한국경영정보학회
96년 춘계학술대회

Development of an Idea Categorizer for Group Decision Support System.

Soung Hie Kim, Jae Kwang Lee
Dept. of Management Engineering, Graduate School of Management, KAIST, Seoul, Korea
Sun Uk Kim
Dept. of Industrial Engineering, DanKuk University, CheonAn, Korea
Hung Kook Park
Dept. of Telecommunication Systems Management, SangMyung University, Seoul, Korea

ABSTRACT

Important matters in an organizations are decided by a group which consists of numerous specialists rather than an individual. Researches on group decisions and group decision support systems have been increased rapidly over the past few years. One of the important research field in GDSS, is on effectiveness of GDSSs. Research on effectiveness according to the various cultural diversity is considered necessary to develop an appropriate GDSS for its own cultural context.

Although various issues were raised in empirical research using an existing GDSS, we try to solve an issue on idea categorizing in this study. Idea categorizing in an existing GDSS was performed in a top-down procedure and mostly by participants' manual works. This resulted in long idea categorizing time as much as idea generating time, clustering an idea in multiple categories, and identifying almost similar redundant categories. To solve these problem, we suggest an interactive idea categorizing method which is a bottom-up approach. This method consists of steps of presenting idea using keywords, identifying keyword's affinity, computing similarity among ideas, and clustering ideas. This method allows iterative interaction to participants for clear manifestation of ambiguous ideas. We also develop an prototype interactive idea categorizer(IIC) to test this bottom-up approach.

I. INTRODUCTION

1.1 Group Decision Support Systems and Idea Categorizing

Recently, in organizations many major decisions are being made by groups. Group or work group refer to two or one more individuals whose mission is to perform some task and who act as one unit. The group can be permanent or temporary. The group can be in one location or in several locations, and it can meet concurrently or at different times. A group can be a committee, a review panel, a task force, an executive board, a team, or a permanent unit.

There are many potential benefits for making decisions in groups. The groups are better at understanding problems. It is better at catching errors. And a group has more information or knowledge than any one member and as a result, more alternatives for problem solving. Synergy may be produced. Working in a group may stimulate the participants and the process. The group members will have their ego embedded in the decision, so they will be committed to the implementation. The participation of the members in a decision means less likelihood to resist implementation.

Although there are many potential benefits to working in groups, there are also many dysfunctions. So many decisions that are made by groups are considered ineffective or inefficient. Getting a group together in one place and at one time can be difficult and expensive. Furthermore, group meetings can take a long time and the resulting decisions may be mediocre.

For many years there were attempt to improve the work of groups. If we can eliminate or lessen some of the phenomena that cause the dysfunctions, then the benefits can greatly be enhanced. Behavioral scientists, personnel experts, efficiency experts, and others have developed many approaches to the solution of the problems. For example, the Nominal Group Technique and the Delphi Method. They are manual methods. So they may be expensive, slow, and not so effective. Therefore attempts to improve the situation with the aid of information technology appear under several names, such as Groupwere, electronic meeting systems, collaborative systems, Computer-Supported Cooperative Work(CSCW), and group decision support systems.

Here are some typical definitions of GDSS: "A GDSS consists of a set of software, hardware, language components, and procedures that support a group of people engaged in a decision-related meeting"[Huber 84]. "A GDSS is an interactive, computer-based system that facilitates the solution of unstructured problems by a set of decision makers working together as a group"[DeSanctis and Gallupe 85]. Components of a GDSS include hardware, software, people, and procedures. These components are arranged to support a group of people, usually in the context of a decision-related meeting. A GDSS mainly supports the process of decision making rather than the solution of a specific problem.

GDSS is considered a subset of broader field titled *electronic meeting systems(EMS)* or Group Support Systems. EMS is defined by Dennis et al., [Dennis 88]as: An information technology(IT)-based environment that supports group meetings, which may be distributed geographically and temporally. EMSs support more tasks than just decision making, they focus on communication, moving beyond the GDSS decision room. The EMS concept is presented in detail in several publications from the University of Arizona [Dennis 91].

In generally, the group meetings have activities as followings; idea generation, idea organization, prioritization, and summarizing. When ideas were generated, *idea organization* activity involves taking ideas and refining, rearranging, and consolidating the items of idea based on some organizing principle or logical framework. [GS 92A] There are idea categorizer, idea organizer, and group outliner.

1.2 Objectives and Scope of the Research

The main concern of this research is to develop more effective idea organizer and categorizer for GDSS. In group meetings which are supported by GDSS, there are a lot of ideas which are generated rather than non-supported group meetings'. So a better idea categorizer is required to support the ideas organizing and categorizing.

In this research, we assume that the idea organization activity begin when idea generation was finished. The previous generated ideas were stored database. Under those circumstances, this research suggests an interactive idea categorizing method to support idea organizing in group meeting using GDSS. The suggest method carry out items of idea categorizing based on automatic document classification methodologies and information retrieval models.

The scope of the research is organized as follows. The related researches with idea categorizing are briefly surveyed in chapter 2. Chapter 3 propose an interactive idea categorizing method and its key algorithms. In chapter 4, implementation and demonstration of IIC based on theoretical foundations of suggested method is showed. Finally, the conclusions are discussed in chapter 5.

II. LITERATURE REVIEW

2.1 Prior Methods of Idea Categorizing

Categorizer allows the group to organize a file, e.g. the output from Electronic Brainstorming, into categories or themes. The group creates a list of categories, then attaches the information from the file into these categories. This tool can also be used to build a list. This is used for idea generation and idea organization. [GS 92A]

Idea Organization is a multi-faceted tool that can be used for idea generation and idea organization. Idea Organization has several levels of power and complexity. In its basic form, it allows participants to build a list of key ideas or categories and attach comments to list items. Comments may be created in the tool or taken from a pre-existing file that contains information you want to organizer, as in Categorizer. The file may be the output of an Electronic Brainstorming session, another tool report, or any ASCII text file. The list can be consolidated, rearranged, and edited. The supporting comments can also be edited.[GS 92B]

Group Outliner allows you to enter or import a simple list of topics or an elaborate multi-level set of topics. Levels of subordination are represented by a graphical outline structure. All or some portion of the outline is made available to all or selected participants, who can comment on the topics and view comments entered by others. It supports the group activity of idea generation and idea organization. [GS 92B]

2.2 Information Retrieval

A specific document is retrieved by similarity evaluation of given query and documents set with threshold value. In information retrieval system, it is a retrieval model that is the method with criteria to retrieve information . It is query that is the inputted question with form.

2.2.1 Boolean Retrieval Model

The Boolean model have been most widely used among commercially available Information Retrieval systems due to efficient retrieval and easy query formulation[Bookstein 81, Salton 89]. The query that represents information requirement using Boolean logic is simple and clear. Because, the query is consist of query terms and their logical relationships. In addition to that, computer processing is convenient.

The relationship of query terms in Boolean query is represented by logic operator. Using logic operator are *AND, OR,* and *NOT.* The Boolean retrieval logic using Boolean algebra based on logic and set theory.

The Boolean model is simple and easy to use. In the conventional Boolean retrieval model, however, various similarity cannot be computed between queries and documents. There have been many works in the past to overcome this problem. In Section 2.2.3 and 2.2.5 two elegant approaches such as the fuzzy set model and the extended Boolean model, which are logical extensions of the conventional Boolean model.

2.2.2 Probabilistic Retrieval Model

The probabilistic retrieval model is based on two main parameters, Pr(*relevance*), the probability of relevance, and Pr(*nonrelevance*), the probability of nonrelevance of record. If relevance is assumed to be a binary property, Pr(*nonrelevance*) = 1- Pr(*relevance*). In addition, two cost parameters are used, designated a_1 and a_2, representing the loss associated with the retrieval of a nonrelevant record and the nonretrieval of a relevant record, respectively. Because the retrieval of a nonrelevant record carries a loss of $a_1 \bullet [1-\Pr(relevance)]$, and the rejection of a nonrelevant item has an associated loss factor of $a_2 \bullet \Pr(relevance)$, the total loss caused by a given retrieval process will be minimized if an item is retrieved whenever

$$a_2 \bullet \Pr(relevance) \geq a_1 \bullet [1-\Pr(relevance)] \qquad (2.1)$$

Equivalently, a retrieval function g may be introduced, where

$$g = \frac{\Pr(relevance)}{1-\Pr(relevance)} - \frac{a_1}{a_2} \qquad (2.2)$$

and an item may then be retrieved whenever its g value is nonnegative.

Function *g* cannot be evaluated without relating it to other design parameters. In particular, the relevance properties of records must be related to the relevance properties of various terms attached to records. In particular, using Bayes' theorem to replace $\Pr(relevance\,|\,x)$ with $Pr(x\,|\,relevance)/Pr(x)$ for each record *x*, and assuming that the loss parameters a_1 and a_2 are equal, one obtains

$$\log g(x) = \log \frac{\Pr(x|relevance)}{\Pr(x|nonrelevance)} + \log \frac{\Pr(relevance)}{\Pr(nonrelevance)} \qquad (2.3)$$

where *Pr(relevance)* and *Pr(nonrelevance)* are the priori probabilities of relevance or nonrelevance of any record.[Salton 89]

2.2.3 Fuzzy Set Model

The conventional fuzzy set model based on the MIN and MAX operators support the similarity evaluation for Boolean retrieval systems by using document term weights[Buell 82, Radecki 79, Tahani 76, Salton 87]. In the conventional fuzzy set model the retrieval function *R(Q, D)* is defined to be equal to the function *F(D, Q)*. The function *F(D, Q)* is a membership function of the document *D* in the set of documents is retrieved by the given query *Q*.

Boolean Formulation	**Evaluation Formula**
$F(D, t_1\ AND\ t_2)$	$MIN(F(D, t_1), F(D, t_2))$
$F(D, t_1\ OR\ t_2)$	$MAX(F(D, t_1), F(D, t_2))$
$F(D, NOT\ t_1)$	$1 - F(D, t_1)$

Figure 2.1 Query-document similarity evaluation in the conventional fuzzy set model.

Many methods have been proposed to use the query weight weights [Radecki 79]. Suppose p is a term or clause in a given query and w is a weight for the term or clause. The value of *f(F(D, p), w)* is defined in the three ways shown in Figure 2.2

(a) $f(F(D, p), w) = w \bullet F(D, p)$

(b) $$\begin{cases} f(F(D,\ p),\ w) \ = \ w \bullet \ F(D,\ p) \ \ \textit{when p appears in an OR clause} \\ f(F(D,\ p),\ w) \ = \ \dfrac{1}{w} \bullet \ F(D,\ p) \ \ \textit{when p apears in an AND clause} \end{cases}$$

(c) $$\begin{cases} f(F(D,p),w) \ = \ F(D,p) & \textit{if} F(D,p) \geq w \\ f(F(D,p),w) \ = \ 0 & \textit{if} F(D,p) < w \end{cases}$$

Figure 2.2 Evaluation of weighted Boolean expression in the fuzzy set model.

2.2.4 Vector Space Model

The vector-space model is simplest to use and in some way the most productive. The vector-space model assumes that an available term set is used to identify both stored records and information requests. [Salton 83B, Raghavan 86] Both queries and documents can be represented as term vectors of the form

$$D_i = (a_{i1}, a_{i2}, \ldots\ldots, a_{it}) \qquad (2.4)$$

and

$$Q_j = (q_{j1}, q_{j2}, \ldots\ldots, q_{jt}) \qquad (2.5)$$

where the coefficients a_{ik} and q_{jk} represent the values of term k in document D_i or query Q_j, respectively. Typically a_{ik} (or q_{jk}) is set equal to 1 when term k appears in document D_i (or in query Q_j), and to 0 when the term is absent from the vector. Alternatively, the vector coefficients could take on numeric values, the size of the coefficient depending on the importance of the term in the respective document or query.

Consider now a situation in which *t* distinct terms are available to characterize record content. Each of the t terms can then be identified with a term vector *T*, and a vector space is defined whenever the *T* vectors are linearly independent. In such a space, and vector can be represented as linear combination of the t term vectors. Hence the *r*th document D_i can be written as

$$D_r = \sum_{i=1}^{t} a_{ri}\, T_i \qquad (2.6)$$

where the a_{ik}s are interpreted as the components of D_r along the vector T_i.

In vector space, the similarity between document and query can be computed as

$$D_r \bullet Q_s = \sum_{i,j=1}^{t} a_{ri}\, q_{sj}\, T_i \bullet T_j . \qquad (2.7)$$

Computing the similarity values of expression(2.4) thus depends on a specification of the document and query components. Assuming that the terms are uncorelated, the term-document similarity computation of expression(2.4) is reduced to the simple sum-of-products form of expression (2.5):

$$Sim(D_r \bullet Q_s) = \sum_{i,j=1}^{t} a_{ri}\, q_{sj} \qquad (2.8)$$

A similar computation can then be used to obtain pair-wise similarity measurements between documents, the latter forming a basis for certain document-clustering systems:

$$Sim(D_r \bullet D_s) = \sum_{i,j=1}^{t} a_{ri}\, q_{sj} \qquad (2.9)$$

The vector-space model can be used to obtain correlations, or similarities, between pairs of stored documents, or between queries and documents, under the assumption that the t term vectors are orthogonal, or that the term vectors are linearly independencies, so that a proper basis exists for the vector space.

The disadvantages of the vector-processing model relate to the assumed orthogonality, and hence independence between terms, and the lack of theoretical justification for some of the vector-manipulation operations operations. [Salton 83B] Some of the advantages are the model's simplicity, the ease with which it accommodates weighted terms, and its provision of ranked retrieval output in decreasing order of query-document similarity.

2.2.5 Extended Boolean Model

The extended Boolean model represents a unifying retrieval model in which the conventional Boolean model, the conventional fuzzy set model and the vector space model are special cases [Salton 83A, Salton 89]. The query-document similarity defined in the extended Boolean model is based on L_p vector norm computations, and is controlled by a parameter p, $1 \le p \le \infty$, providing a special interpretation for the Boolean operators. As the value of p decreases, the interpretation of the Boolean operators is relaxed more and more. The distinction between the Boolean *AND* and *OR* operators is lost completely when the p value reaches its lower limit.

The extended Boolean model provides a way of evaluating the query and document weights. The coument value can be calculated by the same procedure as the fuzzy set model. The only difference between two retrieval models is the similarity evaluation formulas calculating the similarity between a weighted Boolean query and a weighted document. In the extended Boolean model the retrieval function *R(q, D)* is computed with the similarity evaluation rules given in Figure 2.4

$$F(D,(t_1,w_1)\ AND\ (t_2,w_2)) = 1 - \left[\frac{w_1^p(1-F(D,t_1))^p + w_2^p(1-F(D,t_2))^p}{w_1^p + w_2^p}\right]^{1/p}$$

$$F(D,(t_1,w_1)\ OR\ (t_2,w_2)) = 1 - \left[\frac{w_1^p F(D,t_1)^p + w_2^p F(D,t_2)^p}{w_1^p + w_2^p}\right]^{1/p}$$

$$F(D, NOT(t_1,w_1)) = 1 - w_1 F(D,t_1)$$

Figure 2.3 Similarity evaluation rules in the extended Boolean model.

2.3 Automatic Document Classification

The conventional document classification had been carried out manually. But it has been tried to classify automatically since in the 1960's. The document classification is a grouping task using similarity between the contents of documents. It has been carried out conventionally according to a classification table [Jeong 93].

In automatic document classification, there are two approaches. One is to use of already fixed classification table. This is to allocate documents among the given categories. The other is to build according to similarity between the contents of documents instead of a priori classification table.

2.3.1 Classification

The automatic document classification is defined that is to classify automatically among the given categories or the generated categories by experience, which is using a priori classification table. [Hamill 80, Borko and Bernick 83]

There is an example of experiment which is the automatic document classification. In this experiment, it was used the frequency of appeared keywords in the documents. Figure 2.4 is shown the calculation equation of probability of the document include word *i*, given *j*th category.

$$p(c_j | w_k \cdot w_m \cdots w_s) = kp(c_j) \cdot p(w_k | c_j) \cdot p(w_m | c_j) \cdots p(w_s | c_j)$$

w_i : word in document,

c_j : jth subject category,

$p(w_i\ c_j)$: probability of the document include word i, when the document is included jth category ,

$p(c_j)$: probability that a document belongs to jth category,

k : scaling factor such as $\sum_j p(c_j | w_k \cdot w_m \cdots w_s) = 1$.

Figure 2.4 Equation for probability calculation.

A disadvantage of conventional automatic document classification methodology, using a priori classification table is that in many cases a priori classification table is not exist or difficult to build classification category.

2.3.2 Clustering

Two main strategies can be used to carry out the cluster generation. First, a complete list of all pairwise item similarities can be constructed, in which case it is necessary to employ a grouping mechanism capable of assembling into common clusters items with sufficiently large pairwise similarities. Alternatively, heuristic methods can be used that do not require pairwise item similarities to be computed.

Hierarchical Cluster Generation

When cluster generating does depend on pairwise item similarities, a term-document matrix is conveniently used as a starting point, followed by a comparison of all distinct pairs of matrix rows to be used for document clustering(or of matrix columns to be used for term clustering). The pairwise comparison of matrix columns produces *N(N-1)/2* different pairwise term similarity coefficients for the documents(or *t(t-1)/2* different pairwise term similarities). No matter what specific clustering method is used, the clustering process can be carried out either divisively or agglomeratively. In this case, the complete collection is assumed to represent one complete cluster that is subsequently broken down into smaller pieces. In the latter, individual item similar are used as a starting point, and a gluing operation collects similar items, or groups, into larger groups[Griffiths, Robinson and Willett 84]. Several methods have been proposed to generate several cluster, using graph theory. There are single-link clustering, complete-link clustering, and group-average clustering.

Heuristic Clustering Methods

The hierarchical clustering strategies just described are based on prior knowledge of all pairwise similarities between items. Therefore the corresponding cluster-generation methods are relatively expensive to perform. In return, these methods produce a unique set of well-formed clusters for each set of data, regardless of the order in which the similarity pair are introduced into the clustering process.

2.3.3 Measures of Similarity

When clustering, we can usually use some measure for evaluating similarity coefficient between items. A list of typical similarity measures appears in Figure 2.5. [Salton 83B]

1. Dice's coefficient	$\frac{2\lvert X \cap Y\rvert}{\lvert X\rvert+\lvert Y\rvert}$
2. Jaccard's coefficient	$\frac{\lvert X \cap Y\rvert}{\lvert X \cup Y\rvert}$
3. Cosine coefficient	$\frac{\lvert X \cap Y\rvert}{\lvert X\rvert^{1/2} \times \lvert Y\rvert^{1/2}}$
4. Overlap coefficient	$\frac{\lvert X \cap Y\rvert}{\min(\lvert X\rvert,\lvert Y\rvert)}$
5. Tanimoto coefficient	$\frac{\lvert X \cap Y\rvert}{\lvert X\rvert+\lvert Y\rvert-\lvert X \cap Y\rvert}$

$|X|$, $|Y|$ is number of keywords

$|X \cap Y|$ is number of terms appearing jointly in X and Y.

Figure 2.5 Measures of similarity.

III . THE INTERACTIVE METHOD OF IDEA CATEGORIZER

3.1 Criticism on the Previous Categorizing Methods

In section 2.1 there are several methods and tools for idea organizing and categorizing in group meeting using GDSS. All described previous methods have many disadvantages.

All procedures of described methods carry out manually. We have to assume that almost participants are not accustomed to use meeting system. It has too much workload on participants. Specially those tools are difficult to use for beginners. In addition, the procedure is top down approach. This approach can be useful for idea classification. But it is not useful as a lot of ideas were generated.

In previous idea organization and categorizing methods, all participants who are present at group meeting using GDSS build category respectively. Therefore various categories which have similar mean can then be built by each participant. The categorizer have no functions to resolute redundancy between idea categories. In addition, an idea can be classified to different categories by the different participants. Those occur the problem that requires additional efforts.

So we need a better method to support idea organization or categorizing in group meeting using GDSS. In information retrieval, there are some models. And these models have the useful concepts for idea categorizing. Also, In clustering, there are some methods for similarity measuring and document clustering.

3.2 Information Retrieval and Idea Categorizing

3.2.1 Vector Representation of Idea

The generated *n* number of ideas can then be represented as keyword vectors of the form

$$I_i = (a_{i1}, a_{i2}, \ldots\ldots, a_{in}) \qquad (3.1)$$

where the coefficients a_{ik} represent the values of keyword *k* in idea I_i. Typically a_{ik} is set of equal to 1 when keyword *k* appears in idea I_i, and to 0 when keyword is absent from the vector. Figure 3.1 present a typical idea vector in vector space.

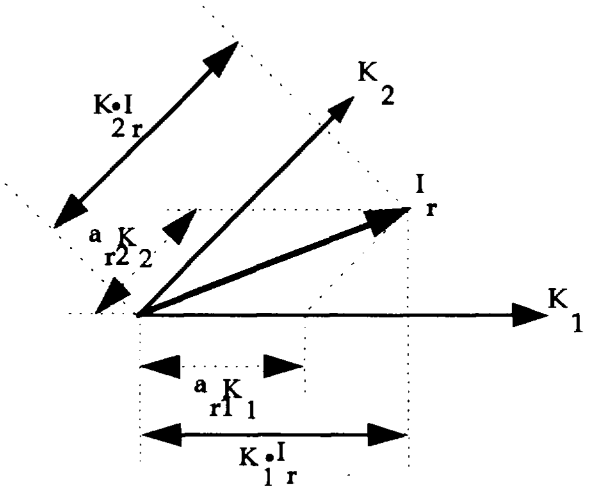


Figure 3.1 Representation of a typical idea vector in vector space.

3.2.2 Procedure of the Method

The procedure of this method can be explained two parts, calculation of the similarity between ideas and interactive clustering. The first, all generated ideas in Electronic Brainstorming are loaded from database. The user input keywords. The keyword's affinity are generated by user's input and priori knowledge. All ideas are extended using the keyword's affinity. All extended ideas are weighted using appearance frequency of keywords. The similarities between ideas are computed from weighted ideas. After similarity calculating between ideas, interactive clustering procedure is carried out as shown Figure 3.2. The procedure search candidate ideas pair. If candidate ideas pair is found than the procedure carry out alternative operation using critical region. If critical ideas pair is occurred than the procedure receive user's choice. The selected ideas pair will be linked each other. The previous loop will be continue until there is no candidate ideas pairs. After interactive clustering procedure, the user give a name to generated clusters.

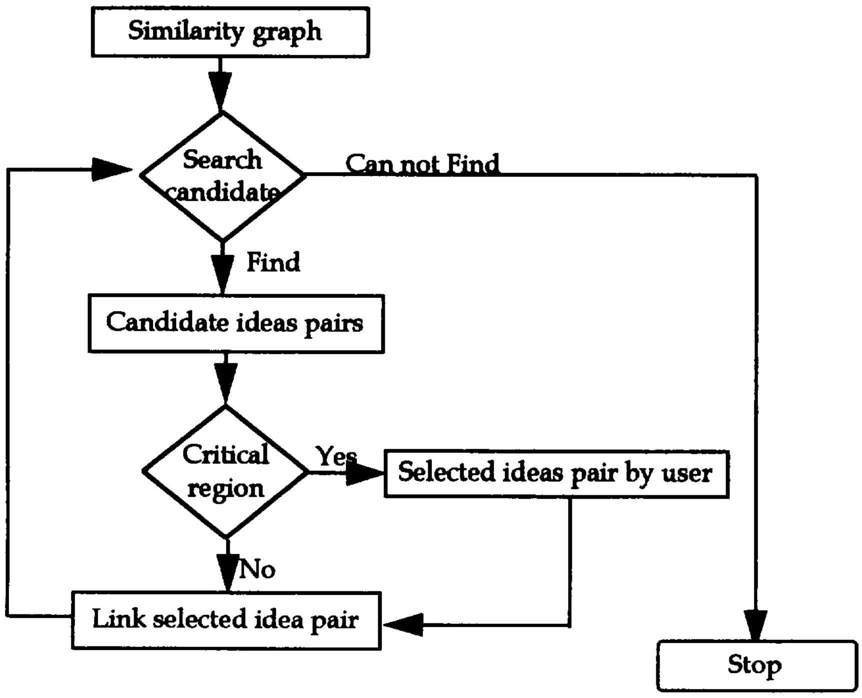


Figure 3.2 The procedure of interactive idea clustering

3.3 Key Algorithms of the Method

This section contain key algorithms such as keyword affinity matrix generation, weighted idea matrix generation, similarity matrix generation, and interactive clustering.

3.3.1 Keyword Affinity Matrix

The keyword affinity matrix is generated using knowledge base which contain degrees of affinity between keywords. If there is not exist all affinity degrees, user input the missed affinity degree. All participants of group meeting select a value from given set of degrees. The set consist of several degrees such as Strongly Similar, Similar and Not Similar. Input degrees are converted to fuzzy values.

Keyword affinity matrix can then be represented as

$$K = \left\| k_{ij} \right\|_{n \times n} \quad 0 \le k_{ij} \le 1 \qquad (3.2)$$

The generating algorithm is presented as fellows.

$$Affinity(T_i, T_j) = Max\{Min[Affinity(T_i, T_k), Affinity(T_k, T_j)]\} \; k = 1, n \quad (3.3)$$

3.3.2 Weighted Idea Matrix

Extended idea matrix

The ***extended idea matrix R*** is generated from keywords affinity matrix. It can then be represented as keyword vectors of the form

$$R = \left\| r_{ij} \right\|_{m \times n} \qquad (3.4)$$

There is the generating algorithm which is presented by C language.

```
float **gen_r_matrix(byte **i_matrix, float **a_matrix)
{
        int m, n, l;
        float **r_matrix;
        r_matrix = malloc(sizeof(float *) * i_cnt);
        for(l = 0; l < i_cnt; l++)
        r_matrix[l] = (float *) malloc(sizeof(float) * k_cnt);
        for(m = 0; m < i_cnt; m++)
           for(n = 0; n < k_cnt; n++)
                    r_matrix[m][n] = 0.;
        for(m = 0; m < i_cnt; m++)
           for(n = 0; n < k_cnt; n++)
                    if(i_matrix[m][n] != 0)
                             for(l = 0; l < k_cnt; l++)
                                      if(i_matrix[m][l] <= a_matrix[n][l])
                                               r_matrix[m][l] = a_matrix[n][l];
        return r_matrix;
}
/*      i_matrix[i][j] : the element of initial idea matrix I              */
/*      a_matrix[i][j] : the element of keyword affinity matrix K                    */
/*      r_matrix[i][j] : the element of extended idea matrix               */
```

Weight matrix of keywords

The weight matrix of keywords is generated from appearance frequency of keywords. The keyword list Y is vector that have elements y_i , number of kinds of keywords. W is weight matrix of keywords. They are presented the forms

$$Y = \{y_i\}_{i=1,n} \quad y_i \geq 0 \tag{3.5}$$

$$W = \|w_{ij}\|_{m \times n} \quad 0 \leq w_{ij} \leq 1 \tag{3.6}$$

The generating algorithm is presented as fellows.

```
float **gen_w_matrix(float **r_matrix)
{
        float **w_matrix;
        int m, l;
        int cnt = 0;
        w_matrix = malloc(sizeof(float *) * i_cnt);
        for(l = 0; l < k_cnt; l++)
           w_matrix[l] = (float *) malloc(sizeof(float) * k_cnt);
        for(m = 0; m < i_cnt; m++)
           for(l = 0; l < k_cnt; l++)
                    w_matrix[m][l] = 0.;
        for(m = 0; m < i_cnt; m++){
           for(l = 0; l < k_cnt; l++)
```

```
                        if(r_matrix[m][l] != 0)
                                cnt += keywordlist[l]->frequency;
                for(l = 0; l < k_cnt; l++)
                        w_matrix[m][l] = (float)keywordlist[l]->frequency / cnt;
            }
            return w_matrix;
}
/*      r_matrix[i][j] : the element of extended idea matrix                    */
/*      w_matrix[i][j] : the element of  weight matrix W                        */
```

Weighted idea matrix

The weighted idea matrix is generated from extended idea matrix and weight matrix. D is weighted idea matrix . The matrix can be presented as fellows.

Let $D = W \otimes R$, then $\|d_{ij}\|_{m\times n} = \|w_{ij} \cdot r_{ij}\|_{m\times n} \quad 0 \le d_{ij} \le 1$ (3.7)

$$R = \|r_{ij}\|_{m\times n}, \quad W = \|w_{ij}\|_{m\times n}$$

3.3.3 Similarity Matrix

The similarity matrix is generated using weighted idea matrix. S is idea similarity matrix, which is presented as fellows.

$$D = \{D_i\}_{i=1,m} \quad D_i = (d_{ij})_{j=1,n} \tag{3.8}$$

$$S = \|Sim(D_i, D_j)\|_{m\times m} \tag{3.9}$$

$$Sim(D_i, D_j) = \frac{D_i \cdot D_j}{|D_i| \cdot |D_j|} = \cos\theta \qquad 0 \le \theta \le \tfrac{p}{2} \tag{3.10}$$

Hence, the similarity between ideas is represented by cosine value of vector D_i, D_j as shown Figure 3.3. Figure 3.4 present similarity graph.

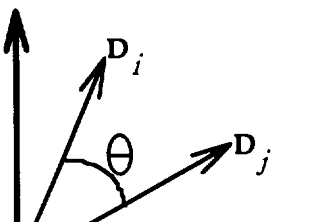
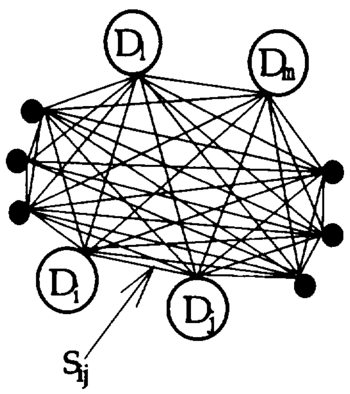


Figure 3.3 $Sim(D_i, D_j)$, cosine value of vector D_i, D_j



$0 \le S_{ij} \le 1$

Figure 3.4 The similarity graph

3.3.4 Interactive Clustering

The interactive idea clustering is carried out based on single linked clustering method. We add the alternative operation for interaction with user. The alternative operation is look for candidate ideas pair which is contained critical region. Therefore, we have to define critical region. The critical region is defined as difference of current threshold value and critical level. We will call critical ideas pair that contained critical region.

The interactive idea clustering is carried out as fellows.

Step 1 : Initialize critical level

We can use given critical level but we can change the critical level.

Step 2 : Look for candidate ideas pair

Look for candidate ideas pair which has the highest value of similarity.

If there is no candidate ideas pair than go to Step 6.

Step 3 : Look for critical ideas pair

Compute critical region.

If there is critical ideas pair than go to Step 5.

Step 4 : Link the ideas pair each other

Link selected ideas pair each other than go to Step 2.

Step 5 : Dialog(receive user's choice)

Display critical ideas pairs to user.

Receive user's choice than go to Step 4.

Step 6 : Input category name

Receive category name from user.

Step 7 : Saving category and ideas

Save the result.

End.

3.4 Conclusions

In this chapter, we developed an interactive method to categorize ideas using keyword's affinity and idea's similarity. The suggested method is bottom up approach. It offer to solve that idea overlapping and category redundancy problems.

Further research should focus on automatic keyword generating and idea understanding. To improve efficiency, the knowledge was managed more effectively.

IV. AN INTERACTIVE IDEA CATEGORIZER: IIC

4.1 Architecture of IIC

The main role of IIC is to support task which is to generate some reasonable idea clusters for facilitator. Hence, IIC is developed for achieving the idea categorizing in such circumstances.

The architecture of IIC consists of Similarity Calculator between ideas, Interactive Cluster Generator, Database, Knowledge base, and User Interface as shown in Figure 4.1.

Similarity Calculator

Similarity Calculator include idea matrix generator, keyword affinity matrix generator, extended idea matrix generator, weight matrix generator, weighted idea matrix generator, and similarity matrix generator. Idea matrix generator brings ideas and keywords which are stored in database and then build idea matrix using keyword list.

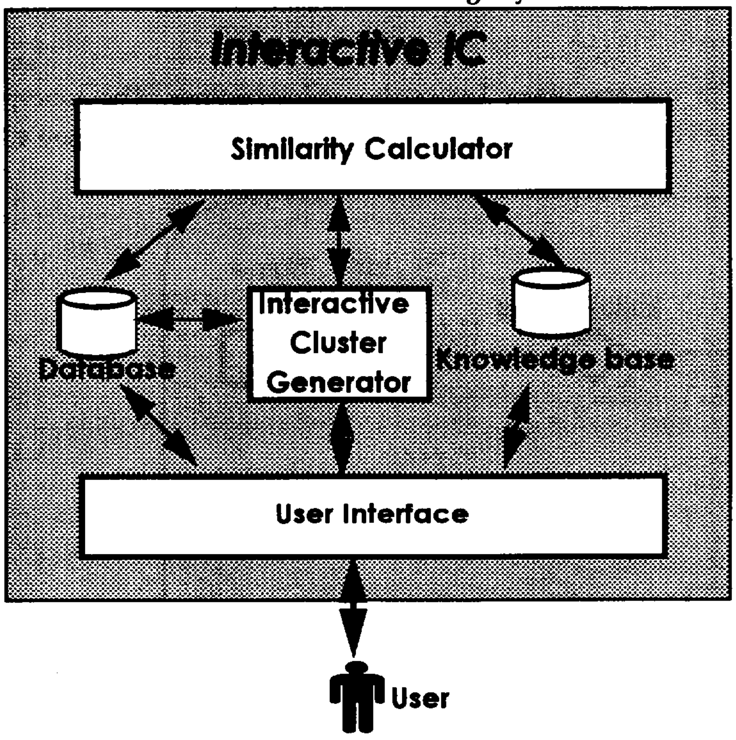


Figure 4.1 Architecture of IIC

Keyword affinity matrix generator searches affinity between keywords which are stored in knowledge base. In this process, keyword affinity matrix generator can infer affinity between keywords using fuzzy operation. If affinity of keywords can not find than keyword affinity matrix generator demand to input informations' of new keyword affinity from user.

Extended idea matrix is generated from idea matrix and keyword affinity matrix. Weight matrix generator brings ideas and keywords which are stored in database and then build weight matrix using appearance frequency of keywords in those ideas. Weighted idea matrix is generated from element's production of extended idea matrix and weight matrix. In finally, similarity matrix generator calculates similarity between ideas which are represented vectors, using pairwise comparison.

Similarity Calculator is implemented on Borland C++. This is implemented for the computational efficiency by C language.

Interactive Cluster Generator

Interactive Cluster generator is programmed algorithm base on single linked clustering. It is that offer user a flexibility and authority of idea categorizing. Interactive Cluster generator has an alternative operation which needs critical level, and look ahead function. The initial critical level is given by program. But, the value can be changed on the interactive clustering by user. It is that can offer interaction between user and system through those functions. Interactive Cluster generator has the number of clusters. Also, It can be used to interact between user and system through changing the number of clusters.

Interactive Cluster generator is implemented on Borland C++. This is implemented by C language.

Database

Database is implemented to manage ideas, keywords, and categories. The conceptual design database for IIC is designed using Entity-Relationship Diagram as shown in Figure 4.2.

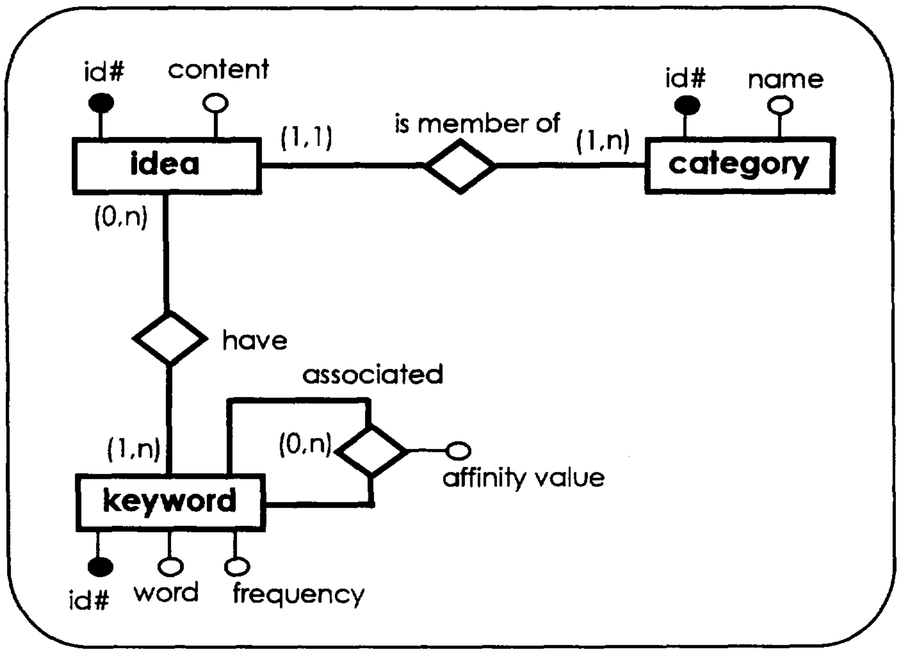


Figure 4.2 Entity-Relationship Diagram of Database for IIC

E-R diagram of Database for IIC is consist of three entity and three relationships. The attributes of idea entity are identifier number and contents. The attributes of category entity are identifier number and category name. The attributes of keyword entity are identifier number, word, and appearance frequency. The relationship of idea entity and category entity have many to one cardinality. The relationship of idea entity and keyword entity have many to many cardinality. The keyword entity has a recursive relationship which represent affinity between keywords. Hence, the relationship has a attribute, affinity value. For logical database design, we can transform E-R diagram into five relations. But, this is a part of database for integrated Group Decision Support System. Database is implemented on Borland dBASE V for Windows. dBASE offer developer a 4GL and script language. It offers three things that set it apart from traditional language; Object-Oriented Programming, Inheritance and Properties, and Message-Driven programming. In this system, we use dBASE to manage ideas, keywords, and generated categories.

Knowledge base

Knowledge base is implemented to manage affinity between keywords. We need divide the part of keywords affinity from Database. If this system is used by members of same organization than we can assume that they use the keywords which have same mean. It is important. Because the knowledge of keywords' affinity can be reused in this case. Hence, we need to store the knowledge of keywords' affinity. After, we can use the knowledge in the categorizing session of another meeting. The keywords' affinity can be inferred by keyword affinity matrix generator, using fuzzy operation which was described in Chapter 3.

In this system, we use dBASE to manage the knowledge of keywords' affinity. But, keyword affinity matrix generator of Similarity Calculator has inference engine.

User Interface

The user interface is very important issue in design of Group Support System. The design of user interface is important because the IIC has the interaction with user for reaching satisfied idea categorizing. In this system, we implement screens to show generated ideas and categorized ideas. The screen for keyword affinity matrix generator is implemented matrix stile. The idea's pair which will be linked from candidate idea's pairs is chosen simply by user.

4.2 Demonstration with an Example

In this section, we will describe IIC among an example. IIC is started by loading stored ideas from database.

We use a simple example which can be categorized exactly by human's common. Also, we will describe the inside of the IIC for easily understand. Used example for demonstration of IIC has 6 ideas which are generated in brainstorming session by participants as shown as Figure 4.3.

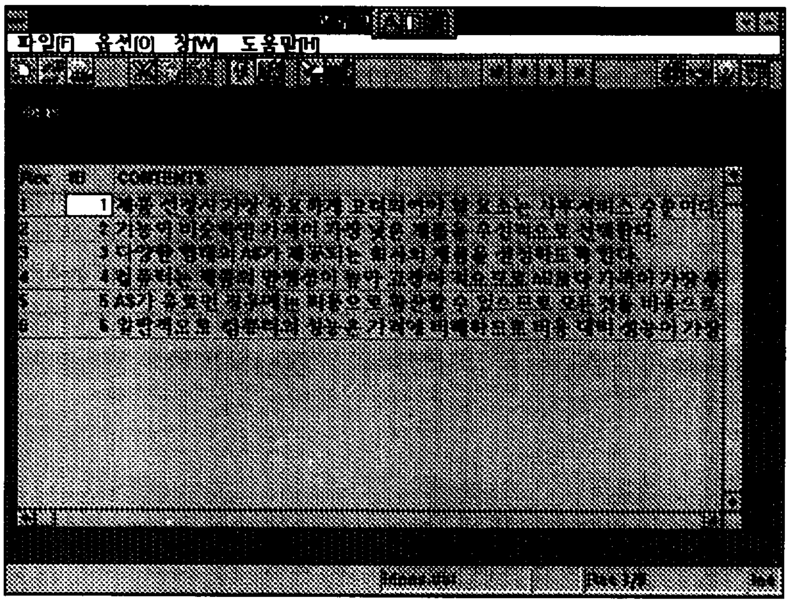


Figure 4.3 The generated ideas of the example

	k1	k2	k3	k4	k5	k6	k7	k8
idea # : 1	1	0	0	0	0	0	0	0
idea # : 2	0	1	1	0	0	0	0	0
idea # : 3	0	0	0	1	0	0	0	0
idea # : 4	0	0	1	1	1	0	0	0
idea # : 5	0	0	0	1	0	1	1	0
idea # : 6	0	0	1	0	0	1	0	1

Figure 4.4 Initial ideas matrix

Idea matrix generator of Similarity Calculator generates initial idea matrix as shown Figure 4.4. The value of elements has integer number, 1 or 0. The value is 1 set equal to when a keyword appears in the idea, and to 0 when the keyword is absent from the idea.

Next, keyword affinity matrix generator infer affinity between keywords using fuzzy operation from knowledge base. If affinity of keywords can not find than keyword affinity matrix generator demand to input informations' of new keyword affinity from user.

Extended idea matrix is generated based on initial idea matrix and keyword affinity matrix by extended matrix generator of Similarity Calculator as shown Figure 4.5. The value of elements has real number between 0 and 1.

	k1	k2	k3	k4	k5	k6	k7	k8
idea # : 1	1.0	0.0	0.0	0.9	0.1	0.0	0.0	0.0
idea # : 2	0.0	1.0	1.0	0.0	0.0	0.8	0.7	0.0
idea # : 3	0.0	0.0	0.0	1.0	0.1	0.1	0.0	0.0
idea # : 4	0.0	0.0	1.0	1.0	1.0	.8	0.0	0.0
idea # : 5	0.0	0.0	0.0	1.0	0.1	1.0	1.0	0.7
idea # : 6	0.0	0.0	1.0	0.0	0.0	1.0	0.7	1.0

Figure 4.5 Extended ideas matrix

Weight matrix is generated to build weighted idea matrix using keyword's appearance of each idea. Weight matrix generator of Similarity Calculator finds keyword's appearance from the example as shown Figure 4.6.

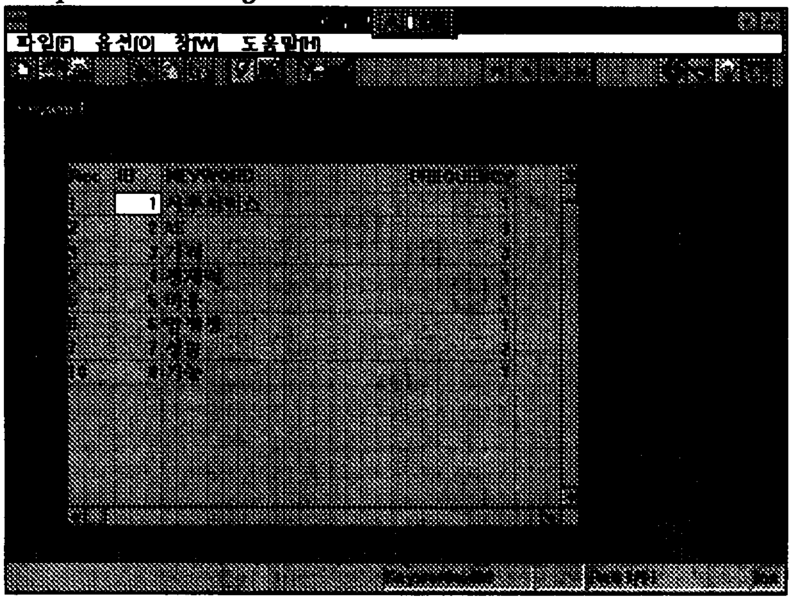


Figure 4.6 The keywords appearance of the example

Weighted idea matrix is generated based on extended idea matrix and weight matrix by weighted idea matrix generator of Similarity Calculator. Next, idea similarity matrix is generated based on weighted idea matrix by similarity matrix is generator as shown Figure 4.7. The value of elements has real number between 0 and 1. Idea similarity graph is generated based on idea similarity matrix as shown Figure 4.8.

	1	2	3	4	5	6
1	1.000	0.000	0.997	0.667	0.655	0.000
2		1.000	0.000	0.571	0.658	0.903
3			1.000	0.660	0.670	0.000
4				1.000	0.818	0.552
5					1.000	0.642
6						1.000

Figure 4.7 Idea similarity matrix

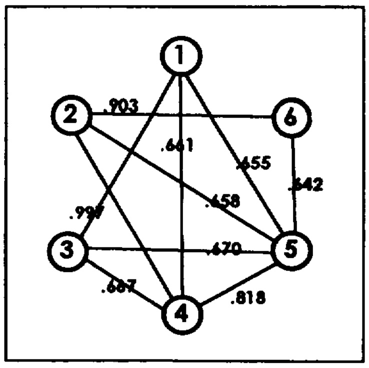


Figure 4.8 Idea similarity graph

The ideas is clustered by Interactive Cluster Generator. The line which is shown dots present candidate idea's pair.

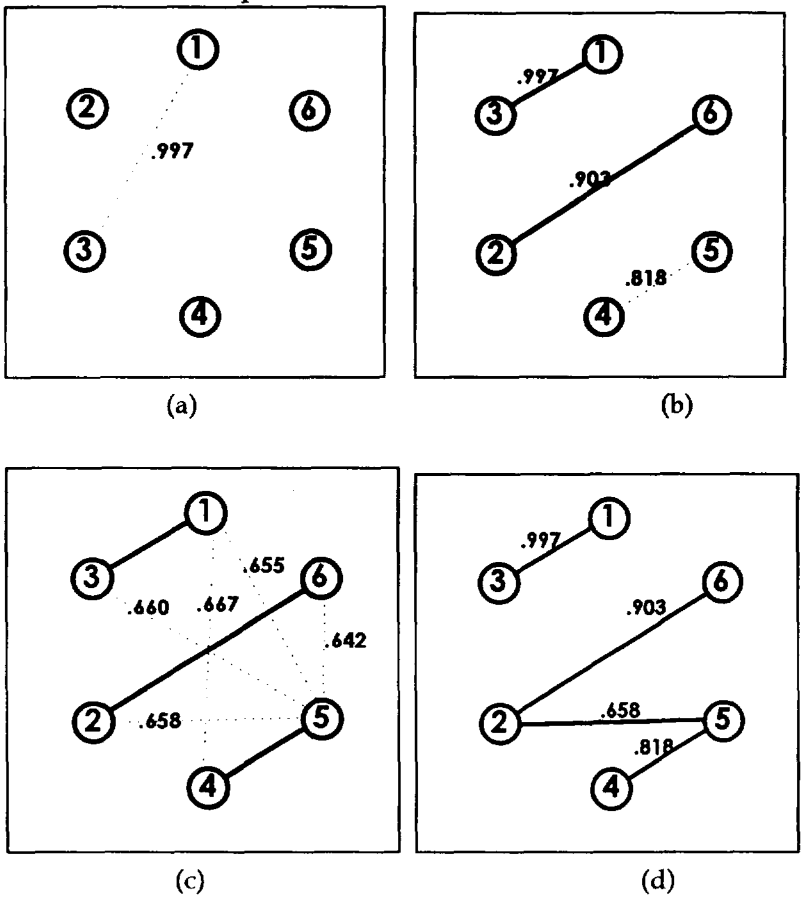
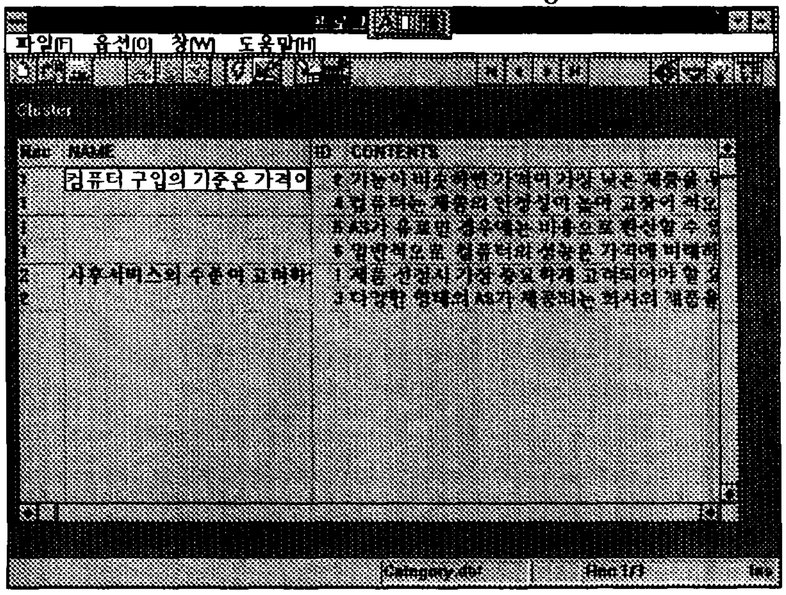


Figure 4.9 The Process of interactive idea clustering



Rat	NAME	ID	CONTENTS
1	컴퓨터 구입의 기준은 가격이	9	기능이 비슷하면 가격이 가장 낮은 제품을
1		4	컴퓨터는 제품의 안정성이 높아 고장이 적은
1		5	AS가 유료인 경우에는 비용으로 환산할 수 있
1		6	일반적으로 컴퓨터의 성능은 가격에 비례하
2	사후서비스의 수준이 고려하	1	제품 선정시 가장 중요하게 고려되어야 할 요
2		3	다양한 형태의 AS가 제공되는 회사의 제품을

Figure 4.10 Result of categorizing

In this example, the critical value is 0.05. In Figure 4.9, interactive clustering procedure is shown for this example. The multiple candidate idea's pairs were occurred as shown (c) by alternative operation. User chose the idea's pair (2,5). The line which is shown bold stile present clustered ideas. The Interactive Cluster Generator was stooped. Because the candidate idea's pair was not generated. The result of clustering is presented in (d). Finally, the cluster is named by user as shown Figure 4.10.

4.3 Comparison

In this section, We will describe the result of an experiment. We observed the accuracy of idea categorizing and categorizing time for Categorizer of GroupSystems, the without interactive categorizer, and IIC, respectively. The observation of the idea categorizing accuracy is a difficult problem. Hence, we compare with categorizing result which was categorized by human under enough time that was 2 hours for categorizing as shown Figure 4.11 and its graph is present in Figure 4.12.

# of Ideas	GroupSystems	W/O Interactive	Interactive IC
6	overlapped ideas: 0 # of categories: 2	incorrect classified ideas : 2 # of categories: 2	incorrect classified ideas : 0 # of categories: 2
15	overlapped ideas: 1 # of categories: 3	incorrect classified ideas : 4 # of categories: 3	incorrect classified ideas : 1 # of categories: 3
24	overlapped ideas: 3 # of categories: 4	incorrect classified ideas : 5 # of categories: 4	incorrect classified ideas : 2 # of categories: 2
50	overlapped ideas: 7 # of categories: 6	incorrect classified ideas : 8 # of categories: 7	incorrect classified ideas : 5 # of categories: 6

Figure 4.11 Comparison of the idea categorizing accuracy

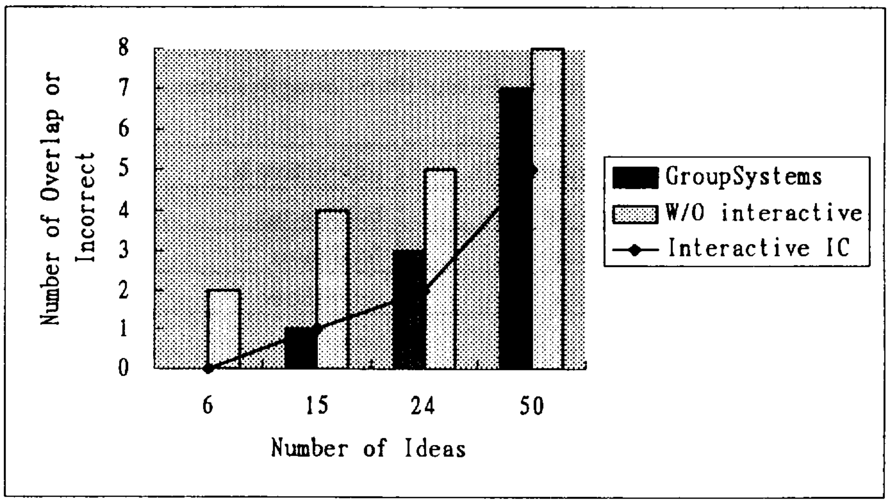


Figure 4.12 Comparison of the idea categorizing accuracy

The result is shown that IIC has high accuracy. The without interactive categorizing method is fast than the methods but it has low accuracy. The categorizing time of GroupSystems is long than the other methods as shown Figure 4.13. and its graph is present in Figure 4.14.

# of Ideas	GroupSystems	W/O Interactive	Interactive IC
6	3.5 minutes	2.5 minutes	3.0 minutes
15	9.0 minutes	4.5 minutes	5.5 minutes
24	14.0 minutes	6.6 minutes	8.3 minutes
50	21.0 minutes	8.5 minutes	12.5 minutes

Figure 4.13 Comparison of the idea categorizing time

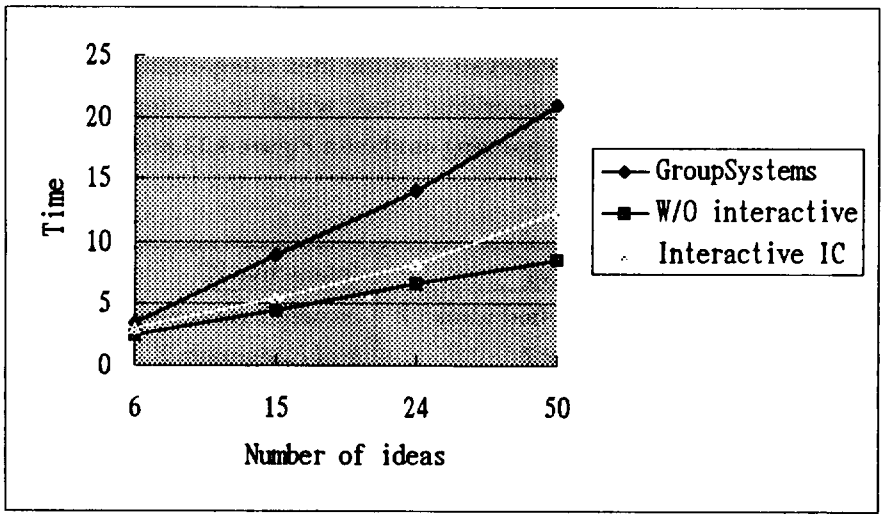


Figure 4.14 Comparison of the idea categorizing time

The previous comparison between methods is rough. It is difficult to compare the method's efficiency and effectiveness. In finally, we described the property of the categorizing methods as shown Figure 4.15. IIC is bottom approach and GroupSystems is top down approach. GroupSystems carry out idea categorizing manually. IIC carry out idea categorizing interactively. IIC has a good advantage which is the reuse of keyword's affinities. It offer the efficiency. In a lot of ideas were generated, IIC is more efficient than GroupSystems.

	GroupSystems	W/O Interactive	Interactive IC
Approach	top down	bottom up	bottom up
User	participant	facilitator	facilitator
Classification	manual	automatic	interactive
Keyword's affinity	don't use	use	use
Idea overlapping	allow	doesn't allow	doesn't allow
Category redundancy	allow	doesn't allow	doesn't allow
Information	user preference information	keyword's affinity	keyword's affinity, user preference information
Reuse	impossible	possible	possible
Interaction	frequently	almost never	sometimes

Figure 4.15 Comparison of the property of the categorizing methods

4.5 Conclusions

In this chapter, we developed the IIC which is a idea categorizer for Group Decision Support System. IIC is based on the similarity computing and the interactive algorithm. The interactive algorithm which is used IIC is more efficient.

Further researcher should be focus on natural language understanding. It is to improve intelligence of idea categorizer. It offer more general idea categorizing to be able to applied in real word.

V. CONCLUSIONS

This research is an effort to solve the idea categorizing problem in an previous GDSS. The previous idea categorizing is performed mostly by participants' manual works. In comparison with this existing idea categorizing, an interactive idea categorizing suggested in this paper reduced 40% of the categorizing time. The quality of categorized result was also acceptable in view that there is hardly any difference compared to the manual categorizing work which was performed for enough time. Also, because the suggested IIC doesn't allow categorizing an idea in multiple categories and generating almost similar redundant categories, There need not any additional step to adjust the initial categorizing result. Knowledge in regard to affinities among keywords are accumulated and used later as a knowledge base. Owing to the flexibility of the system, participants can influence their original intentions interactively for any ambiguous ideas.

In an previous categorizing method, participants should classify ideas manually and intuitively looking at whole generated ideas in an top-down manner. IIC solve this burden and inconvenience. IIC also has an important feature that even an novice on computer can use the system without any difficulty.

Although bottom-up categorizing approach suggested in this paper has a lot of advantages, there also need to be supplemented through further researches. First, it is necessary to study further on natural language processing and natural language understanding to identify keywords more completed and automatically from the generated ideas. These may be helpful to translate the contextual meaning of an idea more accurately. The human computer interface is another important factor in the interactive system research, but passed over without any special notice. And, this prototype version developed for validation test of the bottom-up approach should be up graded to be integrated in a full version GDSS.

Finally, a well designed representing tool for the categorizes ideas is important no less than categorizing ideas well. Idea categorizing is a factor of the systematic Idea settlement. Therefore, further research of this paper should be extended to the methodology and the system development which can support full process from idea generation, to idea categorizing and applying to the next step.

References

[Aiken 94] M.Aiken, J.Krosp, A.Shirani and J.Martin, "Electronic Brainstorming in Small and Large Groups," *Information & Management*, Vol.27, pp.141-149, 1994.

[Bookstein 81] A.Bookstein, "A Comparision of Two Systems of Weighted Boolean Retrieval," *Journal of the American Society for Information Science (JASIS)*, Vol.32, No.4, pp.275-279, 1981.

[Borko and Bernick 83] H.Borko and M.Bernick, "Automatic Document Classification," *Journal of the Association for Computing Machinery*, Vol.10, No.1, pp.151-162, 1983.

[Bruandet 89] M.F.Bruandet, "Outline of A Knowledge-Base Model for An Intelligent Information Retrieval System," *Information Processing & Management*, Vol.25, No.1, pp.89-115, 1989.

[Buell 82] D.Buell, "An Analysis of Some Fuzzy Subset Applications to Information Retrieval Sysems," *Fuzzy Sets and Systems*, Vol.17, No.3, pp.35-42, 1982.

[Dennis 88] Dennis, A., et al. "Information Technology to Support Electronic Meetings." *MIS Quarterly*, December 1988.

[Dennis 91] Dennis, A., et al. "Group, Sub-Group and Nominal Group Idea Generation in an Electronic Meeting Enviroment," *System Science*, Vol.3, pp573-579, 1991.

[DeSanctis and Gallupe 85] DeSanctis, G. and B. Gallupe. "Group Decision Support Systems: A new Frontier," *Management Science*, May 1987.

[Griffiths, Robinson and Willett 84] A.Griffiths, L.A.Robinson, and Willett, "Hierarchic Agglomerative Clustering Methods for Automatic Document Classification," *Journal of Documentation*, Vol.40, No.3, pp.175-205, 1984.

[GS 92A] "*GroupSystems V: Basic Tools Manual,*" Ventana Corporation, 1992.

[GS 92B] "*GroupSystems V: Advanced Tools Manual,*" Ventana Corporation, 1992.

[Hamill 80] K.A.Hamill and A.Zamora, "The Use of Titles for Automatic Document Classification," *Journal of the American Society for Information Science (JASIS)*, Vol.31, No.6, pp.396-402, 1980.

[Huber 84] Huber, G.P. "Issues in the Design of Group Decision Support Systems," *MIS Quarterly*, September 1984.

[Jardine 71] N.Jardine and C.J.van Rijsbergen, "The Use of Hierarchic Clustering in Information Retrieval," *Information Storage and Retrieval*, Vol.7, No.5, pp.217-240, 1971.

[Jeong 93] Y.M.Jeong, "*The Theory of Information Retrieval,*" Inc. kumi trade press, 1993.

[Leonard 93] Leonard M.Jessup and Joseph S.Valacich, "*Group Support Systems,*" Macmillan Publishing Company, 1993.

[Nunamaker 91] J.Nunamaker, A.Dennis, J.Valacich, D.Vogel and J.George, "Electronic Meeting

System to Support Group Work," *Communication of ACM*, Vol.34, No.7, pp.30-39, 1991.

[Radecki 79] T.Radecki, "Fuzzy Set Theoretical Approach to Document Retrieval," *Information Processing & Management*, Vol.15, No.5, pp.247-259, 1979.

[Raghavan 86] V.V.Raghavan and S.K.M.Wang, "A Critical Analysis of the Vector Space Model for Information Retrieval," *Journal of the American Society for Information Science (JASIS)*, Vol.37, No.5, pp.279-287, 1986.

[Salton and Wong 78] G.Salton and A.Wong, "Generation and Search of Clustered Files," ACM Transations on Database Systems, Vol.3, No.4, pp.321-346, 1978.

[Salton 83A] G.Salton, E.A.fox and H.Wu, "Extended Boolean Information Retrieval," *Communication of the ACM*, Vol.26, No.11, pp.1022-1036, 1983.

[Salton 83B] G.Salton and M.J.McGill, "*Intorduction to Modern Information Retrieval*," McGraw-Hill, 1983.

[Salton 87] G.Salton, "Historical Note: The Past Thirty Years in Information Retrieval," *Journal of the American Society for Information Science*, Vol.38, No.5, pp.375-380, 1987.

[Salton 89] G.Salton, "*Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*," Addison Wesley, 1989.

[Tahani 76] V.Tahani, "A Fuzzy Model of Document Retrieval Systems," *Information Processing & Management*, Vol.12, No.3, pp.177-187, 1976.

[Willett 81] P.Willett, "A Fast Procedure for the Calculation of Similarity Coefficients in Automatic Classification," *Information Processing & Management*, Vol.17, No.2, pp.53-60, 1981.

[Yu 85] C.T.Yu, C.M.Suen, K.Lam and M.K.Siu, "Adaptive Record Clustering," *ACM Transaction on Database Systems*, Vol.10, No.2, pp.180-204, 1985.