PERGAMON

Expert Systems with Applications 17 (1999) 105–113

Expert Systems with Applications

www.elsevier.com/locate/eswa

Intelligent query agent for structural document databases ☆

M.F. Jiang, S.S. Tseng*, C.J. Tsai

Department of Computer and Information Science, National Chiao Tung University, Hsinchu 300, Taiwan, ROC

Abstract

Querying a database for document retrieval is often a process close to querying an answering expert system. In this work, we apply the expert system techniques to the intelligent query agent establishment and regard the structural document database as the expertise which can be the objective of the knowledge acquisition. A new knowledge representation, named *Structural Documents* (SDs), is proposed to be the base of our model, and a transformation process from the raw data to the format of a database is applied. Based on the SDs, more suitable results could be inferenced rapidly by inference engine, and the flow of inference is also described. For implementation, an intelligent Chinese information retrieval system for personnel regulations by integrating knowledge-based and full-text searching techniques is proposed. In our experiments, the structural information of the documents can be acquired from the database using the knowledge extraction module. By observing the operating process of users, we found the query process of users are simplified. © 1999 Published by Elsevier Science Ltd. All rights reserved.

Keywords: Information retrieval; Document database; Intelligent agent; Expert system; Hierarchy clustering

1. Introduction

Nowadays, database systems are useful in business and industrial environments due to their multiple applications. A lot of database systems are built for storing documents, say document databases, and deserves more attention. Recently, due to the rapid growth of the Internet and hardware performance, research relating to digital library has become an important issue. A major portion in the digital library is the electronic books which have the advantage of saving a lot of retrieval time. There has been a recent trend to publish electronic books excepting hard copy. Especially in the professional field, the reference manuals are usually preserved as the document databases in order to increase the convenience to query.

If there is a database system, which can be modeled as shown in Fig. 1, users could formulate the query to retrieve documents, and reformulate the query when they see the results, and so on, until satisfied with the answer (Navarro & Baeza-Yates, 1997). The query effectiveness depends upon user's knowledge about the query language.

In order to improve the convenience of query for the traditional database system, we wanted to design a query agent which can be used to transform user's demand into a query format, coordinate with a user's request and revise the query interactively. For example, querying the personnel regulations for civil servant in Taiwan seems to be very sophisticated, and a lot of access time is required. Building a query agent is one of the solutions to such problems. Moreover, we hope that the agent provided an adaptability for different document databases. Therefore, we built an intelligent query agent (IQA), which is illustrated in Fig. 2, to assist users generating suitable queries and adjust the queries according to user's demand (Riecken, 1994).

Since querying a database for document retrieval is often a process close to querying an answering expert system (ES) (Celentano, Fugini, & Pozzi, 1995), in this work, we apply the ES techniques to the IQA establishment. In building IQA by the ES approach, we are concerned about the construction of the knowledge base, including the knowledge representation and the method of knowledge acquisition.

A document database consists of a large number of electronic books. In addition to the electronic form of content stored in the database, some structural information, i.e. the chapter/section/paragraph hierarchy, may also be embedded in the database. Classical information retrieval usually allows little structuring (Navarro & Baeza-Yates, 1997), since it retrieves information only on data. However, the structural information is useful in querying the document database, for instance, most people always read books with a chapter-oriented concept. In accordance with the

☆ This work was supported by the National Science Council of the Republic of China under Grant No. NSC88-2213-E-009-078.

* Corresponding author. Tel: + 886-3-5712121, ext. 56658; fax: + 886-3-5721490.

E-mail address: sstseng@cis.nctu.edu.tw (S.S. Tseng)

0957-4174/99/$ - see front matter © 1999 Published by Elsevier Science Ltd. All rights reserved.
PII: S0957-4174(99)00028-7

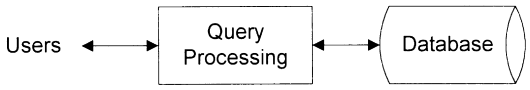

Fig. 1. The model of the database system.

document structure, including the index and the table of content, it can be regarded as the expertise and also the objective of the knowledge acquisition.

In order to elicit the knowledge embedded in the document structure, we first proposed a new knowledge representation, named *Structural Documents* (SD) (Jiang, Tseng, & Tsai, 1999), to be the basis of IQA. Second, our idea is to design a process of transforming the documents into a set of structural documents, which merge two documents with similarity greater than the given threshold into one structural document. Based on this idea, we developed a clustering-based approach to construct the SD in this work. Moreover, baed on the SD, more suitable results could be inferenced rapidly by the inference engine, and the flow of inference is also described. Besides, the architecture of the IQA and whole structural document database system is also proposed.

As we know, a sound legal system and complete regulations are usually of great importance for a government by law. Right now, the personnel regulations for civil servant in Taiwan seem to be very sophisticated. Although several kinds of reference books about personnel regulations have been provided for the general public to inquiry, they are not easy to use and a lot of access time is required.

Therefore, we design an intelligent Chinese information retrieval system for personnel regulations by integrating ES-based and full-text searching techniques. Our system may help users to retrieve the required personnel regulations. As mentioned above, a transformation process from the raw data to the format of a database is first applied. The embedded knowledge of the resulting data are then elicited by applying clustering techniques. By this way, the semantic indexes of the raw data can be established, and suitable results may be obtained. In our experiments, the structural information of the documents can be acquired from the database using the knowledge extraction module. By observing the operating process of users, we found that the query processes of users are simplified.

The organization of the rest of paper is described as follows. Reviews of related work are first described in Section 2. The knowledge extractor process we proposed is presented in Section 3. Section 4 presents the inference process of the IQA. Section 5 demonstrates the implementations of this approach. Finally, the conclusion and future work are discussed in Section 6.

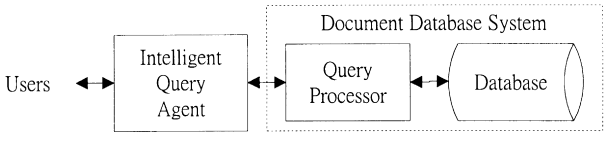

Fig. 2. An intelligent query agent for the document database system.

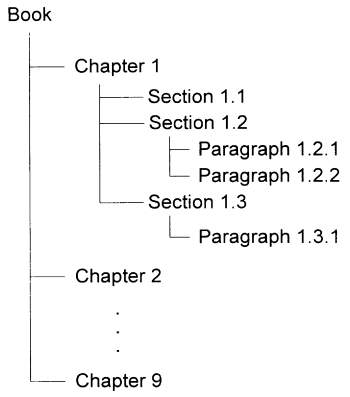

Fig. 3. The chapter structure of a book.

2. Related works

An ES is a knowledge-base system to solve problem at the level of a human expert (Giarratano & Riley, 1993), which generally consists of three modules: user interface, knowledge base and inference engine. User interface helps users to easily communicate with the ES, knowledge base contains the knowledge which forms the basis of inference and inference engine generates some conclusions according to the knowledge bases.

Generally, the process of building the knowledge base, which is called knowledge acquisition, is interviewing experts by knowledge engineers. However, it may cost a lot of time, as the domain expert may not have any sense of the computer techniques or the knowledge engineer is not familiar with the domain knowledge (Hwang & Tseng, 1990). In order to achieve the goal of decreasing the intervention of experts and the goal of smoothing the knowledge acquisition, we established an adaptive process to transfer domain knowledge to the format of a knowledge base.

The approach undertaken in IQA is based on the assumption that the structure of reference books is a hierarchy structure. Let us take the book's structure, shown in Fig. 3, as an example.

In Fig. 3, there are nine chapters, where Chapter 1 consists of three sections and some section may be composed of paragraphs. As most of the readers usually refer the books following the tree-like hierarchy, considering the book's structure seems to influence in query constructing. Intuitively, the documents in the same chapter seem to have very likely higher similarity rather than the documents in other chapters. Similarly, for the documents in the same chapter, the documents in the same section have very likely higher similarity rather than the documents in other sections. So, the information of the structural hierarchy may be useful in retrieving document databases.

Besides, the semantic meaning of the documents in the

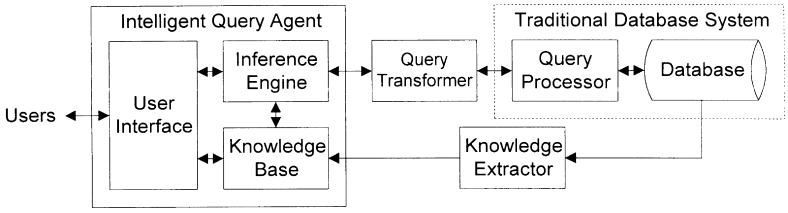

Fig. 4. Overview of the system.

book is expressed in the contents including words, phrases, etc. Therefore, we propose a new knowledge representation mixing contents and the structure of document database in building the IQA.

3. System structure

In this section, we will first introduce our system model and then the major modules of the system will be described in detail.

3.1. System overview

Fig. 4 shows the overview of the system, which consists of three components: IQA; query transformer (QT) module and knowledge extractor (KE) module; and a traditional database system. IQA and QT are used to transform the user's demand into an accessible query for the database system, and the KE module is used to extract the knowledge from the database and then store the extracted knowledge into the knowledge base of IQA.

Based on the knowledge stored in the format of SD, which is the new knowledge representation proposed in this work, the IQA module infers and revises the suitable query for user's demand, and the QT module transforms the result of IQA into an accessible query following the query syntax supported by the database system.

3.2. IQA and KE modules

The IQA module consists of user interface, inference engine and knowledge base. The user interface helps users communicate easily with the IQA using a web-based approach. The inference engine implies the results based on knowledge and the process is addressed in Section 4. Knowledge base stores the knowledge for inference, uses a new knowledge representation called SD, which is formally defined in Definition 3.1. As mentioned above, the knowledge is extracted from the database by the KE module. The flow of the knowledge extracting is described as follows as shown in Fig. 5. The content of the document database is first transformed into sets of words by partition and transformation. Besides, the index of the book are further considered as the knowledge source to identify the set of keywords for each document in the partition and transformation step. By computing each pair of documents, the similarity matrix for documents are obtained, and used as the basis for the subsequent clustering to find a similar pair of documents. After clustering, the hierarchy of structural documents can be obtained.

In this figure, the two kinds of existing resources, including the index and the table of contents for the books are used as the knowledge source in the KE module. All the three procedures in the KE module described in detail are as follows.

3.2.1. Partition and transformation

The KE module is capable of processing English or Chinese document database. As there is no obvious word boundary in the Chinese text, an identification process for identifying each possible disyllabic word from target database is needed. After the disyllabic words are identified from the sentence by applying the association measure, each document can be transferred to a set of keywords. That is, for two different Chinese characters, c_i and c_j, the association (Liang, 1995; Sproat and Shih, 1990) of the two

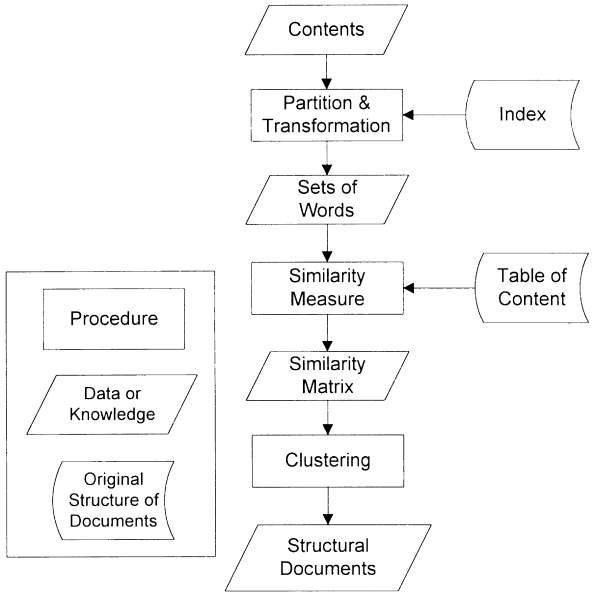

Fig. 5. Flowchart of knowledge extracting.

characters can be computed by:

$$\log_2 \frac{\frac{freq(C_iC_j)}{n}}{\frac{freq(C_i)}{n}\frac{freq(C_j)}{n}},$$

where the value of the function *freq*( ) is the occurrence frequency of any character or two successive characters and *n* is the number of all the characters in the target database. Since the association formula is originally designed for a large corpus, the obtained associations may not good enough to identify words. Therefore, further manual turning may be required.

After the disyllabic words are identified from the sentence by applying the association measure, each document can be transferred to a set of words. For example, the sentence "工 友 之 管 理 考 核,待 遇 應 由 何 單 位 辦 理" may be segmented into '工友', '管理', '考核', '待遇', '應由', '單位', and '辦理'.. It should be noted here that there is no need to identify words in English text, and so the above segmented method should be skipped for English text.

In our approach, the indexes of the book are also used to identify the set of keywords for each document. After the execution of the above two steps, each document is transformed into a set of keywords and words. Therefore, the size of both sets varies depending on the document itself, because the words are identified by the association measure and the keywords are identified by comparing with the index of the reference books.

3.2.2. Similarity measure

To measure the similarity between two documents, we use the following heuristics. The similarity between two documents in the same chapter is higher than that in two different chapters, and the similarity between two documents in the same section is higher than that in two different sections. Without losing the generality, we assumed the whole reference book to be divided into a three-tier hierarchy, including chapter, section and paragraph. Based upon these heuristics, the *Hierarchy Dependence* (HD) between two documents can be easily computed by the following procedure:

Step 1: If two documents are not in the same chapter, HD ← 0 and stop.
Step 2: If two documents are not in the same section HD ← (1/*s*), where *s* is the total number of sections in this chapter, and stop.
Step 3: If two documents are not in the same paragraph, HD ← 0.5 + (1/*p*) , where *p* is the total number of paragraphs in this section, and stop.
Step 4. HD ← 1

.

Let the two documents be denoted as D_i and D_j. In the KE module, the similarity of D_i and D_j, denoted by $S(i,j)$, is computed by the following formula:

$$S(i,j) = (1 - \delta) * same(i,j) + \delta * \mathrm{HD}(i,j),$$

where $same(i,j)$ means the number of words and keywords which appear both in documents D_i and D_j. The value is normalized by dividing the total number of keywords in documents D_i and D_j, $\mathrm{HD}(i,j)$ means the hierarchy dependence of documents D_i and D_j and δ is an adaptive weight value with $0 \le \delta \le 1$.

The default value of δ is 0.5, which can be adjusted by the number of chapters for a given book. For example, when the number of chapters is near to 1 or *n* for a book divided into *n* documents, we set δ as the value closed to 0, as there is not much meaning in the structure.

Example 3.1. Let D_i = {'管理', '考核', '待遇', '單位'}, $|D_i| = 4$, and D_j = {'管理', '考核', '機構'}, $|D_j| = 3$. Therefore, the number of the words which appear both in documents D_i and D_j is 2, and we have $same(i,j) = |D_i \cap D_j| / (|D_i| + |D_j|) = 2/(4 + 3)$.

Example 3.2. Let D_i = {'intelligent', 'query', 'agent'}, $|D_i| = 3$, and D_j = {'database', 'query',}, $|D_j| = 2$. Therefore, the number of the words which appear both in documents D_i and D_j is 1, and we have $same(i,j) = 0.2$.

By computing the similarity measure of all two different documents D_i and D_j, a similarity matrix $[S_{ij}]$ can be formed by letting $S_{ij} = S(i,j)$. To simplify our further discussion, we assign $S_{ij} = 0$ for $i \ge j$. The matrix becomes an upper triangular matrix and the diagonal elements are 0's.

3.2.3. Clustering and structuring documents

Clustering is an important step in the KE module for the purpose of structuring documents. To transfer the documents into a set of structural documents, the algorithm for similarity matrix updating in hierarchy clustering (Jain & Dubes, 1988) is used. In hierarchy clustering, a lot of approaches are considered in measuring similarity of clusters, including the single-link and the complete-link methods. It seems that the clustered hierarchy generated by the complete-link method is more balanced than the one generated by the single-link method. In this work, the complete-link method is implemented. To represent the knowledge in clustering process, the definition and notation of SD are formally defined.

Definition 3.1. The *SD with level l* is defined recursively as follows:

- A document D_i is a *SD with level* 0, denoted as SD_i which also can be denoted as $({}_0D_i)_0$.
- A pair of SD, denoted as $({}_lSD_i, SD_j)_l$, with the greatest

similarity measure, which is greater than a threshold θ, among all the different pairs is also a *SD with level l*, where $l = \text{Maximum}(m, n) + 1$, m is the level number of SD_i and n is the level number of SD_j.

Definition 3.2. The *similarity* S' between the structural documents SD_i and SD_j is defined as follows:

1. If $\text{SD}_i = (D_i)$ and $\text{SD}_j = (D_j)$, the similarity $S'(\text{SD}_i, \text{SD}_j) = S(i, j)$.
2. The similarity between $(\text{SD}_i, \text{SD}_j)$ and SD_k is defined as $S'(\text{SD}_k, (\text{SD}_i, \text{SD}_j)) = \Theta\{S'(\text{SD}_k, \text{SD}_i), S'(\text{SD}_k, \text{SD}_j)\}$, where the operator Θ means 'minimum' for the complete-link method, or 'maximum' for the single-link method.

Assume we have n documents, and then an $n \times n$ similarity matrix can be generated by our method. First, each document is assigned to be a SD, i.e. $\text{SD}_i = (D_i)$ for document D_i. Moreover, the similarity matrix for documents is transferred to the initial similarity matrix for SDs. The elements of the similarity matrix $[S'_{ij}]$ is the similarity measure of the structural documents SD_i and SD_j, i.e. $S'_{ij} = S_{ij}$. Let $\Psi = \{\text{SD}_i\}$ be the set of all SD_i. Based upon the Johnson's algorithms (1967) for hierarchy clustering (Jain & Dubes, 1988), we propose the following procedure for updating similarity matrix.

Algorithm 3.1.

Step 1: Find the most similar pair of structural documents in the current similarity matrix, say pair $\{p, q\}$, where, $S'_{p,q} = \text{Maximum}\{S'_{i,j}$, for any $i, j\}$.
Step 2: Merge structural documents SD_p and SD_q into a new single structural document (SD_p, SD_q).
Step 3: Delete the structural documents SD_p and SD_q from the set Ψ, and insert the new structural document $(\text{SD}_p, \text{SD}_q)$ into the set, i.e. $\Psi \leftarrow \Psi - \{SD_p\} - \{SD_q\} + \{({}_lSD_p, SD_q)_l\}$, where $l = \text{Maximum}(m, n) + 1$, m is the level number of SD_p and n is the level number of SD_q.
Step 4: Update the similarity matrix by deleting the rows and columns related to structural documents SD_p and SD_q, and adding a row and a column corresponding to the new structural document $(\text{SD}_p, \text{SD}_q)$.
Step 5: If there are no two structural documents with similarity greater than θ, stop. Otherwise, go to Step 1.

After clustering, the hierarchy of structural documents can be obtained according to the set Ψ.

3.3. An example

Let $\theta = 0.1$, assume we have six documents, $\{D_1, D_2, D_3, D_4, D_5, D_6\}$, and the similarity matrix is:

	D_1	D_2	D_3	D_4	D_5	D_6
D_1	0	0.8	0.4	0.5	0.4	0.1
D_2		0	0.5	0.6	0.3	0.2
D_3			0	0.7	0.8	0.1
D_4				0	0.9	0.3
D_5					0	0.2
D_6						0

Before clustering, each document is assigned to be a structural document, i.e. $\text{SD}_i = ({}_0D_i)_0$ for document D_i. The set of all SD_i is written as $\Psi = \{\text{SD}_1, \text{SD}_2, \text{SD}_3, \text{SD}_4, \text{SD}_5, \text{SD}_6\}$, and the initial similarity matrix for SDs is generated as:

	SD_1	SD_2	SD_3	SD_4	SD_5	SD_6
SD_1	0	0.8	0.4	0.5	0.4	0.1
SD_2		0	0.5	0.6	0.3	0.2
SD_3			0	0.7	0.8	0.1
SD_4				0	0.9	0.3
SD_5					0	0.2
SD_6						0

In the first iteration, the structural documents, SD_4 and SD_5, are merged into $(\text{SD}_4, \text{SD}_5)$, as the elementary value of the 4th row and the 5th column is the maximum. The set of all SD_i is written as $\Psi = \{\text{SD}_1, \text{SD}_2, \text{SD}_3, ({}_1\text{SD}_4, \text{SD}_5)_1, \text{SD}_6\}$. After the similarity matrix is updated, we get a new similarity matrix after Step 4:

	1	2	3	4, 5	6
1	0	0.8	0.4	0.4	0.1
2		0	0.5	0.3	0.2
3			0	0.7	0.1
4, 5				0	0.2
6					0

In the second iteration, we merge the structural documents SD_1 and SD_2 into $({}_1\text{SD}_1, \text{SD}_2)_1$, as the elementary value of the 1st row and 2nd column is the maximum. We have the new $\Psi = \{({}_1\text{SD}_1, \text{SD}_2)_1, \text{SD}_3, ({}_1\text{SD}_4, \text{SD}_5)_1, \text{SD}_6\}$. After the similarity matrix is updated, the new similarity

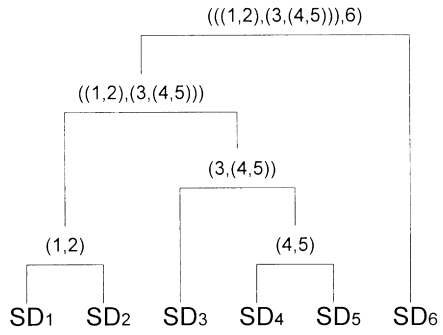

Fig. 6. The hierarchy of structural documents according to the clustering result.

matrix is transformed to:

$$\begin{array}{c} \\ 1,2 \\ 3 \\ 4,5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccc} 1,2 & 3 & 4,5 & 6 \end{array} \\ \begin{bmatrix} 0 & 0.4 & 0.3 & 0.1 \\ & 0 & 0.7 & 0.1 \\ & & 0 & 0.2 \\ & & & 0 \end{bmatrix} \end{array}$$

Similarly, we merge the structural documents SD_3 and $({}_1SD_4, SD_5)_1$ into $({}_2SD_3, ({}_1SD_4, SD_5)_1)_2$ in the third iteration, since the elementary value of the 2nd row and 3rd column is the maximum. We have the new $\Psi = \{({}_1SD_1, SD_2)_1, ({}_2SD_3, ({}_1SD_4, SD_5)_1)_2, SD_6\}$ and the new similarity matrix is transformed to:

$$\begin{array}{c} \\ 1,2 \\ 3,4,5 \\ 6 \end{array} \begin{array}{c} \begin{array}{ccc} 1,2 & 3,4,5 & 6 \end{array} \\ \begin{bmatrix} 0 & 0.3 & 0.1 \\ & 0 & 0.1 \\ & & 0 \end{bmatrix} \end{array}$$

In the last two iterations, we have the new $\Psi = \{({}_3({}_1SD_1, SD_2)_1, ({}_2SD_3, ({}_1SD_4, SD_5)_1)_2)_3, SD_6\}$ and $\{({}_4({}_3({}_1SD_1, SD_2)_1, ({}_2SD_3, ({}_1SD_4, SD_5)_1)_2)_3, SD_6)_4\}$.

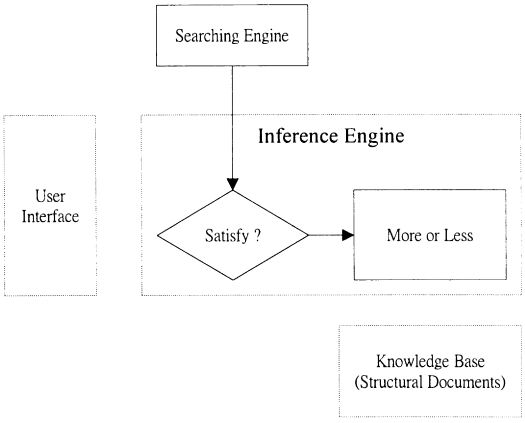

Fig. 7. The flowchart of the inference process.

According to the set Ψ, the hierarchy of SDs can be obtained as Fig. 6.

4. Inference process based on the structural documents

After the knowledge extracting process, in the previous section, the SD Ψ have been built for the retrieving documents. Based on the SD, more suitable results could be rapidly inferenced by an inference engine. The flow of the inference process is described in Fig. 7.

The query result R is generated by some searching engine. However, in most cases, the result may be not satisfied very well for the user's demand, likely more or less. Two algorithms are designed to solve the problem. In this section, the detail of inference process will be illustrated. First, we state the notations and definitions for the following algorithms.

To easily represent the set of documents, a data structure *bit_map* is defined as:

$$bit\_map = b_1 b_2 \cdots b_n,$$

where $b_k = 1$, if document D_k belongs to some set of documents, and $b_k = 0$, otherwise.

Based on the data structure *bit_map*, for a given query result, a set of documents generated by some searching engine can be written as:

$$R = b_1 b_2 \cdots b_n,$$

such that $b_k = 1$, if document D_k belongs to the query result, and $b_k = 0$, otherwise.

In the format of SDs, a pair of structural documents (SD_i, SD_j) means the similarity between SD_i and SD_j is greater than the other SDs. For any structural document SD_j, the most similar structural document SD_k can be found when SD_i, a pair of structural documents (SD_j, SD_k), has been found. To easily describe the above idea, the definitions of *immediate successor* and *successor* are illustrated as following.

Definition 4.1. A structural document SD_i is said to be an *immediate successor* of structural documents SD_j, if $SD_i = ({}_lSD_j, SD_k)_l$ or $SD_i = ({}_lSD_k, SD_j)_l$ for some structural document SD_k, and level l.

Definition 4.2. A structural document SD_i is said to be a *successor* of a structural document SD_j, if there exists an immediate successor SD_k of SD_j, such that SD_i is also a successor of SD_k or SD_i is an immediate successor of SD_j.

The key process, that is to find the *immediate successor*, is to directly scan the structural document Ψ, and described in the following algorithm.

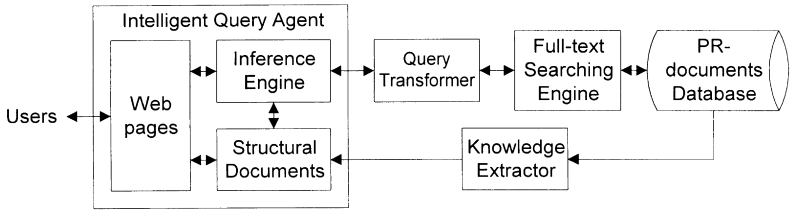

Fig. 8. The architecture of CPRCS.

Algorithm 4.1 ((Find_immediate_successor)).

Input: Ψ, SD_i
Output: The immediate successor of SD_i
Step 1: Find SD_i in the set Ψ.
Step 2: Check the next element of SD_i with two cases possibly existing.

Case 1: The next element of SD_i is ",",
Find the structural document SD_j, for some $j > 0$, in the next element of "SD_i," and return $({}_k\ SD_i, SD_j)_k$, for $k = MAX(i, j) + 1$.
Case 2: The next element of SD_i is "$)_l$" for some l,
Find "$({}_l$" in the preceding element of SD_i and return $({}_l SD, SD_i)_l$, for some structural document SD_j.

In the Step 2 of Algorithm 4.1, there are two cases possibly existing. In Case 1, output $= (SD_i, SD_j)$ for some $SD_j \in \Psi$ and in Case 2, output $= (SD_j, SD_i)$ for some $SD_j \in \Psi$. By Definitions 3.1 and 4.1, it can be easily seen that output is the immediate successor of SD_i.

Now, a function *bit_set* is defined to convert the structural document SD_i into a string of bits, such that the kth bit is 1, if the document D_k belongs to SD_i, and is 0, otherwise.

Example 4.1. Assume the amount of all documents is 5, $bit\_set((((D_1), (D_2)), (D_3))) =$ "11100".

The ith cover for all n documents is defined, following the definition of *bit_map*:

$$C_i = b_1 b_2 \cdots b_n$$

Algorithm 4.2.

Input: Any structural document SD_i, number k
Output: $C_k, C_{k+1}, \ldots, C_{k+m}$, where the number m is the minimum integer such that $R \subseteq C_{k+m}$
Procedure $Cover(k, SD_i)$

$SD_j \leftarrow$ Find_immediate_successor (Ψ, SD_i)
$C_k \leftarrow bit\_set(SD_j)$
If NOT_EQUAL(R, AND (R, C_k)) Then Call $Cover(k + 1, SD_j)$

End_Procedure

Example 4.2. Assume we have six documents, and the hierarchy of SDs are shown in Fig. 6 and $\Psi = \{(((SD_1, SD_2), (SD_3, (SD_4, SD_5))), SD_6)\}$. Now, for a user's query, database return the result, $\{D_2, D_3, D_4\}$ and $R = 011100$. When $Cover(2, (D_3))$ is called, we get $C_2 = 001110$ and $C_3 = 111110$.

Algorithm 4.3. / * The users want to get a set of documents which can be deleted from the query result. * /

Input: The query result R, Ψ
Output: The set of documents which can be deleted from the query result.
Procedure Less

Select a D_i according to R

$SD_j \leftarrow (D_i)$
$C_1 \leftarrow \{D_i\}$

Call $Cover(2, SD_j)$
Case COUNT_ONE(AND(R, C_{k-1}))-
COUNT_ONE(AND(R, SUB(C_k, C_{k-1})))

< 0: RETURN AND(R, C_{k-1})
> 0: RETURN AND(R, SUB(C_k, C_{k-1}))
$= 0$: RETURN AND(R, C_{k-1}) or
AND(R, SUB(C_k, C_{k-1})) determined by user

End_Case

End_Procedure

Fig. 9. The main window of the CPRCS.

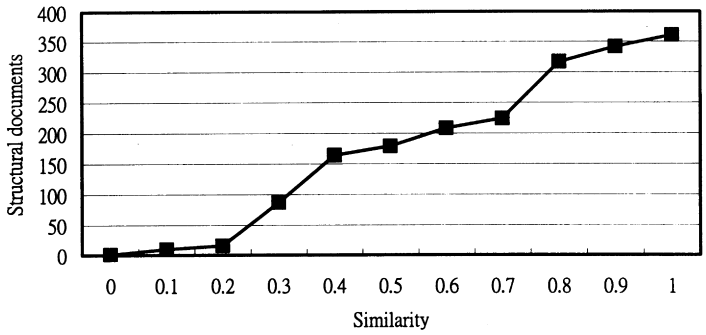

Fig. 10. The amount of structural documents of similarity. For instance, the amount of structural documents is 224, when the similarity between any two documents in the same SD is greater than 0.7.

Algorithm 4.4. / * The users want to get a set of documents, which could offer more information. * /

Input: The query result R, Ψ
Output: A set of documents, which could offer more information.
Procedure More

Select a D_i according to R

$\mathrm{SD}_j \leftarrow (D_i)$
$C_1 \leftarrow \{D_i\}$

Call *Cover*$(2, \mathrm{SD}_j)$
While COUNT_ONE(AND$(R, C_k)) \neq 0$ do

Case COUNT_ONE(AND(R, C_{k-1}))-
COUNT_ONE(AND$(R, \mathrm{SUB}(C_k, C_{k-1}))$)
< 0: $R \leftarrow$ AND$(R, \mathrm{SUB}(C_k, C_{k-1}))$ and
Call More
> 0: $R \leftarrow$ SUB$(R,$ AND$(R, \mathrm{SUB}(C_k, C_{k-1})))$,
$k \leftarrow k - 1$
$= 0$: RETURN AND(R, C_{k-1}) or
AND$(R, \mathrm{SUB}(C_k, C_{k-1}))$ determined by
user and stop.

End_Case
RETURN C_k

End_While

End_Procedure

Example 4.3. Assume we have six documents, and the hierarchy of structural documents are shown in Fig. 6 and $\Psi = \{(((\mathrm{SD}_1, \mathrm{SD}_2), (\mathrm{SD}_3, (\mathrm{SD}_4, \mathrm{SD}_5))), \mathrm{SD}_6)\}$. Now, for a user's query, database return the result, $\{D_3, D_4, D_6\}$. For the result, we have the corresponding structural documents $\{\mathrm{SD}_3, \mathrm{SD}_4, \mathrm{SD}_6\}$ generated by the IQA. When the user feels insufficiency about the result, the IQA may generate the set $\{\mathrm{SD}_3, \mathrm{SD}_4, \boxed{\mathrm{SD}_5}, \mathrm{SD}_6\}$ and asks database to return document $\{D_5\}$. When the user feels insufficiency again, the IQA may generate the set $\{\boxed{\mathrm{SD}_1}, \boxed{\mathrm{SD}_2}, \mathrm{SD}_3, \mathrm{SD}_4, \mathrm{SD}_5, \mathrm{SD}_6\}$ and the documents $\{D_1, D_2\}$ are returned.

On the other hand, if the user feels the amount of result $\{D_3, D_4, D_6\}$ is too many, IQA generate the set $\{\mathrm{SD}_3, \mathrm{SD}_4\}$ and deletes the document $\{D_6\}$ from the query result. If the user feels the amount of result is too many again, the IQA generates the set $\{\mathrm{SD}_4\}$, and delete the document $\{D_3\}$ from the query result.

Example 4.4. For another user's query, database return the result, $\{D_3, D_4, D_5\}$. The user feels the amount of result is too many, IQA generates the set $\{\mathrm{SD}_4, \mathrm{SD}_5\}$ and deletes the document $\{D_3\}$ from the query result.

Example 4.5. For another user's query, database return the result, $\{D_2, D_3, D_4\}$. The user feel the amount of result is too many, IQA generates the set $\{\mathrm{SD}_3, \mathrm{SD}_4\}$ and deletes the document $\{D_2\}$ from the query result.

5. Implementation

As we know, a sound legal system and complete regulations are usually of great importance for a government by law. Right now, the PR for civil servant in Taiwan seem to be very sophisticated. Although several kinds of reference books about personnel regulations have been provided for the general public to inquiry, they are not easy to use and a lot of access time is required. Therefore, improvement of the methods of inquiry and annotation of the personnel regulations has become an important issue. Besides, by comparing the experimental results between the traditional database model and our new model, we may verify the performance of the model based on SDs for the PR document databases. We implemented a database prototype, named CPRCS (Chinese Personnel Regulations Consultation System), for PR documents and used the IQA to assist the querying process. The CPRCS architecture is followed by the system structure of Fig. 4 and shown in Fig. 8.

In CPRCS, users operate the system through the web pages interface, which can be easily promoted to be

extended. Two different modes, with or without IQA, are provided for users as they please. By observing the operating process of users, we found the query process of users are simplified. Fig. 9 is the main window of the system.

Furthermore, the relation between the chapter hierarchy and the amount of the SDs is explained in the following experiment. There are 365 documents in the target database for the experiment. All the documents are divided into 11 chapters and the total amount of sections is 171. After the processing of the KE module, the amount of SDs of different similarity is shown in Fig. 10.

The figure shows the merging situation in the clustering process. The results observed from the figure are discussed in the following way. When the similarity value is between 0 and 0.2, the amount of structural documents is about 11, which is the amount of chapters. Similarly, when the similarity value is between 0.4 and 0.7, the amount of structural documents is about 171, which is the total amount of sections. The situation says that the hierarchy of structural documents is similar to the chapter/section hierarchy for the book. The structural information of the documents can be acquired from the database using the KE module.

6. Conclusion

Classical information retrieval usually allows little structuring. However, the structural information is useful in querying the document database, for instance, most people always read books with chapter-oriented concept. In accordance with the document structure, including the index and the table of content, it can be regarded as the expertise and also can be the objective of the knowledge acquisition. Since querying a database for document retrieval is often a process close to querying an answering expert system, in this work, we apply the ES techniques to the IQA establishment. In building IQA by the ES approach, we are concerned about the construction of the knowledge base, including the knowledge representation and the method of knowledge acquisition. Therefore, a new knowledge representation, named SDs, is defined to construct the acquisition model for IQA, and proposed the KE model to transform the data of database into the knowledge storing in IQA. For comparing the convenience of IQA, an intelligent retrieval system, CPRCS, is implemented. By observing the operating process of users, we found the that query processes of users are simplified. Besides, the experimental result has shown the structural information of the documents can be acquired from the database using the KE module.

Future research will focus on several areas. First, better similarity measurements are necessary for increasing the performance of clustering. The analysis for the influence of different clustering methods is not covered in this work. The general formula proposed by Jain and Dubes (1988) includes most of the commonly hierarchical clustering method which is the basis for future work. Another significant focus on this work is to extend the model based on SD, and the goal is to allow any different kinds of documents that can be merged into the same knowledge structure in order to increase the practicability of the system.

References

Celentano, A., Fugini, M. G., & Pozzi, S. (1995). Knowledge-based document retrieval in office environments: the Kabiria system. *ACM Transactions on Information System, 13* (3), 237–268.

Giarratano, J., & Riley, G. (1993). *Expert systems*, 2. PWS Publishing Company.

Hwang, G. J., & Tseng, S. S. (1990). EMCUD: A knowledge acquisition method which captures embedded meanings under uncertainty. *International Journal of Man Machine Studies, 33*, 431–451.

Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data*, Englewood Cliffs, NJ: Prentice Hall pp. 58–86.

Jiang, M.F., Tseng, S.S., Tsai, C.J. (1999). Discovering structure from document databases, *The Third Pacific-Asia Conference on Knowledge Discovery and Data Mining*, PAKDD-99, Beijing, China.

Liang, T. (1995). The Study of Character-based Signature Methods in Chinese Text Retrieval, PhD thesis, National Chiao Tung University, Taiwan.

Navarro, G., & Baeza-Yates, R. (1997). Proximal nodes: a model to query document database by content and structure. *ACM Transactions on Information Systems, 15* (4), 400–435.

Riecken, D. (1994). Intelligent agent. *Communications of ACM, 37* (7), 18–21.

Sproat, R., & Shih, C. (1990). A statistical method for finding word boundaries in Chinese text. *Computer Proceedings of Chinese and Oriental Languages, 4* (4), 336–351.