http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 7 (5): 746-753, 2008
ISSN 1812-5638

Clustering Search Results Based on Formal Concept Analysis

Y. Zhang and B. Feng
School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an, 710049, China

Abstract: This study propose a new method based on Formal Concept Analysis (FCA) to group and organize search results. After formal concepts are extracted using FCA, the concepts most relevant to the query are selected and extracted and then a two-level hierarchy is built and presented to the user. We refer the proposed algorithm to as CHC (Conceptual and Hierarchical Clustering). Evaluating the quality of the clustering results is a non-trivial task. Two improved objective metrics of clustering quality, ANCE@K and ANCE@K, are proposed based on NMI (normalized mutual information) and NCE (normalized complementary entropy) metrics but eliminating the biases existed in them. We compare CHC with three other Search Results Clustering (SRC) algorithms: Suffix Tree Clustering (STC), Lingo and Vivisimo, using a comprehensive set of documents obtained from the Open Directory Project hierarchy as benchmark. In addition to comparison based on objective measures, we also subjectively analyze the properties of cluster labels produced by different SRC algorithms. The experimental results show that our method outperforms the other three SRC algorithms and is helpful to the user for browsing and locating the results of interests.

Key words: Search results clustering, formal concept analysis, concept lattice, conceptual and hierarchical clustering, clustering evaluation

INTRODUCTION

With an enormous growth of Internet, it has become more and more difficult for users to find relevant documents. In response to the user's query, currently available search engines return a ranked list of documents along with their partial content (snippets). However, the ranked list paradigm is highly inefficient since the number of retrieved search results can be in the thousands for a typical query. Most users just view the top results displayed in the first page and therefore might miss relevant information. Moreover, the criteria used for ranking may not reflect the needs of the user. A majority of the queries tend to be short, thus making them non-specific or imprecise. The inherent ambiguity in interpreting a word or a phrase in the absence of its context means that a large percentage of the returned results can be irrelevant to the user.

One of the alternative approaches is to automatically group similar document together in order to facilitate presentation of results in a more compact form and enable thematic browsing of the results set. Hearst and Pedersen (1996) showed that relevant documents tend to be more similar to each other, thus the clustering of similar search results helps users find relevant results. In fact, Search Results Clustering (SRC) algorithms have been studied quite deeply in Information Retrieval (Zamir and Etzioni, 1999; Zhang and Dong, 2004; Kummamuru *et al*., 2004; Osinski *et al*., 2004). Vivisimo (www.vivisimo.com) is an example of a successful commercial application of the clustering idea.

This study propose a new method for search results clustering using Formal Concept Analysis (FCA). FCA is a mathematical approach to data analysis based on Lattice Theory, which was invented by Rudolf and Wille (1982). As a conceptual clustering technique, FCA has some advantages over standard document clustering algorithms for clustering web search results: it provides an intrinsic description of each cluster, which makes clusters more interpretable; intent and extent of formal concept are uniform which insures grouping similar documents together to generate coherent clusters; concept lattice reflects the relation of all of the concepts which provides a richer and more flexible way to browse the document space.

In this method, FCA is used to model the document space returned by a certain search engine and a concept lattice is generated. Based on the generic/specific relations among concepts uncovered by the generated lattice, a heuristic algorithm is proposed to select the concepts which most relevant to the user query and then a hierarchical structure of search results is constructed and presented to the user. We refer to the proposed algorithm as CHC (Conceptual and Hierarchical Clustering).

Corresponding Author: Y. Zhang, School of Electronics and Information Engineering, Xi'an Jiaotong University, Xi'an, 710049, China

We have tested the clustering effectiveness of CHC against other there SRC algorithms --- STC (Zamir and Etzioni, 1999), Lingo (Osinski *et al*., 2004) and Vivisimo --- using a set of snippets obtained from the Open Directory Project hierarchy as benchmark. We also propose two improved metrics of clustering quality, ANCE@K and ANCE@K, based on two commonly used metrics, NMI (normalized mutual information) and NCE (normalized complementary entropy), but eliminating the biases existed in them. Experimental results show that this method outperforms the other three SRC algorithms. In addition to comparison based on objective measures, we also subjectively analyze the properties of cluster labels produced by different SRC algorithms.

The idea of search results clustering was first introduced in the Scatter/Gather system (Hearst and Pedersen, 1996), which was based on a variant of the classic K-Means algorithm. Scatter/Gather was followed by Grouper (Zamir and Etzioni, 1999), which was the first publicly available software to address the SRC problem. The main feature of Grouper is the introduction of a phrase-analysis algorithm called Suffix Tree Clustering (STC), in which snippets sharing the same sequence of words were grouped together. The Semantic On-Line Hierarchical Clustering (SHOC) (Zhang and Dong, 2004) algorithm used Singular Value Decomposition to group search results in oriental languages like Chinese according to the latent semantic relationships between the snippets. Yet another algorithm called DisCover (Kummamuru *et al*., 2004) clustered search results in such a way as to maximize the coverage and distinctiveness of the clusters. With their respective advantages such as speed and scalability, all these algorithms share one significant shortcoming: none of them explicitly addresses the problem of cluster description quality. This, unfortunately, leads these algorithms to knowing that certain documents should form a group and at the same time being unable to concisely explain to the user what the group's documents has in common.

Descriptive clustering was put forward recently, which focus on providing good, descriptive clusters labels for groups of semantically related documents. The representative algorithm is Lingo (Osinski *et al*., 2004). The main idea behind Lingo was to reverse the usual order of the clustering process: Lingo first identified meaningful cluster labels using the Singular Value Decomposition (SVD) factorization and only then assigned documents to these labels to form proper clusters.

Different from the research, FCA is a conceptual clustering technique, which generates clusters and cluster labels simultaneously. For its natural advantages on search results clustering, some systems applying FCA have been presented such as Credo (Carpineto and Romano, 2004) and JBraindead (Juan *et al*., 2004). However, both of them applied FCA to organize the search results and presented the resulting lattice to the user directly. Unlike them, the method proposed in this study is further exploring the properties of the lattice and mining the concepts most relevant to the user query to construct a two-level hierarchical structure of search results, which is presented to the user in the end.

APPLY FCA TO WEB SEARCH RESULTS

In this study, we intend to analyze the resulting documents of ranked items returned by search engines based on FCA. The basic idea of FCA model is to use formal context of objects and attributes to describe formal concept of a domain. Here, we will explore formal context between documents in the resulting documents firstly and then extract formal concepts and construct the concept lattice.

Textual document formalization: In order to construct the formal context, documents are preprocessed and normalized firstly. As we know, a ranked list of documents is returned by a particular search engine for a given query. In our system, the top N documents (generally, N = 200 if the number of documents retrieved is greater than 200) in the ranked list are examined to extract, from the terms in the documents, a set of k optimal descriptors to construct the formal context.

The construction of formal context of documents and terms is similar to the process of representation of documents in an IR system. To automatically extract a set of index terms describing each document, the following steps can be followed:

- **Text segmentation:** The individual words occurring in a text collection are extracted, ignoring punctuation and case.
- **Word stemming:** Stemming reduces the amount of indexable terms. Each word is reduced to word-stem form. We used the Porter (1980) stemming algorithm to stem the words in our system.
- **Stopword elimination:** A stop list (Fox, 1990) is used to delete from the texts the words that are insufficiently specific to represent content.
- **Word weighting:** This step is necessary to perform word selection, described in step (5); we employed the classical tf*idf weighting method in our experiment.
- **Word selection:** This last step is not necessary for IR systems performing full-text indexing but is very important for FCA-based systems to facilitate the subsequent process of lattice construction. Generally, when the weight of a term exceeds a pre-defined threshold, the term can be used as an index term.

Table 1: The formal context

C	Jaguar	Car	Vehicle	Model	Sports	Animal	Leopard
d_1	x	x					
d_2	x	x		x			
d_3		x			x		
d_4	x					x	
d_5	x		x				
d_6	x	x					
d_7			x	x			
d_8	x					x	x
d_9	x	x			x		

After extracting the optimal k descriptors from the documents, we will get the formal context between the documents and terms, the definition is (we denote the N retrieved documents as D and the k optimal keywords as T):

Definition 1: A (formal) context of documents and terms in D is a triple (D, T, R) where $D = \{d_1, d_2, \ldots, d_N\}$ is set of retrieved documents, $T = \{t_1, t_2, \ldots, t_n\}$ is a set of index terms in D and R is the incident relation between D and T where $R \subseteq D \times T$.

Table 1 shows a sample context table of the formal context (D, T, R) for query jaguar.

Construction of concept lattice: After constructing the formal context the concepts and concept lattice can be extracted using FCA. We define the formal concept of the context (D, T, R) as follows:

Definition 2: A pair (A, B) is a (formal) concept of (D, T, R) if and only if

$$A \subseteq D, B \subseteq T, A' = B \wedge B' = A$$

Where:

$$A' = \{t \in T \mid (d, t) \in R \text{ for all } d \in D\}$$

$$B' = \{d \in D \mid (d, t) \in R \text{ for all } t \in T\}$$

In a formal concept, the set of documents A is referred to as the extent of the concept (A, B) and the set of terms B corresponding to those documents is referred to as the intent of the concept (A, B). For example, ($\{d_1, d_2, d_9, d_9\}$, {Jaguar, car}) is a concept of the context shown in (Table 1), $\{d_1, d_2, d_6, d_9\}$ are the concept's extent and {Jaguar, car} are the concept's intent. ($\{d_9\}$, {Jaguar, car, sports}) is another concept of this context.

A partial ordering can be defined over the concepts of a context. Specifically,

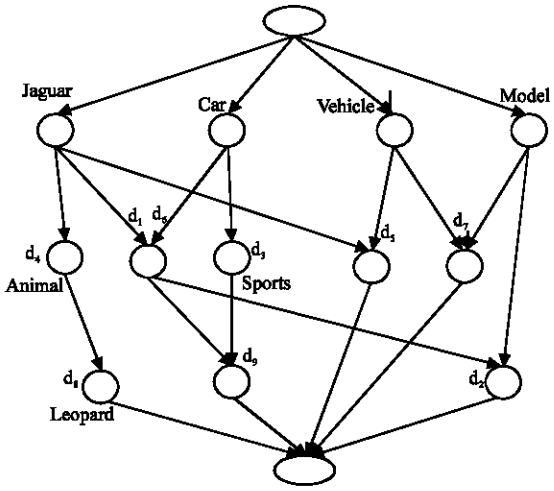

Fig. 1: The concept lattice

$$(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2$$

Equivalently,

$$(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow B_1 \supseteq B_2$$

In the example, ($\{d_9\}$, {jaguar, car, sports}) $\leq$ ($\{d_1, d_2, d_6, d_9\}$, {jaguar, car}). This can be thought of as a subconcept-superconcept relation. Thus, formal concepts are partially ordered with regard to inclusion of their extents or (which is equivalent) to inverse inclusion of their intent. The set of all formal concepts of a given context with the subconcept-superconcept relation ($\leq$) is always a complete lattice, called the concept lattice. Figure 1 is a Hasse diagram depicting the concept lattice for the context of Table 1. In the lattice diagram, each document is described by exactly those terms that are attached to nodes that are above the document node. Each term belongs to exactly those documents that are attached to nodes below the term node.

In the applications of FCA, building concept lattice efficiently is an important task, for which various algorithms (Lindig, 2000; Godin *et al.*, 1995; Carpineto and Romano, 1996) have been developed. Here we have chosen to use (Godin *et al.*, 1995) incremental algorithm due its high efficiency in generating concept lattice. We implemented Godin's algorithm in Java for this study.

The lattice helps us to visualize the relationships between concepts, objects (documents) and/or attributes (terms), which has many advantages over simple list technology. For example, users can easily get an overview of the retrieved document set and locate the interested documents. However, forming useful visualizations of graph structures is notoriously difficult due to the

conflicting issues of size, layout and legibility on limited screen area. The problem is further compounded by the fact that the concept lattices of real applications are usually very large and hence difficult to be used for practical browsing purposes.

By examining the generated lattice, it was found that the concept lattice of the retrieved documents may contain many irrelevant concepts resulting from spurious combinations of the document terms. This problem is especially relevant to the set of coatoms (i.e., the lower neighbors of the top element of the lattice), which must be shown to the user at the beginning of the interaction and should give an immediate idea of the main topics into which the results can be grouped. Different from expanding the lattice into tree hierarchical structure directly, we select those concepts which most relevant to the user query to build a hierarchical structure. To help users navigating and browsing the retrieved documents, generally the hierarchy is limited to 2-3 levels. Too deep hierarchy is not only unnecessary for the user, but difficult to get the meaningful labels. We limit the hierarchy to 2 levels in our system and the overlapping clusters are permitted. The details will be explained in the next section.

OBTAIN CLUSTERS FROM CONCEPT LATTICE

In this method, the conceptual clusters of documents are organized according to the concepts extracted from the concept lattice. We identified the sets of documents (extensions of the formal concepts) as clusters of documents and the sets of terms (intensions of the formal concepts) are considered as the cluster labels that represent their corresponding document clusters. To select the concepts most relevant to the user query from the lattice, several properties are defined for each concept firstly.

Concept importance: This property is used to indicate how important a concept is, denoted by CI. For each concept in the lattice, the more documents included in the extent of a concept, the more times the terms (intent) of the concept occurred in the resulting documents. We define the number of the documents in the extent of a concept as the TF feature of the concept. On the other hand, each concept in the concept lattice has its own descendant-concepts. The more descendant-concepts a concept has, the more generic the concept is. We define the number of descendant-concepts of a concept as the DF feature of the concept. We borrow the idea of tf*idf from the IR field and define CI as follows:

$$CI(C_i) = TF(C_i) * (1 + \log(\frac{M}{DF(C_i)})) \quad (1)$$

where, TF (C_i) is the number of the objects in the extent of the concept C_i, DF (C_i) is the number of the descendant-concepts of the concept C_i' M is the total concept number in the concept lattice. We will calculate the CI value of each concept in concept lattice by (Eq. 1) and sort them accordingly.

Concept similarity: This property indicates the similarity between two concepts, denoted by CS. It is evaluated based on the extent of each concept. Jaccard's similarity coefficient, defined as the ratio between the size of the intersection of two sets and the union of those sets, $|A \cap B| / |A \cup B|$, is used here:

$$CS(C_1, C_2) = \frac{|extent(C_1) \cap extent(C_2)|}{|extent(C_1) \cup extent(C_2)|} \quad (2)$$

where, (C_i) is the extent of the concept C_i, that is, the set of objects of C_i. We use CS property to avoid selecting similar concepts as different categories of the same level and a similarity threshold CS-Threshold is needed.

Step 1: Compute the CI value of each concept in the concept lattice and then sort them accordingly.
Step 2: Construct the two-level hierarchy structure.

Assume the top n node in the concept lattice has been processed and the two-level hierarchy structure H is built; Now we will process the n+1 node C_{n+1} and add it to H: if(CC all node in the first level of H)> CC-Threshold) exit; flag = false;
foreach node C_i in the first level of H {
if(CS(C_i, C_{n+1})>CS-Threshold) {
flag = true;
if(CC(all child nodes of C_i)>CC-Threshold) {
discard C_{n+1};
continue;
}
foreach childnode C_j of C_i {
if(CS(C_j, C_{n+1})> CS_Threshold) {
discard C_{n+1};
break;
}
}
set C_{n+1} as the child of C_i;
continue;
}
}
if (flag) set Cn+1 as
the next node in the
first level of H;

Fig. 2: The procedure to construct the two-level hierarchy based on concept lattice

ConceptSet coverage: The coverage of a concept set is used to control the number of child nodes of a certain node, denoted as CC.

$$CC(C_1,C_2,...,C_k) = \frac{\left|\bigcup_{i=1}^{k} \text{extent}(C_i)\right|}{\left|\text{extent}(C_s)\right|} \quad (3)$$

where, C_1, C_2,...., C_k are the already selected child nodes of Cs. The selection of child concepts for a certain parent concept will be stopped when the CC value exceeds a predefined threshold CC-Threshold.

The procedure of the proposed algorithm is shown in Fig. 2.

EXPERIMENTAL SETUP

The baseline: To compare the clustering capabilities of our method, we have chosen three other methods as baseline: STC, Lingo and Vivisimo. STC is a classical SRC algorithm and Lingo is the latest SRC algorithm we have found in the academic field, which leads to a search results clustering system Carrot2. Carrot2 is an open source search results clustering engine, more info can be acquired in http://project.carrot2.org/.

Different from STC and Lingo, Vivisimo is considered as an industrial standard in terms of clustering quality and user satisfaction and in 2001 and 2002 it has won the best meta-search-award assigned annually by the on-line magazine SearchEngineWatch.com. Vivisimo thus represents a particularly difficult baseline and it is unknown whether its clustering quality only depends on an extremely good clustering algorithm, or rather on the use of external knowledge or custom-developed resources. Here we chose Vivisimo as a baseline because we will construct the experimental datasets based on it.

The ground truth: As a metasearcher, Vivisimo has an advanced searching feature that allows retrieving search results from one or more different search engines. For the purpose of our experiment we restrict our source of snippets to the Open Directory Project (ODP) directory. The ODP is a searchable Web-based directory consisting of a collection of a few million web pages (as of today, ODP claims to index 4.6 million web pages) pre-classified into more than 590,000 categories by a group of volunteer human experts. In ODP, documents are organized as a tree, where the categories are internal nodes and pages are leaf nodes. The classification induced by the ODP labeling scheme gives us an objective ground truth against which we can compare the clustering quality of different SRC algorithms.

Queries are submitted to Vivisimo, asking it to retrieve pages only from ODP. This is done to ensure that all the SRC algorithms operate on the same set of snippets, hence to ensure full comparability of the results. The resulting set of snippets is collected and given as input to different SRC algorithms. Meanwhile, the ODP-categories of these results are established.

Clustering quality measure: In this research we measure the effectiveness of a clustering system by the degree to which it is able to correctly re-classify a set of pre-classified snippets into exactly the same categories without knowing the original category assignment. In other words, given a set $C = \{C_1,....C_k\}$ of categories and a set Θ of N snippets pre-classified under C, the ideal clustering algorithm is the one that, when asked to cluster Θ into k groups, produces a grouping $C' = \{C'_1,....C'_k\}$ such that, for each snippet $s_j \in \Theta$, $s_j \in s_i$ if and only if $s_j \in s'_i$. The original labeling is thus viewed as the latent, hidden structure that the clustering system must discover.

NMI (normalized mutual information) and NCE (normalized complementary entropy) are employed (Geraci *et al*., 2006) to compare the effectiveness of different SRC algorithms, which are defined as follows:

$$NMI(C,C') = \frac{2}{\log|C||C'|}\sum_{c\in C}\sum_{c'\in C'} P(c,c')\cdot\log\frac{P(c,c')}{P(c)\cdot P(c')} \quad (4)$$

Where:

$$P(c) = \frac{|c|}{N}, P(c') = \frac{|c'|}{N}, P(c,c') = \frac{|c\cap c'|}{N}$$

$$NCE(C,C') = \sum_{k=1}^{|C'|}\frac{|c_k'|}{N'}NCE(c_k',C) \quad (5)$$

Where:

$$NCE(c_k',C) = 1 - \frac{2}{\log|C|}\sum_{l=1}^{|C|} - \frac{P(c_k',c_l)}{P(c_l)}\log\frac{P(c_k',c_l)}{P(c_l)}$$

$$N' = \sum_{k=1}^{|C'|}|c_k'|$$

Higher values of NMI mean better clustering quality. NMI is designed for non-overlapping clustering therefore, we eliminate from the ground truth the snippets which present in multiple clusters produced by different SRC algorithms. NCE is designed for considering overlapping clusters. Note that when clusters may overlap it holds that $N' \geq N$. NCE ranges in the interval [0, 1] and a greater value implies better quality of clustering result.

We employed the same metrics in our experiments at first, but soon we found that there are many biases on the two metrics. Note that documents are organized as a tree in ODP, thus we can use different levels of the tree as the original categories in our experiments. On the other hand, different SRC algorithms can generate different numbers of clusters. To find the standard original category in ODP and find the comparative resulting clusters by different SRC algorithms, we did a lot experiments. Here are some biases of the two metrics found in our experiments: (1) If the original categories are fixed, the more clusters generated by a certain SRC algorithm, the higher value of the NMI and NCE obtained; (2) If the clusters which need to be compared are fixed, the more groups in the original categories, the higher value of the NMI obtained; (3) When comparing the resulting clusters generated by two different SRC algorithms with NCE or NMI, the performance may be reverted if using different original categories. For example, when the categories in a certain level of the ODP tree are used as original categories, the algorithm A is better than the algorithm B; the case may arise that B is better than A instead when use different level of the ODP tree as original categories.

Considering the above biases of the two metrics, we design two improved metrics: ANMI@K and ANMI@K. K means the number of clusters compared; for each SRC algorithm, we choose the top K clusters in the first level and compare the resulting clusters with the original categories; thus we can eliminate the bias caused by (1). ANMI and ANCE means average NMI and average NCE; we choose the first three levels of ODP tree as the original categories separately and compare the resulting clusters with each original category; then we compute the average NMI value obtained from the three comparisons; thus the biases caused by (2) and (3) can be eliminated. In this experiment, we set K as 5, 10 and 20 separately. In the authors' opinion, the top 5, 10 or 20 clusters are the most concerned by a user when he/she interacts with a clustering search engine. Also we compare all the resulting clusters obtained by different SRC algorithms. Generally, the number of resulting clusters of an algorithm is set to 30 if needed.

In addition, note that the above measures are used as the clustering quality measures; they cannot be used to assess the quality of a cluster label. We employ subjective cluster label quality judgments further.

EXPERIMENTAL RESULTS

Outcome of the comparative experiments: We selected 14 queries randomly in this experiment, which are: jaguar, apple, matrix, java, iraq, london, harry potter, world cup, Clinton, Mozart, ipod, travel, music, health. According to the classification of queries in Zeng *et al*. (2004), the above 14 queries can be cursorily divided into 4 ambiguous queries (the first four), 4 general queries (the last four) and 6 entity names (the middle six). Thus, we can say that these queries are representative enough to support the conclusions of the experiment.

Table 2: Clustering algorithm comparison: ANMI@K

	ANMI@5	ANMI@10	ANMI@20	ANMI@all
Vivisimo	0.0775	0.1653	0.2537	0.2818
CHC	0.0937	0.1956	0.2562	0.3384
Lingo	0.0710	0.1102	0.1707	0.1847
STC	0.0645	0.1055	0.1548	0.2114

Table 3: Clustering algorithm comparison: ANCE@K

	ANCE@5	ANCE@10	ANCE@20	ANCE@all
Vivisimo	0.7762	0.7796	0.8196	0.8381
CHC	0.7562	0.7793	0.7911	0.8331
Lingo	0.8392	0.8494	0.8667	0.8699
STC	0.6347	0.6726	0.7091	0.7331

Several statistics about the 14 datasets of the above queries are as follows: on average, there are 152 snippets returned for each query and the average numbers of categories of the top three levels in ODP are 10.75, 32.25 and 48.75, respectively.

Table 2 presents the average ANMI@5, ANMI@10, ANMI@20 and ANMI@all results for the resulting clusters of Vivisimo, CHC, Lingo and STC of the 14 queries. We can see that CHC performs best and provides an improvement of 20.81, 18.36, 1.03 and 20.07% with respect to Vivisimo and of 45.22, 85.40, 65.51 and 60.08% with respect to STC in ANMI@5, ANMI@10, ANMI@20 and ANMI@all. Table 3 presents the average ANCE@5, ANCE@10, ANCE@20 and ANCE@all results for the resulting clusters of the four SRC algorithms on the 14 datasets. We can see that CHC performs similar to Vivisimo and slightly worse than Lingo, while provides an improvement of 19.14, 15.86, 11.58 and 13.64% with respect to STC in ANCE@5, ANCE@10, ANCE@20 and ANCE@all.

Subjective label quality judgments: Figure 3 shows the labels of the top 10 clusters produced by Vivisimo, CHC, Lingo and STC for given query jaguar and world cup. By analyzing the labels carefully, we can find that there are many meaningless labels created by STC, such as Includes, Including etc. The difference between Events and Upcoming Events are ambiguous and thus the user will become bewildered on them.

While cluster labels produced by Lingo are generally readable and informative, there are still existed few meaningless labels, such as Und. Moreover, each cluster generated by Lingo only consists of few documents (which is indicated by the number followed each label), which means lots of documents are ungrouped into the top 10 clusters of Lingo.

Fig. 3: Clustering algorithms comparison: cluster labels

In the author's opinion, the labels produced by Vivisimo are better than the others. It should be noted that Vivisimo uses a proprietary algorithm, not in the public domain, which might make extensive use of external knowledge. By analyzing the labels and related documents, we find that snippets grouped in Photos also include the documents about picture and photography, thus we infer that Vivisimo at least treats synonyms specially.

Different from the above three methods, the labels produced by our method is not based on phrase but based on keyword. Each label is composed by one or more keywords. This is an interesting feature of concept lattice-based mining of web results, because it permits the discovery of deterministic or causal associations between words that hold in the set of results. Each label in the given example contains the corresponding query word, which is not designed artificially, but generated automatically by the algorithm. This point shows the query-related concepts are selected and thus demonstrates the effectiveness of our method.

The examples show the utility of our method to disambiguate a user query and to quickly focus on the documents relevant to the intended meaning. Further research will be performed on replacing the label with a meaningful phrase which contains all the keywords in the label. In fact, researches on the generation of the cluster labels and its effective measures are issues that remain largely unaddressed in the literature.

CONCLUSION AND FUTURE WORK

In this study we propose a FCA-based method named CHC to group and classify search results. The top N search results for a given query are retrieved from search engines and modeled as formal context, then the

formal concepts are extracted and the concept lattice is constructed. The unrelated concepts are eliminated and the concepts most relevant to the user query are selected. A two-level hierarchy is constructed and presented to the user finally. We analyze the biases in NMI and NCE and design two improved metrics, ANCE@K and ANCE@K, to evaluate the effectiveness of the resulting clusters generated by different SRC algorithms. Our experiments revealed that CHC significantly outperforms the other three baseline SRC algorithms (Vivisimo, Lingo and STC) with respect to ANCE@K, while maintaining the same level with Vivisimo on ANCE@K. In addition to comparison based on objective measures, we have also compared the quality of cluster labels generated by different SRC algorithms subjectively. Experimental results show that the labels generated by our method can improve user's browsing efficiency through search results.

We will further investigate several problems on search results clustering. First, we will try to extract the phrases in the search results and replace the labels generated by our method with meaningful phrases which contain all the keywords in the original labels. Second, a combination of some external sources of knowledge, such as WordNet (Fellbaum, 1998), might be helpful in this application.

REFERENCES

Carpineto, C. and G. Romano, 1996. A lattice conceptual clustering system and its application to browsing retrieval. Mach. Learn., 24: 95-122.

Carpineto, C. and G. Romano, 2004. Exploiting the potential of concept lattices for information retrieval with CREDO. J. Univ. Comput. Sci., 10: 985-1013.

Fellbaum, C., 1998. WordNet: An Electronic Lexical Database. 1st Edn., The MIT Press, Christiane Fellbaum, ISBN: 026206197X.

Fox, C., 1990. A stop list for general text. SIGIR Forum, 24: 19-21.

Geraci, F., M. Pellegrini, M. Maggini and F. Sebastiani, 2006. Cluster generation and cluster labeling for web snippets: A fast and accurate hierarchical solution. Proceedings of the 13th Symposium on String Processing and Information Retrieval, October 11-13, Glasgow, UK., pp: 25-36.

Godin, R., R. Missaoui and H. Alaoui, 1995. Incremental concept formation algorithms based on Galois (concept) lattice. Comput. Intell., 11: 246-267.

Hearst, M.A. and J.O. Pedersen, 1996. Reexamining the cluster hypothesis: Scatter/gather on retrieval results. Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1996, Zurich, Switzerland, pp: 76-84.

Juan, M.C., G. Julio and P. Anselmo, 2004. Browsing search results via formal concept analysis: Automatic selection of attributes. Proceedings of the 2nd International Conference on Formal Concept Analysis, February 19, Springer Berlin/Heidelberg, pp: 74-87.

Kummamuru, K., Kummamuru, R. Lotlikar, S. Roy, K. Singal and R. Krishnapuram, 2004. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. Proceedings of the 13th International Conference on World Wide Web, May 17-20, ACM New York, USA., pp: 658-665.

Lindig, C., 2000. Fast concept analysis. Working with Conceptual Structures-Contribution to the 8th International Conference on Conceptual Structures, August 2000, Darmstadt, Germany, pp: 152-161.

Osinski, S., J. Stefanowski and D. Weiss, 2004. Lingo: Search results clustering algorithm based on singular value decomposition. Proceedings of the International Conference on Intelligent Information Systems (IIPWM), May 17-20, Zakopane, Poland, pp: 359-367.

Porter, M.F., 1980q. An algorithm for suffix stripping. Program, 14: 130-137.

Zamir, O. and O. Etzioni, 1999. Grouper: A dynamic clustering interface to web search results. Proceedings of the 8th International World Wide Web Conference, May 1999, Toronto, Canad, pp: 1361-1374.

Zeng, H., Q. He, Z. Chen, W. Ma and J. Ma, 2004. Learning to cluster web search results. The 27th Annual International ACM SIGIR Conference on Research and Development in Informing Retrieval, July 25-29, Sheffield, South Yorkshire, UK., pp: 210-217.

Zhang, D. and Y. Dong, 2004. Semantic, hierarchical, online clustering of web search results. Proceedings of 6th Asia-Pacific Web Conference (APWEB), April 14-17, Hangzhou, China, pp: 69-78.