Proceedings of the ASME 2012 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2012
August 12 – August 15, 2012, Chicago, Illinois, USA

DETC2012-70995

REPRESENTATION: EXTRACTING MATE COMPLEXITY FROM ASSEMBLY MODELS TO AUTOMATICALLY PREDICT ASSEMBLY TIMES

Eric Owensby
Research Assistant
Mechanical Engineering
Clemson University
Clemson, SC 29634-0921
jowensb@clemson.edu

Essam Z. Namouz
Research Assistant
Mechanical Engineering
Clemson University
Clemson, SC 29634-0921
enamouz@clemson.edu

Aravind Shanthakumar
Research Assistant
Mechanical Engineering
Clemson University
Clemson, SC 29634-0921
ashanth@clemson.edu

Joshua D. Summers, PhD
Associate Professor & CoES IDEaS Professor
Mechanical Engineering
Clemson University
Clemson, SC 29634-0921
joshua.summers@ces.clemson.edu
(corresponding author)

ABSTRACT

The work in this paper uses neural networks to develop a relationship model between assembly times and complexity metrics applied to defined mate connections within SolidWorks assembly models. This model is then used to develop a Design for Assembly (DFA) automation tool that can predict a product's assembly time using defined mate connections within SolidWorks assembly models. The development of this new method consists of: creating a SolidWorks (SW) Add-in to automatically extract the mate connections from SW assembly models, parsing the mate connections into graphs, implementing a new complexity training algorithm to predict assembly times based on mate graphs, and evaluating the effectiveness of the new method. The motivation, development, and evaluation of the new automated DFA method are presented in this paper. Ultimately, the method that is trained on both fully defined and partially defined assembly models is shown to provide assembly time prediction results that are typically within 25% of target time, but with one outlier at 95% error, suggesting that a more robust training set is needed.

Keywords: Design for Assembly, DFA, Automated DFA, Assembly Time

1 MOTIVATION / BACKGROUND

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[1, 2]. The original work used a regression analysis to relate a products physical connection complexity to assembly times. A continuation of this worked used neural networks to develop a relationship between a products physical connection complexity and assembly times. The issue with these methods is that they have to be manually implemented.

This paper presents the development of a semi-automated Assembly Time Prediction method that extracts defined mates from SolidWorks Assembly models and uses the mate complexity to determine the products assembly time. The concept of Design for Assembly (DFA) originated in the 1960's when designers were given basic guidelines to improve their products with regards to manufacturing and assembly[3]. In the 1980's these guidelines were integrated into systematic DFA analysis tools that would help the designer predict the products assembly time[4]. This provided the designer with a tool that would help identify the products assembly cost and measure design improvements with respect to the assembly times[3]. Some examples of DFA methods are the: Boothroyd Dewhurst method[5], the Methods-time Measurement (MTM) method[6], the Assembly Evaluation Method of Hitachi[7], and the Lucas Method[8].

DFA methods have been industry tested and proven to be advantageous by reducing the products: total part count, manufacturing cost, production lead time, inventory, assembly time, and assembly cost [4] [9][10]. Even with the proven benefits achieved by applying DFA methods they still have limitations. Many of these methods are tedious, time consuming, and subjective which leads to reluctance to implement [11][8]. These methods also require design details such as dimensions, securing methods, or assembly motions, which are not generally known until the detailed design stage of the design process. DFA methods will yield the greatest benefits if they can be applied to products early in the design process [12][8]. Figure 1 shows a simplified version of a

systematic design process along with where DFA methods are currently used and where they would ideally be used[13].

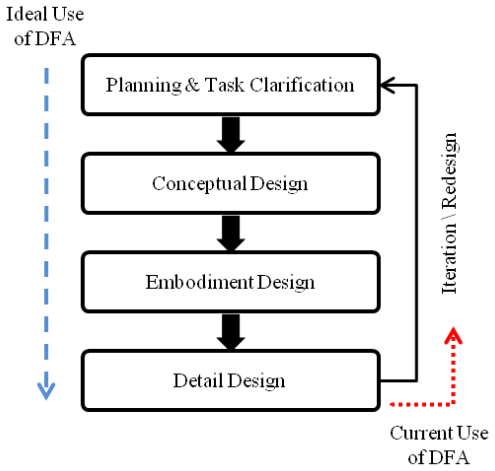

Figure 1: DFA in the Design Process (Adapted from[13])

DFA methods were originally intended to be applied throughout the design process in order to maximize cost savings based on the fact that 60% to 80% of a products cost is determined during the early phases of the design process [9][8]. Due to the issues mentioned 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[14]. For example the Boothroyd and Dewhurst DFMA software is one of the most widely published and used methods in industry today. 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[15]. All 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 new product.

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[8]. This would eliminate the tediousness, subjective, time consuming issues that reduce current DFA implementation. Attempts at automating current DFA methods have been inhibited since they often use a variety of subjective information which is difficult to program.

One attempt at solving the subjective issues of existing DFA methods which prevents their automation was the development of a Connectivity Complexity DFA method[1]. This method predicts assembly times from products inter connectedness complexity. This method was developed using linear regression to identify a relationship between a products assembly time and the complexity of the inter part connections. The advantage of this method over existing 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[1].

The Connectivity Complexity DFA method was later evaluated and compared to the Boothroyd Dewhurst DFMA software based on five criteria: approximate time to conduct the analysis, predicted assembly time, amount of required input information, amount of subjective information, and number of redesign features provided to the user[16]. The results of this evaluation and comparison determined that the Boothroyd Dewhurst DFMA software required the user to answer forty nine questions per part, sixteen of which were subjective. The Connectivity Complexity method only required the users to answer five questions per part none of them being subjective. The predicted assembly times of the Connectivity Complexity method ranged from 13.11% to 49.71% lower than the predicted times of the DFMA software which was considered to be the baseline. Both methods required similar amounts of time to implement. The evaluation suggests that the Boothroyd DFMA software is effective while requiring extensive subjective user inputs which would be difficult to program. Based on this evaluation the Connectivity Complexity method should be automatable since it only requires objective information but its accuracy could be improved[16].

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. Figure 2 shows a flow chart of the continued development of the Connectivity Complexity DFA method. The original work, shown by the top row in Figure 2, 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 2, implemented the ANN training to improve the accuracy of the predicted assembly times[2]. The work presented in this paper relates to the third version evolution in the attempt to develop an objective and automated assembly time estimation tool.

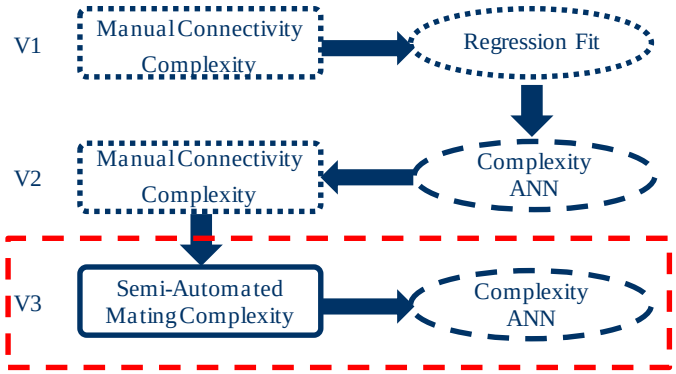

Figure 2: Connectivity Complexity DFA development flow chart

The focus of this paper is shown in the third step, V3, (boxed level in Figure 2) 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 CAD models which are generated as early as the conceptual design phase giving it the potential to be applied throughout the design process[17]. Extracting the connections from 3D CAD Assembly models would also enable a program to be developed to completely automate this method. The rest of this paper presents the development of a semi automated Connectivity Complexity assembly time prediction method.

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.

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[1]. This means that in order to automate the original connectivity complexity method, the physical part connections would have to be extracted from the assembly models.

3D CAD 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 CAD 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 was explored.

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 (shaft connections, electrical connections, spring connections, etc.)[1]. 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.

Consider three parts (Part A, Part B, and Part C in Figure 3) representing two pillow block bearings and a shaft respectively. If Part A and Part B have holes extruded through them and Part C has a circular cross section of the same size then they may form a shaft connection. To determine if these parts do form a shaft connection, the parts location would have to be identified 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 would be present.

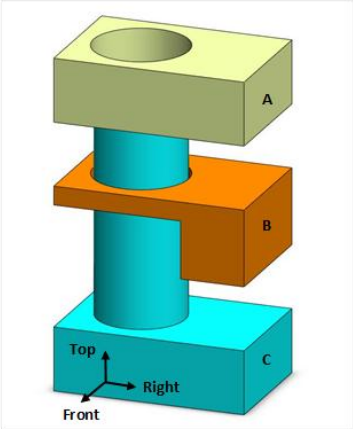

Figure 3: Part A, Part B, and Part C which could be mated or constrained in a variety of ways

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 the 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[18, 19]. Ultimately, the design intent may be captured, but other relationships will also be captured, thereby hiding the underlying intent.

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 SW mates are used to define the relationship between two

components within the assembly[20]. 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 1[20].

Table 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 types, ex: limit mates confine two parts to a given range
Mechanical	6 types, ex: gear mates define the angle of rotation between two gears

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 3 describe in Section 2.2: Part A and Part B which have holes extruded through them and Part C which has a circular cross section. 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 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 one of the research questions for this paper which is: can mate connections as defined in assembly models be used to predict a products assembly time?

2.4. Feature Connections vs. Mate 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 combination 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 3. Table 2 shows two different configurations that can be followed to fully constrain the parts in Figure 3 and achieve the same outcome.

Table 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 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.

3 MATE EXTRACTION SOLIDWORKS ADD-IN

The tool used for extracting mates-information from CAD assembly models was developed using 2010 SolidWorks Application Programming Interface (API) Software Development Kit (SDK). SW is a commercial CAD software package that supports CAD files downloaded from libraries and provides an easy-to-use Graphical User Interface (GUI). The software offers two options to develop the SolidWorks API application, macros and programming[20]. 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 be 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[20]. The C++ programming language was used for its easier implementation of COM objects[21], 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 is shown in Figure 4.

```
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 4: Pseudo-code for Extracting Mate Information

To obtain the mate information from a CAD assembly file the code traverses through the contents of the assembly feature manager tree. It should be noted that within the assembly model all items contained in the feature manager tree are considered features of that assembly, these are different than part features contained in part files. The feature manager tree in SW consists of information such as annotations, co-ordinate planes, part names, parts, sub-assemblies, and mate constraints. The code traverses through this list to a container that has the mate information. Each mate consists of the names of parts between which the mate is defined as relationship between parents. For each mate, the names of both parents are retrieved, and the relationship between the parents indicates the connection between the parts. 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 shows a flow diagram of the SW mate extraction add-in, its required inputs, the information processing steps, and the assembly time output.

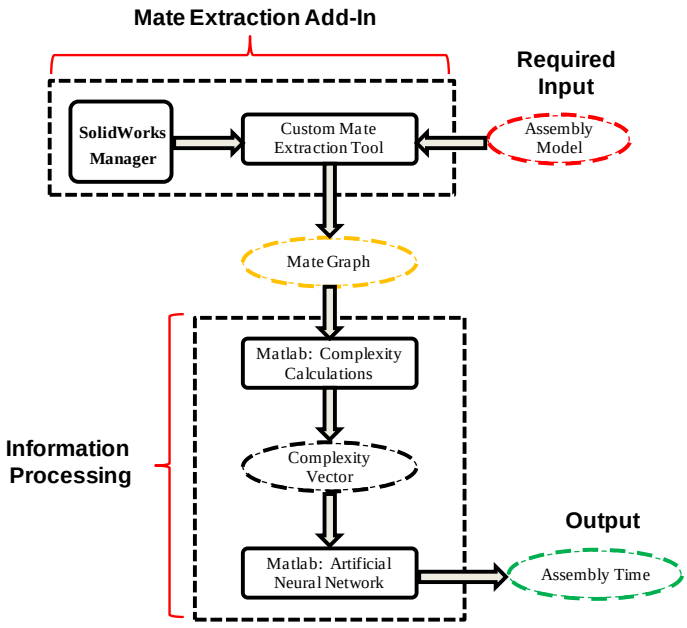

Figure 5: SW mate extraction add-in and information processing

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.

4 CREATION OF ANN TRAINING SET

Before the information extracted from the developed tool can be used, Artificial Neural Networks (ANNs) must be trained. ANN training was presented in detail in the motivating research but since the effectiveness of the ANNs is determined by their training, this topic is reviewed and expanded in this work.

Training of an Artificial Neural Network requires a large set of inputs and respective target values. Since the goal of this work is to identify the relationship between the mate complexity of 3D CAD assembly models and the assembly times these items become the inputs and targets respectively. This means that a collection of 3D CAD assembly models with known assembly times has to be compiled.

4.1. Collecting Product 3D CAD Assembly Models

The original goal was to download 3D CAD assembly models of consumer household products from one online CAD 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 method described in Section 4.2. An extensive search of online CAD databases was conducted to identify one that would contain assembly models that met the desired criteria. A single online CAD database was not identified due to a variety of issue: compatibility issues with SW assembly files, 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, or in many cases a combination of these issues. Without the purchasing of the physical product the assembly time could not be determined.

From this attempt the next method was to download any product assembly models from any CAD 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 products were reverse engineered in order to create respective assembly models. The complete list of product assembly models can be seen in Table 3.

Once ten different assembly models of consumer products were gathered the Mate Extraction tool discussed in Section 0was used to automatically generate the mate connection graphs.

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

Table 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 [22]
7	6" MagLight	SW 3D Content [23]
8	Indoor Electric Grill	SW 3D Content [23]
9	Shift Frame LH	OEM
10	Wide Flag	OEM

4.2. Calculating Product Assembly Times

In order to conduct the neural network training an assembly time was needed so that the twenty nine complexity metrics generated for each product could be given a respective assembly time to target. Since access to the actual assembly times for the products was 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[4]. 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 3. The physical products for items 1-6 in Table 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. 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 in order to apply 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 6.

Generation of the exploded view makes the designer think about the assembly sequence and function of the given parts reducing some of the difficulty of the DFA.

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. 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, since several of the models were reverse engineered in order to create the CAD models some of the answers may have been influenced by the fact that the product had previously been disassembled.

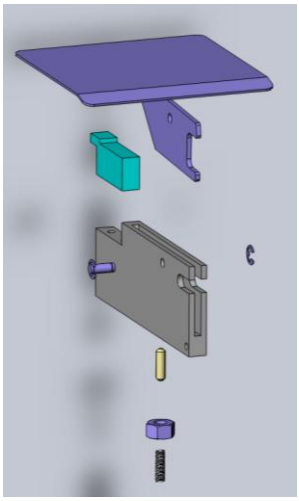

Figure 6: Exploded view of OEM Wide Flag Assembly

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 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.

4.3. Training of Mate Complexity DFA Method

The research motivating the focus of this paper utilized ANNs to increase the accuracy of the original connectivity complexity DFA method. The basic overview of the previous research is that the physical connectivity complexity graphs of twenty four assemblies were manually put together with a respective MTM DFA based assembly time assigned to it. Each of the connectivity graphs was generated through a Matlab algorithm to generate twenty-nine different complexity metrics. The nineteen of the twenty-four complexity metrics for each assembly became the inputs for the generated ANN and the respective MTM assembly times were the targets. The remaining five of the twenty-four assemblies were left out for testing after the Neural Networks were generated. The ANN consisted of 189 neural network architectures using up to 15 neurons, up to 3 layers, and 100 test repetitions[2]. The reason each architecture was trained (repeated) 100 times is that when each architecture is given the same input it could possibly generate a different output. The 100 different outputs for a given architecture were then used to generate a probability density plot to determine the probability that the given architecture would have predicted the target assembly time. Once the ANN was generated it was tested on five connectivity complexity metric test inputs to determine what assembly times the given ANN would predict[2].

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. The architectures generated for this research consisted of 189 with 15 neurons, 3 layers, and 100 tests. The training inputs for the ANN training cases generated in this research consisted of eleven sets of complexity values 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 values for that product would be trained towards the two different assembly times. The Automatic Mate Extraction Add-in discussed in Section 0was used to extract the mate connections from the ten SW assembly models listed in Table 3. These mate connection graphs were then run through a Matlab Complexity Algorithm to generate twenty-nine complexity metric values. Three of the products shown in Table 3 were held back to be used as test inputs once the ANN training cases were created. These three products were the Pencil Compass, the virtual 6 Inch MagLight, and the virtual Black and Decker Drill.

Three separate ANN training cases (Case 1,Case 2, and Case 3) were developed for this research to determine if the number of mates, determined by the level of mate definition, has an effect on the results. Case 1 was generated using complexity metrics 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. Case 2 was generated using complexity metrics 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. Case 3 was generated using both the complexity metrics 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 TESTING

Once the different ANN training cases using the given inputs and targets was generated they had to be tested. The complexity metrics from the three products held back for testing were given to the training cases 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 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 7. The training target is the vertical red line on the plot and +/- 25% of the target which is the desired range is shown by the vertical dashed red lines. Figure 7 shows an example probability density plot generated by a given architecture for a given product. This figure shows that the majority of the predicted assembly times for this product fall within the +/- 25% range.

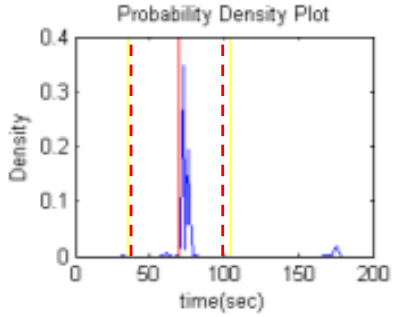

Figure 7: Example Probability Density Plot

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 4 shows the top five architectures selected for the three training schemes: Case 1, Case 2, and Case 3.

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.

Table 4: Selection of top 5 ANN Architectures

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

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 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 bold 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: Comparison of Predicted Assembly Times for Different ANN

Product Test Case	Level of Definition (Test)	Target Time (s)	CAse1 (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.

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 requiring 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 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. 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, 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.

6 CONCLUSIONS AND FUTURE WORK

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 those 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 a different way. To determine if different mating schemes have an effect on the results of the ANN predicted assembly times two different mating schemes were tested. A designer was 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 values were generated for the partially defined assembly. The designer was then asked to continue adding mates until the model was fully defined. The fully and partially defined complexity graphs were then given to the Case 3 to evaluate and compare the predicted assembly times. The predicted assembly times and the percent error for each mating scheme are shown in Table 6. A detailed study with different designers' mating configurations is reserved for future work.

Table 6: Mate configuration comparison

Product	Level of Definition	Target Time (sec)	Predicted Time (sec)	% Error
MagLight Mate Conf. 1.	FD	75.4	85.5	13.4
	PD		75.0	0.4
MagLight Mate Conf. 2	FD		79.8	5.8
	PD		85.0	12.7

While this paper 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 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.

7 REFERENCES

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

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

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

[4] Geoffrey Boothroyd, "Product design for manufacture

and assembly," *Computer-Aided Design*, vol. 26, no. 7, pp. 505-520, 1994.

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

[6] 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.

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

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

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

[10] 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.

[11] 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.

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

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

[14] 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.

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

[16] 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.

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

[18] 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.

[19] 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.

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

[21] Microsoft Corporation. (2012) Microsoft COM: Component Object Model Technologies. [Online]. http://www.microsoft.com/com/default.mspx

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

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

[24] F Ameri, J.D. Summers, G.M. Mocko, and M. Porter, "Engineering design complexity: an investigation of methods and measures," *Research in Engineering Design*, vol. 19, no. 2-3, pp. 161-179, November 2008.

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

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

[27] Dassult Systems. (1995-2012) SolidWorks Help. [Online]. http://help.solidworks.com/2010/english/SolidWorks/sldworks/LegacyHelp/Sldworks/Assem_1/Mates_Overview.htm?id=5e03254f145f404eb2993bae3dc45261#Pg0

