Query Clustering Using User Logs

JI-RONG WEN
Microsoft Research, Asia
JIAN-YUN NIE
University de Montréal
and
HONG-JIANG ZHANG
Microsoft Research, Asia

Query clustering is a process used to discover frequently asked questions or most popular topics on a search engine. This process is crucial for search engines based on question-answering. Because of the short lengths of queries, approaches based on keywords are not suitable for query clustering. This paper describes a new query clustering method that makes use of user logs which allow us to identify the documents the users have selected for a query. The similarity between two queries may be deduced from the common documents the users selected for them. Our experiments show that a combination of both keywords and user logs is better than using either method alone.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*Data mining*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.3.4 [**Information Storage and Retrieval**]: Systems and Software

General Terms: Algorithms, Experimentation, Performance

Additional Key Words and Phrases: Query clustering, web data mining, user log, search engine

1. INTRODUCTION

Despite the fact that keywords are not always good descriptors of contents, most existing search engines still rely solely on the keywords contained in documents and queries to calculate their similarity. This is one of the main factors that affects the precision of the search engines. In many cases, the answers returned by search engines are not relevant to the user's information need, although they do contain the same keywords as the query.

Faced with the increasing requirement for more precise information retrieval devices, a new generation of search engines—or question answering systems—have appeared on the Web (e.g. AskJeeves, http://www.askjeeves.com/). Unlike

Authors' addresses: Ji-R. Wen, Microsoft Research, Asia, 5F, Beijing Sigma Center No. 49, Zhichun Road Haidian District, Beijing, P. R. China; email: jrwen@microsoft.com; Jian-Y, Nie, DIRO, University of Montreal, CP 6128, succursale Center-ville, Montreal (Quebec), H3C 3J7 Canada; email: nie@IRO.Umontreal.CA; Hong-J. Zhang, Microsoft Research, Asia, 5F, Beijing Sigma Center No. 49, Zhichun Road Haidian District, Beijing, P.R. China; email: hjzhang@microsoft.com.

traditional search engines, these new systems try to "understand" the user's question in order to suggest similar questions that other people have asked and for which the system has the correct answers. In fact, the correct answers have been prepared or checked by human editors in most cases. It is then guaranteed that, if one of the suggested questions is truly similar to that of the user, the answers provided by the system will be relevant. The assumption behind such a system is that many people are interested in the same questions—the Frequently Asked Questions (FAQs). It is assumed that if the system can correctly answer these questions, then an important part of users' questions will be answered precisely.

The queries submitted by users are very different, however, and they are not always well-formed questions. (We will still use the term FAQ to refer to the queries frequently raised by users). In order to cluster queries, two related problems have to be solved: (1) How can human editors determine which questions/queries are FAQs? (2) How can a system judge if two questions/queries are similar?

The first question has to be answered in order to identify possible FAQs so that human editors can prepare/check their answers. It would be a tedious task to do this manually. What is needed is an automatic tool that helps the editors to identify FAQs. This is the goal of the present study—to cluster similar queries/questions together in order to discover FAQs.

The second question is closely related to the first one. This question has to be answered before implementing any automatic tool. It represents the core problem that we will deal with in this paper.

The classic approach to information retrieval (IR) would suggest a similarity calculation between queries according to their keywords. However, this approach has some known drawbacks due to the limitations of keywords. In the case of queries, in particular, the keyword-based similarity calculation will be very inaccurate (with respect to semantic similarity) due to the short lengths of the queries.

In this paper, we propose a new approach to query clustering based on user logs. In particular, we make use of cross-references between the users' queries and the documents that the users have chosen to read. Our hypothesis is that there is a stronger relationship between the queries and the selected documents (or clicked documents) than between these queries and other documents. Our query clustering approach is based on the following principles: (1) If users clicked on the same documents for different queries, then these queries are similar. (2) If a set of documents is often selected for the same queries, then the terms in these documents are; to some extent, related to the terms of the queries. The first principle has been applied in our query clustering approach. The second principle is being used in our work on the construction of a live thesaurus. Both principles are used in combination with the traditional approaches based on query contents (i.e. keywords). Our experimental results on query clustering show that many similar queries are actually clustered together by using our approach. In addition, we notice that many similar questions would have been separated into different

clusters by traditional clustering approaches because they do not share any common keywords. This study demonstrates the usefulness for a search engine, of user logs for query clustering, and the feasibility of an automatic tool to detect FAQs.

2. THE NEED FOR QUERY CLUSTERING AND OUR APPROACH

The current study has been carried out on the Encarta encyclopedia (http://encarta.msn.com/), which can be accessed on the Web. The Encarta editors have the problem of determining the FAQs of the users. This research work is launched specifically to answer the need of Encarta editors. However, the application of the approach described in this paper is not limited to Encarta. The idea of using user clicks is very general. It can be applied to any search engine where a large number of user logs are available.

Encarta contains a large number of documents organized in a hierarchy. A document (or article) is written on a specific topic, for instance, a country. It is further divided into sections and subsections on different aspects of the topic. For example, there is usually a section about the population of a country, another section on its culture, and so forth. In contrast with other documents on the Web, the quality of the documents is well controlled. The current search engine used on Encarta employs some advanced strategies to retrieve relevant documents using keywords. The answer to a query/question is a set of documents (articles, sections or subsections). Although the search engine suffers from the common problems of search engines using keywords, the result list does provide a good indication of the document contents. Therefore, when a user clicks on one of the documents in the result list, he/she has some idea about what can be found in the document. This provides us with a solid basis for user clicks as a valid indication of relevance.

Unlike a printed encyclopedia, Encarta is not static. It evolves in time. In fact, a number of human editors are working on the improvement of the encyclopedia so that users can find more information from Encarta and in a more precise way. Their work concerns, among other things, the following two aspects: (1) If Encarta does not provide sufficient information for some often asked questions, the editors will add more documents to answer these questions. (2) If many users asked the same questions (FAQ) in a certain period of time (a hot topic), then the answers to these questions will be checked manually and directly linked to the questions. This second aspect is also found in many new-generation search engines, such as AskJeeves.

Our first goal is to develop a clustering system that helps the editors quickly identify the FAQs. The key problem is to determine a similarity function that will enable the system to form clusters of truly similar queries. Our second goal is to create a live online thesaurus that links the terms used by the users to those found in the documents, again through an analysis of user logs. Given the large number of user logs, the relationships created could help the system match user queries and relevant documents despite differences in the terms they use.

3. REVIEW OF WORK RELATED TO QUERY CLUSTERING

Although the need for query clustering is relatively new, there have been extensive studies on document clustering, which is similar to query clustering. In this section, we give a review of some approaches related to query clustering.

3.1 Using Keywords

The first group of related clustering approaches is certainly those that cluster documents using the keywords they contain. In these approaches, in general, a document is represented as a vector in a vector space formed by all the keywords [Salton and McGill 1983]. Researches have been concerned mainly with the following two aspects:

—similarity function

—algorithms for the clustering process

Typically, people use either cosine-similarity, Jaccard-similarity or Dice-similarity [Salton and McGill 1983] as similarity functions. The edit-distance is also used in some other approaches [Gusfield 1997]. As to clustering algorithms, there have been mainly two groups: hierarchical and non-hierarchical. Hierarchical agglomerative clustering (HAC) algorithm and k-means are representatives of the two groups [Dubes and Jain 1988].

Keyword-based document clustering has provided interesting results. One contributing factor is the large number of keywords contained in documents. Even if some of the keywords of two similar documents are different, there are still many others that can make the documents similar in the similarity calculation. However, since queries, especially the queries submitted to the search engines, typically are very short, in many cases it is hard to deduce the semantics from the queries themselves. Therefore, keywords alone do not provide a reliable basis for clustering queries effectively.

In addition, words such as "where" and "who" are treated as stopwords in traditional IR methods. For questions, however, these words (if they occur) encode important information about the user's need, particularly in the new-generation search engines such as AskJeeves. For example, with a "who" question, the user intends to find information about a person. So even if a keyword-based approach is used in query clustering, it should be modified from that used in traditional document clustering.

Special attention is paid to such words in question answering (QA) [Kulyukin et al. 1998] [Srihari and Li 1999], where they are used as prominent indicators of question type. The whole question is represented as a template in accordance with the question type. During question evaluation, the question template may be expanded using a thesaurus (e.g. WordNet [Miller 1990]), or morphological transformations. In our case, we found that well-formed natural language questions represented only a small portion of queries. Most queries are simply short phrases or keywords (e.g. "population of U.S."). The approach used in QA is therefore not completely applicable to our case. However, if words denoting a question type do appear in a complete question, these words should be taken into account.

3.2 Using Hyperlinks

Because of the limitations of keywords, people have been looking for additional criteria for document clustering. One of them is the hyperlinks between documents. The hypothesis is that hyperlinks connect similar documents. This idea has been used in some early studies in IR [Garfield 1983] [Kessler 1963]. More recent examples are Google (http://www.google.com) and the authority/hub calculation of Kleinberg [1998]. Although Google does not perform document clustering explicitly, its PageRank algorithm still results in a weighting of hyperlinks. For a document, it is then straightforward to know the documents that are the most strongly related to it according to the weights of the hyperlinks to/from the document. Therefore, we can see PageRank as an implicit clustering approach. Google's use of hyperlinks has been very successful, making it one of the best search engines currently available.

The same idea is difficult to apply to query clustering, however, because there is no link between queries.

3.3 Using Cross-reference between Queries and Documents

In the hyperlink approaches, document space and query space are still separated. Our next question is whether it is possible to exploit the cross-reference between documents and queries in query/document clustering. By cross-reference, we mean any relationship created between a query and a document. The intuition of using cross-references is that similarity between documents can be transferred to queries through these references, and vice versa.

Relevance feedback in IR is a typical exploitation of cross-references. It is typically used to reformulate the user's query [Salton and McGill 1983]. It is also suggested that relevance feedback may be used as follows: if two documents are judged relevant to the same query, then there are reasons to believe that these documents talk about the same topic, and therefore can be included in the same cluster. Incorporating user judgments, in this way, may solve some of the problems in using keywords. However, in a traditional IR environment, the amount of relevance feedback information is too limited to allow for a reasonable coverage of documents.

In the Web environment, the choice of a particular document from a list by a user is another kind of cross-reference between queries and documents. Although it is not as accurate as explicit relevance judgment in traditional IR, the user's choice does suggest a certain degree of "relevance" of that document to his information need. In fact, users usually do not make the choice randomly. The choice is based on the information provided by the search engine on the returned documents (e.g. titles). Therefore, if many people select a common set of documents for the same query, this is a strong indication that these of documents are similar. This hypothesis has been used in several existing search engines. For example, in Citeseer (http://citeseer.nj.nec.com/), once a document is identified, the system also indicates a set of documents that other people have accessed together with that document. This is a form of implicit document clustering based on cross-references using the above hypothesis. Amazon

(http://amazon.com) exploits the same hypothesis to suggest books similar to those selected by users.

A similar idea can be used for query clustering—if a set of queries often lead to the same or similar document clicks, then these queries are similar, to some extent. This is the idea that we will further explore in this paper.

Similar ideas have been used in some work in IR. Voorhees et al. [1995] try to evaluate the quality of an IR system for a cluster of queries on different document collections. Their goal is to determine an appropriate method for database merging according to the quality estimation. The assumption used in query clustering is that if two queries retrieve many documents in common, they are on the same topic. In Lu and McKinley [2000], a measure of query similarity is used to route queries to proper collection replicas. In their work, two queries are considered to be related to the same topic if the top 20 documents they retrieve completely overlap.

The most similar work is that of Beeferman and Berger [2000], which exploits the same hypothesis for document and query clustering. However, while exploiting cross-references, Beeferman and Berger reject the use of the contents of queries and documents. They consider that keyword similarity is totally unreliable. We believe that content words provide some useful information for query clustering that is complementary to cross-references. Therefore, our approach tries to combine both content words and cross-references in query clustering. We will see in Section 6 that better results can be obtained using such a combination.

4. OUR APPROACH

Let us now describe the approach we use for query clustering. We will first reiterate the principles of our approach, then we will analyze them.

4.1 Clustering Principles

Our approach is based on two criteria: one is on the queries themselves, and the other on cross-references. We formulate them as the following principles:

Principle 1 (*using query contents*): If two queries contain the same or similar terms, they denote the same or similar information needs.

Obviously, the longer the queries, the more reliable is principle 1. However, as queries are short, this principle alone is not sufficient. Therefore, the second criterion is used as a complement.

Principle 2 (*using document clicks*): Two queries are similar if they lead to the selection of the same or similar document.

Document selections (or document clicks) are comparable to user relevance feedback in a traditional IR environment, except that document clicks denote implicit and not always valid relevance judgments.

The two criteria have their own advantages. In using the first criterion, we can group together queries of similar compositions. In using the second criterion, we benefit from user's judgments.

4.2 Discussion

As we mentioned before, principle 1 alone is not sufficient to cluster semantic-related queries. The main reason is that keywords and meanings do not strictly correspond. The same keyword does not always represent the same information need (e.g. the word "Java" may refer to "Java" the island, coffee, or a programming language); and different keywords may refer to the same concept (e.g. all the terms "atomic bomb", "Manhattan project", "nuclear weapon" and "Nagasaki" are related to the concept "atomic bomb"). Therefore, calculated similarity does not necessarily reflect semantic similarity. This is particularly the case for queries which are very short (usually 1–3 words).

On the other hand, one may also reject the use of the second principle as a reliable criterion. In fact, a document click is not a relevance judgment. Users may well click on irrelevant documents. Therefore, document clicks, if considered as relevance judgments, are noisy. In addition, a document can describe more than one specific topic, and can correspond to different queries. Putting together the queries corresponding to the same document may result in a cluster containing very different queries. These facts may raise reasonable doubt about the reliability of document clicks as a clustering criterion.

While it is true that choices made by a small number of users are likely to be unreliable, the large amount of information available in Encarta user logs makes this less of a problem; we assume that users are more consistent in their choices of relevant documents than irrelevant ones. Cross-references confirmed by many users are therefore taken to be reliable.

One more piece of supporting evidence is the results of pseudo-relevance feedback in IR [Xu and Croft 1996], which show that in reformulating queries using the top-ranked documents as if they are relevant, the retrieval effectiveness can be increased significantly. This shows that useful information about the connection between queries and documents in IR is not restricted to explicit relevance judgments. Even very coarse and noisy indications can help. Our situation is better than in pseudo-relevance feedback—not only are the documents presented to the user the top-ranked documents, but there is a further selection by the user. So document clicks are a more reliable indication than that used in pseudo-relevance feedback. Therefore, we can only expect better results. This expectation is supported by the quantitative evaluations given in Section 6.

Examinations of the Encarta user logs showed that most queries are followed by only one or two document selections. One may doubt whether such a small number of document clicks can provide a reliable basis for query clustering. This is very similar to the "short query" problem. Both content words and document clicks can partially reflect the semantics of the queries. Because of the small numbers of keywords and document clicks, however, neither of them is sufficient to be used alone. Therefore, we defined some combined measures to take advantage of both strategies. Our experiments (Section 6) show that the combined measures can be used to cluster more queries in a more precise way.

5. IMPLEMENTATION

In this section, we give some implementation details for our tool developed for Encarta.

5.1 Data and Pre-processing

The available data is a large set of user logs from which we extracted query sessions. A query session is defined as follows:

$$\text{session} := \text{<query text>}\ [\text{clicked document}]^*$$

Each session corresponds to one query and the documents the user clicked on. A query may be a well-formed natural language question, or one or more keywords or phrases. In Encarta, once a user query is input, a list of documents is presented to the user, together with the document titles. Because the document titles in Encarta are carefully chosen, they give the user a good idea of their contents. Therefore, if a user clicks on a document, it is likely that the document is relevant to the query, or at least related to it.

About 1 million queries are made each week, and about half of query sessions have document clicks. Out of these sessions, about 90% have 1–2 document clicks. Even if some of the document clicks are erroneous, we can expect that most users do click on relevant/strongly related documents.

5.2 Clustering Algorithm

To determine an appropriate clustering method, one first has to choose an appropriate clustering algorithm. There are many clustering algorithms available to us. The main characteristics that guide our choice are the following:

—As query logs usually are very large, the algorithm should be capable of handling a large data set within reasonable time and space constraints.

—The algorithm should not require manual setting of the resulting form of the clusters, for example, the number or the maximal size of clusters. It is unreasonable to determine these parameters in advance.

—Since we only want to find FAQs, the algorithm should filter out those queries with low frequencies.

—Due to the fact that the log data changes daily, the algorithm should be incremental.

The density-based clustering method DBSCAN [Ester et al. 1996] and its incremental version Incremental DBSCAN [Ester et al. 1998] satisfy the above requirements. DBSCAN does not require the number of clusters as an input parameter. A cluster consists of at least the minimum number of points—MinPts (to eliminate very small clusters as noise); and for every point in the cluster, there exists another point in the same cluster whose distance is less than the distance threshold Eps (points are densely located). The algorithm makes use of a spatial indexing structure (R*-tree) to locate points within the Eps distance from the core points of the clusters. All clusters consisting of less than the minimum number of points are considered as “noise” and are discarded.

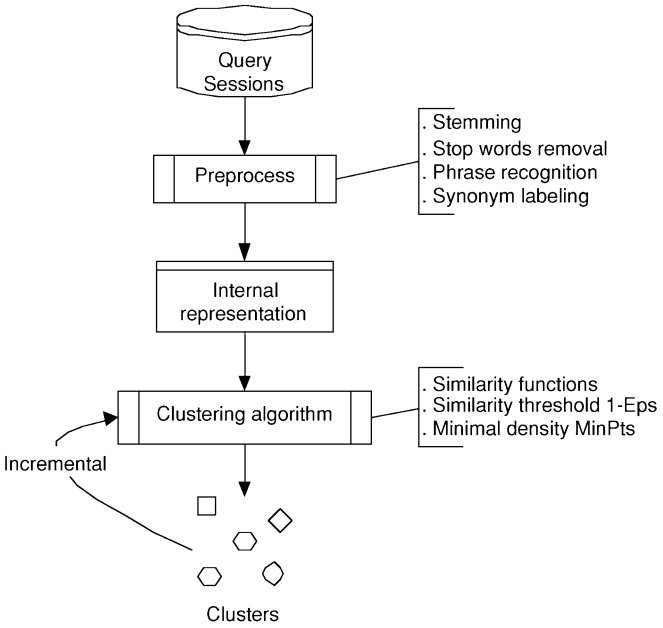

Fig. 1. Flow chart of the clustering process.

The average time complexity of the DBSCAN algorithm is O(n*logn). Previous experiments showed that DBSCAN outperforms CLARANS [Ng and Han 1994] by a factor of between 250 and 1900, which increases with the size of the data set. In our experiments, it only requires 3 minutes to deal with one-day user logs of 150,000 queries. The ability of Incremental DBSCAN to update incrementally is due to the density-based nature of the DBSCAN method, in which the insertion or deletion of an object only affects the neighborhood of this object. In addition, based on the formal definition of clusters, it has been proven that the incremental algorithm yields the same results as DBSCAN. Performance evaluations show Incremental DBSCAN to be more efficient than the basic DBSCAN algorithm.

We adopted DBSCAN and Incremental DBSCAN as the core algorithms of our query clustering tool. This clustering tool is organized as shown in Figure 1. Notice that we substitute the distance threshold Eps with a similarity threshold, which is equal to 1-Eps.

Once a clustering algorithm has been chosen, the next crucial problem is the definition of similarity functions. In the following, we will focus on this problem.

5.3 Similarity Based on Query Content Words

There are different ways to represent query contents: keywords, words in their order, and phrases. They provide different measures of similarity.

5.3.1 *Similarity Based on Keywords or Phrases.* This measure comes directly from IR studies. Keywords are all words, except function words included in a stop-list. All the keywords are stemmed using the Porter algorithm

[Porter 1980]. The keyword-based similarity function is defined as follows:

$$similarity_{keyword}(p,q) = \frac{KN(p,q)}{Max(kn(p), kn(q))} \quad (1)$$

where $kn(.)$ is the number of the keywords in a query, $KN(p, q)$ is the number of common keywords in two queries.

If query terms are weighted, the cosine similarity [Salton and McGill 1983] can be used instead:

$$similarity_{w\text{-}keyword}(p,q) = \frac{\sum_{i=1}^{k} cw_i(p) \times cw_i(q)}{\sqrt{\sum_{i=1}^{m} w_i^2(p)} \times \sqrt{\sum_{i=1}^{n} w_i^2(q)}} \quad (2)$$

where $cw_i(p)$ and $cw_i(q)$ are the weights of the *i-th* common keyword in the query p, and q respectively and $w_i(p)$ and $w_i(p)$ are the weights of the *i-th* keywords in the query p and q respectively. In our case, we use tf^*idf for keyword weighting.

The above measures can easily be extended to phrases. Since phrases are a more precise representation of meaning than a single word, if we can identify phrases in queries, we can obtain a more accurate calculation of query similarity. For example, the two queries "history of China" and "history of the United States" are very close queries (both asking about the history of a country). Their similarity is 0.33 on the basis of keywords. If we can recognize "the United States" as a phrase and take it as a single term, the similarity between these two queries is increased to 0.5. The calculation of phrase-based similarity is similar to formula (1) and (2). We only need to recognize phrases in queries. There are two main methods for doing this. One is by using a noun phrase recognizer based on syntactic rules and statistics [De Lima and Pederson 1999; Lewis and Croft 1990]. Another way is to use a phrase dictionary. In Encarta, there is such a dictionary, containing a large number of phrases and proper nouns that appear in Encarta documents. This dictionary provides us with a simple way to recognize phrases in queries. Our current phrase recognition is limited to the use of this dictionary. However, it may not be complete. In the future, it will be supplemented by an automatic phrase recognizer based on a syntactic and statistical analysis.

5.3.2 *Similarity Based on String Matching.* This measure uses all the words in the queries for similarity estimation, even the stopwords. Comparison between queries becomes an inexact string-matching problem, as formulated by Gusfield [1997]. Similarity may be determined by the edit distance, which is a measure based on the number of edit operations (insertion, deletion, or substitution of a word) necessary to unify two strings (queries). In our case, we use the maximum number of words in the two queries to divide the edit distance so that the value can be constrained within the range of [0, 1]. The similarity is inversely proportional to the normalized edit distance:

$$similarity_{edit}(p,q) = 1 - \frac{Edit\_distance(p,q)}{Max(wn(p), wn(q))} \quad (3)$$

where $wn(.)$ is the number of the words in a query.

The advantage of this measure is that it takes into account the word order, and also words that denote query types such as "who" and "what" if they appear in a query. This method is more flexible than those used in QA systems, which rely on special recognition mechanisms for different types of questions. It is therefore more suitable to our situation.

In our experiments, we found that this measure is particularly useful for long and complete questions in natural language. Below are some questions that have been grouped into one cluster:

—Query 1: Where does silk come from?

—Query 2: Where does lead come from?

—Query 3: Where does dew come from?

This cluster contains questions of the form "Where does X come from?"

In the similarity calculations described above, we can further incorporate a dictionary of synonyms, which also exists in Encarta. If two words/terms are synonyms, their similarity is set at a predetermined value (0.8 in our current implementation). It is then easy to incorporate this similarity between synonyms into the calculations (1), (2) and (3).

5.4 Similarity Based on Cross-references

Let $D(p)$ and $D(q)$ be the set of documents the system presents to the user as search results for the queries p and q respectively. The document set that users clicked on for the queries p and q may be seen as follows:

$$D\_C(p) = \{d_{p1}, d_{p2}, \ldots, d_{pi}\} \subseteq D(p)$$
$$D\_C(q) = \{d_{q1}, d_{q2}, \ldots, d_{qj}\} \subseteq D(q)$$

Similarity based on cross-references follows the following principle: If $D\_C(p) \cap D\_C(q) = \{d_{pq1}, d_{pq2}, \ldots, d_{pqk}\} \neq \emptyset$, then documents $d_{pq1}, d_{pq2}, \ldots, d_{pqk}$ represent the common topics of queries p and q. Therefore, a similarity between the queries p and q is determined by $D\_C(p) \cap D\_C(q)$.

Encarta documents may be considered either in isolation or in terms of their position in the Encarta hierarchy.

5.4.1 *Similarity through a Single Document.* A first similarity measure based on cross-references considers each document in isolation. This similarity is proportional to the shared number of clicked (or selected) documents, taken individually, as follows:

$$similarity_{single\_doc}(p, q) = \frac{RD(p, q)}{Max(rd(p), rd(q))} \tag{4}$$

where $rd(.)$ is the number of clicked documents for a query, and $RD(p, q)$ is the number of document clicks in common.

In spite of its simplicity, this measure demonstrates a surprising ability to cluster semantically related queries that contain different words. Below are some queries from one such cluster:

—Query 1: atomic bomb
—Query 2: Nagasaki
—Query 3: nuclear bombs
—Query 4: Manhattan Project
—Query 5: Hiroshima
—Query 6: nuclear fission
—Query 7: Japan surrender
—.....

They all correspond to the document "ID: 761588871, Title: Atomic Bomb" in Encarta. It is obvious that no approach that uses only the keywords in the queries could create such a cluster.

In addition, this measure is also very useful in distinguishing between queries that happen to be worded similarly but stem from different information needs. For example, if one user asked for "law" and clicked on the articles about legal problems, and another user asked "law" and clicked the articles about the order of nature, the two cases could be easily distinguished through user clicks. This kind of distinction can be exploited for sense disambiguation in a user interface. This is an aspect we will investigate in the future.

5.4.2 *Similarity through Document Hierarchy.* Documents in Encarta are not isolated; they are organized into a hierarchy that corresponds to a concept space. The hierarchy contains 4 levels. The first level is the root. The second level contains 9 categories, such as "physical science & technology", "life science", "geography", and so forth. These categories are divided into 93 subcategories. The last level (the leaf nodes) is made up of tens of thousands of documents. Roughly, this document hierarchy corresponds to a domain/concept hierarchy. This hierarchy allows us to extend the previous calculation by considering a conceptual distance between documents. This distance is determined as follows: the lower the common parent node shared by two documents, the shorter the conceptual distance between the two documents, and the higher their conceptual similarity. Let $F(d_i, d_j)$ denote the lowest common parent node for documents d_i and d_j, $L(x)$ the level of node x, L_Total the total levels in the hierarchy (i.e. 4 for Encarta). The conceptual similarity between two documents is defined as follows:

$$s(d_i, d_j) = \frac{L(F(d_i, d_j)) - 1}{L\_Total - 1} \tag{5}$$

In particular, $s(d_i, d_i) = 1$; and $s(d_i, d_j) = 0$ if $F(d_i, d_j) =$ root.

Now let us incorporate this document similarity measure into the calculation of query similarity. Let $d_i (1 \le i \le m)$ and $d_j (1 \le j \le n)$ be the clicked documents for queries p and q respectively, and $rd(p)$ and $rd(q)$ the number of document clicks for each query. The hierarchy-based similarity is defined as follows:

$$similarity_{hierarchy}(p, q) = \frac{1}{2} \times \left(\frac{\sum_{i=1}^{m} \left(max_{j=1}^{n}\, s(d_i, d_j) \right)}{rd(p)} + \frac{\sum_{j=1}^{n} \left(max_{i=1}^{m} s(d_i, d_j) \right)}{rd(q)} \right) \tag{6}$$

Table I. Query Sessions

No.	Query Text	Clicked Documents
1.	law of thermodynamics	ID: 761571911 Title: Thermodynamics ID: 761571262 Title: Conservation Laws
2.	conservation laws	ID: 761571262 Title: Conservation Laws ID: 761571911 Title: Thermodynamics
3.	Newton law	ID: 761573959 Title: Newton, Sir Isaac ID: 761573872 Title: Ballistics
4.	Newton law	ID: 761556906 Title: Mechanics ID: 761556362 Title: Gravitation

Using formula (6), the following two queries are recognized as being similar:

—Query 1: <query text> image processing
<clicked documents> ID: 761558022 Title: Computer Graphics

—Query 2: <query text> image rendering
<clicked documents> ID: 761568805 Title: Computer Animation

Both documents in the example have a common parent node "Computer Science & Electronics". According to formula (5), the similarity between the two documents is 0.67. If these were the only two documents selected for the two queries, then the similarity between the queries would also be 0.67 according to formula (6). In contrast, their similarity based on formula (4) using common clicks is 0. Hence, this new similarity function can recognize a wider range of similar queries.

5.5 Combination of Multiple Measures

Similarities based on query contents and cross-references represent two different points of view. In general, content-based measures tend to cluster queries with the same or similar terms; but similar terms could be used to represent different query requirements because of the ambiguity of words. Measures based on cross-references tend to cluster queries related to the same or similar topics; but a document usually contains more than one topic or "interest point", thus queries with different intentions may lead to the same document.

Since user's information needs may be partially captured by each of the above criteria, we would like to define a combined measure that takes advantage of both strategies. A simple way to do this is to combine both measures linearly, as follows:

$$similarity = \alpha^* similarity_{content} + \beta^* similarity_{cross\_ref} \tag{7}$$

This raises the question of how to set parameters α and β. We believe that these parameters should be set according to the editor's objectives. It is difficult to determine them in advance; they can be adjusted over time and in light of the system's use. In what follows, we give a simple example to show the possible effects of different measures, as well as their combination.

Let us consider the four queries shown in Table I. Assume that the similarity threshold is set at 0.6. According to the queries and the corresponding document

Table II. Similarities between Documents

	①	②	③	④	⑤	⑥
①Thermodynamics	1.0	0.66	0.33	0.33	0.66	0.66
②Conservation laws	0.66	1.0	0.33	0.33	0.66	0.66
③Newton, Sir Isaac	0.33	0.33	1.0	0.33	0.33	0.33
④Ballistics	0.33	0.33	0.33	1.0	0.33	0.33
⑤Mechanics	0.66	0.66	0.33	0.33	1.0	0.66
⑥Gravitation	0.66	0.66	0.33	0.33	0.66	1.0

clicks, we can guess that the expected result would be to group queries 1 and 2 in a cluster, and queries 3 and 4 in another.

1. If the keyword-based measure is applied (formula (1)), the queries are divided into 3 clusters:
   —Cluster 1: Query 1
   —Cluster 2: Query 2
   —Cluster 3: Query 3 and query 4
   Queries 1 and 2 are not clustered together.
2. If we use the measure based on individual documents (formula (4)), we obtain:
   —Cluster 1: Query 1 and query 2
   —Cluster 2: Query 3
   —Cluster 3: Query 4
   In this case, queries 3 and 4 are not judged to be similar.
3. Now we use the measure based on document hierarchy. The document similarities according to formula (5) are shown in Table II.
   Applying formula (6), we can group the queries as follows:
   —Cluster 1: Query 1, query 2, and query 4
   —Cluster 2: Query 3
   Thus we have not been able to separate query 4 from queries 1 and 2.
4. Now let us use formula (7) with $similarity_{content} = similarity_{keyword}$, and $similarity_{cross\_ref} = similarity_{hierarchy}$. Both α and β are set to 0.5. The queries are now clustered in the desired way:
   —Cluster 1: Query 1 and query 2
   —Cluster 2: Query 3 and query 4

The purpose of this example is to show that each similarity calculation focuses on a specific aspect. By combining them in a reasonable way, better results can be obtained. Therefore, by trying different combinations, the editors will have a better chance of locating desired FAQs. Our current implementation includes all of the similarity measures described above. An interface allows the editors to choose different functions and to set different combination parameters (see Figure 2).

6. EVALUATION

This section provides empirical evidence as to how different similarity functions affect the query clustering results. We collected one-month user logs (about 22

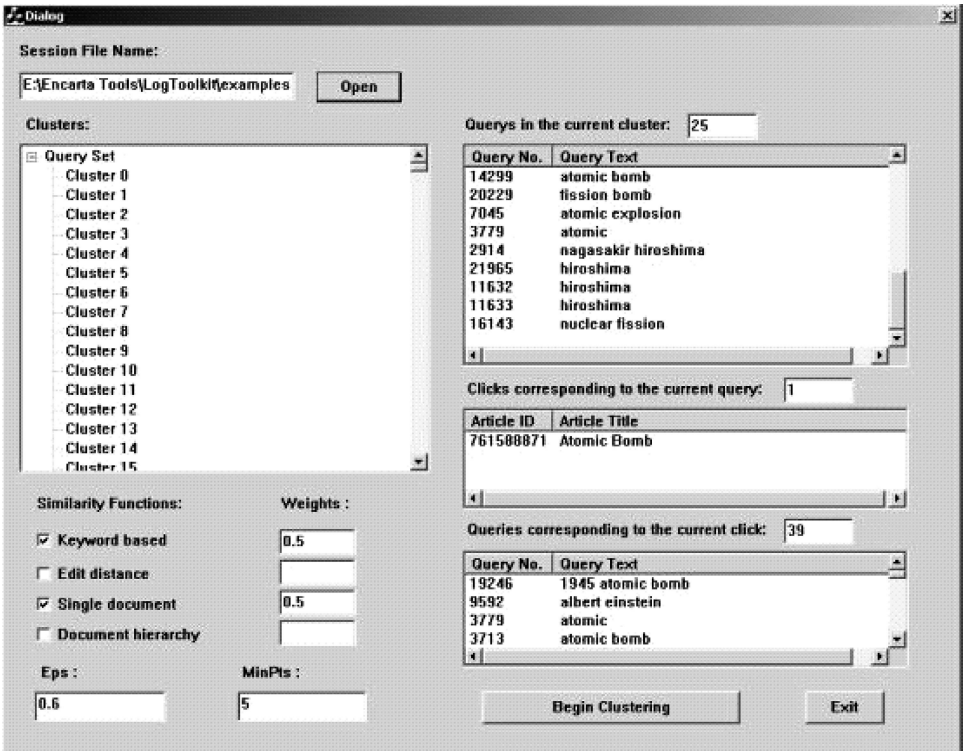

Fig. 2. Interface of the query clustering tool.

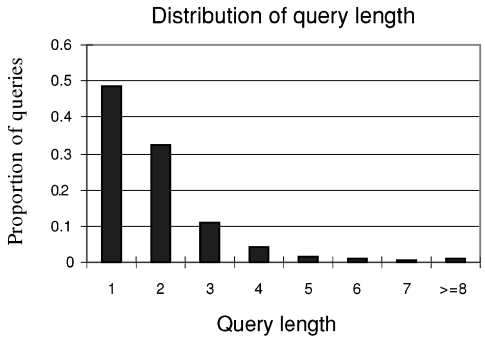

Fig. 3. Distribution of query length.

GB) from the Encarta web site. From these logs we extracted 2,772,615 user query sessions. Figure 3 illustrates the distribution of query lengths in terms of the number of words. We notice that 49% of the queries contain only one keyword and 33% of the queries contain two keywords. The average length of all queries is 1.86. The distribution of query length is similar to those reported by others. Because the number of queries is too big to conduct detailed evaluations, we randomly chose 20,000 query sessions for our evaluations.

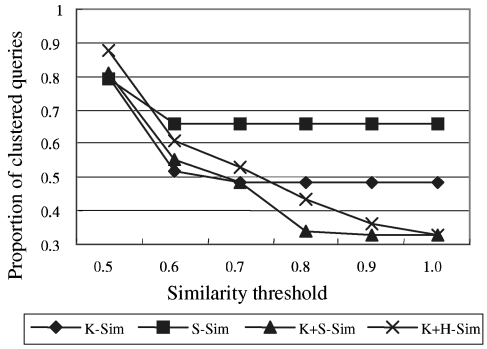

Fig. 4. Proportion of clustered queries vs. similarity threshold.

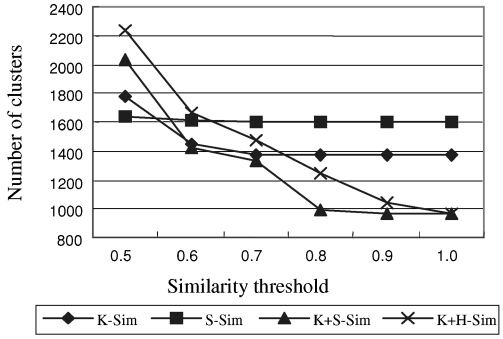

Fig. 5. Number of clusters vs. similarity threshold.

6.1 Experimental Results

We tested the following four similarity functions on the 20,000 query sessions:

—Keyword similarity (K-Sim),

—Cross-reference similarity using single documents (S-Sim),

—Keyword + cross-reference similarity using single documents (K + S-Sim), and

—Keyword + cross-reference similarity using document hierarchy (K + H-Sim).

The minimal density parameter (MinPts) was uniformly set to 3, which means that only those clusters containing at least 3 queries were kept. Then we varied the similarity threshold (=1–Eps) from 0.5 to 1.0. We assigned weight 0.5 to both α and β.

6.1.1 *Verification of the FAQ Concept.* By varying the similarity threshold we obtain different proportions and numbers of clustered queries (Figures 4 and 5). When using K-Sim to cluster all 20,000 queries, the proportions of clustered queries decrease from 0.80 to 0.48 (Figure 4) and the number of clusters decreases from 1778 to 1368 (Figure 5), along with the change of similarity threshold from 0.5 to 1.0. It is interesting to observe the threshold at 1.0 (where queries in the same cluster are formed with identical keywords). We see

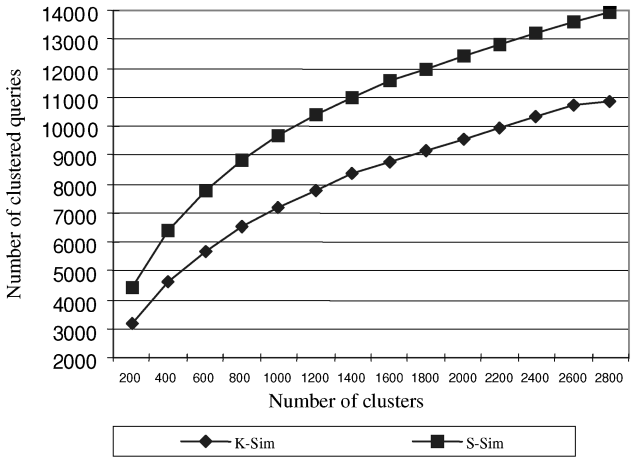

Fig. 6. Correlation between number of clustered queries and number of clusters.

(Figure 4) that 48% of queries are formed with the same keywords and appear at least three times. The average number of queries per cluster in this case is 7.1.

The proportions of clustered queries of S-Sim decrease from 0.79 to 0.66 and the number of clusters decrease from 1632 to 1602 when similarity threshold is varied from 0.5 to 1.0. When similarity threshold is 1.0, 66% of queries are put into 1756 clusters and the average number of queries per cluster is 8.24.

The two figures show that many users' interests focus on a relatively small number of topics—they often use a small set of words, and they choose to read a small set of documents. This confirms the hypothesis behind the FAQ approach—that many users are interested in the same topics (or FAQs).

The comparison between the K-Sim and S-Sim curves shows that clusters using S-Sim cover more queries than K-Sim. This suggests that there are more divergences in query words than in document selections.

Both combinations shown in the figures change more than single-criterion functions. The small proportion of clustered queries at threshold = 1.0 shows that the joint conditions of identical words and identical document selections are difficult to satisfy completely. However, when the threshold is low (0.5), there may be more queries clustered in the combined approach (K + H-Sim) than in the single-criterion approaches; but the size of clusters is smaller (because there are many clusters).

To further verify this hypothesis, we draw the following figure which shows the correlation between number of clustered queries and number of clusters (Figure 6). The clusters in this figure are obtained with the threshold set at 1.0, that is, they contain identical queries—queries with identical keywords (K-Sim) or queries leading to the same document clicks (S-Sim).

In this figure, we can see that quite a number of identical queries appear in a small number of clusters (the biggest clusters). For example, the K-Sim curve shows that the 4500 most popular queries (22.5% of the total number) are grouped into only about 400 clusters, which further confirms the FAQ concept,

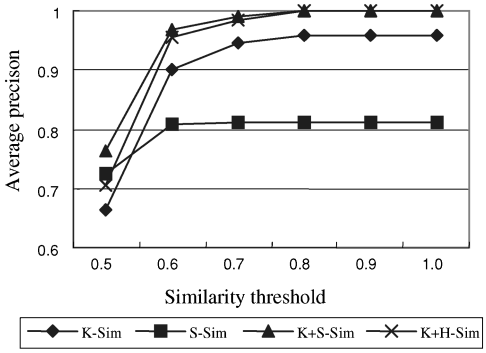

Fig. 7. Precision for four kinds of similarity functions.

that is, many users tend to use similar or identical queries in a given period of time. Moreover, the S-Sim curve shows that only 200 clusters are needed to cover the 4500 top queries, which confirms that many users are interested in a small number of documents, that is, there is a similar concept of frequently asked documents (FAD). In addition, through the comparison of the two curves, we can see that there is a stronger concentration in document clicks than in common keywords.

6.1.2 *Quality of Clustering Results.* We borrow two IR metrics to measure the quality of clustering results:

—Precision—the ratio of the number of similar queries to the total number of queries in a cluster.

—Recall—the ratio of the number of similar queries to the total number of all similar queries for these queries (those in the current cluster and others).

For every similarity function, we randomly selected 100 clusters. For each cluster, we calculated precision by manually checking the queries. Since we do not know the actual intentions of users with their queries, we simply guess, taking into account both queries and clicked documents. We report the average precision of the 100 clusters in Figure 7, where all four functions are shown, with similarity threshold varying from 0.5 to 1.0.

We first observe that the combinations of keywords and cross-references (K + S-Sim and K + H-Sim) result in higher precision than the two criteria separately. When similarity threshold is equal to or higher than 0.6, K + S-Sim and K + H-Sim have very high precision (above 95%). When the similarity threshold is higher than 0.8, the precision for both similarity functions reaches 1.0.

For clustering using single criteria, we observe that the highest precision that can be reached by K-Sim is about 96%, when all queries in a cluster contain identical keywords. This means that the ambiguity of keywords will bring in only about 4% errors. This number is much lower than our expectation. A possible reason is that users are usually aware of word ambiguity and like to use more precise queries. For example, instead of using “Java”, users use “Java island” or “Java programming language” to avoid ambiguities.

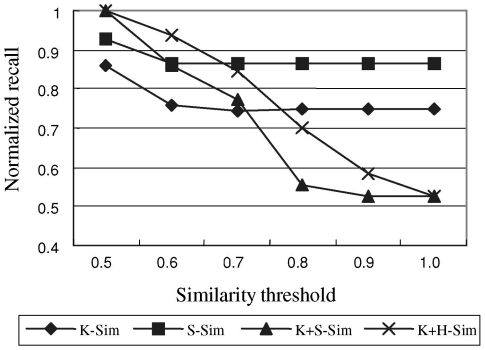

Fig. 8. Recall for four kinds of similarity functions.

It is difficult to use the recall metric directly for clustering because no standard clusters or classes are available. Therefore, we use a different measure to reflect, to some extent, the recall factor—normalized recall. This factor is calculated as follows:

—For any similarity function, we collect the number of correctly clustered queries in all 100 clusters. This is equal to the total number of clustered queries times the precision.

—Then we normalize this value by dividing it by the maximum number of correctly clustered queries. In our case, this number is 12357 which is obtained by K + H-Sim when the similarity threshold is 0.5. This normalization aims to obtain a number in the range [0, 1].

Figure 8 shows the normalized recalls for the four similarity functions when similarity threshold varies from 0.5 to 1.0.

We observe that when similarity threshold is below 0.6, K + H-Sim and S + H-Sim result in better normalized recall ratios than using two other functions on single criteria. This shows that both functions can take advantage of both criteria by combining them. However, when similarity threshold increases, normalized recall ratios drop quickly. On the other hand, there is almost no change for S-Sim and K-Sim for threshold higher than 0.6. Again, this is due to the small number of keywords per query and document clicks per query.

Although the precision of K + H-Sim is very close to K + S-Sim (Figure 7), the normalized recall of K + H-Sim is always higher than K + S-Sim with a margin of about 10%. This shows that, when combined with keywords, the consideration of document hierarchy is helpful for significantly increasing recall without decreasing precision.

In order to compare the global quality of the four functions, we use the F-measure [Van Rijsbergen 1979] as metric (in which the recall ratio is replaced by our normalized recall). Although the modified F-measure is different from the traditional one, it does provide some indication on the global quality of different similarity functions. Figure 9 shows this evaluation.

We can see that within the threshold range of [0.5, 0.7], K + H-Sim and K + S-Sim are better than the single-criterion functions. In particular, when

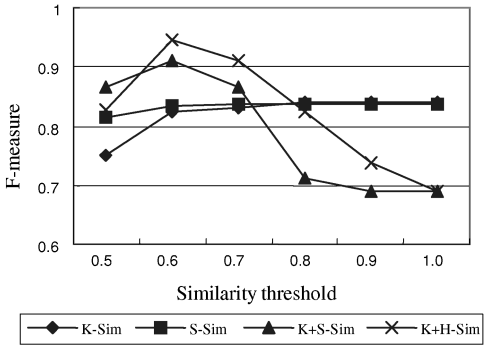

Fig. 9. F-measures for four kinds of similarity functions.

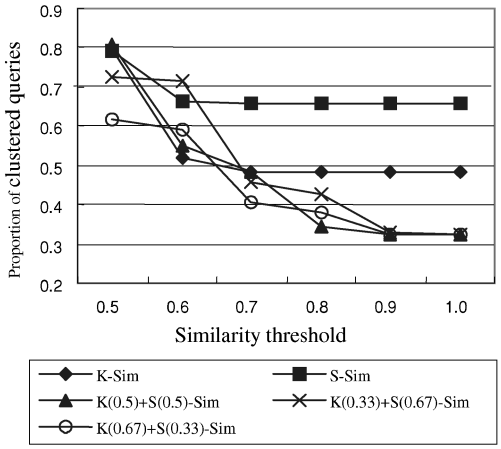

Fig. 10. Correlation between weights and clustering results.

similarity threshold is equal to 0.6, K + H-Sim reaches the highest F-measure value (0.94).

All the above experiments show that it is beneficial to combine keywords and user document clicks in query clustering.

6.1.3 *The Impact of Combination Parameters.* To investigate the correlation between clustering results and the setting of parameters α and β, we tested three different combinations within K + S-Sim: (1) $\alpha = 0.5$ and $\beta = 0.5$; (2) $\alpha = 0.67$ and $\beta = 0.33$; (3) $\alpha = 0.67$ and $\beta = 0.33$. Figure 10 shows the proportions of clustered queries for these three settings with respect to the similarity threshold (in comparison with the two single-criterion functions).

We observe that to some extent, the setting influences the behavior of the algorithm. The general trend with respect to the threshold of all the three settings is similar—they decrease when the threshold is higher, and their change is larger than in the single-criterion cases. Between (0.5, 0.9), we do observe some difference among the three settings. This shows that the setting of the two parameters has a noticeable impact on the behavior of the clustering algorithm.

It is interesting to observe that at some point (threshold = 0.6) that when S-Sim and K-Sim are respectively supplemented by the other criterion, even more queries can be clustered. Therefore, the addition of a new criterion does not uniformly decrease the number of clustered queries.

This experiment shows that by varying the parameters α and β, we can obtain very different clustering results. This offers a certain flexibility to the editors in their exploration of the query sessions.

7. CONCLUSION

The new generation of search engines for precise question answering requires the identification of FAQs, so that human editors can prepare the correct answers for them. The identification of FAQs is not an easy task. It requires a proper estimation of query similarity. Given the different forms of queries and user intentions, the similarity of queries cannot be accurately estimated through an analysis of their contents (e.g. keywords). In this paper, we suggested user logs (recording user document clicks) as a supplement. A new clustering principle is proposed—if two queries correspond to the same or similar document clicks, they are similar. Our analysis of the clustering results suggests that this clustering strategy can group similar queries together more effectively than using keywords alone. It provides assistance for human editors in finding new FAQs. Indeed, the clustering tool is currently in use by the Encarta editors who have reported the successful identification of a considerable number of true FAQs.

The project is still ongoing. There are several avenues for further improving and extending the clustering tool:

—The identification of phrases may be extended by using an automatic recognizer based on syntax.

—The calculation of query similarity can incorporate more relationships between terms, in addition to the synonyms stored in the Encarta dictionary. One possibility is to use a co-occurrence analysis on documents in Encarta to calculate term similarity. Another possible solution is to extract strong relationships between query terms and the terms in the clicked documents using a statistical analysis on user logs. This will allow us to facilitate user queries by narrowing the gap between query space and document space.

—In this paper, to calculate query similarity, we made use of a limited similarity of documents based on the document hierarchy. In fact, it is possible to use any document similarity in this process. The intuition is that if two queries often lead to similar documents, then they are also similar. Document similarity based on keywords or hyperlinks can be further exploited for this purpose. This will enable us to take advantage of the richer contents of documents for the calculation of query similarity. It is also possible to apply the dual of our principle 2—If the same or similar queries often lead to the selection of two documents, then these two documents are similar.

—Our method of query similarity calculation can also be used in different contexts. It can be extended to provide a word disambiguation tool by considering

each clicked document as a possible meaning of a query word. It is also possible to use it in query expansion—a short query can be expanded by its similar queries and/or their answers. Such an expansion has been proposed in several studies, for example, Fitzpatrick and Dent [1997] and De Lima and Pederson [1999]. Our calculation of query similarity can provide a new way to determine similar queries and then use them in query expansion process.

In summary, the availability of large numbers of user logs provides new possibilities for search engines. In particular, it allows trends in user searching behavior to be spotted, thus helping builders of search engines and editors responsible for content to improve their system. The present study is only a first step in this direction.

ACKNOWLEDGMENTS

The authors would like to thank Jon Rosenberg, Kevin Cretin, Bob O'Brien and Harish Jayanti of the Microsoft Encarta group for providing us with the data and giving us much helpful feedback. We are also grateful to the anonymous referees for their valuable comments and suggestions. A special thank to Graham Russell of RALI lab for his careful reading and correction of this paper.

REFERENCES

BEEFERMAN, D. AND BERGER, A. 2000. Agglomerative clustering of a search engine query log. In *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (August). Acm Press, New York, NY, 407–416.

DUBES, R. C. AND JAIN, A. K. 1988. *Algorithms for Clustering Data.* Prentice-Hall, Englewood Cliffs, NJ.

ESTER, M., KRIEGEL, H., SANDER, J., AND XU, X. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*. 226–231.

ESTER, M., KRIEGEL, H., SANDER, J., WIMMER, M., AND XU, X. 1998. Incremental clustering for mining in a data warehousing environment. In *Proceedings of the 24th International Conference on Very Large Data Bases*, 323–333.

FITZPATRICK, L. AND DENT, M. 1997. Automatic feedback using past queries: social searching? In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 306–312.

DE LIMA, E. AND PEDERSEN, J. 1999. Phrases recognition and expansion for short, precision-biased queries based on a query log. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 145–152.

GARFIELD, E. 1983. *Citation Indexing: Its Theory and Application in Science, Technology and Humanities*, 2nd ed. The ISI Press, Philadelphia, PA.

GUSFIELD, D. 1997. Inexact matching, sequence alignment, and dynamic programming. In *Algorithms on Strings, Trees, and Sequences Computer Science and Computational Biology*, Cambridge University Press.

KESSLER, M. M. 1963. Bibliographic coupling between scientific papers. In *American Documentation, 14*, 1, 10–25.

KLEINBERG, J. 1998. Authoritative sources in a hyperlinked environment. In *Proceedings of the 9th ACM SIAM International Symposium on Discrete Algorithms*. ACM Press, New York, NY, 668–677.

KULYUKIN, V. A., HAMMOND, K. J., AND BURKE, R. D. 1998. Answering questions for an organization online. In *Proceedings of AAAI 98*. 532–538.

LEWIS, D. D. AND CROFT, W. B. 1990. Term clustering of syntactic phrases. In *Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM Press, New York, NY, 385–404.

LU, Z. AND MCKINLEY, K. 2000. Partial collection replication versus caching for information retrieval systems. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 248–255.

MILLER, G. A., ED. 1990. WordNet: an on-line lexical database, *Int. J. Lexico. 3*, 4.

NG, R. AND HAN, J. 1994. Efficient and effective clustering method for spatial data mining. In *Proceedings of the 20th International Conference on Very Large Data Bases*. 144–155.

PORTER, M. 1980. An algorithm for suffix stripping. *Program*, *14*, 3, 130–137.

SALTON, G. AND MCGILL, M. J. 1983. *Introduction to Modern Information Retrieval.* McGraw-Hill New York, NY.

SRIHARI, R. AND LI, W. 1999. Question answering supported by information extraction. In *Proceedings of TREC8*, 75–85.

VAN RIJSBERGEN, C. J. 1979. *Information Retrieval*. 2nd ed, Butterworths, London.

VOORHEES, E., GUPTA, N. K., AND JOHNSON-LAIRD, B. 1995. Learning collection fusion strategies. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 172–179.

XU, J. AND CROFT, W. B. 1996. Query expansion using local and global document analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, 4–11.

Received April 2001; revised August 2001, and September 2001; accepted September 2001