Identifying Product Families Using Data Mining Techniques in Manufacturing Paradigm

Israt J. Chowdhury and Richi Nayak
School of Electrical Engineering and Computer Science
Queensland University of Technology
Brisbane, Australia
{israt.chowdhury, r.nayak}@qut.edu.au

Abstract

Identifying product families has been considered as an effective way to accommodate the increasing product varieties across the diverse market niches. In this paper, we propose a novel framework to identifying product families by using a similarity measure for a common product design data BOM (Bill of Materials) based on data mining techniques such as frequent mining and clustering. For calculating the similarity between BOMs, a novel Extended Augmented Adjacency Matrix (EAAM) representation is introduced that consists of information not only of the content and topology but also of the frequent structural dependency among the various parts of a product design. These EAAM representations of BOMs are compared to calculate the similarity between products and used as a clustering input to group the product families. When applied on a real-life manufacturing data, the proposed framework outperforms a current baseline that uses orthogonal Procrustes for grouping product families.

Keywords: Product families BOM, frequent mining, matrix representation, and clustering.

1 Introduction

Agile manufacturing has resulted in mass customization and product proliferation, which consequently increases the number of products and part variations extensively. Simultaneously the current business climate demands for moving a product quickly from concept-to-market by reducing the product development lead time. A key element of shortening this lead time is the ability to use existing knowledge and designs to generate new variations of existing products, which ensure a reduction in time-to-market (Utterback & Meyer, 1993). Therefore, the concept of grouping product families has been introduced. Besides leveraging product development cost, this grouping can offer multiple benefits including reduction in new product launching risks, improved ability to upgrade products, and enhanced flexibility and responsiveness of manufacturing processes (Sawhney, 1998). For example, if two products have 45 out of 50 parts common, design of the similar parts can be reused and positioned for assembly early so that the remaining five parts can be added to the assembly when an order for a specific assembly has arrived. Exploring similarity among products may lead to the redesign of some parts.

Nowadays, with the advent of cheap storage and fast computer, a huge amount of data is generated during product design and development in a manufacturing system. The ability beyond human is required to process this huge amount of complex data into useful knowledge such as common product family information. The identification of product families is a non-trivial task due to the volume and complexity of the available data. A well-known historical approach of grouping product families is Group Technology (GT) (Harhalakis, Kinsey & Minis, 1992; Marion, Rubinovich & Ham, 1986). However, the practical acceptance of GT has been limited in modern manufacturing (Romanowski & Nagi, 2002; Romanowski & Nagi, 2005), as it requires enormous effort to do groupings due to the involvement of manual intermittent steps for developing a "coding system" to summarize the key design and other attributes. Some efforts have been made towards automation (Iyer & Nagi, 1997), but acceptable performance is not reached yet, especially for situations where the sheer volume of data becomes overwhelming for both human and systems.

Data mining techniques have been specifically designed to deal with massive amount of data automatically (i.e. without human intervention) and to identify meaningful patterns and dependencies hidden behind the data. However, due to the complex nature of the data generated in product design domain, existing data mining algorithms require modifications. Although data mining algorithms have been specifically written to effectively analyse large datasets, the product design data often cannot be simply "plugged in" to these programs (Romanowski, Nagi & Sudit, 2006).

Bill of Materials (BOM) is a common product design data used in various domains like mechanical, electrical, electronic and civil/infrastructure. BOM is a hierarchical, structured representation of the products that details information such as parts descriptions, raw materials, quantities, manufacturing details, production times, etc. (Romanowski & Nagi, 2005). Researchers and practitioners have started using BOM specifications more commonly to represent their data (Matías, Garcia, Garcia & Idoipe, 2008). It has become essential to propose similarity measures for BOM data to determine similarity between product designs, which will eventually lead to find effective groupings of product families.

For BOM data, the critical information lays in the recursive parent-child relationships between the end item, its components or subassemblies, and the raw (or

purchased) materials they contain. This information can naturally be depicted in rooted labelled unordered tree format. In this paper we represent BOM data as unordered trees and introduce a novel matrix form called Extended Augmented Adjacency Matrix (EAAM) for equivalent tree representation. This representation facilitates search for similar designs and thus reduces the time consumption between concept and product launch. Our approach is to utilize the data mining techniques like frequent mining and clustering for ensuring efficient similarity calculation and reducing the search space for finding similar groups. Using frequent mining allows finding frequent structural dependencies like parent-child in a particular database, which gives the list of most occurred BOM parts or components relations. This information is then used with other content and topological information such as optimal part encoding, hierarchical position or level, and part quantity, in clustering. Using EAAM representations of BOM data, cosine similarity measure is used to generate a similarity matrix that becomes input to a clustering algorithm for identifying the product families.

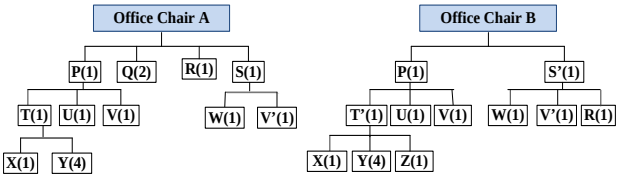
When applied on a real-life manufacturing data, the proposed framework including the BOMs similarity measure method has proven to excel in solving the problem of grouping product families automatically. The results are also compared with a current baseline that uses orthogonal Procrustes (Shih, 2011) for finding the product families and the proposed framework clearly outperforms.

Road map: In the following section, the related work is discussed. In section 3, the background knowledge is presented. In section 4, the proposed method for BOM similarity measure and the framework for identifying product families are given. The results are discussed in section 5. In section 6, the conclusion is drawn.

2 Related Works

Many efforts have been made for grouping the product families based on similarity schemes with emphasis on the different design areas and manufacturing. Most of them have focused on the historical approach of grouping individual parts into families, called as Group Technology (GT) (Harhalakis, Kinsey & Minis, 1992; Marion, Rubinovich & Ham, 1986)). The practical acceptance of GT has remained limited due to the expensive coding system development for summarizing the key product design and manufacturing attributes for doing the grouping. The main limitation of GT is the manual coding system. Though some efforts have been made towards automation, still more improvements are needed. Later, Authors in (Kao & Moon, 1991) used a back-propagation neural network based method for the product family grouping, but kept the existing GT classification and coding system. Another automated retrieval and ranking process for finding similar parts was proposed by authors in (Iyer & Nagi, 1997), but again based on GT coding. Authors in (Lee-Post, 2000) employed genetic algorithm to form the families, however, this approach also required to use the existing classification and coding scheme.

Instead of using information derived from a fixed GT code; some methods proposed similarity based on product functional features. Authors in (Chen, Chen, Wang & Chen, 2004) used the adaptive resonance theory (ART) neural network to develop a functional feature-based similarity method for grouping product families. Authors in (Liu, Yang, Bai & Tan, 2008) introduced another functional similarity-based combinatorial design method to produce a variety of products that satisfy various customer requirements in time. However, these functional feature-based schemes did not consider the hierarchical product design features. Authors in (Shih, 2011) attempted to calculate the similarity between BOMs considering the shape or geometrical structure, where a matrix representation and orthogonal Procrustes method were used to calculate the similarity score for grouping the product families. But BOMs are very flexible in shape, since there is no common rule or template to follow for generating them, therefore looking for geometrical or exact shape difference may give false similarity score. Emphasis should be put on the significant structural dependencies, hierarchical positions and other important contents during similarity calculation. The proposed framework in this paper focuses on the above for identifying the product families. To our best of knowledge, this is the first work on BOM data to determine product families using data mining.



Nodes meanings: P=Seat, Q=Elbow rest, R=Lumbar support, S and S'=Back variation, T and T'=Under frame variation, U=Seat frame, V and V'=Upholstery variation, W=Back frame, X =Standard, Y=Wheel, Z=Footrest

Figure 1: Variants of office chair

3 Background Knowledge

3.1 Bill of Materials (BOMs)

BOM represents hierarchical relations between various product parts with necessary details of manufacturing a particular product. It is a structural representation of a product including its required subassemblies, components and parts at various levels of production (Clement, Coldrick & Sari, 1992). To understand the proposed framework, following definitions need to be considered.

Definition 1 (*End Items*): The entities that are sold directly to the customer without any further value added manufacturing step. End items usually contain several subassembly parts and raw materials and appear at the top of the BOM hierarchal position.

Definition 2 (*Subassemblies*): These are the entities that cannot be sold to the customer. Subassemblies may contain manufactured or purchased part or other subassemblies, and therefore, are appeared at a level of BOM hierarchy which is positioned neither at the top nor at the bottom.

Definition 3 (*Purchased Parts*): The raw materials which are the initial entities for finishing a final product. Purchased parts are positioned at the bottom level of the BOM structure.

Definition 4 (*Quantity Representation*): In BOM, repeated subassemblies or parts are represented by a quantity per value. This value is the number of the part required per one unit of the part's parent.

Definition 5 (*Part Number*): This is an alphanumeric string that uniquely identifies an end item, subassembly and a purchased part. Each number corresponds to a specific item with specific characteristics.

Properties of BOM: BOM structures can be different for the identical end items, as each end item may be designed by a different company. Moreover, the product design is the result of human made input and developed completely based on individuals' understandings of how the product is manufactured or assembled. Similar BOMs may have different structures with same parts appearing at different level. However they will share similar components or parts and, most importantly, the structural dependencies among them will be usually kept same (Figure 1). BOMs substructures are unordered which means that the order of components is not significant. For instance, it does not matter if we say a chair has a seat, elbow rest and wheel, or a chair has a wheel, seat and elbow rest. In this paper we depicted BOM as rooted labelled unordered tree.

Definition 6 (*Unordered Tree*): A rooted labelled unordered tree has an identical root node and preserves only ancestor-descendant or parent-child relationships among nodes. There is no left-to-right order among the sibling nodes.

3.2 Data Mining Techniques Used

To satisfy the need of mass customization and agile manufacturing, we need to apply techniques that will extract implicit, previously unknown, potentially useful and understandable pattern from a large database (Fayyad, Piatetsky-Shapiro & Smyth, 1996), thus the product design and manufacturing system will have substantial improvement. Using data mining techniques in advance manufacturing is becoming popular (Choudhary, Harding & Tiwari, 2009). In the proposed framework, we have used frequent mining and clustering, two well-known data mining techniques for finding similarities between products and grouping them into families.

Frequent mining is used to extract interesting patterns from a database using a specified support (Chowdhury & Nayak, 2014a, 2014b). Support determines how often a pattern is applicable to or appears to a given data set. It represents the probability that a database instance contains that pattern.

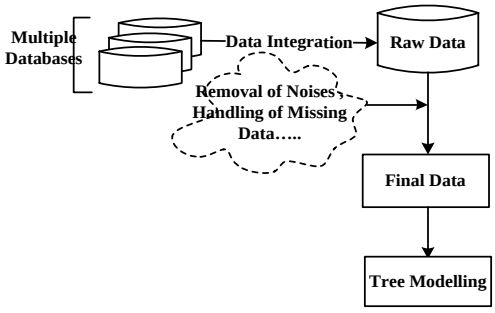


Figure 2: Data Pre-Processing Steps

BOM consists of structural dependencies like parent-child and ancestor-descendant relations between the end item, its components or subassemblies, and the raw (or purchased) materials they contain. The main challenge in BOM data analysis is dealing with the flexibility in its representation. It is very hard to put BOM data into a common format, thus the accurate analysis like similarity comparison can be carried out. Apparently, in BOM no other information keeps constant except the structural dependencies. So, instead of considering geometrical structure and shape, understanding structural dependencies is crucial for BOM similarity comparison. We utilise frequent mining to extract common structural dependencies in a database, which can be used as important representational component of the BOM data. These common structures can be input to clustering along with other information about the BOM data.

Clustering is an unsupervised data mining technique that can group objects based on their common characteristics, without the presence of any prior information about classification (Algergawy, Mesiti, Nayak & Saake, 2011; Kutty, Tran, Nayak & Li, 2008). Without using domain knowledge and GT coding based classification, the identification of product families can be possible using clustering. Clustering is now commonly used in manufacturing domain for doing unsupervised grouping (Ye & Gershenson, 2008). To apply clustering, a similarity measure value needs to be calculated based on commonality of the features. In this work, we utilise cosine similarity (Cha, 2007) to determine a similarity matrix based on the equivalent Extended Augmented Adjacency Matrix (EAAM) of a BOM dataset.

4 Proposed BOM Similarity Framework for Identifying Product Families

In this section a method of similarity measure between two BOM data instances is presented. A framework is then proposed integrating the similarity measure for identifying product families.

4.1 Data Pre-processing

To begin with our approach it is necessary to pre-process BOM data in order to make it useful for knowledge discovery. Figure 2 shows the tasks, which are used in this process.

4.1.1 Final Data

A company's database generally consists of a lot of data records. Only those records that correlate closely with the mining purpose are taken into account. Mostly BOM records are found in a tabular form, which typically contains the part name, part no, part revisions, part manufacturing description and the quantities required building a product assembly (as shown in Figure 3). Usually, the BOM input is given by human in spreadsheet, that can be formatted however one likes, but as anyone can format them, it often results in inconsistencies across a company's BOMs. Hence for mining BOM data, these inconsistencies need to be removed. Moreover not all of the information comprised by BOMs is necessarily mined for knowledge discovery. Therefore, once received the raw data through integration

of multiple databases, the final data sets should be identified involving such data cleaning and filtering tasks as removal of noises, handling of missing data files, etc.

Unordered Tree	BOM data
Root node	End item
Parent or ancestor node	End item and subassemblies
Child or descendant node	Subassemblies and purchased parts
Leaf node	Purchased parts
Parent-child or, ancestor-descendant relationships	End item-subassembly or, end item-purchased part or, subassembly-purchased part relationships
Node label	Part number

Table 1: Considered Mapping for BOM to Unordered Tree Representation

4.1.2 Unordered Tree Representation

After identifying final BOM data, tree modelling is done to support the EAAM construction. This modelling is carried out by using unordered tree structure scheme as template, where only parent-child and ancestor-descendent relationships are important. The BOM data can naturally be represented as unordered tree. By considering the parent-child and ancestor-descendant relationships between end item, subassemblies and purchased parts, a mapping can be derived.

Table 1 shows a general mapping that can be used to represent the BOM data as unordered tree. The end item, or finished product, can be considered a root of the tree; manufactured or assembled components can become the nodes; purchased parts or raw materials can be the leaf nodes. For example, in figure 3 the tabular or indented BOM of an ABC Lamps Product-LA01 (Fogarty, Blackstone & Hoffmann, 1991) is given, where the lamp is the end product, and the parts given under first column are different subassemblies and purchased parts. For constructing a tree from this BOM only the relationships among various parts are important, such as B100, S100 and A100 are the children of the end item; 1100, 1200, 1300, 1400 are the children of B100, representing descendants of the end item.

For node labelling, part numbers are used. If we compared two BOMs of product Lamp, using part numbers as labels, two BOMs would only match where the part numbers were exactly the same. For instance, suppose part S-14 is a shade with I.D. = 14” (inch). Part S-18 is a shade with I.D. = 18” (inch). These two shades would not be matched because of the unique part numbers. However, we are interested in finding BOMs of similar nature even if they do not share exact content and topology. For this reason, we replace the part numbers with general node labels derived from the part characteristics and types. In the case of these two parts, we would replace the unique part labels with a single label S for the class of shades.

4.2 Finding Frequent Structural Relationship

The objective of the proposed framework is to form the product families based on the existing product models (BOMs). Due to the vast flexibility in BOM data, characterizing structural relationships based on frequent occurrence is essential to include in the global similarity calculation as in some cases, frequent-infrequent decision are used as a scale to measure the importance of the structural relations (Chi, Muntz, Nijssen & Kok, 2004). We consider these relationships as a representational component for the BOM dataset. We explain next how these relationships are derived.

ABC Lamp Company, Indented Bill of Materials, Lamp LA01.

Part number	Description	Quantity for each assembly	Unit of measure
B100	Base assembly	1	Each
1100	Finished shaft	1	Each
2100	3/8" Steel tubing	26	Each
1200	7"-Diameter steel plate	1	Inches
1300	Hub	1	Each
1400	1/4–20 Screws	4	Each
S100	14" Black shade	1	Each
A100	Socket assembly	1	Each
1500	Steel holder	1	Each
1600	One-way socket	1	Each
1700	Wiring assembly	1	Each
2200	16-Gauge lamp cord	10	Feet
2300	Standard plug terminal	1	Each

Figure 3: ABC Lamps Product-LA01 (Fogarty, Blackstone, & Hoffmann, 1991)

4.2.1 Tree traversal

Prior to implement frequent subtree mining algorithm, an optimal traversal (Chowdhury & Nayak, 2013) algorithm is used to ensure unique identity or canonical form (Valiente, 2002) of each product model, which is in unordered tree form. Optimal traversal is included as it ensures optimality by providing unique encoding within minimum computation time (Chowdhury & Nayak, 2013).

4.2.2 Frequent Mining Algorithm

Once the canonical form is built, the frequent mining can now be applied that permits not only to explore the relationships and dependencies but also to handle a huge amount of data in an optimal way (Chowdhury & Nayak, 2014a, 2014b). However, such algorithms are sometimes limited to the memory because of its size and calculations that they perform. The candidate frequent subtrees generation can be exponential in large databases (Chi et al., 2004).

We propose to apply the BOSTER algorithm (Chowdhury & Nayak, 2014b) which allows setting the subtree length equal to 1 and retrieves only single relationships exhibiting between parent-parts. This algorithm has proved to be memory efficient and exhibits limited computational complexity (Chowdhury & Nayak, 2014b). A support threshold is needed for frequent subtree mining process. A minimum support is set by trial and error, as it is a data specific parameter that prunes the infrequent subtree.

4.2.3 Characterizing Structural Relationships

Based on the result of the frequent subtree mining algorithm the structural relationships are characterized. If a subtree is frequent then the inherent parent child relation is considered as mandatory. Once all mandatory parent-child relationships are identified, the remaining parent-child relationships are classified as optional. During the EAAM representation a weighted value of 1

and 0 are used to represent the mandatory and optional relationship respectively.

4.3 Extended Augmented Adjacency Matrix (EAAM) Representation

In this paper a new matrix representation called EAAM is introduced. Although, EAAM is an extension of Augmented Adjacency Matrix (AAM) representation (Chowdhury & Nayak, 2013), but to our best knowledge, this is the first matrix, where the frequent structural relationship is included as one of the representational components. The rest of the components are:

- Optimal part sequence of BOM using optimal traversal.
- Part level information from BOM interface.
- Quantity representation (q) representing the number of the part required per unit of the part's parent.

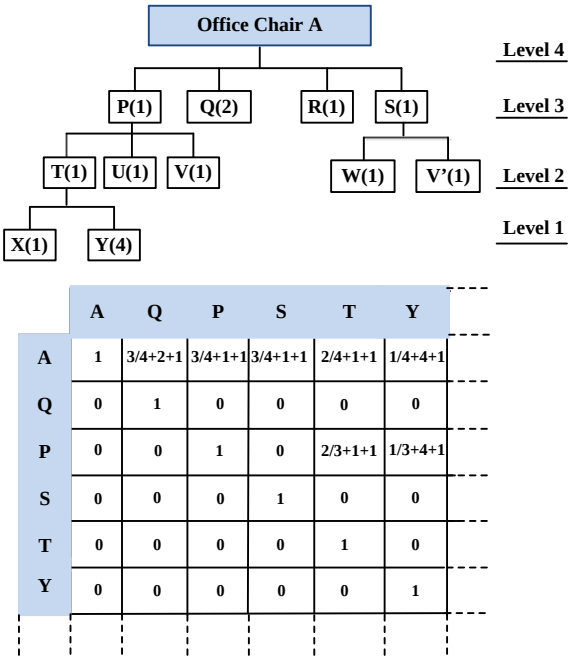
An adjacency matrix of a tree is based on the ordering chosen for the nodes (Rosen, 2011). For EAAM the ordering is achieved using optimal traversal (Chowdhury & Nayak, 2013) which ensures unique encoding of BOM represented in unordered tree form. For populating the cell of EAAM mainly structural relationship importance weight, level information and quantity representation are used.

Let a BOM, B is depicted as a rooted labelled unordered tree $B = (I, R)$, where $I = \{i_0, i_1, i_2, \ldots, i_n\}$ denotes the set of items with i_0 as end item, and other set elements as subassembly and purchased items, $R = \{(i_1, i_2)|i_1, i_2 \in I\} = \{r_1, r_2, \ldots, r_{n-1}\}$. The number of each item is given as $\{q_0, q_1, q_2, \ldots, q_n\}$. For B, the EAAM representation can be formulated in which a cell, a_{cd} is populated as follows:

$$a_{cd} = \begin{cases} 1 & \text{if } i_c \text{ is a node of } B \\ \dfrac{L(B,i_d)}{L(B,i_c)} + q_d + 1 & \text{if } i_c \text{ is an ancestor or parent of } i_d \text{ and the relation is frequent} \\ \dfrac{L(B,i_d)}{L(B,i_c)} + q_d + 0 & \text{if } i_c \text{ is an ancestor or parent of } i_d \text{ and the relation is not frequent} \\ 0 & \text{otherwise} \end{cases}$$

These four components are explained as follows:

1. To represent the presence of each part in a BOM, each diagonal cell is populated with 1.
2. If the part is parent or ancestor of the other respective part, and the parent-part relation is frequent then the cell is populated with level information (fraction of level of corresponding two parts), quantity representation of the child or descendant node and the mandatory structural relationship weight value equals 1.
3. If the part is parent or ancestor of the other respective part, and the parent-part relation is not frequent then the cell is populated with level information (fraction of level of corresponding two parts), quantity representation of the child or descendant node and the optional structural relationship weight value equals 0.
4. If none of these are true, then the cell receives a value of 0.



	A	Q	P	S	T	Y
A	1	3/4+2+1	3/4+1+1	3/4+1+1	2/4+1+1	1/4+4+1
Q	0	1	0	0	0	0
P	0	0	1	0	2/3+1+1	1/3+4+1
S	0	0	0	1	0	0
T	0	0	0	0	1	0
Y	0	0	0	0	0	1

Figure 5: EAAM Construction

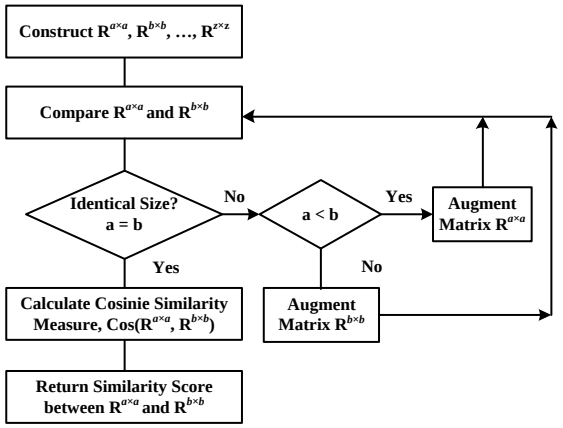


Figure 6: The Flow Chart of Calculating Similarity

Example: From figure 1, we consider the first example BOM of product model "office chair *A*" to explain the EAAM construction. Consider a BOM database that only consists of two BOM trees given in figure 1, and the minimum support is two. It means that if a subtree appears twice or more in the database, it will be considered as a frequent sub-tree. Based on this, *A-Q*, *A-R*, *A-Z*, *T-Z* and *S-R* are found infrequent relationships and considered as optional. The order of the nodes for constructing EAAM is derived using optimal traversal. Consider the cell between nodes *A* and *Q*. For this BOM tree, *A* is the parent of part *Q*, therefore the level information is added as 3/4, where the level of *A* is 4 and the level of *Q* is 3. For the child part *Q*, the quantity representation value is 2, which is added after the fraction of level into that cell. Finally, the frequent parent-part relation adds a value 1 to indicate the mandatory relationship. The overall calculated value for this cell is 3/4+2+1. The rest of the cell values are calculated following the same way.

4.4 BOM Similarity Measure

After constructing EAAMs, we use cosine similarity for matrix comparison for measuring the similarities between a BOM pair (Chowdhury & Nayak, 2013) as follows:

$$\cos(\mathrm{A},\mathrm{B}) = \frac{\sum_{x=1}^{n}\sum_{y=1}^{n} \mathrm{A}_{xy}\,\mathrm{B}_{xy}}{\sqrt{\sum_{x=1}^{n}\sum_{y=1}^{n} \mathrm{A}_{xy}^2}\sqrt{\sum_{x=1}^{n}\sum_{y=1}^{n} \mathrm{B}_{xy}^2}}$$

Where, A and B are two (n×n) matrices.

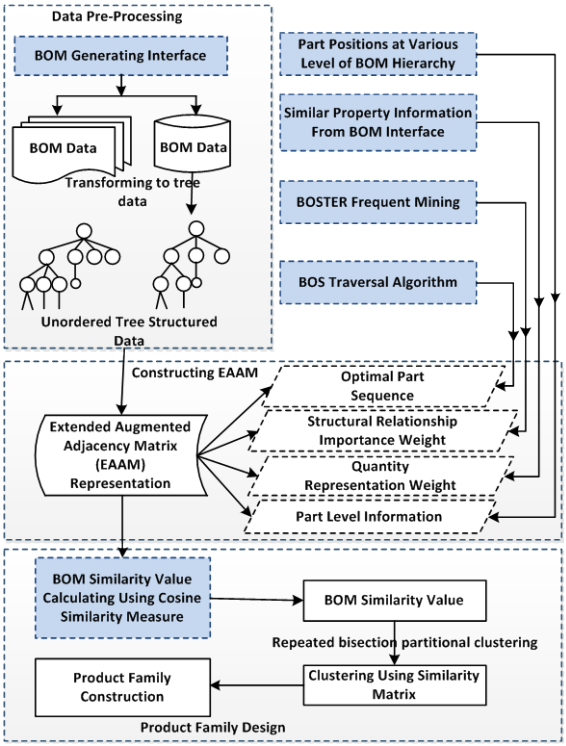


Figure 7: Framework for Product Family Design

If sizes of the two BOM trees are not same, then additional columns and rows with zero elements are padded to the smaller matrix for making the size of both matrices equal, this is called the augmentation of matrix. These two square matrices can be considered as two $|B|\times|B|$ (where $|B|$ = max $\{B1, B2\}$; B_1, B_1 are two BOM trees) dimensional vectors. The overall procedure for similarity measure is given in figure 6 using a flow chart, where matrix is represented as $\mathbf{R}^{a\times a}$, where a is the size of that matrix representing the number of the components or parts in a BOM tree.

4.5 The Proposed Framework

The proposed framework for grouping product families has three main phases as shown in figure 7. In the first phase data pre-processing is done. BOM has different storage under different enterprises; some of them store BOM data in database, some in files like XLS file. Some enterprises use part table/relationship table to express BOM, and some enterprises use a single table. All these variations need to save in memory as a BOM generating interphase, from this node the pre-processing will carry out in next.

Next phase covers the EAAM construction where all necessary steps (dotted blue boxes) are implemented for populating the feature weights.

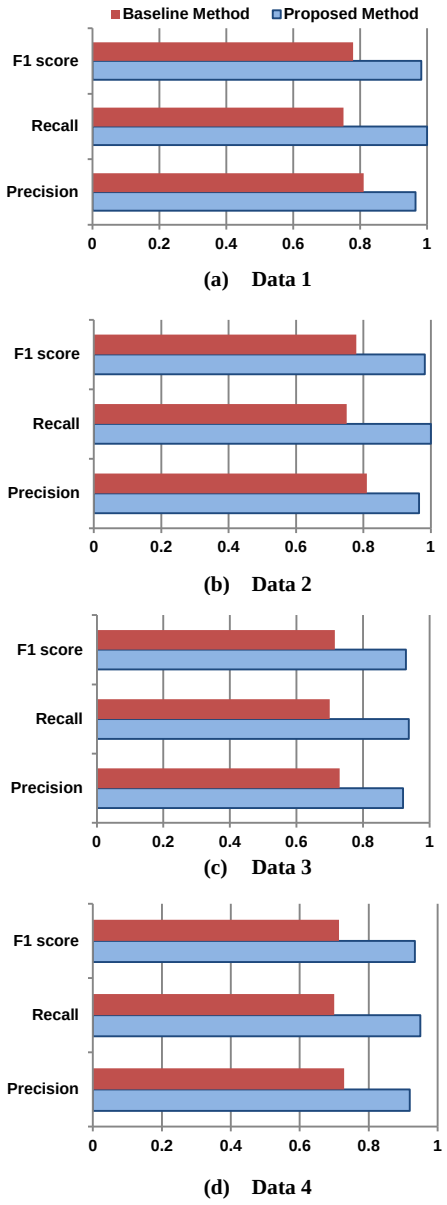


Figure 8: Accuracy Performance over Data 1(a), Data 2(b), Data 3(c) and Data4(c)

In the third and final phase, the pairwise similarity is calculated using the EAAM comparison and a similarity score is calculated between BOM pairs where a similarity score of 0 means completely dissimilar and a score of 1 means exactly similar. Using this similarity values a similarity matrix is constructed which is then employed as an input to a clustering algorithm. Table 2 shows an example of the similarity matrix. We used a well-known clustering algorithm, Repeated Bisection Partitioning (Rasmussen & Karypis, 2004), for grouping the BOMs into families. This algorithm divides trees into two groups and then selects one of the larger groups according to a clustering criterion function and bisects further. This process is repeated until the desired number of clusters is achieved. During each step of bisection, the cluster is bisected so that the resulting 2-way clustering solution locally optimizes a particular criterion function. Other clustering algorithms can also be applied. Finally from the cluster result, the product families will be identified.

	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15
B1	1.00	0.40	0.43	0.57	1.00	0.50	0.61	1.00	1.00	0.64	0.41	0.30	0.41	0.58	0.44
B2	0.40	1.00	0.43	0.47	0.40	0.49	0.37	0.40	0.40	0.42	0.32	0.43	0.32	0.54	0.39
B3	0.43	0.43	1.00	0.65	0.43	0.53	0.43	0.43	0.43	0.45	0.39	0.44	0.39	0.52	0.33
B4	0.57	0.47	0.65	1.00	0.57	0.70	0.60	0.57	0.57	0.71	0.50	0.35	0.50	0.63	0.34
B5	1.00	0.40	0.43	0.57	1.00	0.50	0.61	1.00	1.00	0.64	0.41	0.30	0.41	0.58	0.44
B6	0.50	0.49	0.53	0.70	0.50	1.00	0.62	0.50	0.50	0.71	0.65	0.34	0.65	0.71	0.35
B7	0.61	0.37	0.43	0.60	0.61	0.62	1.00	0.61	0.61	0.58	0.56	0.33	0.56	0.58	0.41
B8	1.00	0.40	0.43	0.57	1.00	0.50	0.61	1.00	1.00	0.64	0.41	0.30	0.41	0.58	0.44
B9	1.00	0.40	0.43	0.57	1.00	0.50	0.61	1.00	1.00	0.64	0.41	0.30	0.41	0.58	0.44
B10	0.64	0.42	0.45	0.71	0.64	0.71	0.58	0.64	0.64	1.00	0.56	0.31	0.56	0.72	0.39
B11	0.41	0.32	0.39	0.50	0.41	0.65	0.56	0.41	0.41	0.56	1.00	0.25	1.00	0.47	0.31
B12	0.30	0.43	0.44	0.35	0.30	0.34	0.33	0.30	0.30	0.31	0.25	1.00	0.25	0.42	0.31
B13	0.41	0.32	0.39	0.50	0.41	0.65	0.56	0.41	0.41	0.56	1.00	0.25	1.00	0.47	0.31
B14	0.58	0.54	0.52	0.63	0.58	0.71	0.58	0.58	0.58	0.72	0.47	0.42	0.47	1.00	0.31
B15	0.44	0.39	0.33	0.34	0.44	0.35	0.41	0.44	0.44	0.39	0.31	0.31	0.31	0.31	1.00

Table 2: BOM Similarity Matrix

5 Evaluation of the Proposed Framework

We implemented the proposed framework on a real manufacturing data to evaluate the performance.

This data is collected from a manufacturer of nurse calling devices (Romanowski & Nagi, 2004). It consists of 404 BOMs with four major product families. From this data set we randomly generated four samples, consisting 100 BOMs each and named them as Data 1, Data 2, Data 3 and Data 4. We used all these four datasets for empirical analysis.

For benchmarking we consider a method that used the orthogonal Procrustes problem to find the orthogonal matrix for two given matrices that will closely map one matrix to another and used this as a geometrical similarities between BOMs and then clustered them into families (Shih, 2011). For the benchmark method we used the same clustering algorithm, but we used the orthogonal Procrustes based similarity measure as input and performed the product grouping. Finally we checked the clustering results with the known product family information and compared the performances.

The main contribution of this paper is the similarity measure method of product BOMs. An efficient grouping of product families largely depends on an efficient similarity measure method. We evaluated our similarity measure approach using the well-known evaluation metrics including precision, recall and F1 score (Goutte & Gaussier, 2005) and performed on all four data samples. For these metrics, the value close to 1 is considered as an indication of better performance. From figure 8, we can see for all four data sets our proposed similarity measure method gives higher accuracy in comparison to the benchmark method. This good accuracy performance should also reflect during the clustering process, as we used this similarity method as an input for an off-the-self clustering algorithm for doing the product family grouping. Table 2 gives a partial view of the similarity matrix generated by our proposed BOM similarity measure method. For clustering we used this similarity matrix for identifying product families.

Table 3 reports the clustering performance results, where we mainly included the number of mis-clustered product BOM for each data by the proposed method and the benchmarked method. The proposed framework outperforms the baseline method.

Method	Data 1	Data 2	Data 3	Data 4
Proposed Framework	2	5	5	6
Baseline Method	19	21	25	35

Table 3: Number of Mis-Clustered BOMs for Different Data Sets

6 Conclusion

A product family is a group of related products based on a product platform, facilitating mass customization by cost-effectively providing a variety of products for different market segments. In this paper we present a data mining approach based framework for grouping various products into families. We introduced a similarity measure method for a common product data type, BOM that can be used to cluster products into families. The benchmarking results confirm the efficiency of the proposed work.

In future work, we intend to expand the study on unifying the families into a single Generic Bill of Material (GBOM) (Hegge & Wortmann, 1991) group.

7 References

Algergawy, A., Mesiti, M., Nayak, R., & Saake, G. (2011). XML data clustering: An overview. *ACM Computing Surveys (CSUR)*, **43**(4): 25.

Cha, S.-H. (2007). Comprehensive survey on distance/similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences,* **1**(4): 300-307.

Chen, Y., Chen, Y., Wang, C., & Chen, T. (2004). Using artificial neural networks to develop a mechanism for functional feature-based reference design retrieval.

Proc. IEEE International Conference on Engineering Management **2**: 829-833. IEEE.

Chi, Y., Muntz, R. R., Nijssen, S., & Kok, J. N. (2004). Frequent Subtree Mining - An Overview. *Fundamental Informatic.*, **66**(1-2): 161-198.

Choudhary, A. K., Harding, J. A., & Tiwari, M. K. (2009). Data mining in manufacturing: a review based on the kind of knowledge. *Journal of Intelligent Manufacturing*, **20**(5): 501-521.

Chowdhury, I. J., & Nayak, R. (2013). A Novel Method for Finding Similarities between Unordered Trees Using Matrix Data Model. *Proc. WISE 14th International Conference on Web Information Systems Engineering*, **8180**: 421-430. Springer Berlin Heidelberg.

Chowdhury, I. J., & Nayak, R. (2014a). BEST: An Efficient Algorithm for Mining Frequent Unordered Embedded Subtrees (*In Press*) *Proc. PRICAI 13th Pacific Rim International Conference on Artificial Intelligence*, Gold Coast, Australia, **8862** Springer Berlin Heidelberg.

Chowdhury, I. J., & Nayak, R. (2014b). BOSTER: An Efficient Algorithm for Mining Frequent Unordered Induced Subtrees. *Proc. WISE 15th International Conference on Web Information Systems Engineering*, Athens, Greece, **8786**: 146-155. Springer Berlin Heidelberg.

Clement, J., Coldrick, A., & Sari, J. (1992). *Manufacturing Data Structures; Building Foundations for Excellence with Bills of Materials and.*. New York, John Wiley & Sons, Inc.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery: an overview. In *Advances in Knowledge Discovery and Data Mining*. 1-34 Fayyad, U. M., Piatetsky-Shapiro, G., Smyth, P. And Uthurusamy, R. (eds). AAAI Press/MIT Press.

Fogarty, D. W., Blackstone, J. H., & Hoffmann, T. R. (1991). *Production & Inventory Management*, South-Western Publishing Company.

Goutte, C., & Gaussier, E. (2005). A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation. In *Advances in Information Retrieval*. Vol. 3408, 345-359 Losada, D., Fernández-Luna, JuanM (eds). Springer Berlin Heidelberg.

Harhalakis, G., Kinsey, A., & Minis, I. (1992). Automated group technology code generation using PDES. *Proc. Third International Conference on Computer Integrated Manufacturing*, 4-14. IEEE.

Hegge, H. M. H., & Wortmann, J. C. (1991). Generic bill-of-material: a new product model. *International Journal of Production Economics*, **23**(1–3): 117-128.

Iyer, S., & Nagi, R. (1997). Automated retrieval and ranking of similar parts in agile manufacturing. *IIE Transactions*, **29**(10): 859-876.

Kao, Y., & Moon, Y. (1991). A unified group technology implementation using the backpropagation learning rule of neural networks. *Computers & Industrial Engineering*, **20**(4): 425-437.

Kutty, S., Tran, T., Nayak, R., & Li, Y. (2008). Clustering XML documents using closed frequent subtrees: A structural similarity approach. In *Focused Access to XML Documents*. 183-194(eds). Springer.

Lee-Post, A. (2000). Part family identification using a simple genetic algorithm. *International Journal of Production Research*, **38**(4): 793-810.

Liu, F., Yang, B., Bai, Z., & Tan, R. (2008). Research on product combinatorial design based on functional similarity. *International Journal of Design Engineering (IJDE)*, **1**(3): 333–356.

Marion, D., Rubinovich, J., & Ham, I. (1986). Developing a group technology coding and classification scheme. *Industrial Engineering*, **18**(7): 90-97.

Matías, J., Garcia, H. P., Garcia, J. P., & Idoipe, A. V. (2008). Automatic generation of a bill of materials based on attribute patterns with variant specifications in a customer-oriented environment. *Journal of Materials Processing Technology*, **199**(1): 431-436.

Rasmussen, M., & Karypis, G. (2004). gcluto: An interactive clustering, visualization, and analysis system, *UMN-CS TR-04* (Vol. 21).

Romanowski, C. J., & Nagi, R. (2004). A data mining approach to forming generic bills of materials in support of variant design activities. *Journal of Computing and Information Science in Engineering*, **4**(4): 316-328.

Romanowski, C. J., & Nagi, R. (2005). On comparing bills of materials: a similarity/distance measure for unordered trees. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, **35**(2): 249-260.

Romanowski, C. J., Nagi, R., & Sudit, M. (2006). Data mining in an engineering design environment: OR applications from graph matching. *Computers & Operations Research*, **33**(11): 3150-3160.

Rosen, K. (2011). *Discrete Mathematics and Its Applications 7th edition*, McGraw-Hill Science.

Shih, H. M. (2011). Product structure (BOM)-based product similarity measures using orthogonal procrustes approach. *Computers & Industrial Engineering*, **61**(3): 608-628.

Valiente (2002). *Algorithms on Trees and Graphs*. New York, Springer, Berlin Heidelberg.

Ye, X., & Gershenson, J. K. (2008). Attribute-based clustering methodology for product family design. *Journal of Engineering Design*, **19**(6): 571-586.