Proceedings of the ASME 2007 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2007
September 4-7, 2007, Las Vegas, Nevada, USA

DETC2007-35742

PROCESS PLATFORM FORMATION FOR PRODUCT FAMILIES

Roger Jianxin Jiao[*]
School of Mechanical and Aerospace Engineering
Nanyang Technological University, Singapore

Linda Lianfeng Zhang
School of Mechanical and Aerospace Engineering
Nanyang Technological University, Singapore

ABSTRACT

In accordance with the product families, process platforms have been recognized as a promising tool for companies to configure optimal, yet similar, production processes for producing different products. This paper tackles process platform formation from large volumes of production data available in companies' production systems. A data mining methodology based on text mining and tree matching is developed for the formation of process platforms. A case study of high variety production of vibration motors for mobile phones is reported to prove the feasibility and potential of forming process platforms using text mining and tree matching.

1 INTRODUCTION

One of the pressing needs faced by manufacturing companies nowadays is to produce quickly a high variety of customized products at low costs. The linchpin for companies to achieve efficiency in the resulting high variety production, thus surviving lies in their ability to production as stable as possible. Stable production can only be achieved by taking advantages of commonality and similarity inherent in various production processes when fulfilling individual products. Process platforms [1] have been recognized as a promising tool for companies to obtain and further maintain stable high variety production through configuring optimal, yet similar, production processes for producing new products while considering their existing manufacturing capabilities.

The authors provided a rigorous definition of process platforms along with the basic constructs and functionalities. However, they did not discuss further how to form process platforms in current manufacturing environment, where large volumes of production data are generated. This paper, thus, discusses process platform formation with existing production data in companies' production systems.

[*] Correspondence: Phone: +65 67904143, Fax: +65 67911859, Email: jiao@ieee.org

Reusing proven knowledge implicitly in existing production data suggests itself as a natural technique to facilitate the handling of production process changes and tradeoffs between design variations and process changes. Data mining excels in discovering previously unknown and potentially useful patterns of information, i.e., knowledge, from past [2]. Towards this end, this chapter introduces a data mining methodology to solve the problems in forming process platforms. To meet the challenges, specific data mining techniques, including text mining and tree matching, are adopted.

2 METHODOLOGY

While a product family refers to a set of individual products, performing a basic function, with a common product structure, a process platform is a set of production processes, assuming a common routing structure, to produce the set of individual products in a family. In accordance with a product family and the corresponding process family, a generic routing structure is fundamental to a process platform [1]. Hence, the generic routing structure is common to all individual production processes in the process family. Thus the problem of process platform formation can be converted to that of generic routing identification from large volumes of production data available in companies' production systems.

The data mining methodology consists of three consecutive parts, as shown in Figure 1. In the first part, the similarity of existing routings is measured. Based on the similarity measure results, in the second part, the similar routings are clustered into same families using a fuzzy clustering procedure. In the last part, the generic routings are formed for the corresponding families using a tree unification procedure developed in this research.

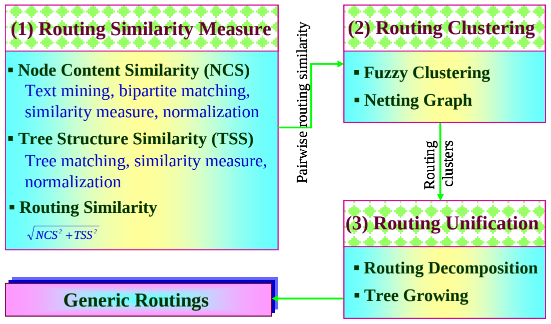

(1) Routing Similarity Measure
▪ Node Content Similarity (NCS)
Text mining, bipartite matching, similarity measure, normalization
▪ Tree Structure Similarity (TSS)
Tree matching, similarity measure, normalization
▪ Routing Similarity
$\sqrt{NCS^2+TSS^2}$
Pairwise routing similarity
(2) Routing Clustering
▪ Fuzzy Clustering
▪ Netting Graph
Routing clusters
(3) Routing Unification
▪ Routing Decomposition
▪ Tree Growing
Generic Routings

Figure 1. Overview of data mining methodology.

A routing consists of a number of product items, operations, and operations precedence relationships. Thus, the similarity of two routings involves two types: operations similarity (involving item similarity) and precedence similarity. In practice, a routing is represented by a tree precedence graph, in which nodes represent operations and arcs indicate the precedence relationships between two connected operations. Thus, measuring routing similarity can be decomposed into measuring node content similarity and tree structure similarity, which correspond to operations similarity and precedence similarity. Since node content similarity and tree structure similarity are two independent similarity measures, the Euclidian distance measure is adopted in this research to compute routing similarity, as shown in Figure 1. In this research, the methodology is applied to the basic representation trees: binary trees, as shown in Figure 2.

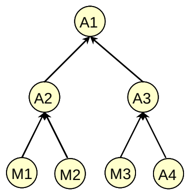


Figure 2. A binary tree representation of a routing.

Besides numerical data, operations are described by categorical data, e.g., specific shapes, materials, pertaining to product and process characteristics. To avoid the subjectiveness and imprecision associated with routings due to the categorical data, fuzzy clustering is adopted in this research to group similar routings into same families. The netting graph method [3] is incorporated in routing fuzzy clustering as well. In the last part, a procedure has been developed to identify the generic routings for the corresponding families obtained in the second part. It is accomplished through unifying other trees with an initial tree, i.e., the seed tree, one by one in a process family. Section 3 presents Part 1: Routing similarity measure; Section 4: Generic routing identification discusses Parts 2 and 3: Routing clustering and unification together.

3 ROUTING SIMILARITY MEASURE

Let $W=\{ROU_1, \ldots, ROU_P\}$ represent a set of existing routings; S_{rs} for the similarity between two routings, ROU_r and ROU_s; S_{rs}^{NC} for node content similarity; and S_{rs}^{TS} for tree structure similarity.

3.1 Node Content Similarity Measure

Node content similarity measures the degree of approximation of two routings in terms of their operations descriptions. To cope with the textual data, the text mining technique is employed. For a set of routings, there are a number of operation types (nodes), $\{O_j\}_N$, from the instances of which each individual routing is constituted. Some routings may not assume all these operations types.

Corresponding to a P number of routings, each operation type assume a maximal number of P specific variants. Let $S_{rs}^{O_j}$ be the similarity of two operations variants, O_{jr}^* and O_{js}^*, $\forall r,s=1,\ldots,P$ and $r\neq s$, corresponding two routings, ROU_r and ROU_s, respectively. Each operation variant is described by materials, $\{F_{O_j}^M\}$, the product, $\{F_{O_j}^P\}$, and resources, $\{R_{O_j}\}$. Accordingly, the operation similarity measure, $S_{rs}^{O_j}$, consists of three elements, namely material similarity, $S_{rs}^{M_j}$, product similarity, $S_{rs}^{P_j}$, and resource similarity, $S_{rs}^{R_j}$.

3.1.1 Material similarity measure

The materials of an operation, O_j, are a set of components, be they are a type of raw materials, intermediate parts, and/or subassemblies, i.e., $F_{O_j}^M=\{Co_{jk}^M \mid \forall k=1,\ldots,K_j\}$. For a labeled binary tree, there are at most two material components, i.e., $K_j=2$. Therefore, $S_{rs}^{M_j}$ is calculated based on he similarity of all their material components. Since some components may be more important than others for an operation, $S_{rs}^{M_j}$ is computed as a weighted sum of individual material component similarity. The weight of each component, $w_{jk}^M \big| \forall k=1,\ldots K_j$, indicates the relative importance of $\{Co_{jk}^M\}_{K_j}$ and $\sum_{k=1}^{K_j} w_{jk}^M=1$. In practice, the weights are determined based on domain knowledge.

Two types of nodes are involved in an ROU tree: leaf nodes (noted as *l*-nodes) and intermediate nodes (referred to as *i*-nodes). Each *l*-node represents a machining or assembly operation that consumes at least one primitive component to produce a compound component. Each *i*-node indicates an assembly operation that produces a subassembly or end product. The materials of an *i*-node are all compound components, i.e., the subassemblies or intermediate parts. For example in Figure 2, operations M1, M2, M3 and A4 are *l*-nodes, whilst operations A1, A2 and A3 are *i*-nodes.

For an *l*-node, the corresponding operation takes at least one primitive component as input materials. If an operation is represented by an *i*-node, its material components are all compound components. Accordingly, the similarity of two component variants, Co_{jkr}^{M*} and Co_{jks}^{M*}, is computed differently under two situations: (1) Co_{jk}^M is a primitive component; and (2) Co_{jk}^M is a compound component.

Text mining is adopted to compute similarity of primitive components. The final result is a number of primitive component similarity matrices, each of which is for primitive components of same types. Based on primitive component similarity matrices, similarity of compound components is computed using bipartite matching, as shown in Figure 3.

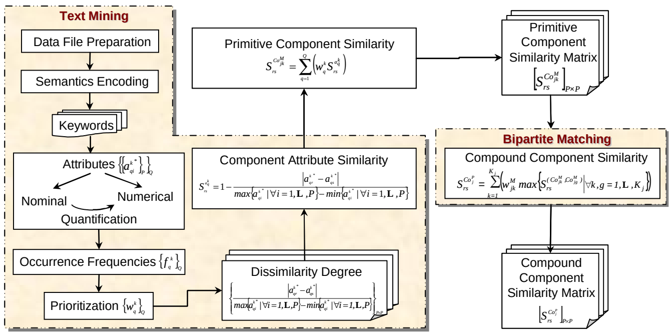

Text Mining
Data File Preparation
Semantics Encoding
Keywords
Attributes $\left\{\left\{a_{qi}^{k*}\right\}_P\right\}_Q$
Nominal
Numerical
Quantification
Occurrence Frequencies $\left\{f_q^k\right\}_Q$
Prioritization $\left\{w_q^k\right\}_Q$
Primitive Component Similarity
$S_{rs}^{Co_{jk}^M}=\sum_{q=1}^{Q}\left(w_q^k S_{rs}^{a_q^k}\right)$
Component Attribute Similarity
$S_{rs}^{a_q^k}=1-\frac{\left|a_{qr}^{k*}-a_{qs}^{k*}\right|}{max\left\{a_{qi}^{k*}\mid\forall i=1,\mathbf{L},P\right\}-min\left\{a_{qi}^{k*}\mid\forall i=1,\mathbf{L},P\right\}}$
Dissimilarity Degree
$\left\{\frac{\left|a_{qr}^{k*}-a_{qs}^{k*}\right|}{max\left\{a_{qi}^{k*}\mid\forall i=1,\mathbf{L},P\right\}-min\left\{a_{qi}^{k*}\mid\forall i=1,\mathbf{L},P\right\}}\right\}_{P\times P}$
Primitive Component Similarity Matrix
$\left[S_{rs}^{Co_{jk}^M}\right]_{P\times P}$
Bipartite Matching
Compound Component Similarity
$S_{rs}^{Co_j^P}=\sum_{k=1}^{K_j}\left(w_{jk}^M\, max\left\{S_{rs}^{(Co_{jk}^M,Co_{jg}^M)}\middle|\forall k,g=1,\mathbf{L},K_j\right\}\right)$
Compound Component Similarity Matrix
$\left[S_{rs}^{Co_j^P}\right]_{P\times P}$

Figure 3. Text mining and bipartite matching for computing material similarity.

Procedure of calculating similarity of primitive components
(1) *Prepare data files.* Operations nodes are sorted by primitive and compound components, which are saved separately in two text files. One contains the descriptions of all primitive components. The other documents the contents of all compound components.

(2) *Encode semantics.* By following practice, a component is described by a list of attribute values with respect to descriptive fields. The basic attribute field is the name or ID of the component type. Figure 4 shows an example of attribute descriptions for a specific bracket b variant, called bb. Two types of attributes can be distinguished: nominal and numerical. Different with norminal attribute values, a specific numerical value alone cannot suggest which attribute field it pertains to. For example, "10mm" can indicate an instance of "length" or "width". Therefore, rather than by listing single values, numerical attributes are described using attribute-value pairs, for example, "weight0.08g" in Figure 4.

Component Description
BRACKET A, SQUARE, BLACK, WEIGHT0.08g

Figure 4. Description of a "bb" variant.

(3) *Extract keywords.* Text mining analysis is applied to each component file. The result is a list of extracted significant words or phrases. The keywords are generated as separated records in three forms. Single words and word combinations constitute phrases, representing the values of nominal attributes. Word-number pairs are related to numerical attribute fields.

(4) *Derive occurrence frequencies.* All extracted keywords are cataloged according to their corresponding attribute fields. The occurrence of each attribute is counted by the actual values assumed. Assume a total number of Q attributes, $\left\{a_q^k \mid \forall q=1,\mathbf{L},Q\right\}$, are used to describe a component Co_{jk}^M. Dividing the number of occurrence by the total number of records in the text file, $\left\{a_{qi}^{k*} \mid \forall i=1,\mathbf{L},P\right\}$, the occurrence frequency, f_q^k, of the attribute a_q^k is determined by the count of active instances, c_q^k, and the total number of records, i.e., the number of P.

(5) *Prioritize attribute fields.* To consider an attribute's different contributions in different components, the relative importance of attributes in relation to occurrence frequencies is introduced. The relative importance of these attributes is indicated by their weights as follows:

$$w_q^k=\frac{f_q^k}{\sum_{q=1}^{Q} f_q^k}, \tag{1}$$

where w_q^k denotes the weight of the attribute, $\sum_{q=1}^{Q} w_q^k=1$.

(6) *Determine scales for nominal values.* To compare nominal values, a semantic scale is necessary in assessing the corresponding attribute type. Usually a number between 0 and 1 is assigned for a specific nominal value, whereby 0 represents no information content and 1 indicates the maximal amount of information content. For example, the semantic scale for attribute "color" may be established by assigning 0.2, 0.3, 0.4 and 0.6 for "yellow", "green" "red" and "blue", respectively. Usually such scales are determined a priori based on domain knowledge. If no domain experts are available, then simply use 1 for exactly the same nominal values and 0 for different ones, regardless of their proximity. With quantification of nominal attributes, both nominal and numerical values can be processed in the same manner, despite of their origins.

(7) *Compare attributes for their similarity.* For an attribute a_q^k, the similarity of its instances is determined by comparing their difference (i.e., dissimilarity) in attribute values, as shown in Figure 3. $S_{rs}^{a_q^k}\in[0,1]$ denotes the similarity of two attribute values, a_{qr}^{k*} and a_{qs}^{k*}.

(8) *Calculate similarity degree.* The similarity of two primitive component variants, $S_{rs}^{Co_{jk}^M}$, is calculated as a weighted sum of similarity measures of all their attributes, as shown in Figure 3. $0\le S_{rs}^{Co_{jk}^M}\le 1$ and $w_q^k\middle|\forall q=1,\mathbf{L},Q$ is computed based on Eq. (1).

(9) *Construct component similarity matrices.* Repeat steps (7)-(8) for all the instances of this component type recorded in the data file. Then a $P\times P$ matrix, $\left[S_{rs}^{Co_{jk}^M}\right]_{P\times P}$, is constructed to present pairwise similarity measures for this primitive component type. Enumerating all the primitive components, a number of such $P\times P$ matrices are constructed in accordance with a total number of K^{Pri} primitive component types contained in the data file.

Procedure of computing similarity of compound components
Each *i*-node operation, O_j, enacts a subtree for producing a compound component, Co_j^P, from primitive components, $\left\{Co_{jk}^M\right\}_{K_j}$. To find the accurate similarity between subtrees, bipartite matching [4] is applied through deriving compound component similarity based on the similarity of primitive components. Further considering that different primitive

components may contribute differently to the compound component, weighted bipartite matching is adopted by introducing different weights to the child nodes. Thus the end result is a weighted sum, as shown in Figure 3. $0 \le S_{rs}^{Co_j^P} \le 1$ suggests the similarity of two compound components of the same type, $Co_{jr}^{P^*}$ and $Co_{js}^{P^*}$, $\sum_{k=1}^{K_j} w_{jk}^{M} = 1$ indicate the relative importance of $\{Co_{jk}^{M}\}_{K_j}$ and can be determined by, e.g., domain experts, and $S_{rs}^{(Co_{jk}^M, Co_{jg}^M)}$ denotes the similarity of a paired child nodes, $Co_{jkr}^{M^*}$ and $Co_{jgs}^{M^*}$.

Since the similarity of compound components depends on the similarity of the material components, be they are primitive components or compound components. Computing similarity of compound component similarity should be carried out in a bottom-up approach along with product structures. In other words, only the similarity of components at lower levels has been obtained, can the similarity of components at higher levels be computed. Same as the result of measuring primitive component similarity, a total number of N compound component similarity matrices, $\left[S_{rs}^{Co_j^P}\right]_{P\times P}$, are constructed.

3.1.2 Product similarity measure

Each product component, $F_{O_j}^{P} = Co_j^P$, is a type of compound components. Thus, the number of N compound component similarity matrices simplifies measuring similarity of same type product components. Similarity of product components is obtained from the corresponding compound component similarity matrices. As a result, a product similarity matrix, $\left[S_{rs}^{P_j}\right]_{P\times P}$, can be constructed for each product component type. By enumerating all the product components contained in the compound data file, a total number of N product similarity matrices are constructed.

3.1.3 Resource similarity measure

The resource description, R_j, of each operation, O_j, includes three attributes: workcenter, W_j, cycle time, T_j, and setup, S_j. While W_j and S_j are nominal attributes, T_j is of the numerical type. Text mining is conducted in a similar manner as that of primitive components. Resource descriptions of all operations (both *l*-nodes and *i*-nodes) are cataloged in a separate text file. Then text mining is carried out with respect to the three attribute fields and thus similarity measures in terms of workcenter ($S_{rs}^{W_j}$), cycle time ($S_{rs}^{T_j}$) and setup ($S_{rs}^{S_j}$) are derived as follows:

$$S_{rs}^{W_j} = 1 - \frac{\left|W_{jr}^* - W_{js}^*\right|}{max\{W_{ji}^* \mid \forall i = 1,\mathbf{L},P\} - min\{W_{ji}^* \mid \forall i = 1,\mathbf{L},P\}}, \qquad (2)$$

$$S_{rs}^{T_j} = 1 - \frac{\left|T_{jr}^* - T_{js}^*\right|}{max\{T_{ji}^* \mid \forall i = 1,\mathbf{L},P\} - min\{T_{ji}^* \mid \forall i = 1,\mathbf{L},P\}}, \qquad (3)$$

$$S_{rs}^{S_j} = 1 - \frac{\left|S_{jr}^* - S_{js}^*\right|}{max\{S_{ji}^* \mid \forall i = 1,\mathbf{L},P\} - min\{S_{ji}^* \mid \forall i = 1,\mathbf{L},P\}}, \qquad (4)$$

where $S_{rs}^{W_j}, S_{rs}^{T_j}, S_{rs}^{S_j} \in [0,1]$, W_{ji}^*, T_{ji}^* and S_{ji}^* stand for the specific values of workcenter, cycle time and setup, respectively. Accordingly, the resource similarity measure of two operation variants, $S_{rs}^{R_j}$, is calculated as a weighted sum of similarity measures regarding all their attributes as follows:

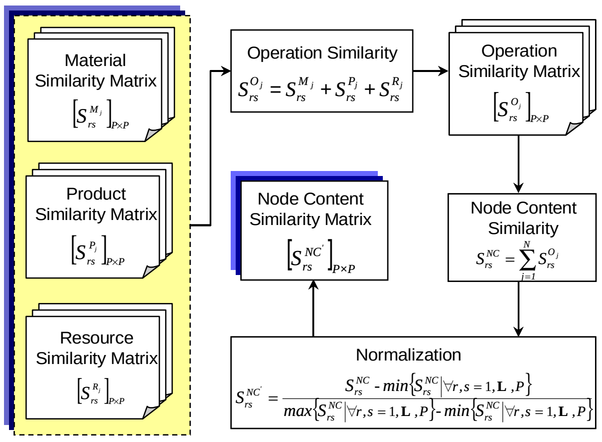
$$S_{rs}^{R_j} = w^{W_j} S_{rs}^{W_j} + w^{T_j} S_{rs}^{T_j} + w^{S_j} S_{rs}^{S_j}, \qquad (5)$$

where $0 \le S_{rs}^{R_j} \le 1$, $w^{W_j} + w^{T_j} + w^{S_j} = 1$, and w^{W_j}, w^{T_j} and w^{S_j} denote the relative importance of workcenter, cycle time and setup attributes, respectively.

By enumerating all instances of resource description, R_j, a resource similarity matrix, $\left[S_{rs}^{R_j}\right]_{P\times P}$, is constructed to present pairwise resource comparisons of all variants of operation O_j. Similarly, a total number of N resource similarity matrices are constructed for all the operations, $\{O_{jk}^*\}_{N\times P}$.

3.1.4 Operation & node content similarity measures

With material similarity, $S_{rs}^{M_j}$, product similarity, $S_{rs}^{P_j}$, and resource similarity, $S_{rs}^{R_j}$, the operation similarity is computed, as shown in Figure 5. Enumerate the above operation similarity calculation for all operations variants of all operations types. Then present pairwise similarity of same types of operations variants in same matrices. A total number of N operations similarity matrices are constructed.



Material Similarity Matrix
$\left[S_{rs}^{M_j}\right]_{P\times P}$
Product Similarity Matrix
$\left[S_{rs}^{P_j}\right]_{P\times P}$
Resource Similarity Matrix
$\left[S_{rs}^{R_j}\right]_{P\times P}$
Operation Similarity
$S_{rs}^{O_j} = S_{rs}^{M_j} + S_{rs}^{P_j} + S_{rs}^{R_j}$
Operation Similarity Matrix
$\left[S_{rs}^{O_j}\right]_{P\times P}$
Node Content Similarity Matrix
$\left[S_{rs}^{NC'}\right]_{P\times P}$
Node Content Similarity
$S_{rs}^{NC} = \sum_{j=1}^{N} S_{rs}^{O_j}$
Normalization
$S_{rs}^{NC'} = \frac{S_{rs}^{NC} - min\{S_{rs}^{NC} \mid \forall r,s = 1,\mathbf{L},P\}}{max\{S_{rs}^{NC} \mid \forall r,s = 1,\mathbf{L},P\} - min\{S_{rs}^{NC} \mid \forall r,s = 1,\mathbf{L},P\}}$

Figure 5. Measuring operation similarity and node content similarity.

The node content similarity between two routings is computed as the sum of their operations similarity, as shown in Figure 5.

Enumerate the above node content similarity calculation for all the *ROUs* in the routing set and obtain the pairwise similarity of node content of all routings.

3.1.5 Normalized node content similarity matrix

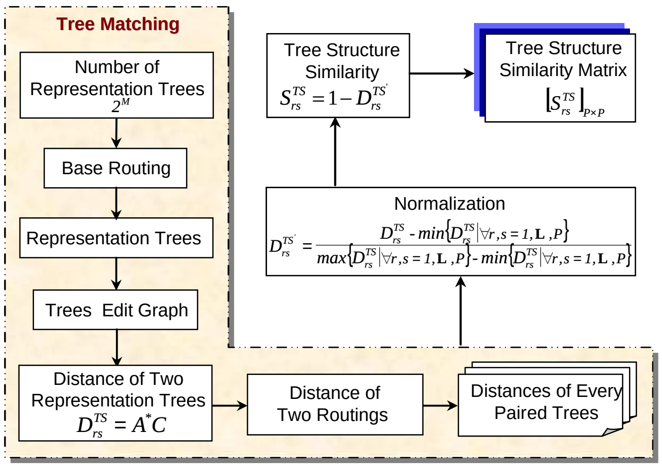
Since $0 \le S_{rs}^{M_j}, S_{rs}^{P_j}, S_{rs}^{R_j} \le 1$, $S_{rs}^{O_j}$ and S_{rs}^{NC} may not suggest a relative measure ranging from 0 to 1. They need to be normalized to achieve a consistent comparison. This research adopts the most common approach: max-min method [5] to convert the node content similarity to a relative magnitude between 0 and 1.

S_{rs}^{NC} and $S_{rs}^{NC'}$ denotes the original and normalized node content similarity between ROU_r and ROU_s, respectively.

Enumerate above normalization for all the $ROUs$ and then present the result in the form of an ROU node content similarity matrix, $\left[S_{rs}^{NC'}\right]_{P\times P}$. Each matrix element indicates the node content similarity of two routing variants corresponding to row and column, respectively.

3.2 TREE STRUCTURE SIMILARITY MEASURE

Tree matching technique is applied to measure tree structure similarity, which measures the degree of commonality of two routings in terms of their operations precedence. Figure 6 shows the procedure of measuring tree structure similarity using tree matching.



Tree Matching
Number of Representation Trees 2^M
Base Routing
Representation Trees
Trees Edit Graph
Distance of Two Representation Trees $D_{rs}^{TS}=A^*C$
Distance of Two Routings
Distances of Every Paired Trees
Normalization
$D_{rs}^{TS'}=\frac{D_{rs}^{TS}-min\{D_{rs}^{TS}\mid\forall r,s=1,\ldots,P\}}{max\{D_{rs}^{TS}\mid\forall r,s=1,\ldots,P\}-min\{D_{rs}^{TS}\mid\forall r,s=1,\ldots,P\}}$
Tree Structure Similarity $S_{rs}^{TS}=1-D_{rs}^{TS'}$
Tree Structure Similarity Matrix $\left[S_{rs}^{TS}\right]_{P\times P}$

Figure 6. Measuring tree structure similarity.

(1) *Determine a base ROU.* Each ROU is a partial order, and shus may possess a number of alternative representation trees. The similarity of two $ROUs$ may vary if different representation trees of them are used for the comparison. It is thus necessary to make decision based on pairwise comparisons of all possible representation trees of two $ROUs$. Owing to the symmetric property of distance measure and cyclic representation of a partial order [6], the pairwise comparisons can be simplified to merely compare an arbitrary tree of one ROU (referred to as a base ROU) with all representation trees of the other $ROUs$. To reduce the total number of pairwise comparisons among $ROUs$, the ROU with the most representation trees should be selected as the base ROU. The number of representation trees of an ROU is given as 2^N, where N is the number of nodes with two child nodes.

(2) *Generate representation trees.* For a number of routings, $\{ROU_r \mid \forall r=1,\ldots,P\}$, each of the first $(P-1)$ routings serves as a base ROU for comparison of tree structure similarity with its immediate next ROU. Thus a total number of $P\times(P-1)/2$ pairwise comparisons are needed. Except for the ROU selected to be the first base ROU, all the rest $ROUs$ are compared with their corresponding base $ROUs$. Thus, all corresponding representation trees need to be generated for each of these $(P-1)$ $ROUs$.

(3) *Establish a tree edit graph.* To overcome the disadvantage of traditional tree transformation using tree editing operations, the tree edit graph [7] is adopted in this research to obtain dissimilarity, and thus similarity of two trees.

(4) *Find the shortest path for distance measure.* In a tree edit graph, there are many paths from the top-left corner to the bottom-right corner. Each such a path suggests a possible way of transforming one tree to another, which carries different costs as well. The distance between two trees should be measured according to the shortest path that requires minimum number of arcs and thus fewest editing operations. The distance measure between every two trees is determined, as shown in Figure 6. A^* is the total number of valid arcs in the shortest path and C is a constant indicating unit cost value associated with each operation, regardless of its type.

Repeat the above procedures for comparing all the representation trees for one ROU with the base ROU. The distance measure between this ROU and the base ROU is determined by the minimum distance among all distance measures between its representation trees and the base ROU. By enumerating all the $(P-1)$ $ROUs$ in the given set, their tree structure distances from the base ROU are reckoned in the same manner.

(5) *Normalize distance data.* The above distance measures are all absolute values instead of relative magnitude. For consistent comparison, they need to be normalized. The max-min method is adopted to convert the absolute distance measure of each ROU pair to a dimensionless value ranging between 0 and 1, as shown in Figure 6. D_{rs}^{TS} and $D_{rs}^{TS'}$ denotes the absolute and normalized distance measures between ROU_r and ROU_s, respectively.

(6) *Calculate tree structure similarity.* According to the normalized distance measure, the similarity can be calculated.

(7) *Construct an ROU structure similarity matrix.* Calculate similarity values for all the $ROUs$ in the routing data set. Then present all pairwise similarity measures in a $P\times P$ matrix, $\left[S_{rs}^{TS}\right]_{P\times P}$. Each matrix element indicates the structure similarity of two $ROUs$ corresponding to row and column, respectively.

3.3 ROU SIMILARITY MEASURE

As node content similarity and tree structure similarity are two independent measures, the overall ROU similarity is obtained by an Euclidian distance, as shown in Figure 1.

Repeat the above routing similarity calculation for all $ROUs$ in the routing set and obtain the pairwise similarity of all $ROUs$. The normalization is applied to obtain the normalized routing similarity, S_{rs} such that $0\le S_{rs}\le 1$, as follows:

$$S_{rs}=\frac{S_{rs}'-min\{S_{rs}'\mid\forall r,s=1,\ldots,P\}}{max\{S_{rs}'\mid\forall r,s=1,\ldots,P\}-min\{S_{rs}'\mid\forall r,s=1,\ldots,P\}}, \qquad (6)$$

Repeat the above normalization for all $ROUs$. Then present pairwise routing similarity in a $P\times P$ matrix, $\left[S_{rs}\right]_{P\times P}$. Each matrix element indicates the normalized similarity of two $ROUs$ corresponding to row and column, respectively.

4 GROU IDNETIFICATION

4.1 ROU CLUSTERING

ROU clustering aims to group similar routings into clusters. Hence, an ROU cluster is a collection of routings that are similar to one another within the same cluster yet dissimilar to the routings in other clusters. Considering the complex data types, e.g., textual data, involved in routings, this research adopts a fuzzy clustering approach by taking its advantage of handling subjectiveness and impression [8].

Procedure of ROU clustering

(1) *Define a fuzzy compatible matrix.* A fuzzy compatible matrix, R, is defined as similarity measures for a given set W of $ROUs$. The R is constructed in a matrix form, that is, $R = [S_{rs}]_{P\times P}$, where $0 \le S_{rs} \le 1$ suggests a pair-wise relationship (similarity grade) between any two ROU instances. In R, it holds true that $S_{rr} = 1$ suggesting that R is reflexive. Also true is $S_{rs} = S_{sr}$ suggesting that R is symmetrical. Therefore, the fuzzy compatible matrix R is identical to routing similarity matrix obtained previously.

(2) *Construct a fuzzy equivalence matrix.* A fuzzy equivalence matrix is defined for W with transitive closure of a fuzzy compatible matrix [8]. To convert a compatible matrix to an equivalence matrix, the max-min composition [9] is adopted.

(3) *Determine a λ-cut of the equivalence matrix.* The λ-cut is a crisp set, R_l, that contains all the elements of W, such that the similarity grade of R is no less than l, that is,

$$R_l = [t_{rs}]_{P\times P}, \tag{7}$$

where $$t_{rs} = \begin{cases} 1 & if\ S_{rs} \ge l \\ 0 & if\ S_{rs} < l \end{cases},\quad S_{rs} \in [0,1], \tag{8}$$

Then each λ-cut is an equivalence matrix representing the presence of similarity among routings to the degree l.

(4) *Identify ROU clusters.* A netting graph method [3] is applied to identify partitions of routing instances with respect to a given equivalence matrix.

4.2 ROU UNIFICATION

ROU unification is to unify all members of an ROU cluster into a generic routing, $GROU$. The major elements of a $GROU$ include a set of master routing elements, such as operations and precedence, and a set of selective routing elements. The structure of a $GROU$, G, is referred to as a generic tree and is developed through a tree growing process, from the sub-general tree structures, referred to as basic trees, $\{T_z\}_Z$, within the ROU cluster. The formation of a $GROU$ involves four major steps, including assorting basic arouting elements, identifying master and selective routing elements, forming basic trees, and tree growing, as discussed below.

4.2.1 Basic routing elements

The first step of routing unification is to breakdown individual routings into operations and precedence elements. For each member of an ROU cluster, $ROU_r \in \{ROU_r \mid \forall r = 1, \ldots, M \le P\}$, the nodes (operations) and arcs (precedence) of the corresponding ROU tree are assorted and categorized by ***l***-nodes or ***i***-nodes. This results in a ***l***-node set, an ***i***-node set, a ***l***-node arc set, and an ***i***-node arc set, corresponding to ***l***-node type $\{LN_j\}_{N^{LN}}$, ***i***-node type $\{IN_j\}_{N^{IN}}$, ***l***-node arc type $\{LA_j\}_{N^{LN}}$, and ***i***-node arc type $\{IA_j\}_{N^{IN}-1}$, respectively, where $N^{LN} + N^{IN} = N$, $\forall LN_j, IN_j \in L$, $\{LN_j\} \cap \{IN_j\} = \varnothing$, $\forall LA_j, IA_j \in \mathbf{f}$, and $\{LA_j\} \cap \{IA_j\} = \varnothing$.

4.2.2 Master and selective routing elements

The second step is to generalize each individual routing element with regard to its original type. This is achieved by replacing the specific name or ID of each specific node or arc with the general name or ID of the operation or precedence class that it belongs to. As a result, each particular routing element is labeled with its class identification. And in turn each operation or precedence class assumes a certain number of occurrences in terms of the number of times individual routing elements are generalized into this class. Such an occurrence count performs as a commonality index revealing to what extent each routing element is reused among individual members of an ROU family.

Given an ROU cluster, if the occurrence count of a precedence class reaches the maximal number of instances of this class contained in the cluster, i.e., it means that all individual routings in the cluster employ this precedence class. Therefore, this precedence class along with the related operation classes suggest themselves to be the master routing elements, i.e., the master precedence and operation classes, respectively. Or else, the related operation and precedence classes are defined as selective operation and precedence classes, respectively, as not all individual variants assume them. In this way, all basic routing elements are grouped into either master or selective routing elements.

4.2.3 Basic tree structures

The third step deals with the generalization of basic trees, each of which is common to several members in an ROU family. Therefore, a basic tree refers to the common tree structure assumed by certain routing variants. A number of $Z \le M$ basic tree structures, $\{T_z\}_Z$, are identified from M member trees of an ROU family.

To track commonality of a basic tree with respect to its represented routing variants, each arc of the basic tree is assigned a weight indicating the degree of repetition of this arc among $M_z \le Z$ routings. Initially, the value of such a weight is set to be the same as the occurrence count of each arc, regardless it is a master or selective precedence. In accordance with the assortment of basic routing elements, a basic tree is specified by a 4-tuple, denoted as:

$$T_z = \left(L^N, I^N, L^A, I^A\right), \tag{9}$$

where $L^N(T_z)$, $I^N(T_z)$, $L^A(T_z)$, and $I^A(T_z)$ are sets of basic routing elements, encompassing all ***l***-node classes, ***i***-node classes, ***l***-node arc classes and ***i***-node arc classes contained in T_z, respectively.

4.2.4 Tree growing

The fourth step aims to form the generic tree by pasting all basic trees one by one. Tree growing starts with the selection of a seed, i.e., an initial generic tree, G_1. Among basic trees,

$\{T_z\}_Z$, the one holding a longest path and possesses the maximal number of ***i***-nodes is recognized as the seed. Such a comprehensive tree encompasses most production conditions occurring among the process family members. The initial generic tree starts to grow by unifying with the other $Z-1$ basic trees one by one, that is,

$$G_i = G_{i-1} \mathbf{U} T_i, \tag{10}$$

where G_i is a growing tree. After all basic trees are unified, the growing tree reaches its final form, G_Z, namely, the tree structure of the $GROU$.

Since the structure of a $GROU$ includes all operations occurred in the ROU cluster, both $L^N(G)$ and $I^N(G)$ are simply union of all node sets contained in basic trees. However, $L^A(G)$ and $I^A(G)$ do not work with simple union operations, because a tree structure has to be maintained throughout the tree growing process.

If an ***l***-node arc exists in T_i but not in G_{i-1}, this arc is of selective type, i.e., La_{ij}^S. Such selective arcs, $\{La_{ij}^S\}_{N_i^{LNS}}$, are pasted to G_{i-1} only when their associated operations, i.e., selective ***l***-nodes, $\{Ln_{ij}^S\}_{N_i^{LNS}}$, do not exist in G_{i-1} at the same time. Except this situation, to include a selective ***l***-node arc of T_i, $La_{ij}^S \in L^A(T_i)$, into G_{i-1} or to put it in A_{AS} depends on the result of comparing its weight, $W_j^{La_{ij}^S}$, with that, $W_j^{La_{(i-1)j}^S}$, of the corresponding arc in G_{i-1}. Whichever assuming higher weight should be included, as a higher weight means more common a selective arc is used. Such a weight results from the sum of the occurrence count of this arc in all member trees and the recorded weight of the same arc in A_{AS}, if it is not empty. Likewise a selective ***i***-node arc, $Ia_{ij}^S \in L^A(T_i)$, does not exist in G_{i-1}. Only when the associated parent ***i***-node, $PIn_{ij}^S \in I^N(T_i)$, and child ***i***-nodes, $CIn_{ij}^S \in I^N(T_i)$., do not exist in G_{i-1} at the same time can this ***i***-node arc be added to G_{i-1}. Otherwise, evaluation of its weight is needed. In essence, arc unification aims to combine the arc sets of T_i and G_{i-1} while removing those less common arcs.

Each arc conveys information regarding two operations and the order of their execution. Tree growing is thus performed based on the search and evaluation of arcs. Any change to an ***l***-node will propagate upwards in the routing tree till to the root node, thus causing changes to all relevant ***i***-nodes and affecting the tree structure as well. Therefore in tree growing, ***l***-node arcs are treated first and then ***i***-node arcs. Moreover, tree growing operates on master ***l***-node arcs and master ***i***-node arcs first, and then selective ***l***-node arcs and selective ***i***-node arcs.

For the master ***l***-node and ***i***-node arc sets of T_i, La_{ij}^M and Ia_{ij}^M, add their weights, $W_j^{La_{ij}^M}$ and $W_j^{Ia_{ij}^M}$, in G_{i-1} by the corresponding weight values in T_i. For the selective ***l***-node arc set, La_{ij}^S of T_i, if they can be found in G_i, then increase their weights $W_j^{Ia_{ij}^S}$ in G_i by the corresponding weight values in T_i.

For the selective ***i***-node arcs of T_i, increase their weights in G_{i-1} by the weight values in T_i if they can be found in G_{i-1},.

Upon completion of the tree growing process, the formed $GROU$ consists of a generic tree structure and an additional arc set. Due to the presence of selective arcs in the generic tree, the $GROU$ is by no means the union of all member trees. Addition and removal of certain arcs according to their weights guarantee that the resulted generic structure is the most common to individual routings in an ROU family.

5 CASE STUDY

The proposed data mining methodology has been applied to high variety production of vibration motors for mobile phones in an electronics company. Due to the many design changes in mobile phones, vibration motors in the company are typically customized. The company has been struggling to produce quickly the diverse motors at low costs. However handling the frequent production changeovers, most of which are caused by improper routings planned subjectively by production engineers, becomes the company's major headache.

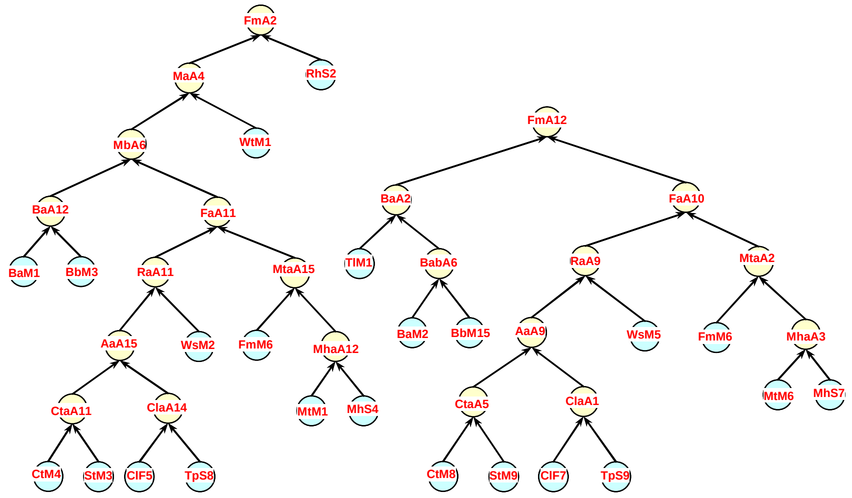


Figure 7. Two routing representation trees.

$$\begin{bmatrix}
1 & .65 & .95 & .67 & .64 & .69 & .65 & .64 & .67 & .88 & .69 & .24 & .85 & .90 & .21 & .74 & .88 & .76 & .32 & .82 & .32 & .78 & .51 & .32 & .74 & .42 & .41 & .43 & .40 & .41 \\
.65 & 1 & .66 & .79 & .81 & .78 & .72 & .74 & .83 & .64 & .84 & .43 & .54 & .54 & .42 & .82 & .64 & .86 & .53 & .63 & .50 & .57 & .13 & .51 & .56 & .11 & .03 & .03 & .10 & .03 \\
.95 & .66 & 1 & .63 & .66 & .71 & .64 & .65 & .68 & .89 & .69 & .23 & .84 & .84 & .21 & .73 & .89 & .81 & .33 & .81 & .31 & .79 & .51 & .33 & .74 & .41 & .41 & .41 & .40 & .41 \\
.67 & .79 & .63 & 1 & .94 & .93 & .86 & .92 & .89 & .62 & .84 & .62 & .53 & .59 & .63 & .83 & .62 & .84 & .52 & .59 & .50 & .61 & .20 & .51 & .56 & .07 & .06 & .05 & .14 & .05 \\
.64 & .81 & .66 & .94 & 1 & .92 & .90 & .90 & .83 & .65 & .87 & .62 & .56 & .56 & .62 & .82 & .70 & .86 & .53 & .62 & .51 & .58 & .13 & .51 & .61 & .10 & .03 & .03 & .11 & .03 \\
.69 & .78 & .71 & .93 & .92 & 1 & .91 & .95 & .86 & .68 & .83 & .63 & .62 & .62 & .61 & .90 & .68 & .90 & .52 & .63 & .51 & .67 & .16 & .52 & .55 & .15 & .08 & .08 & .05 & .13 \\
.65 & .72 & .64 & .86 & .90 & .91 & 1 & .89 & .80 & .60 & .83 & .61 & .67 & .67 & .61 & .88 & .65 & .84 & .51 & .62 & .50 & .61 & .13 & .53 & .54 & .09 & .10 & .11 & .08 & .06 \\
.64 & .74 & .65 & .92 & .90 & .95 & .89 & 1 & .87 & .68 & .81 & .63 & .57 & .62 & .62 & .81 & .64 & .86 & .52 & .59 & .51 & .61 & .10 & .50 & .61 & .02 & .08 & .08 & .11 & .05 \\
.67 & .83 & .68 & .89 & .83 & .86 & .80 & .87 & 1 & .65 & .91 & .52 & .60 & .60 & .53 & .91 & .66 & .95 & .62 & .67 & .61 & .62 & .21 & .61 & .59 & .12 & .15 & .20 & .14 & .11 \\
.88 & .64 & .89 & .62 & .65 & .68 & .60 & .68 & .65 & 1 & .62 & .34 & .76 & .80 & .31 & .61 & .93 & .67 & .43 & .89 & .40 & .84 & .61 & .40 & .88 & .51 & .51 & .51 & .51 & .50 \\
.69 & .84 & .69 & .84 & .87 & .83 & .83 & .81 & .91 & .62 & 1 & .52 & .60 & .60 & .52 & .86 & .68 & .87 & .63 & .62 & .61 & .63 & .20 & .61 & .64 & .09 & .17 & .13 & .15 & .10 \\
.24 & .43 & .23 & .62 & .62 & .63 & .61 & .63 & .52 & .34 & .52 & 1 & .13 & .13 & .96 & .51 & .32 & .52 & .87 & .22 & .88 & .22 & .65 & .86 & .22 & .65 & .57 & .62 & .59 & .52 \\
.85 & .54 & .84 & .53 & .56 & .62 & .67 & .57 & .60 & .76 & .60 & .13 & 1 & .96 & .10 & .71 & .76 & .65 & .21 & .89 & .20 & .87 & .40 & .20 & .84 & .50 & .50 & .50 & .51 & .52 \\
.90 & .54 & .84 & .59 & .56 & .62 & .67 & .62 & .60 & .80 & .60 & .13 & .96 & 1 & .10 & .71 & .76 & .65 & .21 & .89 & .23 & .83 & .40 & .20 & .78 & .50 & .51 & .50 & .50 & .50 \\
.21 & .42 & .21 & .63 & .62 & .61 & .61 & .62 & .53 & .31 & .52 & .96 & .10 & .10 & 1 & .50 & .33 & .51 & .86 & .20 & .81 & .22 & .58 & .88 & .20 & .59 & .53 & .63 & .66 & .60 \\
.74 & .82 & .73 & .83 & .82 & .90 & .88 & .81 & .91 & .63 & .86 & .51 & .71 & .71 & .50 & 1 & .64 & .94 & .61 & .65 & .61 & .63 & .24 & .62 & .57 & .17 & .15 & .12 & .10 & .10 \\
.88 & .64 & .89 & .62 & .70 & .68 & .65 & .64 & .66 & .93 & .68 & .32 & .76 & .76 & .33 & .64 & 1 & .73 & .45 & .87 & .40 & .90 & .61 & .41 & .88 & .50 & .52 & .51 & .51 & .50 \\
.76 & .86 & .81 & .84 & .86 & .90 & .84 & .86 & .95 & .67 & .87 & .52 & .65 & .65 & .51 & .94 & .73 & 1 & .61 & .64 & .61 & .65 & .19 & .61 & .69 & .11 & .12 & .16 & .14 & .12 \\
.82 & .53 & .33 & .52 & .53 & .52 & .51 & .52 & .62 & .43 & .63 & .87 & .21 & .21 & .86 & .61 & .45 & .61 & 1 & .30 & .94 & .31 & .77 & .92 & .34 & .66 & .61 & .64 & .67 & .67 \\
.32 & .63 & .81 & .59 & .62 & .63 & .62 & .59 & .67 & .89 & .62 & .22 & .89 & .89 & .20 & .65 & .87 & .64 & .30 & 1 & .30 & .97 & .51 & .30 & .91 & .61 & .61 & .61 & .61 & .61 \\
.32 & .50 & .31 & .50 & .51 & .51 & .50 & .51 & .61 & .40 & .61 & .88 & .20 & .23 & .81 & .61 & .40 & .61 & .94 & .30 & 1 & .32 & .69 & .95 & .34 & .71 & .65 & .67 & .65 & .68 \\
.78 & .57 & .79 & .61 & .58 & .67 & .61 & .61 & .62 & .84 & .63 & .22 & .87 & .83 & .22 & .63 & .90 & .65 & .31 & .97 & .32 & 1 & .52 & .30 & .96 & .61 & .61 & .61 & .61 & .61 \\
.51 & .13 & .51 & .20 & .13 & .16 & .13 & .10 & .21 & .61 & .20 & .65 & .40 & .40 & .58 & .24 & .61 & .19 & .77 & .51 & .69 & .52 & 1 & .31 & .51 & .81 & .85 & .89 & .82 & .80 \\
.32 & .51 & .33 & .51 & .51 & .52 & .53 & .50 & .61 & .40 & .61 & .86 & .20 & .20 & .88 & .62 & .41 & .61 & .92 & .30 & .95 & .30 & .31 & 1 & .30 & .57 & .62 & .68 & .63 & .56 \\
.74 & .56 & .74 & .56 & .61 & .55 & .54 & .61 & .59 & .88 & .64 & .22 & .84 & .78 & .20 & .57 & .88 & .69 & .34 & .91 & .34 & .96 & .51 & .30 & 1 & .61 & .61 & .61 & .61 & .61 \\
.42 & .11 & .41 & .07 & .10 & .15 & .09 & .02 & .12 & .51 & .09 & .65 & .50 & .50 & .59 & .17 & .50 & .11 & .66 & .61 & .71 & .61 & .81 & .57 & .61 & 1 & .96 & .96 & .90 & .91 \\
.41 & .03 & .41 & .06 & .03 & .08 & .10 & .08 & .15 & .51 & .17 & .57 & .50 & .51 & .53 & .15 & .52 & .12 & .61 & .61 & .65 & .61 & .85 & .62 & .61 & .96 & 1 & .95 & .94 & .93 \\
.43 & .03 & .41 & .05 & .03 & .08 & .11 & .08 & .20 & .51 & .13 & .62 & .50 & .50 & .63 & .12 & .51 & .16 & .64 & .61 & .67 & .61 & .89 & .68 & .61 & .96 & .95 & 1 & .94 & .96 \\
.40 & .10 & .40 & .14 & .11 & .05 & .08 & .11 & .14 & .51 & .15 & .59 & .51 & .50 & .66 & .10 & .51 & .14 & .67 & .61 & .65 & .61 & .82 & .63 & .61 & .90 & .94 & .94 & 1 & .94 \\
.41 & .03 & .41 & .05 & .03 & .13 & .06 & .05 & .11 & .50 & .10 & .52 & .52 & .50 & .60 & .10 & .50 & .12 & .67 & .61 & .68 & .61 & .80 & .56 & .61 & .91 & .93 & .96 & .94 & 1
\end{bmatrix}_{30 \times 30}$$

Figure 9. Routing similarity matrix of 30 routings.

5.1 ROUTING SIMILARITY MEASURE

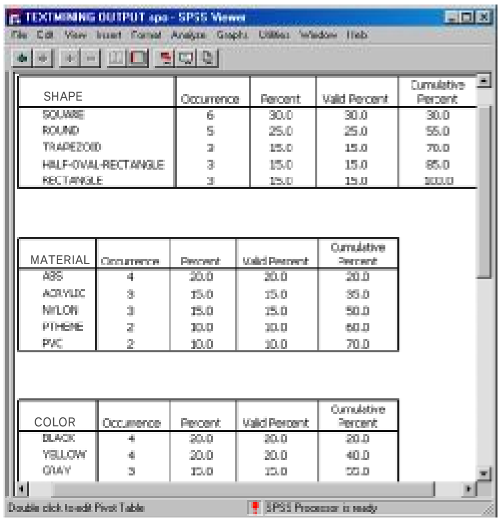
The production data of 30 routing variants for producing 30 motor variants has been used to test the methodology. Figure 7 shows two routing binary representation trees. In each routing tree, the nodes represent specific operations. The label of each node indicates the ID of the operation concerned. For example, "F*m*A2" represents a particular assembly operation for producing the final motor product, and "StM3" denotes a specific variant of shaft machining operation.

The SPSS software package (www.spss.com) has been adopted for text analysis. Three attributes are used to describe the characteristics of each operation: the material, product, and resource types. In preparing data files for text mining, raw materials are described as material components of machining operations. An operation description data file is obtained by enumerating all the operations contained in the 30 routings. Assorting all primitive and compound components for each operation, this data file is separated into two text files, one containing all primitive components and the other containing compound components. Then these two files are input into SPSS for text analysis.

Figure 8 shows the results of text analysis. For illustrative simplicity, a type of primitive component: bracket b (referred to as "bb") is used as an example. The result includes the extracted keywords, i.e., attribute values describing "bb" variants, and their respective occurrence counts.
Based on extracted information, the relevant attributes are identified and their weights are calculated. For the set of attributes identified, shape, color and material are of nominal type whilst weight and thickness are numerical ones. To quantify each nominal attribute, semantic scales are assigned for its specific instances based on domain knowledge. Based on established semantic scales, attribute similarity measures are calculated. Based on the results of attribute similarity, similarity measure of component "bb" among 30 routing variants is derived. The result is presented in a matrix form.

In the same way, the similarity matrices of other primitive components are constructed. Based on the primitive component similarity matrices, the similarity of compound component, and further the similarity matrices of compound components of same types, is obtained. The similarity matrices of compound components provide the similarity of product components. Resource similarity measure proceeds with text analysis in a similar way. At last, the resource similarity matrix for "Bracket Assembly" operation is obtained.

Compiling results of component and resource similarity measures, operation similarity is derived. Similarly, similarity matrices of all operations types involved in the 30 routings are obtained. Finally, the normalized node content similarity measures are calculated. Tree structure similarity of 30 routings is measure by following the developed procedure and the result of pairwise comparison is obtained. Finally, compiling node content similarity and tree structure similarity, the normalized pairwise similarity measures of 30 routings are obtained, as given in Figure 9.


TEXTMINING OUTPUT.spo - SPSS Viewer
File Edit View Insert Format Analyze Graphs Utilities Window Help
Double click to edit Pivot Table
SPSS Processor is ready

SHAPE	Occurrence	Percent	Valid Percent	Cumulative Percent
SQUARE	6	30.0	30.0	30.0
ROUND	5	25.0	25.0	55.0
TRAPEZOID	3	15.0	15.0	70.0
HALF-OVAL-RECTANGLE	3	15.0	15.0	85.0
RECTANGLE	3	15.0	15.0	100.0

MATERIAL	Occurrence	Percent	Valid Percent	Cumulative Percent
ABS	4	20.0	20.0	20.0
ACRYLIC	3	15.0	15.0	35.0
NYLON	3	15.0	15.0	50.0
PTHENE	2	10.0	10.0	60.0
PVC	2	10.0	10.0	70.0

COLOR	Occurrence	Percent	Valid Percent	Cumulative Percent
BLACK	4	20.0	20.0	20.0
YELLOW	4	20.0	20.0	40.0
GRAY	3	15.0	15.0	55.0

Figure 8. Text mining result: extracted keywords and occurrence count.

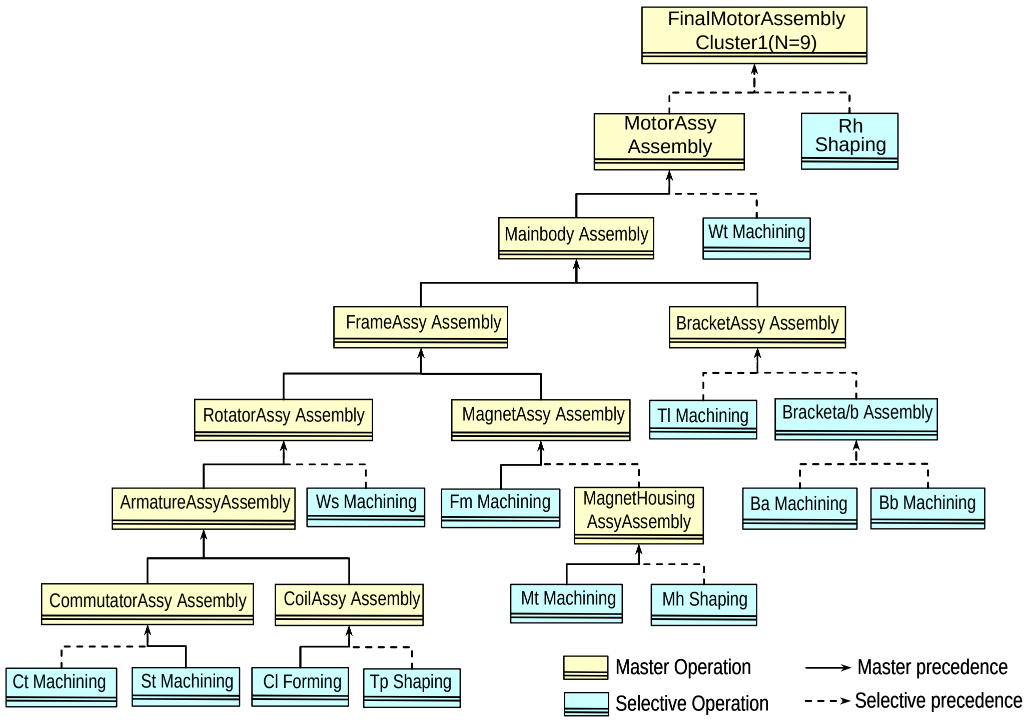


Figure 10. Formed GROU for routing cluster "RC1".

5.2 ROU CLUSTERING AND UNIFICATION

By its property, the routing similarity matrix itself is a fuzzy compatible matrix. Applying the max-min composition, a fuzzy equivalence matrix is obtained. Based on domain knowledge on clustering, a threshold level of 0.85 is decided. Accordingly the λ-cut matrix is obtained. Subsequently, the netted graph is developed, based on which the ROU clusters are derived. Table 1 gives the result of ROU clustering with four ROU clusters identified. For each ROU cluster, one GROU is formed by tree growing. For example, routing cluster "RC1" contains 9 member trees (R1, R3, R10, R13, R14, R17, R20, R22 and R25). The tree structures of these 9 routings are unified as a generic tree. Figure 10 presents the identified GROU for "RC1", which is represented using the unified modeling language.

Table 1. Result of routing fuzzy clustering.

ROU Cluster	ROU Variants
RC1	R1, R3, R10, R13, R14, R17, R20, R22, R25
RC2	R2, R4, R5, R6, R7, R8, R9, R11, R16, R18
RC3	R23, R26, R27, R28, R29, R30
RC4	R12, R15, R19, R21, R24

6 CONCLUSIONS

A generic routing essentially performs as a process platform to support the fulfillment of product families. It contributes to the utilization of commonality underlying process variations. The formation of generic routings coincides with the wisdom of knowledge reuse and economy of repetition. Generating generic routings based on knowledge discovery from past data avails to maintain the integrity of existing product and process platforms, as well as the continuity of the infrastructure and core competencies, hence leveraging existing design and manufacturing investments. The application of data mining, more specifically text mining and tree matching, opens opportunities for incorporating experts' experiences into the projection of production planning patterns from historical data, thereby enhancing the ability to explore and utilize domain knowledge more effectively.

7 REFERENCES

[1] Jiao, J., Zhang, L. and Pokharel, S., 2007, "Process Platform Planning for Variety Coordination from Design to Production in Mass Customization Manufacturing", *IEEE Transactions on Engineering Management*, Vol. **54** (1), pp. 112-129.
[2] Chen, M., Han, J. and Yu, P., 1996, "Data Mining: An Overview from Database Perspective", *IEEE Transactions on Knowledge and Data Engineering*, Vol. **8** (6), pp. 866-883.
[3] Yang, L. and Gao, Y., 1996, *Fuzzy Mathematics: Theory and Applications*, ISBN. 7-5623-0440-8.
[4] Romanowski, C.J. and Nagi, R., 2005, "On Comparing Bills of Materials: A Similarity/Distance Measure for unordered Trees", *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, Vol. **35** (2), pp. 249-260.
[5] Han, J. and Kamber, M., 2001, *Data Mining: Concepts and Techniques*, Morgan-Kanfmann, San Mateo, CA.
[6] Martinez, M.T., Favrel, J. and Ghodous, P., 2000, "Product Family Manufacturing Plan Generation and Classification", *Concurrent Engineering: Research and Applications*, Vol. **8** (1), pp. 12-22.
[7] Valiente, G., 2002, *Algorithms on Trees and Graphs*, Springer-Verlag, Berlin.
[8] Zimmermann, H.J., 2001, *Fuzzy Set Theory and Its Applications*, Kluwer Academic Publishers, USA.
[9] Lin, C.T. and Lee, C.S.G., 1996, *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*, Prentice Hall, NJ, USA.