Automatic query taxonomy generation for information retrieval applications

Shui-Lung Chuang and Lee-Feng Chien

The authors

Shui-Lung Chuang is a Research Associate and **Lee-Feng Chien** is a Research Fellow, both at the Institute of Information Science, Academia Sinica, Taipei, Taiwan.

Keywords

Information retrieval, Worldwide web, Search engines, Query languages, Taxonomy, Cluster analysis

Abstract

It is crucial for information retrieval systems to learn more about what users search for in order to fulfil the intent of searches. This paper introduces query taxonomy generation, which attempts to organise users' queries into a hierarchical structure of topic classes. Such a query taxonomy provides a basis for the in-depth analysis of users' queries on a larger scale and can benefit many information retrieval systems. The proposed approach to this problem consists of two computational processes: hierarchical query clustering to generate a query taxonomy from scratch, and query categorisation to place newly-arrived queries into the taxonomy. The results of the preliminary experiment have shown the potential of the proposed approach in generating taxonomies for queries, which may be useful in various Web information retrieval applications.

Electronic access

The Emerald Research Register for this journal is available at
http://www.emeraldinsight.com/researchregister

The current issue and full text archive of this journal is available at
http://www.emeraldinsight.com/1468-4527.htm

Online Information Review
Volume 27 · Number 4 · 2003 · pp. 243-255
© MCB UP Limited · ISSN 1468-4527
DOI 10.1108/14684520310489032

Introduction

An information retrieval (IR) environment can be depicted as an abstract space composed of three main sub-spaces, as shown in Figure 1. The user space contains people who submit search requests, the document space contains the available documents, and the intermediary space contains the intermediaries that enhance the connection between the users and the documents. In a typical search scenario, large amounts of diverse users in the user space are searching or requesting relevant documents from the document space. Since users and documents do not have the same vocabularies for describing the same concepts, intermediaries – such as information retrieval systems – help to bridge the gap.

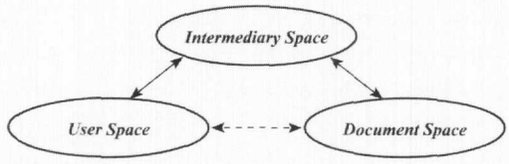
Generally, research on information retrieval is more document-oriented. Much effort has been expended on analysing documents, such as creating effective full-text indexes for documents and manually organising a well-formed taxonomy structure for content categorisation (Sullivan, 2002). Although constructing Yahoo!-like taxonomies can help users locate documents of interest, it is expensive and inapplicable for catching up with updated information and scalability with the rapid growth of documents as on the Web. On the other hand, users' search interests are often dynamic and changing over time. The taxonomies generated using document-based approaches might not be adaptable to users' preferences and usages.

In recent years, a great diversity of Web users' search interests has stimulated the development of user-oriented Web IR systems, in which the information is organised and retrieved more from the users' point of view than from system- or document-oriented perspectives (Belew and Van Rijsbergen, 2001). An analysis reported that users' query terms are different from those terms extracted from Web documents (Buckland *et al.*, 1999). The evolution of the Web has provided rich opportunities for gathering and analysing anonymous log data generated by users' interactions with Web IR systems (Lau and Horvitz, 1999). Such data provide a useful resource for studying users' needs and

Refereed article received 20 March 2003
Accepted for publication 16 April 2003

Online Information Review
Volume 27 · Number 4 · 2003 · 243-255

Figure 1 The abstract space of an information retrieval environment



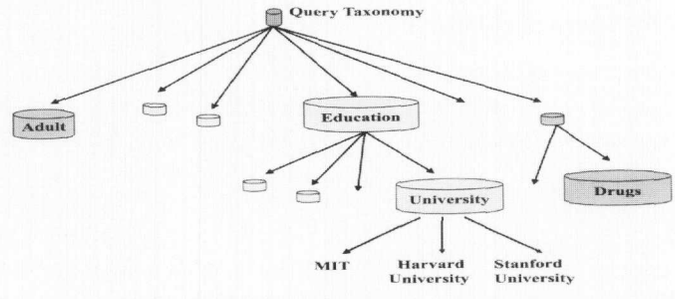
behaviours, and facilitate the design of user-oriented Web IR systems.

For a more user-oriented IR system, it is believed to be equally essential to construct a Yahoo!-like taxonomy, called query taxonomy, for organising users' queries. This can help IR systems understand users' search interests and used vocabularies, help users reformulate more effective queries, and complement the document taxonomies. In fact, users might not be interested in every topic of indexed documents; therefore, the taxonomy used to organise Web documents may not be suitable to organise search vocabularies. The purpose of this paper is to propose a systematic and automatic approach, i.e. query taxonomy generation, to organising queries with their similarity of search interests into a comprehensive hierarchical structure of topic classes. The approach is designed based on mining query logs and search results from Web IR systems for the generation of query taxonomy. The query taxonomy to pursue is defined as a classification tree constituted by the concept hierarchies of users' requests and categorised query terms. An example is illustrated in Figure 2; there, similar queries (as leaf nodes) are grouped to form basic query clusters (as interior nodes such as university), and similar query clusters form super clusters recursively to characterise the associations between the composed clusters. Each of the query clusters can represent a certain concept of users' information requests and some thesaurus information can be obtained within the clusters.

The characteristics of short Web queries and noisy search results have led researchers to investigate the problem of query clustering. Beeferman and Berger (2000) proposed a query-clustering method based on "click-through data", a collection of user transactions with queries and their corresponding clicked URLs, to discover correlations between queries and the clicked pages. Wen *et al.* (2001) developed a similar method that combines the indexed terms of the clicked pages to estimate the similarity between queries, and achieved better performance. However, the number of distinct URLs is often huge in a Web search service. This would cause many similar queries not grouped together because of a lack of common clicked URLs. Moreover, those studies only considered flat, un-nested clustering of queries. In order to have sufficient and effective feature information to characterise the intended search interests of most users' queries, the proposed approach exploits the highly ranked documents retrieved by a query as the source of feature information and designs an effective algorithm for hierarchical query clustering.

As a result of our study, the discovery of topic hierarchies for queries may benefit the design of information retrieval systems in many ways, such as by helping information systems to organise the obtained information in various topic domains, assisting information retrieval systems to organise domain-specific terms into a thesaurus, and enabling question-answering systems to observe users' queries on a larger scale, such as the distributions of the topic

Figure 2 An example of query taxonomy



Online Information Review
Volume 27 · Number 4 · 2003 · 243-255

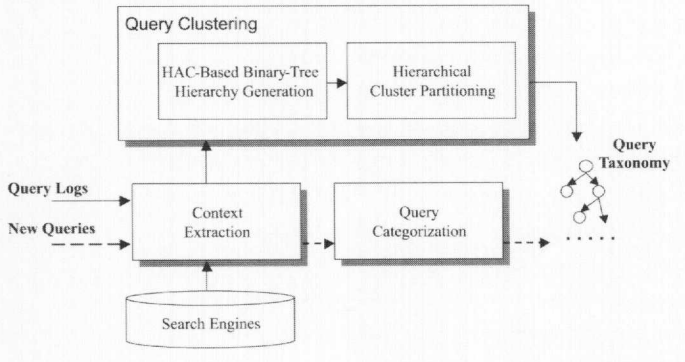
domains and FAQ finding. In the rest of this paper, we first introduce the proposed approach. Then the experiments we conducted and their results are presented. Finally, we discuss some applications and draw conclusions.

Overview of the proposed approach

The approach is based on the mining of search-result snippets from real-world Web search engines to construct the taxonomies for queries. Figure 3 shows the overall concept of the proposed approach, which is composed of three computational modules: context extraction, query clustering, and query categorisation. The query-clustering module is designed to organise queries of different forms, including single terms, Boolean expressions, and natural sentences. To cluster queries with similar information needs, the approach exploits the highly ranked search-result snippets retrieved from online search engines as the feature sources of queries. The context extraction module was, therefore, developed for this purpose.

The proposed query-clustering technique, called HAC + P, is an extension of the hierarchical agglomerative clustering algorithm (HAC). It is combined with a hierarchical cluster-partitioning technique to generate a natural and comprehensive multi-way-tree cluster hierarchy. On the other hand, to enrich the existing taxonomy, the query categorisation module was developed to categorise each newly-arrived query. In the following sections, we will describe the details of each part.

Figure 3 The concept of the proposed approach (abstract)



Context extraction and representation model

Compared with a general document, a query pattern is much shorter and usually contains insufficient feature information in itself for judging its relevance with other target instances in an automatic clustering/categorisation task. Somewhat analogous to determining the sense of a word by means of its context words in linguistic analysis, our approach adopts the query's context information to make up for this insufficiency. The contexts of a query are obtained from highly ranked search-result snippets returned by Web search engines or IR systems, e.g. the titles and descriptions of search-result entries, and texts surrounding matched terms. The features for a query are then extracted from the returned snippets.

Using Web search engines as information sources has disadvantages and advantages. Web contents are usually heterogeneous and noisy, and need careful treatment. However, with the presentation schemes of most search engines, the neighbouring contents surrounding a matched query in Web pages are selectively shown in the returned snippets. Therefore, features are extracted from the corresponding query's contexts instead of the whole Web page. Also a huge amount of pages have been indexed, so most queries can get sufficient results. As a result of recent advances in search technologies, highly ranked documents usually contain documents of interest and can be treated, at least in certain situations, as an approximation of the queries' topic domains. In conclusion, based on a combination of search processes of real-world search engines or IR systems, our clustering hypothesis can be intuitively stated as follows: two queries are clustered together, because they can retrieve similar context contents.

We adopt the vector-space model as our data representation. Suppose that, for each query q, we collect up to $N_{\max}$ search-result entries, denoted as D_q. Each query can then be converted into a bag of feature terms by applying normal text-processing techniques, e.g. removing stop words and stemming, to the contents of D_q. Let T be the feature term vocabulary, and let t_i be the i-th term in T. With simple processing, a query q can be represented as a term vector v_q in a $|T|$-dimensional space, where $v_{q,i}$ is the

Online Information Review
Volume 27 · Number 4 · 2003 · 243-255

weight t_i in v_q. The term weights in this work are determined according to one of the conventional *tf-idf* term-weighting schemes (Salton and Buckley, 1988), in which each term weight $v_{q,i}$ is defined as:

$$v_{q,i} = (1 + \log_2 f_{q,i}) \times \log_2(n/n_i),$$

where $f_{q,i}$ is the frequency t_i occurring in v_q's corresponding feature term bag, n is the total number of queries, and n_i is the number of queries that contain t_i in their corresponding bags of feature terms. The similarity between a pair of queries is computed as the cosine of the angle between the corresponding vectors, i.e.:

$$\text{sim}(v_a, v_b) = \cos(v_a, v_b).$$

For the purpose of illustration, we define the average similarity between two sets of vectors, C_i and C_j, as the average of all pairwise similarities among the vectors in C_i and C_j:

$$\text{sim}_A(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{v_a \in C_i} \text{sum}_{v_b \in C_j} \text{sim}(v_a, v_b).$$

Hierarchical query clustering

The purpose of clustering in our approach is to generate a cluster hierarchy from queries. The hierarchical clustering problem has been studied extensively in the literature, and many different clustering algorithms exist. They are mainly of two major types: agglomerative and divisive. We have adopted HAC as the backbone mechanism and developed a new algorithm, called HAC + P, for our clustering problem (Chuang and Chien, 2002). The algorithm consists of two phases: HAC-based clustering to construct a binary-tree cluster hierarchy and hierarchical cluster partitioning, named min-max partitioning, to generate a natural and comprehensive hierarchy structure from the binary one. The details are introduced as follows.

HAC-based clustering

An HAC algorithm operates on a set of objects with a matrix of inter-object similarities and builds a binary-tree cluster hierarchy in a bottom-up fashion (Mirkin, 1996). At first, each input object is placed in a singleton cluster. Then, at each iteration step, the two most-similar clusters are merged to form a new one, and the whole process halts when there exists only one un-merged cluster, i.e. the root node in the binary-tree hierarchy.

The core of an HAC algorithm is a specific function used to measure the similarity between any pair of clusters C_i and C_j. Here, we consider three well-known inter-cluster similarity functions: (*SL*) the single-linkage function, defined as the largest similarity between two objects in both clusters:

$$\text{sim}_{SL}(C_i, C_j) = \max_{v_a \in C_i, v_b \in C_j} \text{sim}(v_a, v_b);$$

(*CL*) the complete-linkage function, defined as the smallest similarity between two objects in both clusters:

$$\text{sim}_{CL}(C_i, C_j) = \min_{v_a \in C_i, v_b \in C_j} \text{sim}(v_a, v_b);$$

(*AL*) the average-linkage function, defined as the average of all similarities among the objects in both clusters:

$$\text{sim}_{AL}(C_i, C_j) = \text{sim}_A(C_i, C_j).$$

Usually, the clusters produced by the single-linkage method are isolated but not cohesive, and there may be some undesirably elongated clusters. At the other extreme, the complete-linkage method produces cohesive clusters that may not be isolated. The average-linkage method represents a compromise between the two extremes. A comparison of these methods will be made in a later section.

Min-max partitioning

The HAC algorithm produces a binary-tree cluster hierarchy. However, our approach produces a natural and comprehensive hierarchical organisation like that of Yahoo!, in which there are 13-15 major categories, where each category also contains an appropriate number of sub-categories and so on. This broad and shallow multi-way-tree representation, instead of the narrow and deep binary-tree representation, is easier and more suitable for humans to browse, interpret, and perform deeper analysis. To generate a multi-way-tree hierarchy from a binary-tree representation, a top-down approach is used to decompose the hierarchy into several sub-hierarchies first, and then to recursively apply the same decomposing procedure to each sub-hierarchy. Our idea is to determine a suitable level at which to cut the binary-tree hierarchy and create the most appropriate sub-hierarchies; that is, these

Online Information Review
Volume 27 · Number 4 · 2003 · 243-255

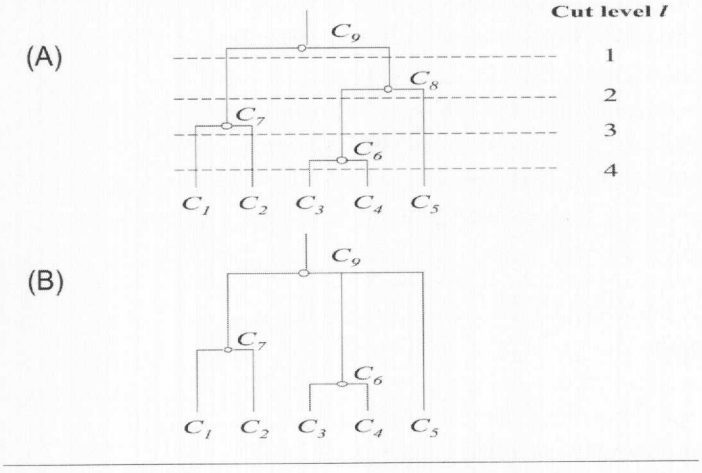
sub-hierarchies are with the best balance of cluster quality and number preference over those produced by cutting at the other levels. Figure 4 demonstrates this idea; Figure 4(A) is the binary-tree hierarchy constructed by an HAC algorithm, and Figure 4(B) shows the resulting hierarchy after level 2 is chosen as the best cut level. Through recursively decomposing the sub-hierarchies, a new multi-way-tree hierarchy can be constructed. In the following, we briefly describe the two criteria used to determine the best cut level.

The generally accepted requirement of "natural" clusters is that they must be cohesive and isolated from the other clusters. Our criterion for determining a proper cut level, given a binary-tree hierarchy of clusters, is to heuristically satisfy this requirement. Let the inter-similarity between two clusters C_i and C_j be defined as the average of all pairwise similarities among the objects in C_i and C_j, i.e. $\mathrm{sim}_A(C_i, C_j)$, and let the intra-similarity within a cluster C_i be defined as the average of all pairwise similarities within C_i, i.e. $\mathrm{sim}_A(C_i, C_j)$. Our partitioning approach finds a particular level that minimises the inter-similarities among the clusters produced at the level and maximises the intra-similarities of all those clusters; this is why the approach is named min-max partitioning. Let C be a set of clusters; our quality measurement of C based on its cohesion and isolation is defined as:

$$Q(C) = \frac{1}{|C|}\sum_{C_i \in C}\frac{\mathrm{sim}_A(C_i, \bar{C}_i)}{\mathrm{sim}_A(C_i, C_i)};$$

where $\bar{C}_i = \bigcup_{k \neq i} C_k$ is the complement of C_i. Note that the smaller the $Q(C)$ value, the better the quality of the given set of clusters, is C.

Figure 4 An illustrative example of cluster partitioning



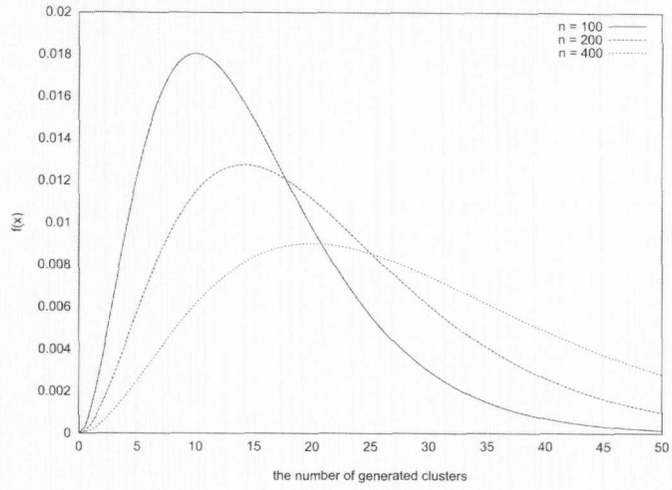
Usually, a partition with neither too few nor too many clusters is preferable. Given n objects, there are at least one cluster and at most n clusters. To have a natural and comprehensive hierarchy structure, we expect that the number of clusters at each level should be appropriate to humans, but a proper number is really hard to anticipate automatically, because we have no idea how many meaningful groups exist among the objects. To make the generated hierarchy structure adaptable to the personal preference of each individual who is going to construct the taxonomy, a parameter N_{clus} on the default expected number of generated clusters at each layer is assumed given by the corresponding taxonomy constructor. Notice that N_{clus} could be a constant value or a function. Then, a simplified gamma distribution function is used to measure the degree of preference on the number of clusters. Its definition is given as follows:

$$f(x) = \frac{1}{\alpha!\beta^{\alpha}}x^{\alpha-1}e^{-x/\beta}; N(C) = f(|C|),$$

where $|C|$ is the number of target clusters in C, α is a positive integer, and the constraint $(\alpha - 1)\beta = N_{\mathrm{clus}}$ is required to ensure that $f(N_{\mathrm{clus}}) \geq f(x), 0 < x \leq n$. The detailed proof is omitted here. The two parameters α and β allow us to tune the smoothness of the preference function, and they are empirically set as $\alpha = 3$ and $\beta = N_{\mathrm{clus}}/2$ in our study. In this work, we empirically define N_{clus} as the square root of the number of objects in each partitioning step. Figure 5 depicts the curves of this cluster-number preference function with respect to different numbers of generated clusters, $f(x)$, on different sizes of objects $n = 100$, 200, and 400. Note that the function favours cluster numbers close to N_{clus}.

Finally, we estimate the quality and the cluster-number preference of all possible cluster sets produced at each level. The suitable cut level is chosen as the level l with the minimum $Q(C_l)/N(C_l)$ value, where C_l is the set of clusters produced after cutting the binary-tree hierarchy at level l. To avoid performing the partitioning procedure on a cluster with too few objects or making the result hierarchy too deep, two constants ϵ and ρ are provided to restrict the size of a cluster and the depth of the hierarchy, respectively, to be further processed.

Figure 5 The cluster number preference function on $n = 100$, 200 and 400



In the literature, several criteria for determining the number of clusters have been suggested (Milligan and Cooper, 1985), but they are typically based on predetermined constants, e.g. the number of final clusters or a threshold for similarity measure scores. Relying on predefined constants is harmful in practice when applying the clustering algorithm, because these criteria are very sensitive to the data set and hard to properly determine. There is another branch of clustering methods, called model-based clustering (Vaithyanathan and Dom, 2000), which has the advantages of offering a way to estimate the number of groups present in the data. However, they suffer from high computational cost and the risk of a wrong assumption of the data model.

Our approach combines an objective function to measure the quality of generated clusters with a heuristically acceptable preference function on the number of clusters. It can automatically determine a reasonable cluster number based on the given data set and still keep the advantage of efficiency of non-parametric clustering methods.

Cluster naming

It is not easy to determine a name for a cluster; there are a variety of methods. In the current stage of our research, we take the most-frequent co-occurred feature terms from the composed instances to name the cluster.

Query categorisation

Query clustering builds a query taxonomy from scratch. To enrich the taxonomy and keep it fresh, an approach to placing newly-arrived queries into the existent taxonomy is demanded; this greatly reduces the heavy load of reconstructing the whole taxonomy (Chuang and Chien, 2003). Suppose plenty of instances have been assigned to appropriate clusters of the hierarchy via either clustering or categorisation processes in an earlier stage. Given a new query q, query categorisation is to determine a set of clusters that are considered as q's related clusters.

With the same scenario described previously, the candidate query q is represented as a feature vector v_q. Here we consider two approaches to this categorisation task.

Mean

Each cluster C_i is represented as its centroid c_i whose j-th feature weight is defined as the average of all its contained instances' j-th feature weights. If a cluster is an interior cluster, its contained instances are those instances contained in itself and its composed clusters. The relevance score between v_q and C_i is defined as:

$$r_{\text{Mean}}(q, C_i) = \text{sim}(v_q, c_i).$$

kNN

kNN has been an effective classification approach to a broad range of pattern

Online Information Review
Volume 27 · Number 4 · 2003 · 243-255

recognition and text classification problems (Dasarathy, 1991). By the kNN approach, a relevance score between q and the candidate cluster C_i is assigned by the following:

$$r_{\mathrm{kNN}}(q, C_i) = \sum_{v_j \in R_k(q) \cap C_i} \mathrm{sim}(v_q, v_j),$$

where $R_k(q)$ are q's k most-similar instances, measured by sim function, in the whole collection.

With the above two approaches, the clusters assigned to q are determined by either a predefined number of most-relevant clusters or a threshold to pick those clusters with scores higher than the specified threshold value. Different threshold strategies have advantages and disadvantages (Yang, 2001). In this study, for the purpose of performance evaluation only, we select the five most relevant clusters as candidates.

Experiment

To assess the performance of the proposed approach, two sets of experiments were conducted. One was conducted to test the feasibility and the performance of query clustering under various similarity measure strategies, and the other was conducted to examine the accuracy of query categorisation.

To have a standard basis for performance evaluation, we used the Yahoo! directory as our major experimental data set. The category names in the top three levels of Yahoo! Computer Science directory were collected. There were 36, 177, and 278 category names in the first, second, and third levels, respectively. These category names were short in length and specifically expressed some subject domain requests; therefore, they could play the role of queries that users might submit. Notice that some category names could be placed in multiple categories; i.e. they have multi-class information.

Besides the Yahoo! directory, we also collected a data set consisting of the academic paper titles from several named conferences. For people in academic research communities, papers are a major search goal, and their titles or author names are often submitted to search engines as query terms. Table I lists the information of the paper data set we collected. Every conference was assigned to the Yahoo! category to which the conference was considered to belong, e.g. AAAI'02 was assigned to CS:AI, and all the papers from that conference unconditionally belonged to that category.

We adopted the F-measure (Larsen and Aone, 1999) as the evaluation metric for the generated cluster hierarchy. The F-measure of cluster j with respect to class i is defined as:

$$F_{i,j} = \frac{2R_{i,j}P_{i,j}}{R_{i,j} + P_{i,j}}$$

where $R_{i,j}$ and $P_{i,j}$ are recall and precision, and are defined as $n_{i,j}/n_i$ and $n_{i,j}/n_j$, respectively, where $n_{i,j}$ is the number of members of class i in cluster j, n_j is the number of members in cluster j, and n_i is the number of members of class i. For the entire cluster hierarchy, the F-measure of any class is the maximum value it attains at any node in the tree, and an overall F-measure is computed by taking the weighted average of all the values of the F-measure as follows:

$$F = \sum_i \frac{n_i}{n} \max\{F_{i,j}\},$$

where the maximum is taken over all clusters at all levels, n is the total number of instances, and n_i is the number of instances in class i.

Table II shows the resulting F-measure values for clustering the 177 second-level category names with the 36 first-level categories as the target classes using various clustering methods and various similarity-measuring strategies. The F-measure values of the binary-tree hierarchies produced by the conventional HAC are provided as the upper-bound reference values for the other HAC variants. HAC+P is HAC with the partitioning procedure, and $\rho = 2$ means the depth of the result hierarchy was constrained to be at most 2. A k-means method was modified to make it hierarchical (HKMeans) and used for comparison. Using HKMeans, all the instances were first clustered into k clusters using k-means, and the same procedure was recursively applied to each cluster until the specified depth ρ was reached. The similarity between an instance and a cluster in k-means was measured using not only the conventional centroid method but also the average- and complete-linkage methods described previously. The single-linkage method was not suitable for incorporation into k means, because, in the single-linkage similarity measure, one instance's most-similar cluster is the one containing the instance itself; therefore, the resulting clusters totally depend

Table I The information of the paper data set

Conference	No. of papers	Assigned category
AAAI'02	29	CS: artificial intelligence
ACL'02	65	CS: linguistics
JCDL'02	69	CS: library and information science
SIGCOMM'02	25	CS: networks

Table II The *F*-measure results of query clustering

	HAC	HAC + P	HAC + P ($\rho = 2$)	HKMeans ($\rho = 2$,$k = \sqrt{n}$)
AL	0.8626	0.8385	0.8197	0.5873
CL	0.8324	0.8116	0.7838	0.2892
SL	0.6848	0.6529	0.2716	N/A
Centroid	0.4177	0.4080	0.2389	0.5873

on the random selection of k initial clusters. In our experiment, the parameter k was automatically set to be the nearest integer of the square root of n, where n is the number of instances to be clustered in each step. The results obtained using HKMeans with complete-linkage were poor because the specified maximum number of iterations was used up and did not converge to a set of stable clusters. Overall, HKMeans did not perform very well in this task.

From the experimental results shown in Table II, we found that the average- and complete-linkage methods performed much better than the single-linkage and centroid ones under the *F*-measure metric, and that the average-linkage method was even slightly better. To provide readers with a more comprehensive and clearer clustering result, Figure 6 shows some selected clusters generated using HAC + P with the average-linkage measure, where each table contains a generated first-level cluster and each row contains a second-level sub-cluster. For each cluster, the Yahoo!'s corresponding category name (notated by users) and the achieved recall/precision rates are presented in the headline row, and the clustered terms are attached with their correct class names. It is easy to see that the terms grouped in the same clusters are most highly relevant. Although there are some terms whose corresponding classes are not matched with generated clusters, their meanings are semantically related to the automatically assigned clusters.

Let us turn now to the generated hierarchy structure. In order to have a clearer structure information of the generated cluster hierarchy, Table III shows some statistics of

Figure 6 Selected examples for clustering Yahoo!'s category names

CS:Linguistics (II) (recall=.23,precision=1)	
Aphasia	CS:Linguistics
Speechreading	CS:Linguistics
Slang	CS:Linguistics
Etymology	CS:Linguistics
Words and Wordplay	CS:Linguistics

CS:Linguistics (I) (recall=.77,precision=1)	
Metaphor	CS:Linguistics
Philosophy of Language	CS:Linguistics
Semiotics	CS:Linguistics
Linguists	CS:Linguistics
Writing Systems	CS:Linguistics
Language Policy	CS:Linguistics
Languages	CS:Linguistics
Lexicography	CS:Linguistics
Translation and Interpretation	CS:Linguistics
Phonetics and Phonology	CS:Linguistics
Dialectology	CS:Linguistics
Sociolinguistics	CS:Linguistics
Computational Linguistics	CS:Linguistics
Natural Language Processing	CS:AI, CS:Linguistics
Language Acquisition	CS:Linguistics
Psycholinguistics	CS:Linguistics

CS:MobileComputing (recall=.57,precision=.80)	
Tablet Computers and Webpads	CS:MobileComputing
Laptop Computers	CS:MobileComputing
Personal Digital Assistants (PDAs)	CS:MobileComputing
Personal Area Networks	CS:Networks
Wearable Computing	CS:MobileComputing

CS:Security (recall=.70,precision=.93)	
Kerberos	CS:Security
S/KEY	CS:Security
Privacy	CS:Security
Cryptography	CS:Security
Encryption Policy	CS:Security
PGP - Pretty Good Privacy	CS:Security
Digital Signatures	CS:Security
RSA	CS:Security
Anonymous Mailers	CS:Networks, CS:Security
Email	CS:Networks
Encrypted Email Providers	CS:Security
Firewalls	CS:Security
Viruses and Worms	CS:Security
Hacking	CS:Security
Software Piracy	CS:Security

CS:AI (recall=.67,precision=.60)	
Turing Machines	CS:Modeling
Turing Test	CS:AI
Higher-Order Logic Theorem Provers	CS:FormalMethods
Set Theory	CS:LogicProgramming
Simulated Annealing	CS:Algorithms
Information Retrieval	CS:Lib&InfoSci
Inference	CS:AI
Ontology	CS:AI
Artificial Life	CS:AI
Cellular Automata	CS:AI, CS:Modeling
Expert Systems	CS:AI
Machine Learning	CS:AI
Neural Networks	CS:AI, CS:NeuralNetworks
Fuzzy Logic	CS:AI
Genetic Algorithms	CS:Algorithms

Table III The information of the generated hierarchy structures

AL	HAC	HAC + P	HAC + P ($\rho = 2$)
Depth	17	6	2
Number of clusters at all levels	352	278	76
Average number of child clusters	2	2.70	5.43

the hierarchy structures generated by clustering Yahoo! CS category names using the average-linkage similarity measure. The reported information includes the depth of the hierarchy, the total number of clusters generated at all levels, and the average of the child cluster numbers among all the clusters, which is a metric that can be used to examine a hierarchy's degree of broadness. Obviously, the decrement of the *F*-measure value caused by partitioning and the constrained hierarchy depth was very small (refer to Table II), but the generated structures were considered more natural and helpful in observing the facts contained. To provide readers with a more comprehensive resulting structure, Figure 7 shows the generated binary-tree hierarchy structure of the instances in Figure 6 using the conventional HAC algorithm with the average-linkage measure, and Figure 8 shows the corresponding multi-way-tree hierarchy generated using HAC + P with two-level depth constraint. Readers can compare the two structures. Although it is hard to have a quantitative approach to measure the worth of a generated hierarchy structure, we believe that the multi-way-tree representation is more natural and easier for humans to browse and interpret than a deep and narrow binary-tree hierarchy.

To examine the performance of our approach on real-world queries, we prepared two testing sets from a three-month query log collected from an online search engine: the 1,000 most frequent queries (HF) and 1,000 randomly selected queries (RD). This query log contains 228,409 distinct queries with total frequency 2,182,960 and has been used in analysis of Web search behaviour and several research studies (Pu *et al.*, 2002). The testing queries have been manually categorised into a two-level taxonomy composed of 14 first-level and 100 second-level categories. Table IV lists some statistics on the hierarchies generated from clustering these two testing sets with the

Figure 7 Example binary-tree hierarchy

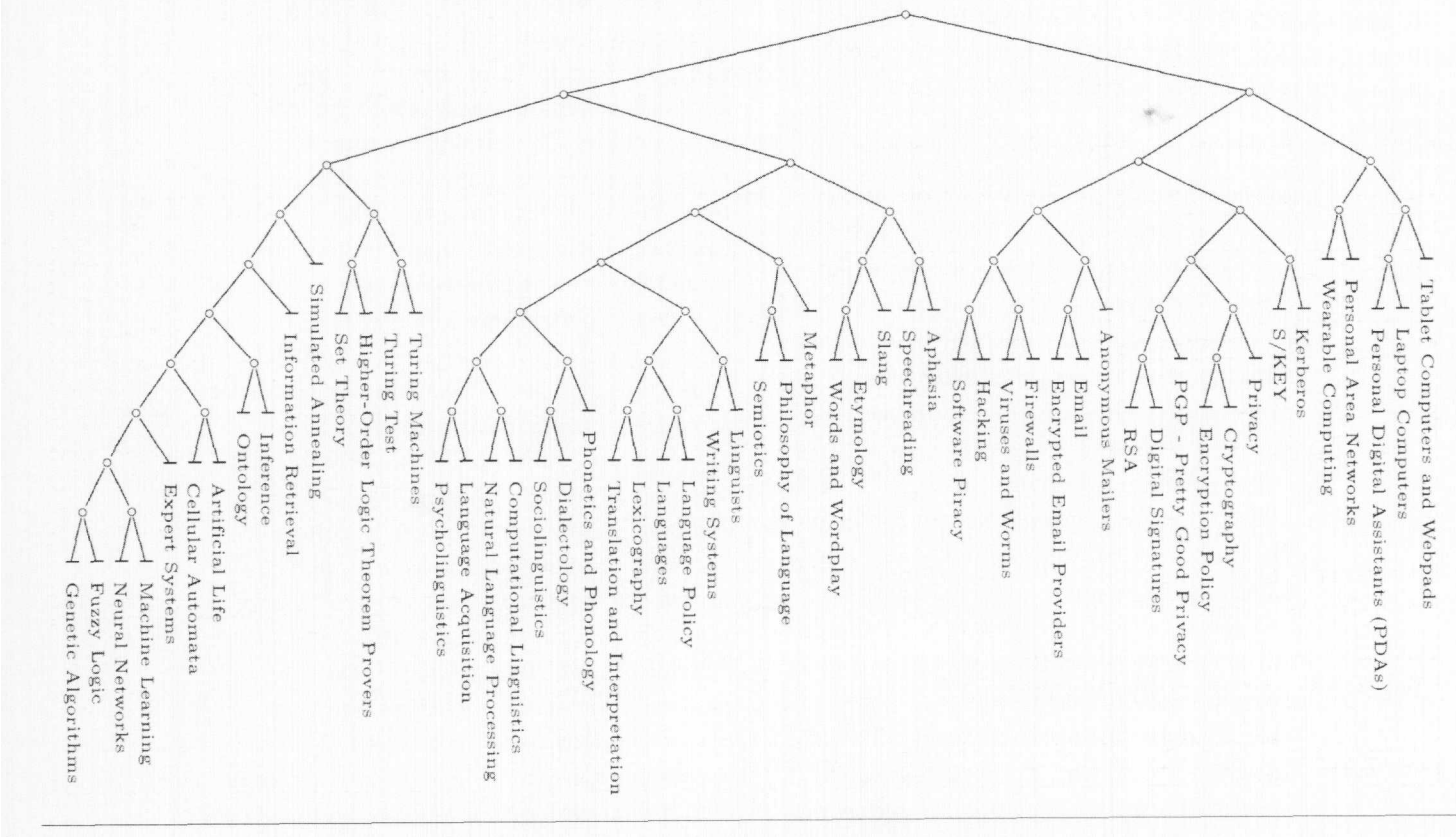


Figure 8 Example multi-way-tree hierarchy

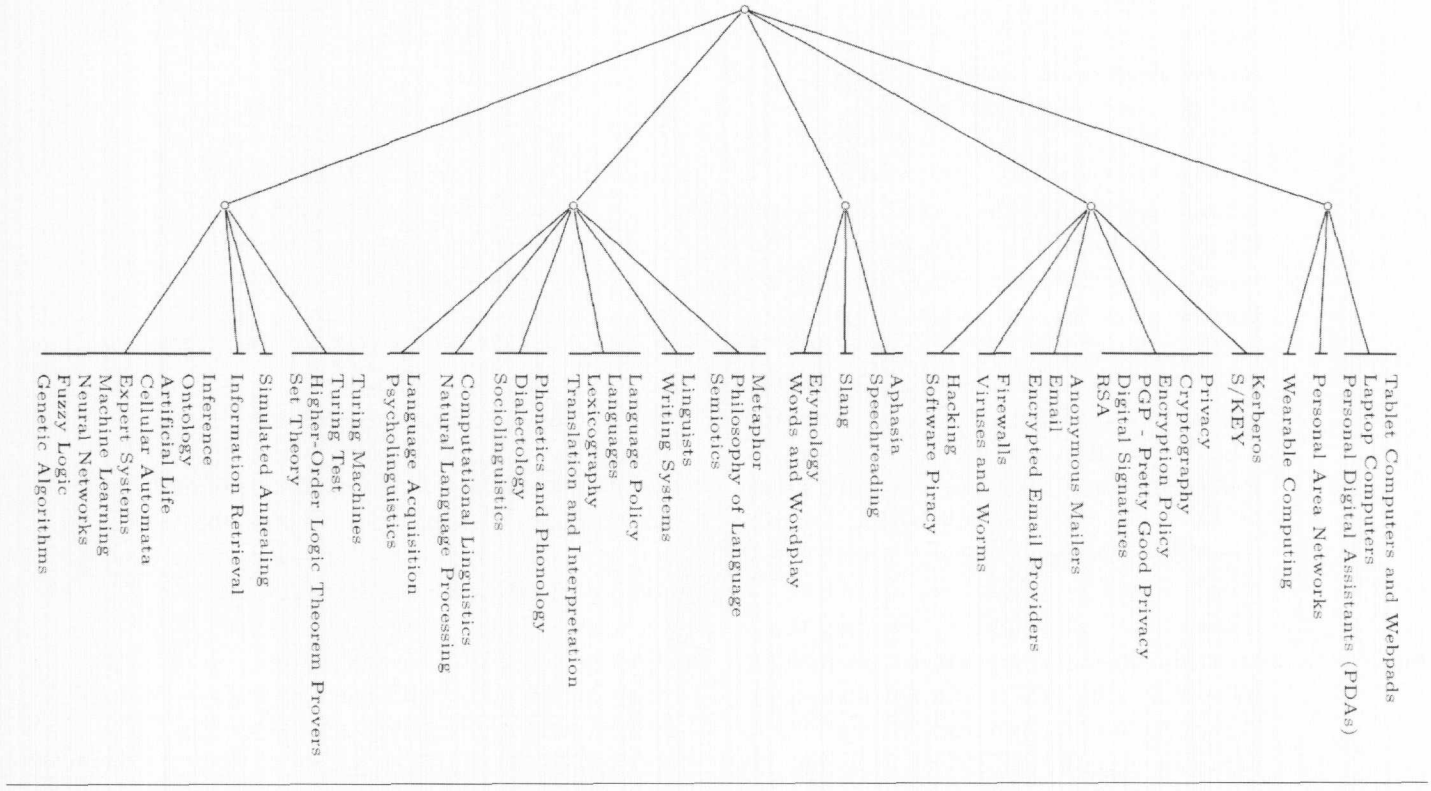


Table IV Results of hierarchical structure generation on real-world queries

	HF	RD
Hierarchy depth	6	5
1st-level clusters	34	35
2nd-level clusters	164	132
3rd-level clusters	206	216
4th-level clusters	79	93
5th-level clusters	25	40
6th-level clusters	14	
F-measure	0.6048	0.5532

average-linkage similarity measure and no depth constraint. Notice that this experiment is on a comparably larger scale than that of Yahoo!'s category names, and the achieved *F*-measure values are considered very good, although they are less than later values. The promising result shows our approach also performs well with real-world data.

Our second experiment was performed to examine the accuracy of query categorisation. We used the first- and second-level category names as the training instances with the first-level categories of Yahoo!'s CS directory as the target classes. Both of the third-level category names and the collected paper titles were used for testing, respectively. Table V lists the resulting top 1-5 inclusion rates, where top n inclusion rate is defined as the rate of instances whose highly ranked n candidate classes contain the correct class(es). The performance of categorising paper titles was comparably poorer than the other, although the achieved accuracy rates were still considered very good. This was mainly because paper titles were specific long queries, and some of them could not get sufficient search-result snippets using current keyword-matching-based search engines. To have a clearer idea, among the total 188 paper titles, there were 61 paper titles obtaining less than 20 search-result entries. This suggests the necessity of further exploring techniques to handle those specific long queries that did not have sufficient and effective feature information.

In addition, Table VI lists the categorisation accuracy of paper titles with respect to different conference categories. Notice that the topic domains of papers from some conferences are concentrative but those from the other conferences are diverse, and this situation is reflected in the performance of categorisation on the papers of each individual conference. Figure 9 lists the categorisation results of several selected mis-categorised paper titles. Although these papers are considered mis-categorised, they seem more appropriate to be placed in those

Table V Top 1-5 inclusion rates for query categorisation

	Top 1	2	3	4	5
Yahoo!					
Mean	0.7219	0.8609	0.8940	0.9073	0.9305
kNN	0.7185	0.8841	0.9238	0.9437	0.9636
paper					
Mean	0.4309	0.6330	0.7128	0.7872	0.8085
kNN	0.4628	0.6277	0.7181	0.7713	0.8085

Table VI Top 1-5 inclusion rates of paper titles with respect to different conferences

	Top 1	2	3	4	5
Conference					
AAAI'02	0.7333	0.8000	0.8000	0.8000	0.8000
ACL'02	0.5152	0.7273	0.8030	0.8182	0.8485
JCDL'02	0.1143	0.3000	0.4714	0.6000	0.6714
SIGCOMM'02	0.9200	1.0000	1.0000	1.0000	1.0000

Figure 9 Selected mis-categorised examples for categorising paper titles

Paper title	Conference	Target Cat.	Top-1	2	3	4	5
A New Algorithm for Optimal Bin Packing	AAAI	AI	ALG	AI	MOD	COLT	DNA
(Im)possibility of Safe Exchange Mechanism Design	AAAI	AI	NET	SC	LG	DB	MD
Performance Issues and Error Analysis in an Open-Domain Question Answering System	ACL	LG	AI	LG	ALG	DC	SC
Active Learning for Statistical Natural Language Parsing	ACL	LG	AI	LG	NN	COLT	ALG
Improving Machine Learning Approaches to Coreference Resolution	ACL	LG	AI	LG	ALG	FM	NN
A language modelling approach to relevance profiling for document browsing	JCDL	LIS	AI	UI	LG	LIS	ALG
Structuring keyword-based queries for web databases	JCDL	LIS	AI	LIS	DB	ALG	ARC
A multilingual, multimodal digital video library system	JCDL	LIS	LG	UI	LIS	ECAD	NET
SOS: Secure Overlay Services	SIGCOMM	NET	SC	NET	MC	OS	DC

Abbreviation List: AI :Artificial Intelligence; ALG :Algorithms; ARC :Architecture; COLT:Computational Learning Theory; DB :Databases; DC :Distributed Computing; DNA :DNA-Based Computing; ECAD:Electronic Computer Aided Design; FM :Formal Methods; LG :Linguistics; LIS :Library and Information Science; MC :Mobile Computing; MOD:Modeling; NET :Networks; NN :Neural Network; OS :Operating Systems; SC :Security; UI :User Interface

highly ranked auto-assigned categories. It is worthy of notice that the promising accuracy also shows its great potential of the proposed approach to classifying paper titles with Yahoo!'s subject directories.

Further discussion

A core of the proposed approach is using Web search results returned by real-world search engines as the information source for the target queries. This gives us several advantages.

First, huge amounts of available documents on the Web have been indexed; this increases the chances of getting sufficient and balanced context information to characterise a query. Second, the Web, whose content is always updating and enlarging, offers the most up-to-date information. Third, we do not have to prepare the information of queries laboriously; existing search engines reduce our load.

However, although using Web search results gives sufficient and effective information for each query, it also entails several weaknesses. The required Web access is obviously a heavy load of the proposed approach, requiring a certain amount of bandwidth. Web contents are usually considered heterogeneous and noisy, and need careful treatment. Moreover, term-vector-based data representation makes our clustering problem a data-sparseness problem with high dimensional feature space. Some noisy features may make the similarity measure not so reliable. This is why HAC performs better than divisive HKMeans, because HAC groups those similar instances, e.g. synonyms or related instances, at very early stages. This is also why the average- and complete-linkage methods, which tend to produce cohesive clusters, perform better.

Online Information Review
Volume 27 · Number 4 · 2003 · 243-255

Another challenge to this clustering problem is the ambiguous nature of queries. Some query terms could be placed in multiple categories; e.g. "natural language processing" is in both AI and linguistics categories. The proposed approach is a hard-clustering algorithm; there, one instance would be placed in only one cluster. Clustering queries with multiple topic domains into multiple clusters is still unexplored. One possible approach is to apply query categorisation to those clustered queries at a later stage to associate them with other related clusters. On the other hand, using highly ranked Web search-result snippets may limit our approach in dealing with those queries with several meanings in cases where those highly ranked snippets contain too little or even no relevant information to be certain of the meanings.

Discussions of possible applications

Our query taxonomy generation approach for real-world queries may be useful in several applications listed below.

Thesaurus construction

A query taxonomy serves as a first step toward the construction of a thesaurus for search purposes. For those queries categorised in the same topic classes, more accurate relationships such as abbreviations, synonyms, related terms, broader terms, narrow terms, and translations might be further discovered. For example, we can apply some similarity estimation techniques to queries in the same class to extract highly related query pairs. This reduces the size of the query set without the disturbance of other irrelevant queries. Also, our work provides a good beginning for constructing a thesaurus that can adapt to users' changing search interests. The vocabulary of query taxonomy can be incrementally updated with the appearance of new queries. Some queries not used for a certain period of time can be removed from the thesaurus. The structure of the taxonomy can be reconstructed after a period of time of service.

FAQ finding and query filtering

It is important to discover common interests among users and to exploit the experience of previous users to help others. The proposed approach can be used to cluster natural language queries in some call-routing applications. The discovered clusters of queries can assist users in reformulating refined queries, discovering other users' FAQs for question-answering systems, performing query expansion and term suggestion, and even constructing a query filter for filtering out sensitive queries, such as pornographic-related terms.

Observation of users' search interests

Knowing more about the information needs of users is valuable for enhancing the effectiveness of information systems and content organisation. With our approach, it is a straightforward matter to construct a system for information systems in order to monitor users' changing search requests, including the distributions of users' search topic classes, and up-to-date frequencies of queries in each class. The proposed approach has been shown in Pu *et al.* (2002) also to be helpful in observing trends of search interests.

Conclusion

This paper has proposed a systematic and automatic Web-mining approach to organising queries into a hierarchical structure of topic classes. Although the hierarchical clustering of queries is generally considered very difficult, the Web provides an alternative way to deal with this problem. With huge amounts of available documents on the Web being indexed by real-world search engines, most queries can get sufficient and effective topic-relevant contextual information. Hence, the approach is designed to combine with the search processes of real-world search engines to extract features from the retrieved highly ranked search-result snippets for each query. Also, a clustering algorithm for generating a natural multi-way-tree cluster hierarchy is developed; the algorithm is a hierarchical agglomerative clustering algorithm, which generates a binary-tree hierarchy at first, followed by a hierarchical cluster-partitioning technique to generate a multi-way-tree hierarchy. The partitioning technique is designed based on a min-max objective function to measure the quality of clusters, and is combined with a heuristically acceptable preference function on the number of clusters to ensure that the produced hierarchy is natural for humans.

Query categorisation is also proposed to enrich the existing taxonomy and keep it fresh while reducing the heavy load of reconstructing the whole taxonomy from scratch. The results of the preliminary experiment on subject terms and paper titles in the domain of computer science have shown the potential of our approach. It is also believed that the proposed approach can benefit various Web information retrieval applications.

References

Beeferman, D. and Berger, A. (2000), "Agglomerative clustering of a search engine query log", *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM Press, New York, NY, pp. 407-16.

Belew, R.K. and Van Rijsbergen, C.J. (2001), *Finding out about: A Cognitive Perspective on Search Engine Technology and the WWW*, Cambridge University Press, Cambridge.

Buckland, M., Chen, A., Chen, H-M., Kim, Y., Lam, B., Larson, R., Norgard, B., Purat, J. and Gey, F. (1999), "Mapping entry vocabulary to unfamiliar metadata vocabularies", *D-Lib Magazine*, Vol. 5 No. 1, available at: www.dlib.org/dlib/january99/buckland/01buckland.html

Chuang, S.-L. and Chien, L.-F. (2002), "Towards automatic generation of query taxonomy: a hierarchical query clustering approach", *Proceedings of the 2002 IEEE International Conference on Data Mining*, IEEE Computer Society Press, Los Alamitos, CA, pp. 75-82.

Chuang, S.-L. and Chien, L.-F. (2003), "Enriching Web taxonomies through subject categorisation of query-terms from search engine logs", *Decision Support Systems*, special issue on Web retrieval and mining, Vol. 35 No. 1, pp. 113-27.

Dasarathy, B.V. (1991), *Nearest Neighbour (NN) Norms: NN Pattern Classification Techniques*, McGraw-Hill Computer Science, IEEE Computer Society Press, Los Alamitos, CA.

Larsen, B. and Aone, C. (1999), "Fast and effective text mining using linear-time document clustering", *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM Press, New York, NY, pp. 16-22.

Lau, T. and Horvitz, E. (1999), "Patterns of search: analysing and modelling Web query refinement", *Proceedings of the Seventh International Conference on User Modelling*, Banff, pp. 119-28.

Milligan, G.W. and Cooper, M.C. (1985), "An examination of procedures for detecting the number of clusters in a data set", *Psychometrika*, Vol. 50, pp. 159-79.

Mirkin, B. (1996), *Mathematical Classification and Clustering*, Kluwer, Dordrecht.

Pu, H.-T., Chuang, S.-L. and Yang, C. (2002), "Subject categorisation of query terms for exploring Web users' search interests", *Journal of the American Society for Information Science and Technology*, Vol. 53 No. 8, pp. 617-30.

Salton, G. and Buckley, C. (1988), "Term-weighting approaches in automatic text retrieval", *Information Processing and Management*, Vol. 24, pp. 513-23.

Sullivan, D. (2002), "Document warehousing and content management: poor search quality in your enterprise information portal?", *DM Review*, available at: www.dmreview.com/master.cfm?NavID=55&EdID=4480

Vaithyanathan, S. and Dom, B. (2000), "Model-based hierarchical clustering", *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence (UAI-2000)*, Stanford, CA, pp. 599-608.

Wen, J.-R., Nie, J.-Y. and Zhang, H.-J. (2001), "Clustering user queries of a search engine", *Proceedings of the 10th International World Wide Web Conference*, Hong Kong, pp. 162-8.

Yang, Y. (2001), "A study on thresholding strategies for text categorisation", *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM Press, New York, NY, pp. 137-45.