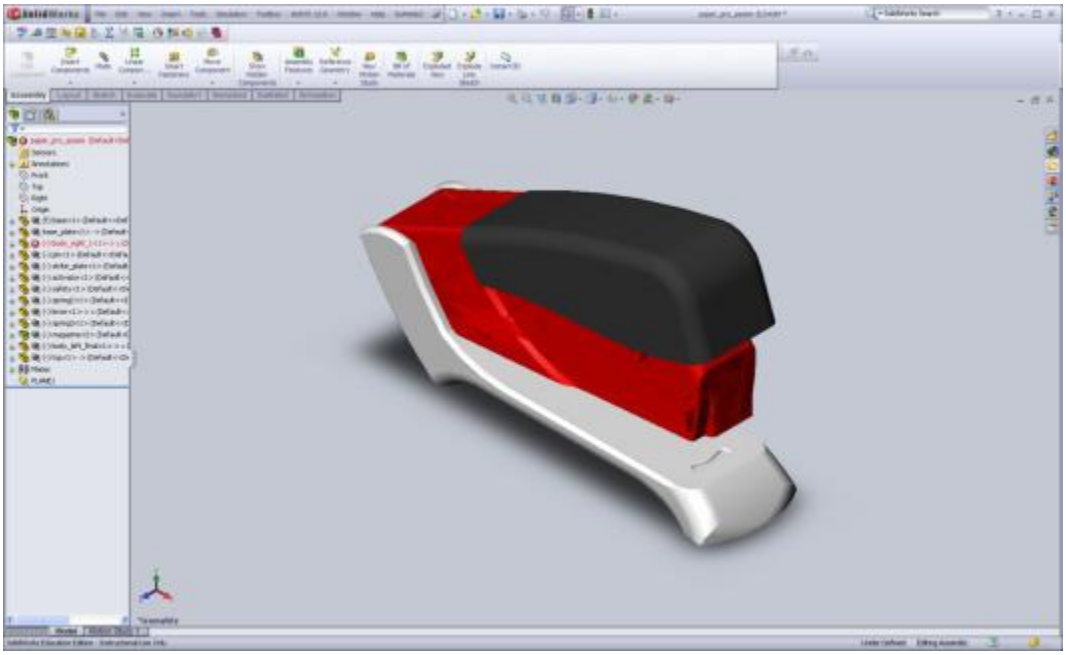
Product Name	Paper Pro Stapler
Constraint Definition	Partially Defined
# of Parts	16
# of Mates	36
Constraint Definition	Fully Defined
# of Parts	16
# of Mates	45
SW Assembly File Origin	GICL Website
Product Structure	Clam Shell
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	135.06
Physical DFA Analysis Time (min.)	68
Boothroyd Dewhurst DFA on Virtual Product (s)	123.51
Virtual DFA Analysis Time (min.)	80

Paper Pro Stapler: Complexity Vector for Partially Defined Model

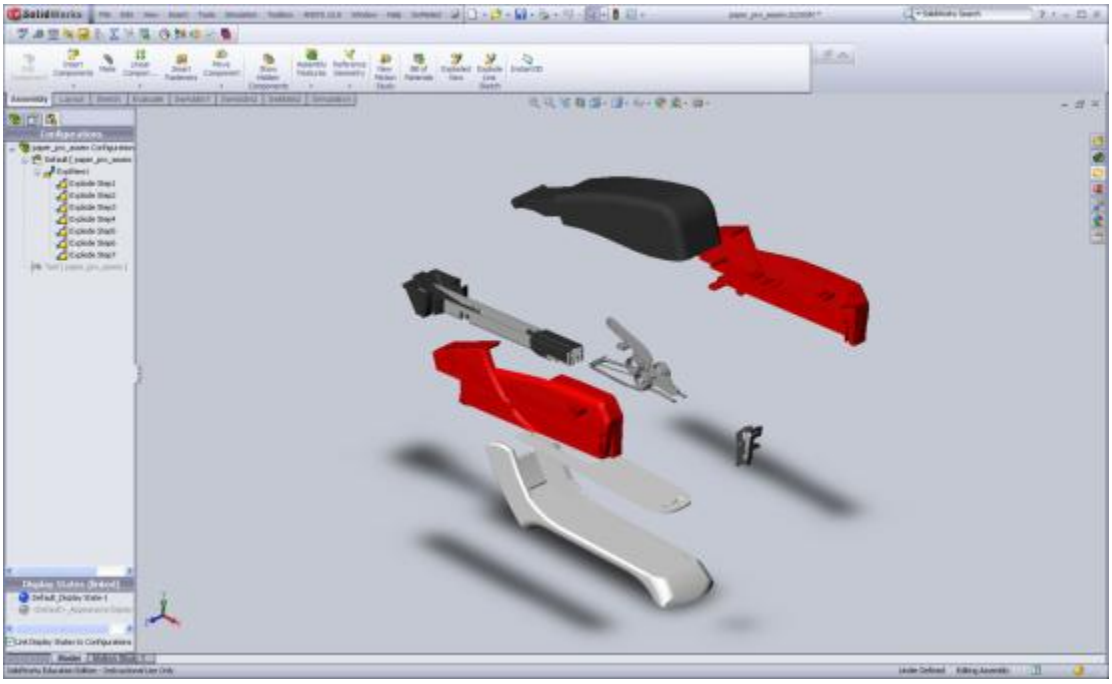
Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	18
			Relationships	36
	Connective		Degrees of Freedom	36
			Connectivity	72
Interconnection	Shortest Path		Total	382
			Maximum	4
			Average	1.248366013
			Density	0.0347
	Flow Rate		Total	564
			Maximum	16
			Average	1.7407
			Density	0.0484
Centrality	Betweenness		Total	188
			Maximum	111.5
			Average	10.44444444
			Density	0.2901
	Clustering Coefficient		Total	6.755555556
			Maximum	1
			Average	0.3753
			Density	0.0104
Decomposition	Ameri-Summers			90
	Core Numbers	In	Total	33
			Maximum	2
			Average	1.833333333
			Density	0.0509
		Out	Total	33
			Maximum	2
			Average	1.833333333
			Density	0.0509

Paper Pro Stapler: Complexity Vector for Fully Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	20
			Relationships	45
	Connective		Degrees of Freedom	45
			Connectivity	90
Interconnection	Shortest Path		Total	1026
			Maximum	6
			Average	2.7
			Density	0.0600
	Flow Rate		Total	858
			Maximum	20
			Average	2.1450
			Density	0.0477
Centrality	Betweenness		Total	646
			Maximum	267.6
			Average	32.3
			Density	0.7178
	Clustering Coefficient		Total	8.257575758
			Maximum	1
			Average	0.4129
			Density	0.0092
Decomposition	Ameri-Summers			215
	Core Numbers	In	Total	37
			Maximum	2
			Average	1.85
			Density	0.0411
		Out	Total	37
			Maximum	2
			Average	1.85
			Density	0.0411



Paper Pro Stapler Assembly Model



Exploded View of Paper Pro Stapler

Solar Yard Light

The Solar Yard Light details for the partially defined assembly model are below.

Solar Yard Light Product and DFA Specifications

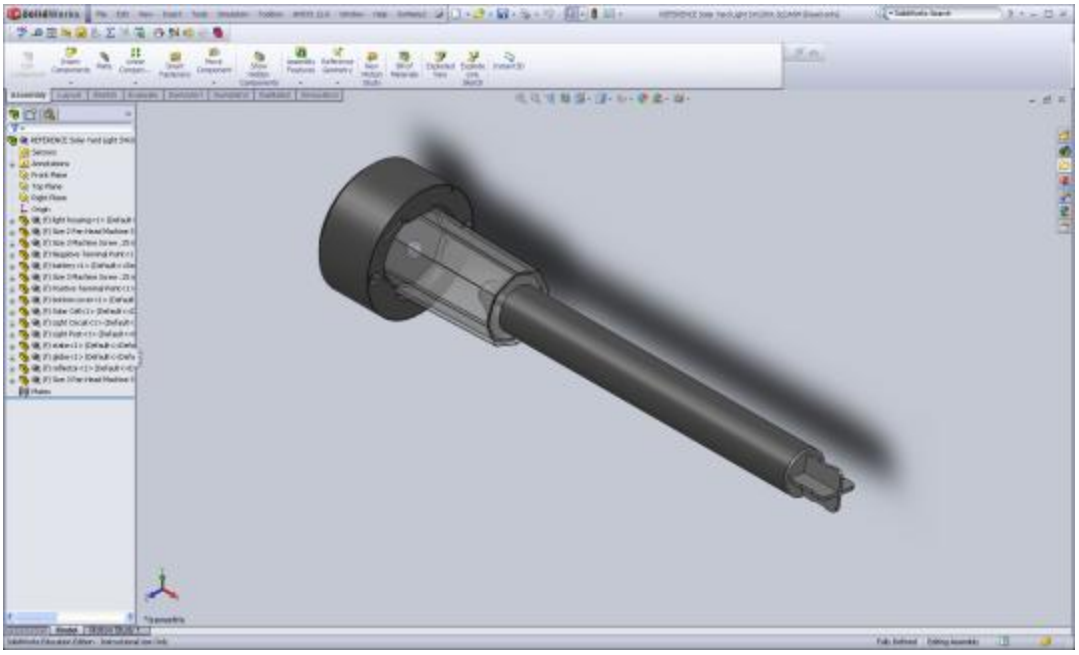
Product Name	Solar Yard Light
Constraint Definition	Partially Defined
# of Parts	17
# of Mates	35
Constraint Definition	Fully Defined
# of Parts	17
# of Mates	42
SW Assembly File Origin	Reverse Engineered by: Eric Owensby
Product Structure	Circular
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	131.23
Physical DFA Analysis Time (min.)	48
Boothroyd Dewhurst DFA on Virtual Product (s)	128.79
Virtual DFA Analysis Time (min.)	55

Solar Yard Light: Complexity Vector for Partially Defined Model

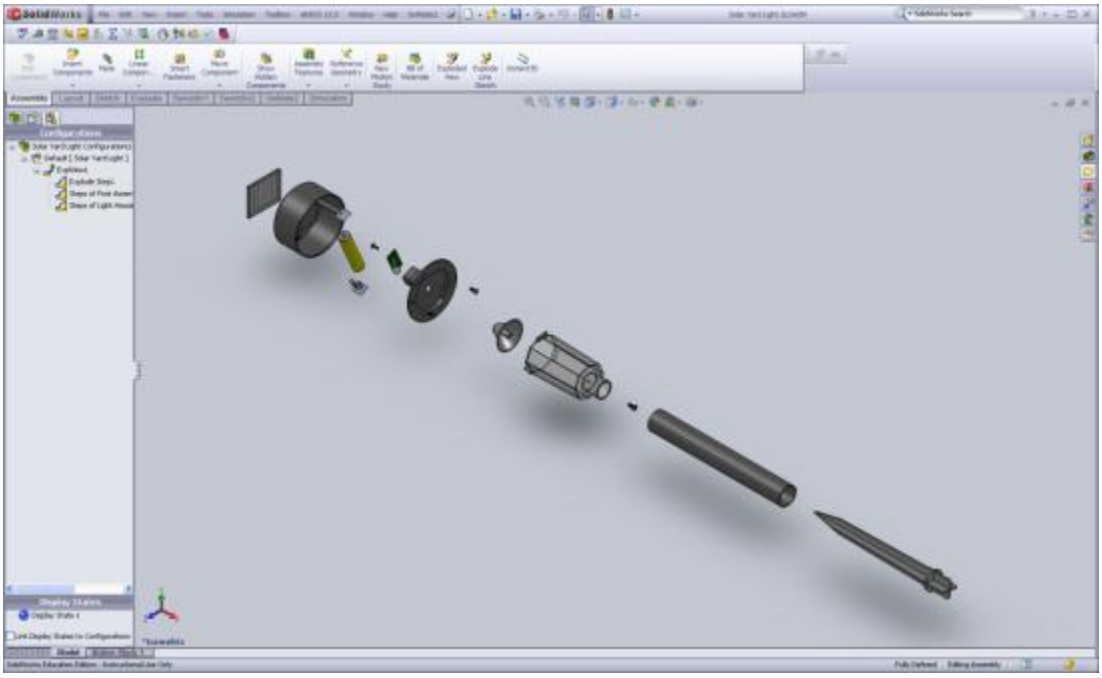
Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	19
			Relationships	35
	Connective		Degrees of Freedom	35
			Connectivity	70
Interconnection	Shortest Path		Total	368
			Maximum	5
			Average	1.076023
			Density	0.0307
	Flow Rate		Total	354
			Maximum	17
			Average	0.9806
			Density	0.0280
Centrality	Betweenness		Total	214
			Maximum	94
			Average	11.26316
			Density	0.3218
	Clustering Coefficient		Total	4.380952
			Maximum	1
			Average	0.2306
			Density	0.0066
Decomposition	Ameri-Summers			143
	Core Numbers	In	Total	25
			Maximum	2
			Average	1.315789
			Density	0.0376
		Out	Total	25
			Maximum	2
			Average	1.315789
			Density	0.0376

Solar Yard Light: Complexity Vector for Fully Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	19
			Relationships	42
	Connective		Degrees of Freedom	42
			Connectivity	84
Interconnection	Shortest Path		Total	734
			Maximum	5
			Average	2.146199
			Density	0.0511
	Flow Rate		Total	792
			Maximum	17
			Average	2.1939
			Density	0.0522
Centrality	Betweenness		Total	460
			Maximum	117
			Average	24.21053
			Density	0.5764
	Clustering Coefficient		Total	5.147619
			Maximum	1
			Average	0.2709
			Density	0.0065
Decomposition	Ameri-Summers			133
	Core Numbers	In	Total	35
			Maximum	2
			Average	1.842105
			Density	0.0439
		Out	Total	35
			Maximum	2
			Average	1.842105
			Density	0.0439



Solar Yard Light Assembly Model



Exploded View of Solar Yard Light

Mag Light

The Mag Light details for the partially defined assembly model are below.

Mag Light Product and DFA Specifications

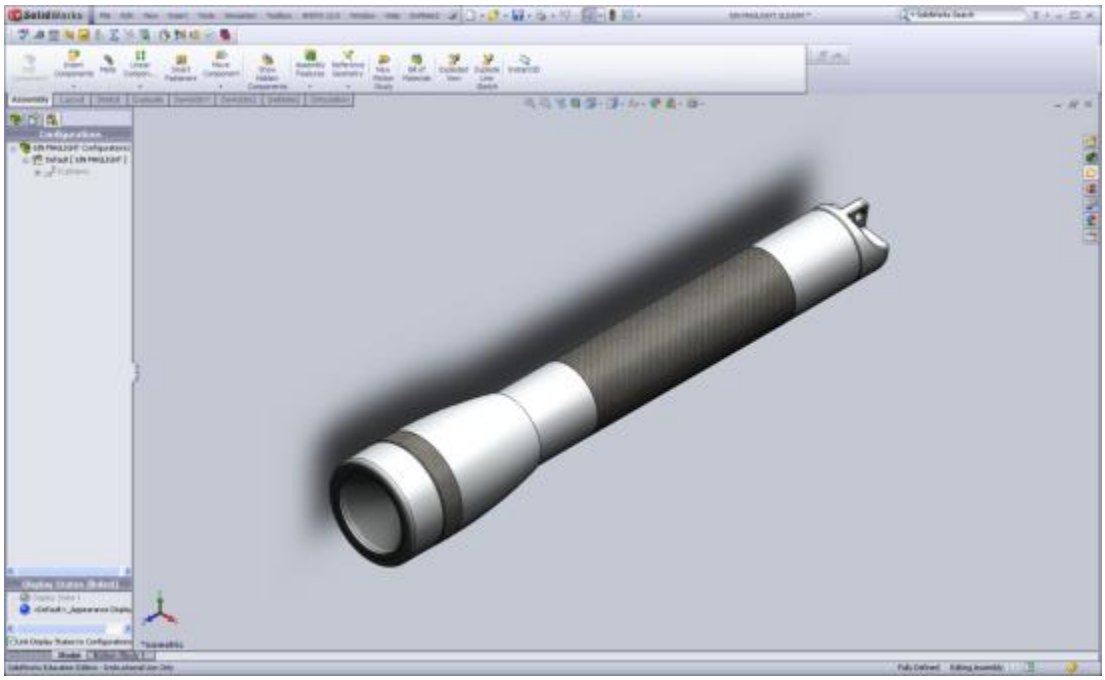
Product Name	Mag Light
Constraint Definition	Partially Defined
# of Parts	14
# of Mates	27
Constraint Definition	Fully Defined
# of Parts	14
# of Mates	29
SW Assembly File Origin	SolidWorks 3D Content
Product Structure	Circular
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	NA
Physical DFA Analysis Time (min.)	NA
Boothroyd Dewhurst DFA on Virtual Product (s)	75.4
Virtual DFA Analysis Time (min.)	32

Mag Light: Complexity Vector for Partially Defined Model

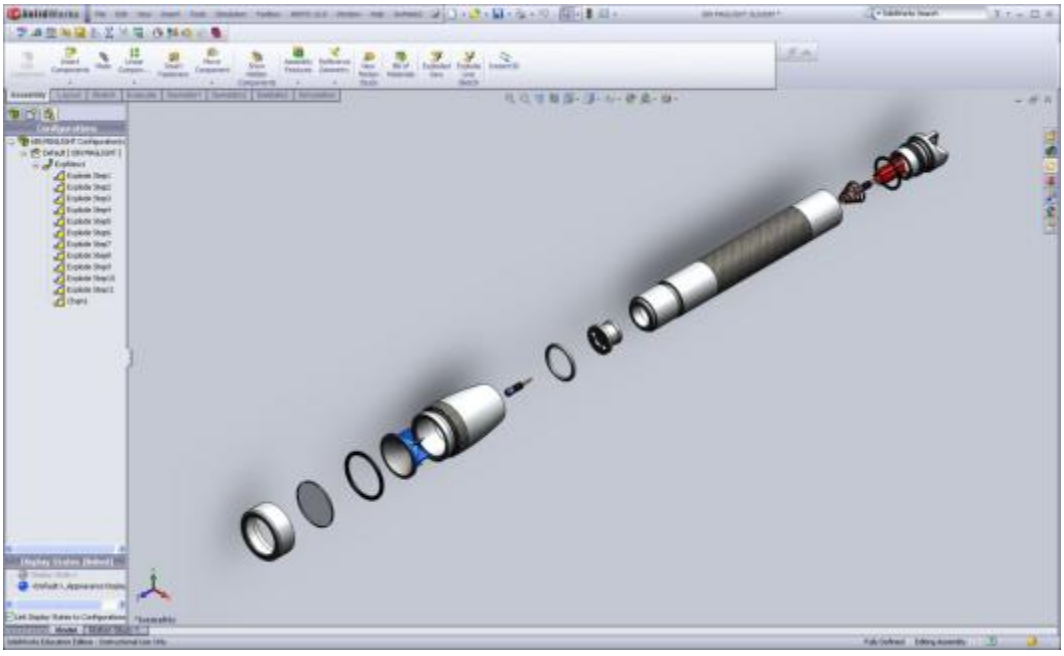
Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	14
			Relationships	27
	Connective		Degrees of Freedom	27
			Connectivity	54
Interconnection	Shortest Path		Total	520
			Maximum	6
			Average	2.8571429
			Density	0.1058
	Flow Rate		Total	380
			Maximum	10
			Average	1.9388
			Density	0.0718
Centrality	Betweenness		Total	338
			Maximum	82
			Average	24.142857
			Density	0.8942
	Clustering Coefficient		Total	1.8666667
			Maximum	1
			Average	0.1333
			Density	0.0049
Decomposition	Ameri-Summers			89
	Core Numbers	In	Total	18
			Maximum	2
			Average	1.2857143
			Density	0.0476
		Out	Total	18
			Maximum	2
			Average	1.2857143
			Density	0.0476

Mag Light: Complexity Vector for Fully Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	14
			Relationships	29
	Connective		Degrees of Freedom	29
			Connectivity	58
Interconnection	Shortest Path		Total	486
			Maximum	5
			Average	2.6703297
			Density	0.0921
	Flow Rate		Total	342
			Maximum	10
			Average	1.7449
			Density	0.0602
Centrality	Betweenness		Total	304
			Maximum	94
			Average	21.714286
			Density	0.7488
	Clustering Coefficient		Total	3.3666667
			Maximum	1
			Average	0.2405
			Density	0.0083
Decomposition	Ameri-Summers			95
	Core Numbers	In	Total	20
			Maximum	2
			Average	1.4285714
			Density	0.0493
		Out	Total	20
			Maximum	2
			Average	1.4285714
			Density	0.0493



Mag Light Assembly Model



Exploded View of Mag Light

Pony Vise

The Pony Vise details for the partially defined assembly model are below.

Pony Vise Product and DFA Specifications

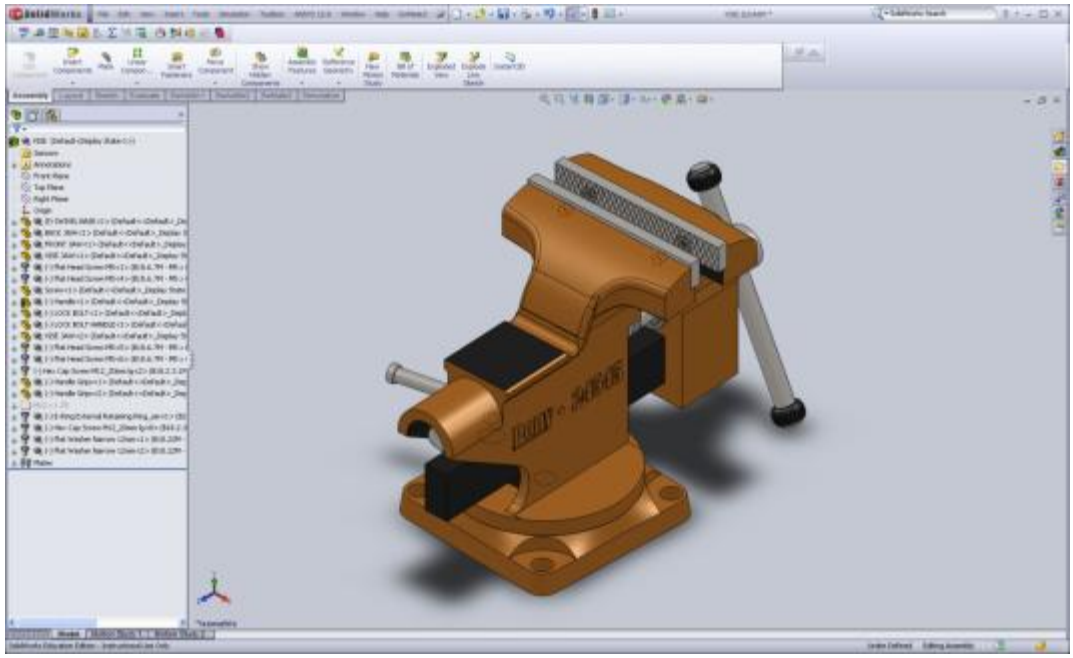
Product Name	Pony Vise
Constraint Definition	Partially Defined
# of Parts	20
# of Mates	45
Constraint Definition	Fully Defined
# of Parts	20
# of Mates	59
SW Assembly File Origin	EG 208 Undergraduate Class
Product Structure	Combination
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	153.3
Physical DFA Analysis Time (min.)	33
Boothroyd Dewhurst DFA on Virtual Product (s)	143.69
Virtual DFA Analysis Time (min.)	48

Pony Vise: Complexity Vector for Partially Defined Model

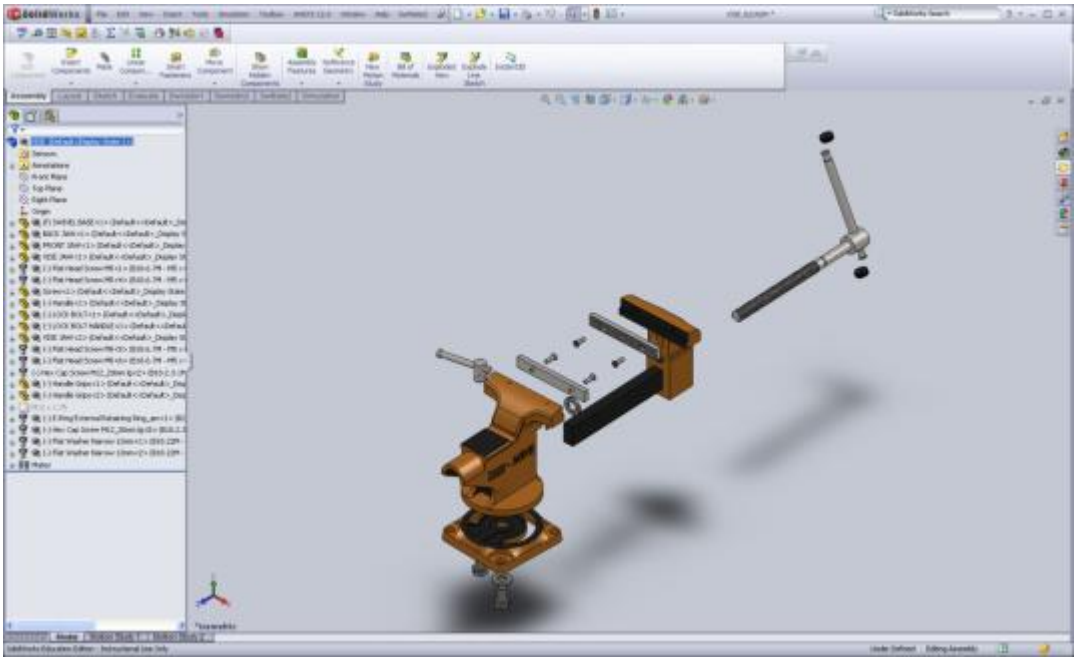
Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	20
			Relationships	45
	Connective		Degrees of Freedom	45
			Connectivity	90
Interconnection	Shortest Path		Total	1146
			Maximum	5
			Average	3.01578947
			Density	0.06701754
	Flow Rate		Total	944
			Maximum	12
			Average	2.36
			Density	0.05244444
Centrality	Betweenness		Total	766
			Maximum	168
			Average	38.3
			Density	0.85111111
	Clustering Coefficient		Total	0.8
			Maximum	0.33333333
			Average	0.04
			Density	0.00088889
Decomposition	Ameri-Summers			161
	Core Numbers	In	Total	30
			Maximum	2
			Average	1.5
			Density	0.03333333
		Out	Total	30
			Maximum	2
			Average	1.5
			Density	0.03333333

Pony Vise: Complexity Vector for Fully Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	22
			Relationships	59
	Connective		Degrees of Freedom	59
			Connectivity	118
Interconnection	Shortest Path		Total	1374
			Maximum	5
			Average	2.97402597
			Density	0.05040722
	Flow Rate		Total	1550
			Maximum	12
			Average	3.20247934
			Density	0.05427931
Centrality	Betweenness		Total	912
			Maximum	213.766667
			Average	41.4545455
			Density	0.70261941
	Clustering Coefficient		Total	3.66666667
			Maximum	1
			Average	0.16666667
			Density	0.00282486
Decomposition	Ameri-Summers			243
	Core Numbers	In	Total	36
			Maximum	2
			Average	1.63636364
			Density	0.02773498
		Out	Total	36
			Maximum	2
			Average	1.63636364
			Density	0.02773498



Pony Vise Assembly Model



Exploded View of Pony Vise

Black and Decker Drill

The Black and Decker Drill details for the partially defined assembly model are below.

Black and Decker Drill Product and DFA Specifications

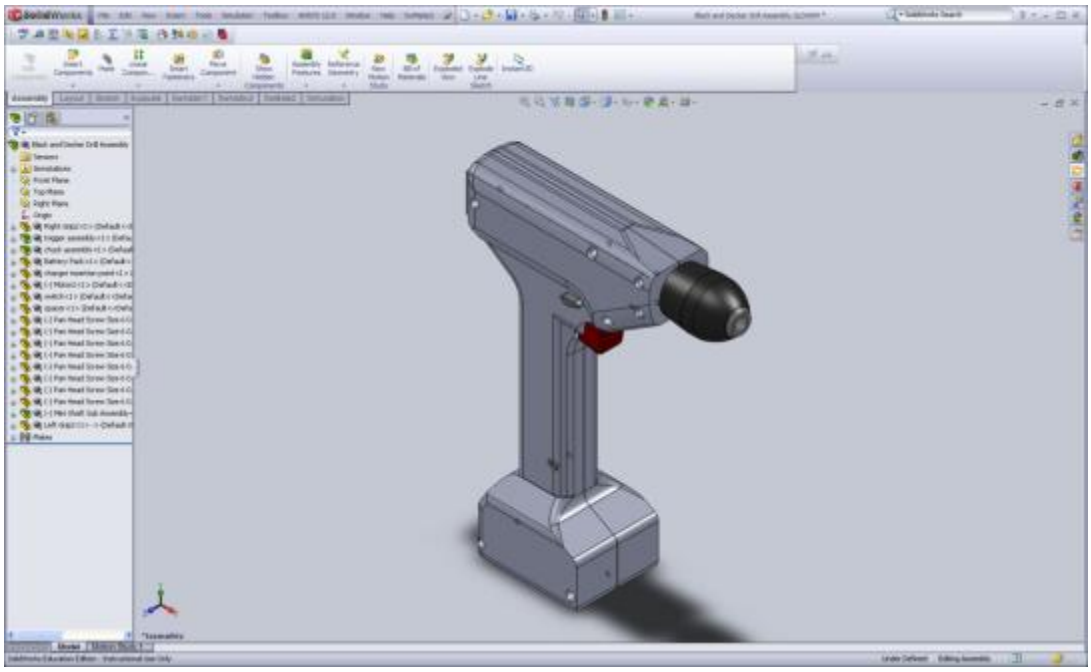
Product Name	Black and Decker Drill
Constraint Definition	Partially Defined
# of Parts	31
# of Mates	64
Constraint Definition	Fully Defined
# of Parts	31
# of Mates	87
SW Assembly File Origin	Reverse Engineered: Eric Owensby
Product Structure	Clam Shell
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	180.2
Physical DFA Analysis Time (min.)	48
Boothroyd Dewhurst DFA on Virtual Product (s)	189.7
Virtual DFA Analysis Time (min.)	42

Black and Decker Drill: Complexity Vector for Partially Defined Model

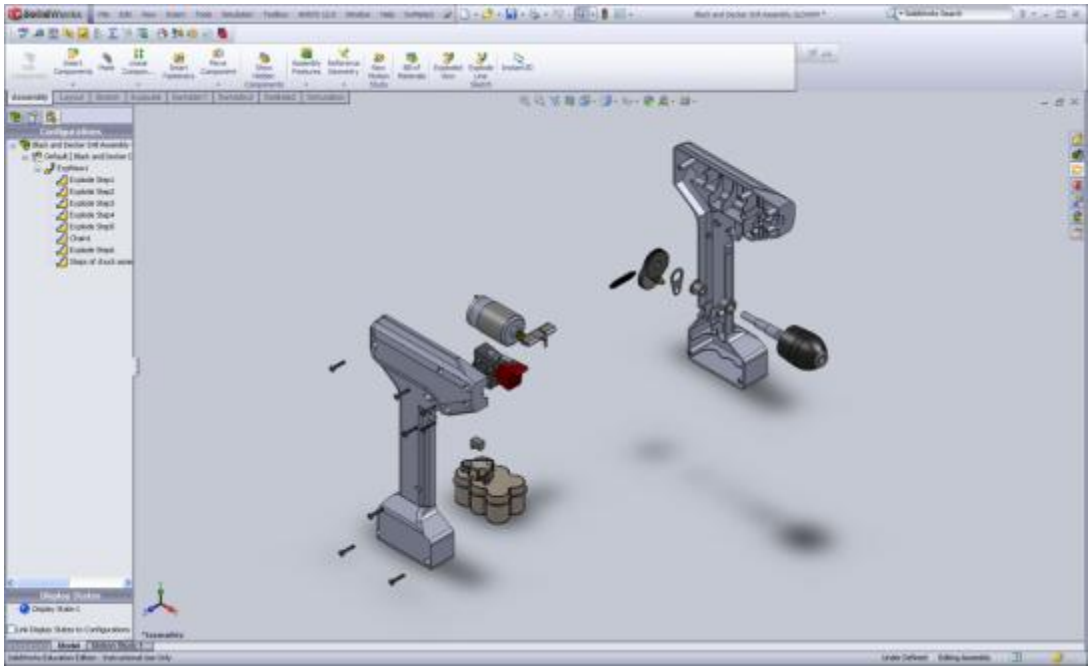
Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	34
			Relationships	64
	Connective		Degrees of Freedom	64
			Connectivity	128
Interconnection	Shortest Path		Total	1574
			Maximum	6
			Average	1.40285205
			Density	0.0219
	Flow Rate		Total	1200
			Maximum	25
			Average	1.0381
			Density	0.0162
Centrality	Betweenness		Total	980
			Maximum	403
			Average	28.8235294
			Density	0.4504
	Clustering Coefficient		Total	6.22727273
			Maximum	1
			Average	0.1832
			Density	0.0029
Decomposition	Ameri-Summers			246
	Core Numbers	In	Total	44
			Maximum	2
			Average	1.29411765
			Density	0.0202
		Out	Total	44
			Maximum	2
			Average	1.29411765
			Density	0.0202

Black and Decker Drill: Complexity Vector for Fully Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	34
			Relationships	87
	Connective		Degrees of Freedom	87
			Connectivity	174
Interconnection	Shortest Path		Total	3032
			Maximum	5
			Average	2.70231729
			Density	0.0311
	Flow Rate		Total	2746
			Maximum	31
			Average	2.3754
			Density	0.0273
Centrality	Betweenness		Total	1910
			Maximum	732.583333
			Average	56.1764706
			Density	0.6457
	Clustering Coefficient		Total	15.2947786
			Maximum	1
			Average	0.4498
			Density	0.0052
Decomposition	Ameri-Summers			211
	Core Numbers	In	Total	63
			Maximum	2
			Average	1.85294118
			Density	0.0213
		Out	Total	63
			Maximum	2
			Average	1.85294118
			Density	0.0213



Black and Decker Drill Assembly Model



Exploded View of Black and Decker Drill

825 Shift Frame LH

The 825 Shift Frame LH details for the partially defined assembly model are below.

825 Shift Frame LH Product and DFA Specifications

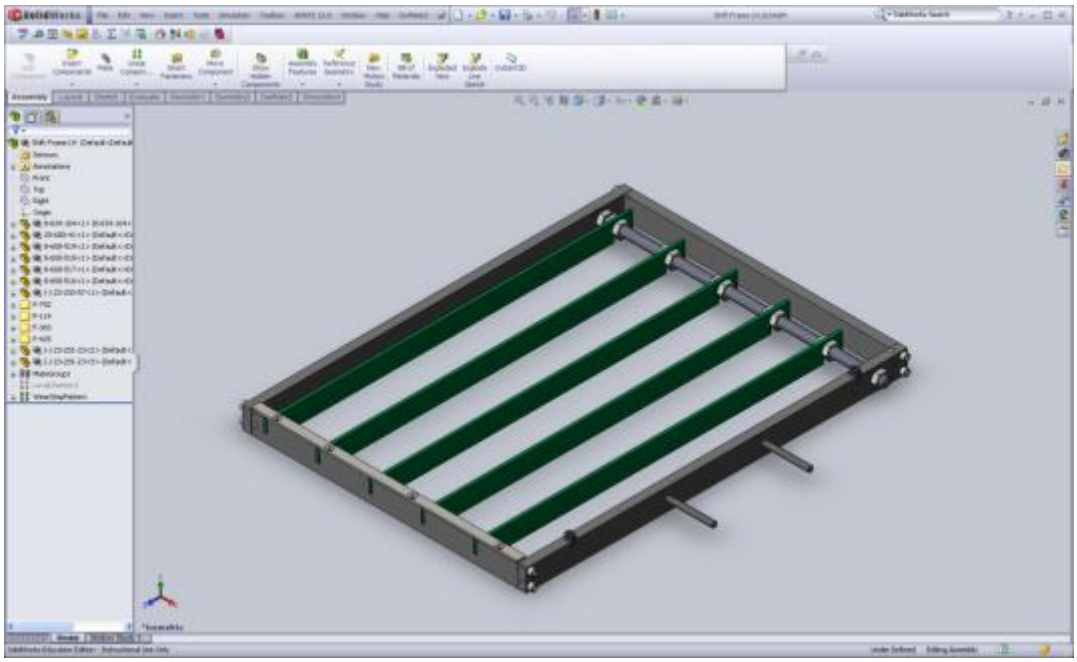
Product Name	825 Shift Frame LH
Constraint Definition	Partially Defined
# of Parts	28
# of Mates	62
Constraint Definition	Fully Defined
# of Parts	28
# of Mates	84
SW Assembly File Origin	OEM
Product Structure	Stackable
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	NA
Physical DFA Analysis Time (min.)	NA
Boothroyd Dewhurst DFA on Virtual Product (s)	313.7
Virtual DFA Analysis Time (min.)	49

825 Shift Frame LH: Complexity Vector for Partially Defined Model

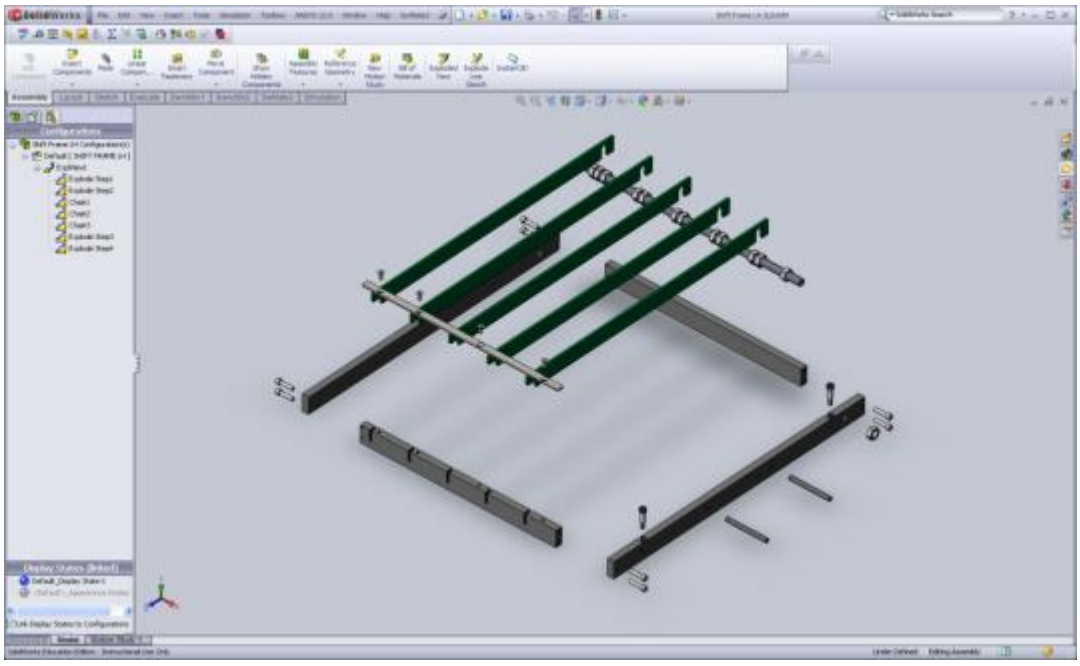
Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	31
			Relationships	62
	Connective		Degrees of Freedom	62
			Connectivity	124
Interconnection	Shortest Path		Total	2454
			Maximum	4
			Average	2.638709677
			Density	0.0426
	Flow Rate		Total	1936
			Maximum	26
			Average	2.0146
			Density	0.0325
Centrality	Betweenness		Total	1524
			Maximum	556.3333333
			Average	49.16129032
			Density	0.7929
	Clustering Coefficient		Total	5.61031746
			Maximum	1
			Average	0.1810
			Density	0.0029
Decomposition	Ameri-Summers			159
	Core Numbers	In	Total	46
			Maximum	2
			Average	1.483870968
			Density	0.0239
		Out	Total	46
			Maximum	2
			Average	1.483870968
			Density	0.0239

825 Shift Frame LH: Complexity Vector for Fully Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	31
			Relationships	84
	Connective		Degrees of Freedom	84
			Connectivity	168
Interconnection	Shortest Path		Total	2408
			Maximum	4
			Average	2.589247312
			Density	0.0308
	Flow Rate		Total	2828
			Maximum	32
			Average	2.9428
			Density	0.0350
Centrality	Betweenness		Total	1478
			Maximum	503
			Average	47.67741935
			Density	0.5676
	Clustering Coefficient		Total	7.946581197
			Maximum	1
			Average	0.2563
			Density	0.0031
Decomposition	Ameri-Summers			211
	Core Numbers	In	Total	49
			Maximum	2
			Average	1.580645161
			Density	0.0188
		Out	Total	49
			Maximum	2
			Average	1.580645161
			Density	0.0188



825 Shift Frame LH Assembly Model



Exploded View of 825 Shift Frame LH

OEM 825 Wide Flag

The OEM 825 Wide Flag details for the partially and fully defined assembly model are below.

OEM 825 Wide Flag Product and DFA Specifications

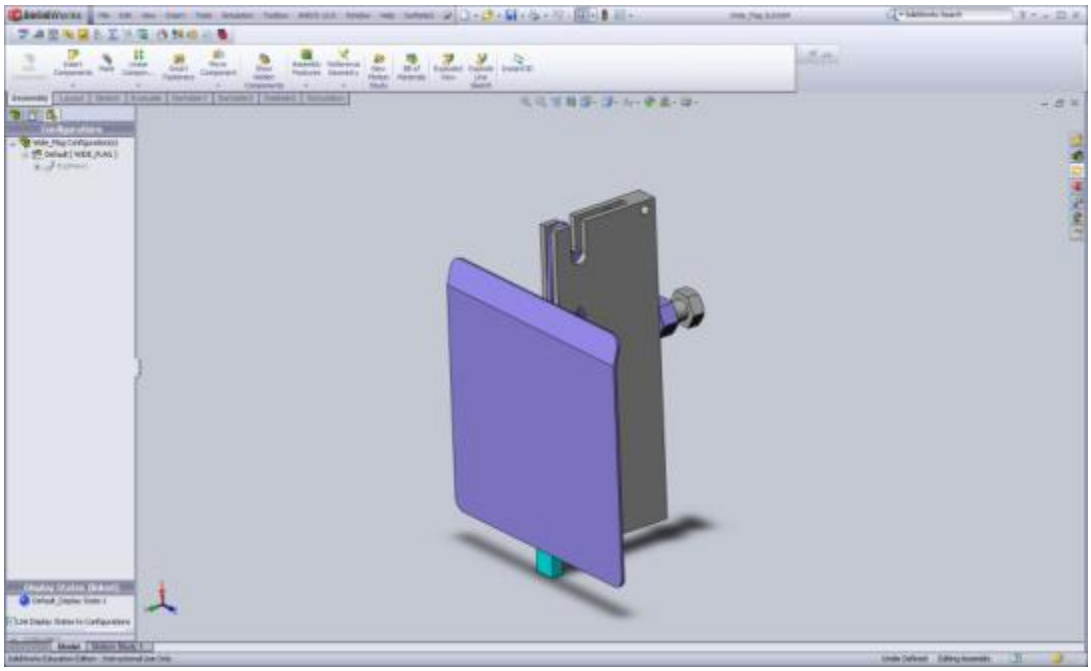
Product Name	OEM 825 Wide Flag
Constraint Definition	Partially Defined
# of Parts	10
# of Mates	21
Constraint Definition	Fully Defined
# of Parts	10
# of Mates	27
SW Assembly File Origin	OEM
Product Structure	Stackable
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	NA
Physical DFA Analysis Time (min.)	NA
Boothroyd Dewhurst DFA on Virtual Product (s)	58.3
Virtual DFA Analysis Time (min.)	21

OEM 825 Wide Flag: Complexity Vector for Partially Defined Model

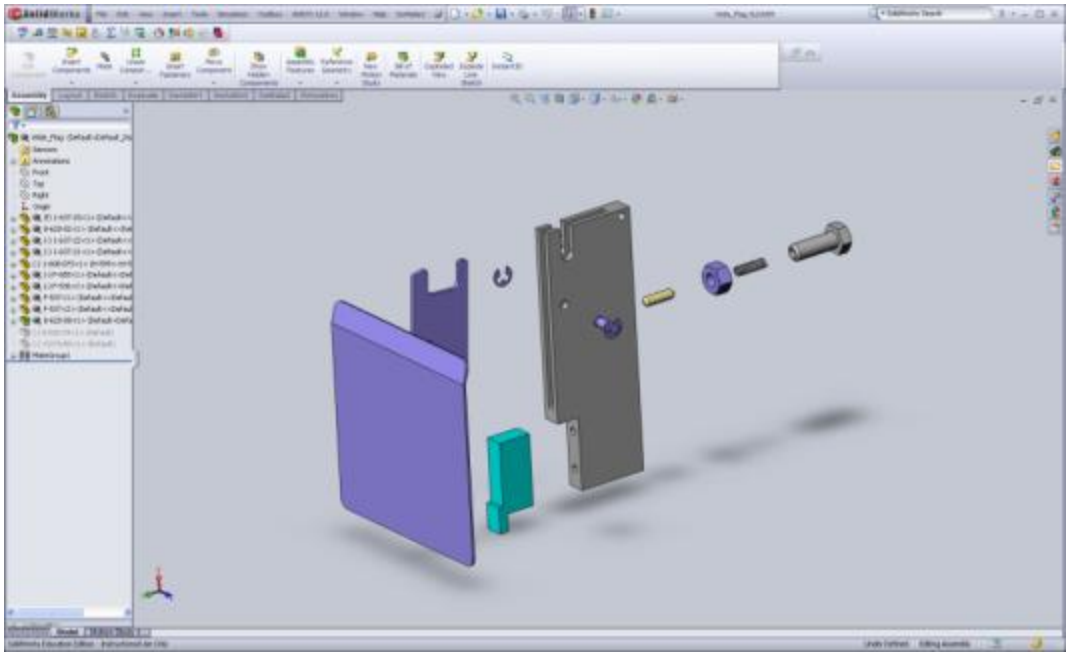
Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	10
			Relationships	21
	Connective		Degrees of Freedom	21
			Connectivity	42
Interconnection	Shortest Path		Total	166
			Maximum	3
			Average	1.844444444
			Density	0.0878
	Flow Rate		Total	252
			Maximum	13
			Average	2.5200
			Density	0.1200
Centrality	Betweenness		Total	76
			Maximum	57
			Average	7.6
			Density	0.3619
	Clustering Coefficient		Total	4.976190476
			Maximum	1
			Average	0.4976
			Density	0.0237
Decomposition	Ameri-Summers			54
	Core Numbers	In	Total	17
			Maximum	2
			Average	1.7
			Density	0.0810
		Out	Total	17
			Maximum	2
			Average	1.7
			Density	0.0810

OEM 825 Wide Flag: Complexity Vector for Fully Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	10
			Relationships	27
	Connective		Degrees of Freedom	27
			Connectivity	54
Interconnection	Shortest Path		Total	148
			Maximum	2
			Average	1.644444444
			Density	0.0609
	Flow Rate		Total	368
			Maximum	16
			Average	3.6800
			Density	0.1363
Centrality	Betweenness		Total	58
			Maximum	53
			Average	5.8
			Density	0.2148
	Clustering Coefficient		Total	7.361111111
			Maximum	1
			Average	0.7361
			Density	0.0273
Decomposition	Ameri-Summers			73
	Core Numbers	In	Total	23
			Maximum	3
			Average	2.3
			Density	0.0852
		Out	Total	23
			Maximum	3
			Average	2.3
			Density	0.0852



OEM 825 Wide Flag Assembly Model



Exploded View of OEM 825 Wide Flag

One Touch Chopper

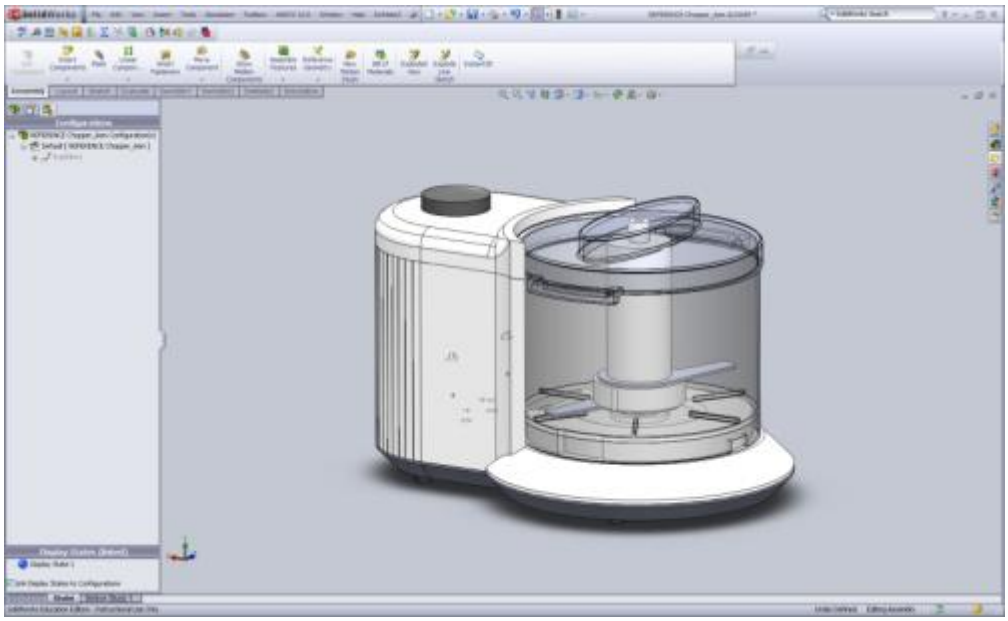
The One Touch Chopper details for the partially defined assembly model are below.

One Touch Chopper Product and DFA Specifications

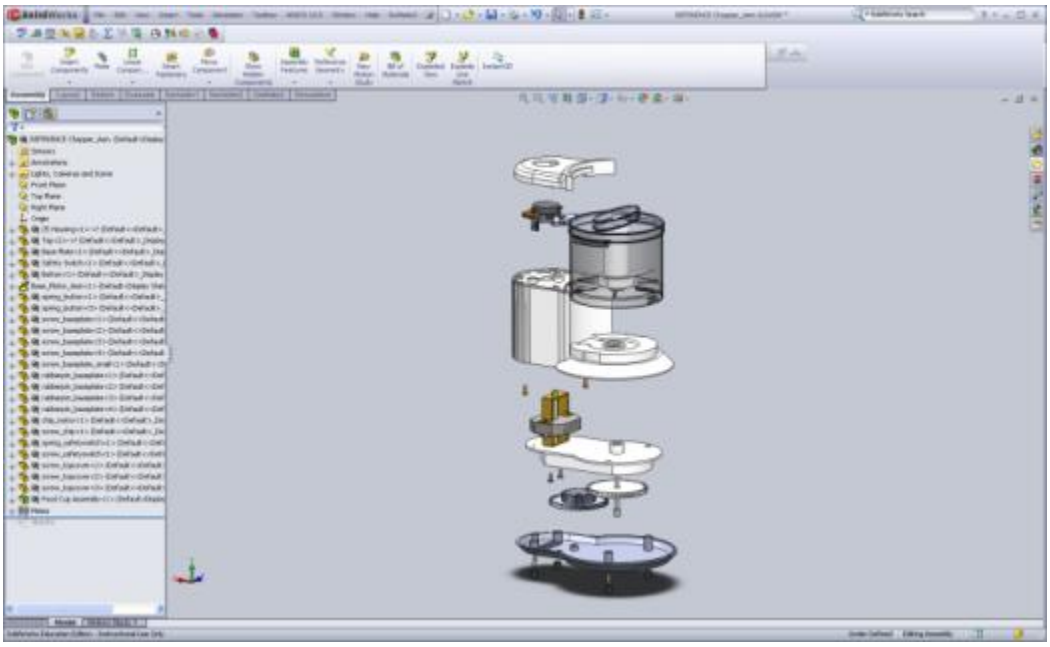
Product Name	One Touch Chopper
Constraint Definition	Partially Defined
# of Parts	43
# of Mates	123
Constraint Definition	NA
# of Parts	NA
# of Mates	NA
SW Assembly File Origin	Reverse Engineered: Aravind Shanthakumar
Product Structure	Combination
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	NA
Physical DFA Analysis Time (min.)	NA
Boothroyd Dewhurst DFA on Virtual Product (s)	316.62
Virtual DFA Analysis Time (min.)	136

One Touch Chopper: Complexity Vector for Partially Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	46
			Relationships	123
	Connective		Degrees of Freedom	123
			Connectivity	246
Interconnection	Shortest Path		Total	3246
			Maximum	6
			Average	1.568115942
			Density	0.0127
	Flow Rate		Total	3066
			Maximum	37
			Average	1.4490
			Density	0.0118
Centrality	Betweenness		Total	2106
			Maximum	507
			Average	45.7826087
			Density	0.3722
	Clustering Coefficient		Total	10.90649351
			Maximum	1
			Average	0.2371
			Density	0.0019
Decomposition	Ameri-Summers			634
	Core Numbers	In	Total	68
			Maximum	2
			Average	1.47826087
			Density	0.0120
		Out	Total	68
			Maximum	2
			Average	1.47826087
			Density	0.0120



One Touch Chopper Assembly Model



Exploded View of One Touch Chopper

Mouse Model

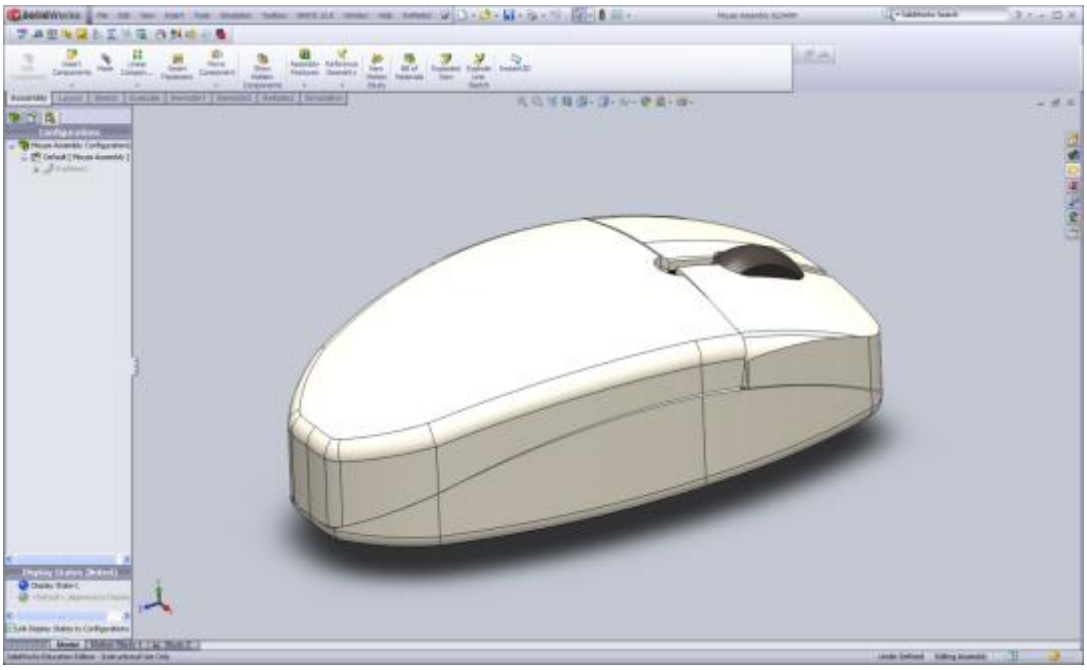
The Mouse Model details for the partially defined assembly model are below.

Mouse Model Product and DFA Specifications

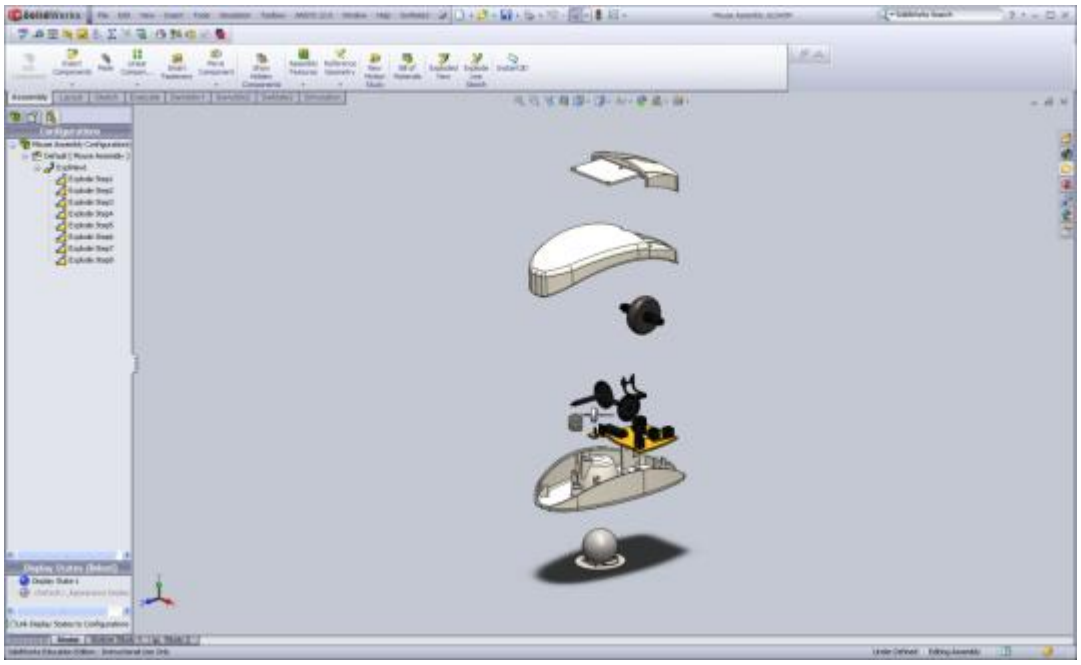
Product Name	Mouse Model
Constraint Definition	Partially Defined
# of Parts	14
# of Mates	30
Constraint Definition	NA
# of Parts	NA
# of Mates	NA
SW Assembly File Origin	Reverse Engineered: Matt Peterson
Product Structure	Combination
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	NA
Physical DFA Analysis Time (min.)	NA
Boothroyd Dewhurst DFA on Virtual Product (s)	81.3
Virtual DFA Analysis Time (min.)	51

Mouse Model: Complexity Vector for Partially Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	14
			Relationships	30
	Connective		Degrees of Freedom	30
			Connectivity	60
Interconnection	Shortest Path		Total	294
			Maximum	4
			Average	1.615384615
			Density	0.0538
	Flow Rate		Total	344
			Maximum	16
			Average	1.7551
			Density	0.0585
Centrality	Betweenness		Total	160
			Maximum	99
			Average	11.42857143
			Density	0.3810
	Clustering Coefficient		Total	2.738095238
			Maximum	1
			Average	0.1956
			Density	0.0065
Decomposition	Ameri-Summers			133
	Core Numbers	In	Total	18
			Maximum	2
			Average	1.285714286
			Density	0.0429
		Out	Total	18
			Maximum	2
			Average	1.285714286
			Density	0.0429



Mouse Model Assembly Model



Exploded View of Mouse Model

Durabrand Blender

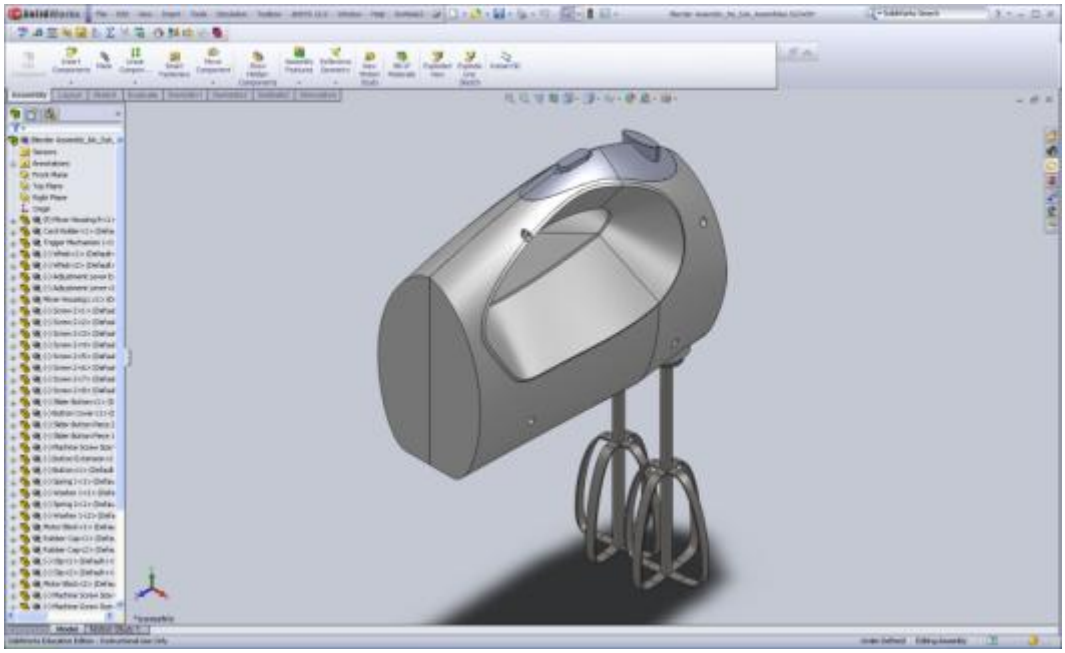
The Durabrand Blender details for the partially defined assembly model are below.

Durabrand Blender Product and DFA Specifications

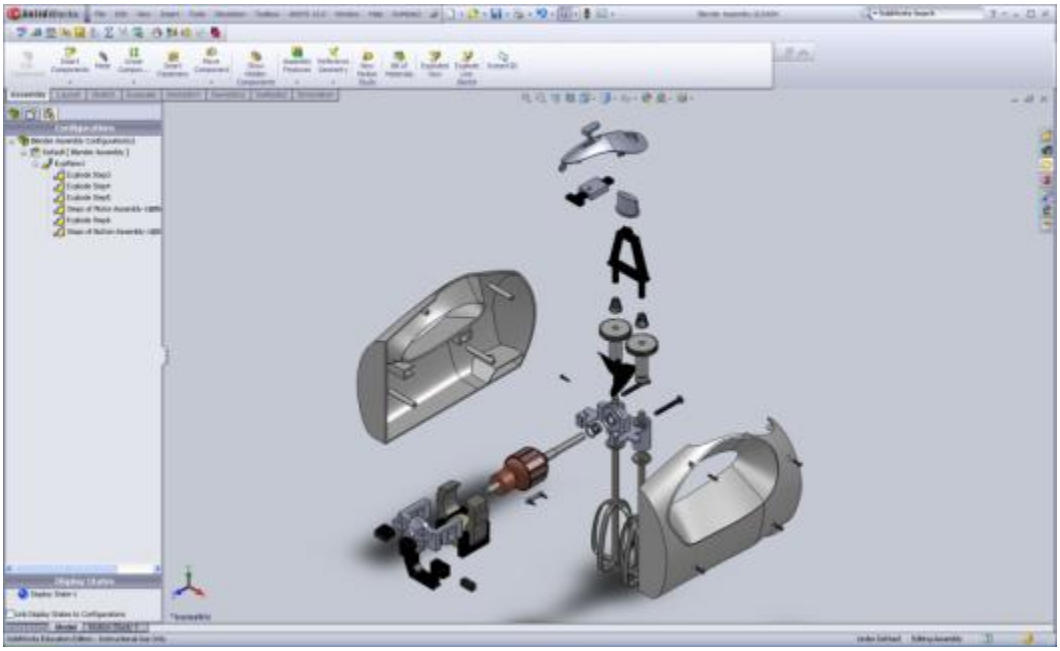
Product Name	Durabrand Blender
Constraint Definition	Partially Defined
# of Parts	45
# of Mates	105
Constraint Definition	NA
# of Parts	NA
# of Mates	NA
SW Assembly File Origin	Reverse Engineered: David Griese
Product Structure	Combination
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	NA
Physical DFA Analysis Time (min.)	NA
Boothroyd Dewhurst DFA on Virtual Product (s)	263.2
Virtual DFA Analysis Time (min.)	139

Durabrand Blender: Complexity Vector for Partially Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	48
			Relationships	105
	Connective		Degrees of Freedom	105
			Connectivity	210
Interconnection	Shortest Path		Total	2296
			Maximum	7
			Average	1.017730496
			Density	0.009692671
	Flow Rate		Total	2242
			Maximum	28
			Average	0.973090278
			Density	0.009267526
Centrality	Betweenness		Total	1442
			Maximum	282
			Average	30.04166667
			Density	0.286111111
	Clustering Coefficient		Total	11.15604396
			Maximum	1
			Average	0.232417582
			Density	0.002213501
Decomposition	Ameri-Summers			395
	Core Numbers	In	Total	76
			Maximum	2
			Average	1.583333333
			Density	0.015079365
		Out	Total	76
			Maximum	2
			Average	1.583333333
			Density	0.015079365



Durabrand Blender Assembly Model



Exploded View of Durabrand Blender

Boothroyd Piston Assembly

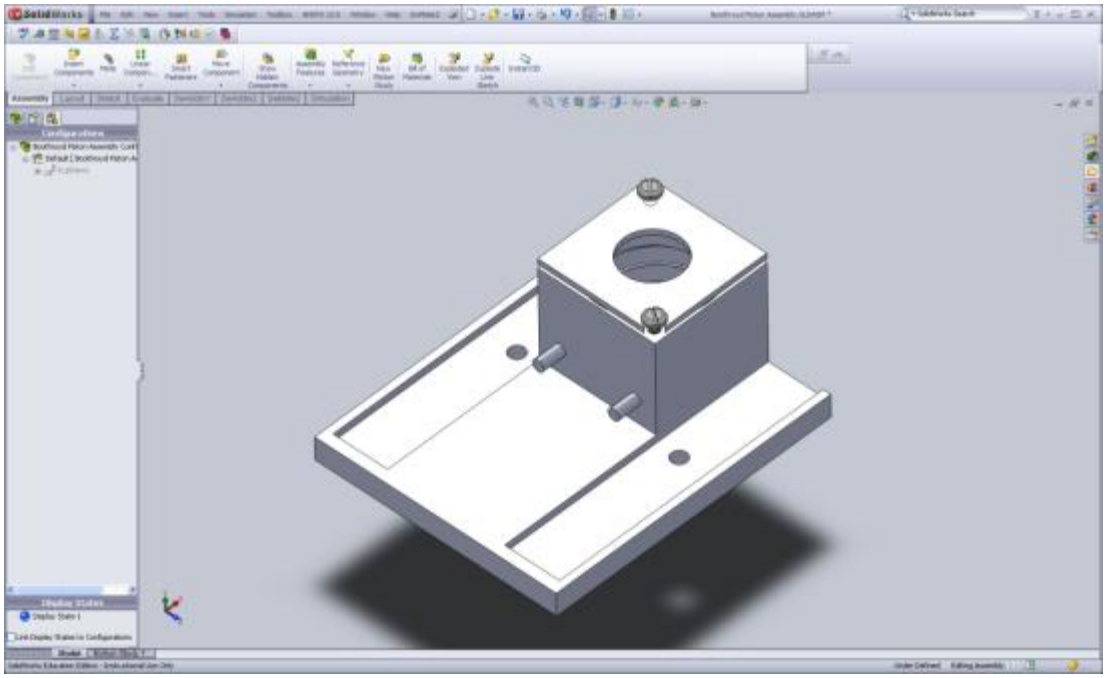
The Boothroyd Piston Assembly details for the partially defined assembly model are below. This piston was modeled using the rough schematic shown of the piston in the Boothroyd Design for Assembly Handbook.

Boothroyd Piston Assembly Product and DFA Specifications

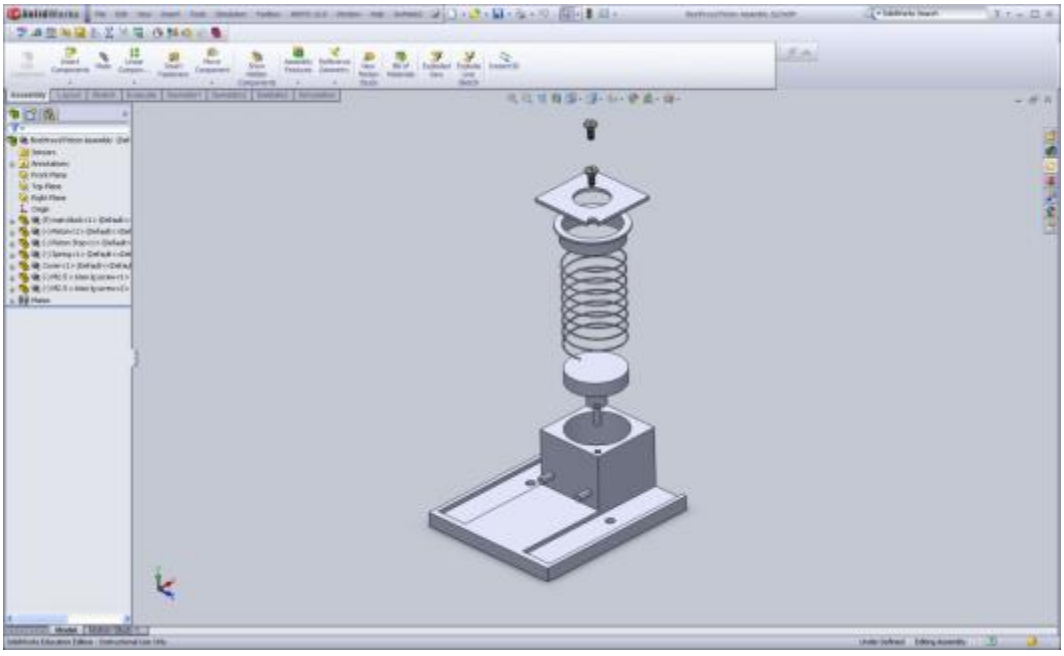
Product Name	Boothroyd Piston Assembly
Constraint Definition	Partially Defined
# of Parts	7
# of Mates	12
Constraint Definition	NA
# of Parts	NA
# of Mates	NA
SW Assembly File Origin	Reverse Engineered: Matt Peterson
Product Structure	Stackable
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	NA
Physical DFA Analysis Time (min.)	NA
Boothroyd Dewhurst DFA on Virtual Product (s)	48.0
Virtual DFA Analysis Time (min.)	12

Boothroyd Piston Assembly: Complexity Vector for Partially Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	7
			Relationships	12
	Connective		Degrees of Freedom	12
			Connectivity	24
Interconnection	Shortest Path		Total	64
			Maximum	2
			Average	1.523809524
			Density	0.126984127
	Flow Rate		Total	118
			Maximum	8
			Average	2.408163265
			Density	0.200680272
Centrality	Betweenness		Total	22
			Maximum	19
			Average	3.142857143
			Density	0.261904762
	Clustering Coefficient		Total	5.766666667
			Maximum	1
			Average	0.823809524
			Density	0.068650794
Decomposition	Ameri-Summers			25
	Core Numbers	In	Total	14
			Maximum	2
			Average	2
			Density	0.166666667
		Out	Total	14
			Maximum	2
			Average	2
			Density	0.166666667



Boothroyd Piston Assembly Assembly Model



Exploded View of Boothroyd Piston Assembly

Hole Punch

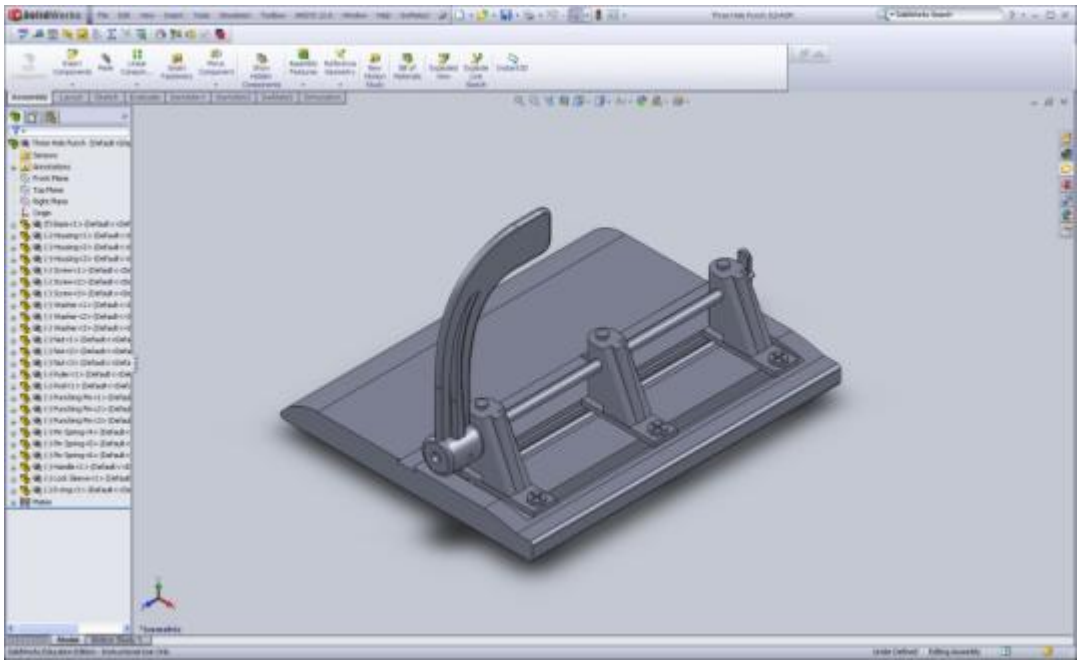
The Hole Punch details for the partially defined assembly model are below.

Hole Punch Product and DFA Specifications

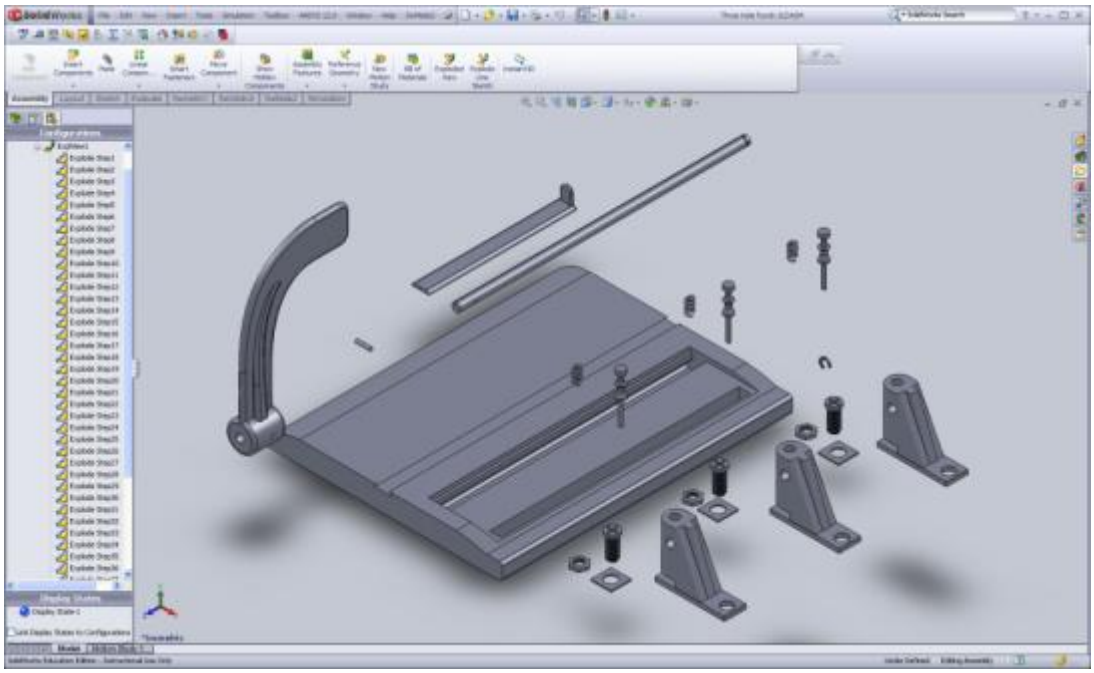
Product Name	Hole Punch
Constraint Definition	Partially Defined
# of Parts	24
# of Mates	52
Constraint Definition	NA
# of Parts	NA
# of Mates	NA
SW Assembly File Origin	EG 208 Undergraduate Class
Product Structure	Combination
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	NA
Physical DFA Analysis Time (min.)	NA
Boothroyd Dewhurst DFA on Virtual Product (s)	145.4
Virtual DFA Analysis Time (min.)	35

Hole Punch: Complexity Vector for Partially Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	25
			Relationships	52
	Connective		Degrees of Freedom	52
			Connectivity	104
Interconnection	Shortest Path		Total	1680
			Maximum	5
			Average	2.8
			Density	0.05384615
	Flow Rate		Total	1468
			Maximum	16
			Average	2.3488
			Density	0.04516923
Centrality	Betweenness		Total	1080
			Maximum	353
			Average	43.2
			Density	0.83076923
	Clustering Coefficient		Total	6.16666667
			Maximum	1
			Average	0.24666667
			Density	0.00474359
Decomposition	Ameri-Summers			188
	Core Numbers	In	Total	43
			Maximum	2
			Average	1.72
			Density	0.03307692
		Out	Total	43
			Maximum	2
			Average	1.72
			Density	0.03307692



Hole Punch Assembly Model



Exploded View of Hole Punch

Electric Knife

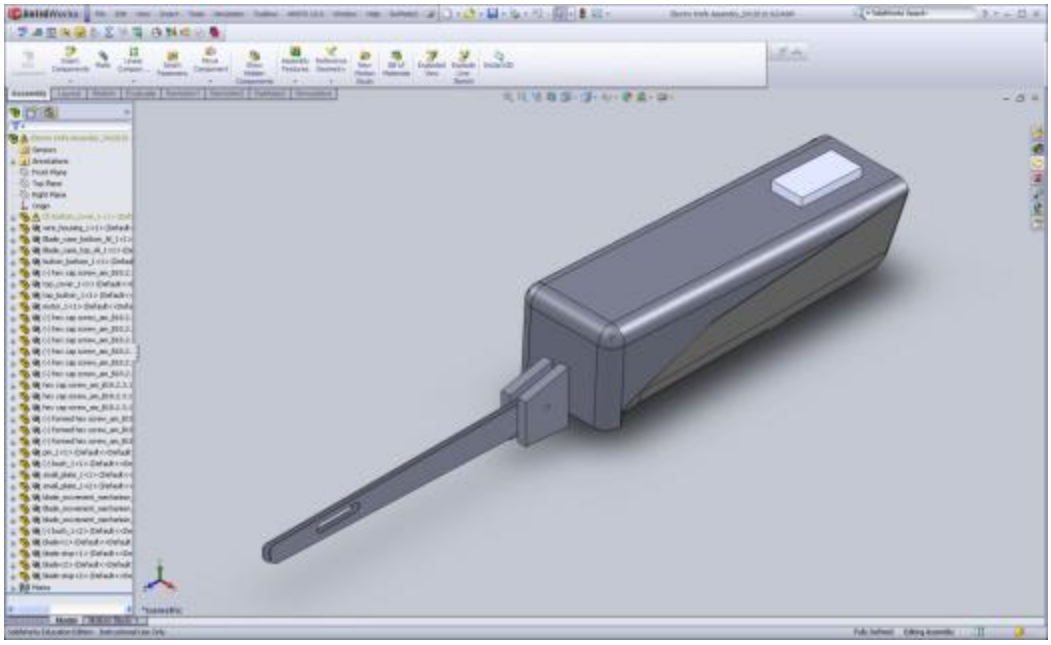
The Electric Knife details for the partially defined assembly model are below. This model was used to test the final automated assembly time prediction tool. The model is not an exact replica of the physical product but it forms a good representation of what the model looks like and it contains enough information to mate the parts and to conduct a virtual Boothroyd DFA analysis.

Electric Knife Product and DFA Specifications

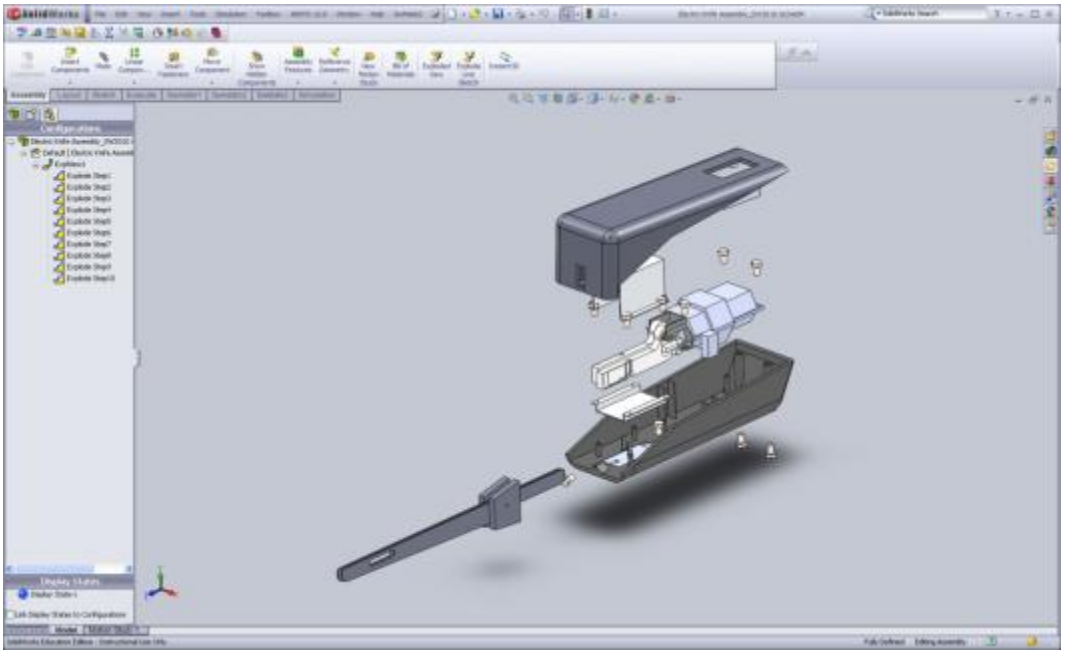
Product Name	Electric Knife
Constraint Definition	Partially Defined
# of Parts	33
# of Mates	80
Constraint Definition	NA
# of Parts	NA
# of Mates	NA
SW Assembly File Origin	Reverse Engineered: Rahul
Product Structure	Clam Shell
DFA Conducted By:	Eric Owensby
Boothroyd Dewhurst DFA on Physical Product (s)	NA
Physical DFA Analysis Time (min.)	NA
Boothroyd Dewhurst DFA on Virtual Product (s)	212.3
Virtual DFA Analysis Time (min.)	33

Electric Knife: Complexity Vector for Partially Defined Model

Class	Type	Dir	Metric	Complexity
Size	Dimensional		Elements	25
			Relationships	80
	Connective		Degrees of Freedom	80
			Connectivity	160
Interconnection	Shortest Path		Total	1746
			Maximum	6
			Average	2.91
			Density	0.0364
	Flow Rate		Total	1928
			Maximum	27
			Average	3.0848
			Density	0.0386
Centrality	Betweenness		Total	1146
			Maximum	352.6667
			Average	45.84
			Density	0.5730
	Clustering Coefficient		Total	4.977778
			Maximum	1
			Average	0.1991
			Density	0.0025
Decomposition	Ameri-Summers			577
	Core Numbers	In	Total	43
			Maximum	3
			Average	1.72
			Density	0.0215
		Out	Total	43
			Maximum	3
			Average	1.72
			Density	0.0215



Electric Knife Assembly Model



Exploded View of Electric Knife