Computer Networks and ISDN Systems 29 (1997) 1217–1224

COMPUTER NETWORKS and ISDN SYSTEMS

Customizable multi-engine search tool with clustering

Chia-Hui Chang *,[1], Ching-Chi Hsu [1]

Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan

Abstract

The dozens of existing search tools and the keyword-based search model have become the main issues of accessing the ever growing WWW. Various ranking algorithms, which are used to evaluate the relevance of documents to the query, have turn out to be impractical. This is because the information given by the user is too few to give good estimation. In this paper, we propose a new idea of searching under the multi-engine search architecture to overcome the problems. These include clustering of the search results and extraction of co-occurrence keywords which with the user's feedback better refines the query in the searching process. Besides, our system also provides the construction of the concept space to gradually customize the search tool to fit the usage for the user at the same time. © 1997 Published by Elsevier Science B.V.

Keywords: WWW; Information extraction; Clustering; Customizable; User feedback

1. Introduction

As it has been arguing that the standard Web search services are far from ideal, many researchers are seeking for a better way to tackle the ever growing WWW. Technologies include software agents, multi-engine search, and distributed cooperating, etc. The intelligent information retrieval agents expect to extract concerning information for users by building an agent theory with features like autonomous, self-learning, planning, customizable, and communication [6,8]. And the multi-engine search utilize the various search engines with the consideration that none of the search services is broad enough to cover the whole World Wide Web [13,4]. Also, research into the distribution of indexing load as in Harvest system [2] and information retrieval protocols and standards like Z39.50 [9] also try to share the indexed data in a tightly-organized corporation among servers.

Each of the applications has its speciality to make up the two main insufficiencies of current search tools.

- First, many search tools rely on robots to discover information published on-line, but the contents of WWW are too massive to be indexed in a single server. Even if we do so, the single server will be overloaded to serve queries from all over the world, and the network delay and traffic will increase remarkably.
- Second, the inverted index database used by most of the current search tools, though efficient and appropriate for keyword-based search model, is not enough to measure the relevance of documents to a specific query for users. This is because most users have little knowledge about the indexed keywords in the database and even the

* Corresponding author.
[1] E-mail: {chia.CCHsu}@ails1.csie.ntu.edu.tw

PII S0169-7552(97)00053-6

domain they are searching. Thus, little information is provided in the query sentences and with the few information given by users, the relevant evaluation may lose their effects.

To solve the first problem, we advocate the deployment of specialized search engines based on geographic locations to distribute the indexing load and the construction of an integrated gateway to these databases are used to provide a way of integration (and possibly incorporation). While the idea is not new, we can see that with more and more searchable tools getting online and the low overlapping of the search results from existing search tools, we do benefit from the adoption of mult-engine search architecture. And with experience from MetaCrawler [13] and SavvySearch [4], the reasonable response shows the possibility of actual use in practice. More than that, releasing ourselves from the burden of indexing the web, we can then focus on the post-processing of the retrieved data of the second problem.

Comparing to the first problem, researches that deal with the second problem are not as promising as the first one. Most search tools still use keywords to specify queries. However, according to Pinkerton's experiment [10], the average number of keywords specified in each query is only 1.5, which definitely leads to thousands of "related" documents. Since some keywords are manifold (e.g., "Java" as a computer language and as a kind of coffee), irrelevant documents are included in the search results and reduce the precision. This situation is further worsen for those keywords that encompass too broad of the search space (e.g., "computer"). Though various ranking algorithms are used to evaluate the relevance of documents, it turns out to be meaningless. To address this problem, category hierarchy of the database of the search engine are provided to narrow down the search space (as in Yahoo and Infoseek). However they demand a lot of man force in the categorization and users are easily confused through the big classification. In fact, the lack of a scalable, customizable references to provide synonyms or related topics for users become the main issue of the problem.

In this paper, we present a new search process to overcome the problems mentioned above: a provision of related words through the extraction of search result, the feedback of users interests to refine the query and a classification of the search results. The insight is that the information about the query that are not specified in the user's input may be embedded in the results of the first query. If we can extract related keywords regarding this query, users can then easily identify the domains relating to their searching and give more information as the searching going on. On the other hand, instead of having users browsing through each singular hypertext, classifying the search results into clusters gives first impression about the hypertexts. This process spotlights on the filtering and grouping of data and saves users from the work of infinite browsing. Therefore, through the feedback of the user in the searching process, a corresponding concept space to the user's knowledge domain is constructed. And the information is reused for related queries in the later process.

2. System architecture

The implementation of the system is based on the multi-engine search architecture. Under this architecture, the system receives a user's query as a list of keywords and redirect the query to other search engines (which include **AltaVista**[2], **Infoseek**[3], **Lycos**[4], **WebCrawler**[5], **Magellan**[6], etc.). While collecting data from the internet, the system collates the data and presents the results of clustering to inform users of the domain they are searching for. The user can then refine the query to direct the searching on what they want precisely. The complete process of the system can be broken down into following steps, and shown in Fig. 1 for reference:

(1) Collect the object-attribute data matrix.
(2) Compute the resemblance and classify the documents.
(3) Extract related words as reference.
(4) Display the clustering results to the user and get the user's feedback.
(5) Refine the query with user feedback.
(6) Repeat steps 1–5.

[2] http://www.altavista.digital.com
[3] http://ultra.infoseek.com
[4] http://lycos.com
[5] http://webcrawler.com
[6] http://searcher.mckinley.com

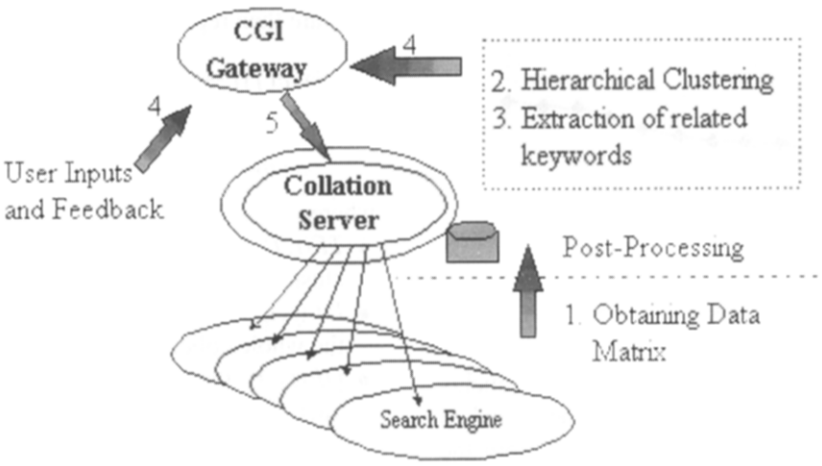

Fig. 1. System architecture.

In the following sections, we will examine each component of the system in detail.

2.1. Obtaining data matrix

For queries new to the system, they are directed to out-side search engines with the display form set to "detail" format (in **AltaVista**[7]) or "summary" mode (in **WebCrawler**[8]). The returned results show where in the internet the related documents exist and the terms that appear in the description are taken as the attributes of the documents. And if the time is available, the original contents of the documents are retrieved from internet. Then with characteristics of hypertexts, words from the page titles, headings, anchor hypertexts, list items, italic and bold face forms are extracted as the representatives of the documents' vectors.

On the other hand, to manipulate various output form of the search engines, the repeat pattern of HTML tags is discovered to find the individual URLs and related information in the search results. For example, when a new search engine is added, the system generate a general query, say "WWW", and examine through the tags of the search results to find the repeat patterns to get the respective URLs. Then, more queries are generated to test the correctness of the pattern and also to find the way to retrieve the subsequent results from the search engine.

[7] http://www.altavista.digital.com
[8] http://webcrawler.com

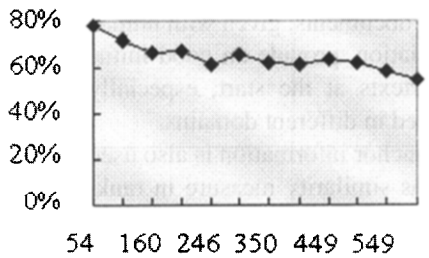

Fig. 2. Percentage of unique locations.

2.2. Initial assignment of documents

In addition to the extraction of keywords from headings or high-lighted texts to be attributes of each document, the hyperlinks in the documents also provide useful information in the clustering. From the experiment, it shows that from an average of 100 URLs, over 30 documents come from the same server (Fig. 2). This is because most of the search engines index at least one to two level deep of an WWW server after the discovery. And since most of the WWW servers focus specifically on some topics, more than one document are usually indexed in the inverted database and retrieved together during the keywords matching. Thus, when someone use search tools to find out where in the internet the related information hides, the option to group documents from the same IP address (server) together will be useful.

Another useful feature of the hypertexts based WWW information browsing are the hyperlinks,

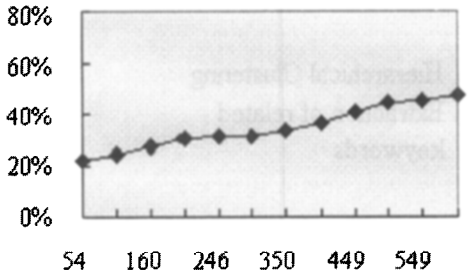

Fig. 3. Percentage of cross-links.

which are intended to be used to link documents that are content-related. In our experiments, when the actual content of an URL is retrieved from the internet, the hyperlinks between the documents (refer to as cross links in the following context) are as high as 20% (Fig. 3). These existing cross links between documents, given with human-identification of association, provide an good initial classification of hypertexts at the start, especially for common terms used in different domains.

The anchor information is also used in WISE [14] system as similarity measure in ranking algorithms as well as grouping of linked documents. In our /system, we use them as initial assignment of hypertexts to clusters. And despite the overlap of the two optional grouping, up to 40% of the documents can be grouped without resemblance computation between documents. The grouped cluster is then represented by terms that appear in the documents of that cluster.

2.3. Resemblance computation

The actual meaning of the data matrix represents the position or vector of each object under the vector model. To compute the resemblance coefficient between two objects, (where objects are represented by a vector of attributes, or terms here), various functions such as the Euclidean distance and the cosine value of the object vectors are the commonly used ones in the clustering literature [3]. And in the field of information retrieval, the $TF \times IDF$, which gives weights to attributes of objects, becomes the mostly used similarity measures. In this paper, we propose an extension of $TF \times IDF$ measure to include learned information from a series of queries. For a new query q, the similarity measure between cluster $c(i)$ and $c(j)$ are then represented as $R(c(i), c(j))$:

$$R(c(i), c(j)) = \sum_{t \in C(i) \cup c(j)} TF(i,t) * TF(j,t) * \left(\frac{DF(t,q)}{QF(t)} \right)$$

where:

- $TF(i,t)$ is the relative frequency of term t in cluster $c(i)$,
- $DF(t,q)$ is the document frequency of term t in the whole collection of query q,
- $QF(t)$ is the frequency of term t appeared in queries so far.

The point is to give higher weights to terms which occur frequently in documents of the query domain, and to those that are unique to other queries. Noted that the weighting of terms $DF(t,q)/QF(t)$ is also adapted to filter related keywords in a query through the users feedback. In which we dynamically re-weight the terms to inflect their significance.

Readers can see here that the resemblance computations are not between the query and documents, such as to give an sorted results of the thousands of documents, but are between documents to give the references for the clustering. The reason is that with average 1.5 query keywords, the ranking of documents to the query can not distinguish well the relevance for the user. On the contrary, resemblance computations between documents with the attached terms do give good indication in the clustering process. And the presentation of classified results shows the advantage of making the query domain more organized.

2.4. Hierarchical clustering with iterative approach

In the practical operation, due to the availability of data from the internet, we need an clustering method differing from the classical ones. [7] With contents coming from the internet on the fly, an iterative approach must be designed to give partial results to the user during the searching process. Given the total number of documents to be got and the number of phase to complete the job (denoted by N and P, respectively), the whole process we construct proceeds as follows: at each phase k, k from 1 to P, once N/P documents are collected, the clustering procedure is activated to accomplish the classification (reference to Fig. 4).

Given: $N, P, r,$
Initialization: $total = 0$; $C(i) = \emptyset$, **for** $i = 0$ **to** P;

for $k = 1$ **to** P **do**
  submit queries to out-side search engines and receive documents;
  collect N/P **documents, then do Clustering;**
  assign each document $d(i)$ **to a proper cluster in** $C(0)$;

Procedure Clustering;
  for $i = 0$ **to** k **do**
    compute the similarity of clusters in $C(i)$;
    group the highest $(|C(i)| * r)$ **groups in** $C(i + 1)$;

Fig. 4. The hierarchical clustering with iterative approach.

At the beginning of the clustering procedure, each new retrieved document is assigned to a proper existing cluster or a new cluster in cluster set $C(0)$ under the principle of initial assignment of documents. Then, the similarity measures between clusters in cluster set $C(i)$ are computed to group clusters with the highest $|C(i)| * r$ similarity measures to construct an hierarchical structure (where r is the grouping ratio discussed below).

There are two points to be noted here. In the processing of iterative clustering, we first want to make sure that each of the documents is compared to others to make sure that the later documents with higher similarity measure are grouped as well. Second, the computation of resemblance between clusters in cluster set $C(i)$ are followed to group related clusters to form cluster set $C(i + 1)$. This corresponds to the computation of similarity between centroids or centers of clusters and give better measures of the resemblance. As for the ratio of grouping, r, it is chosen from 0.4 to 0.6 to give a better division of the whole category. In our experience. two to three major groups are generated to show the main topics of the domain, while others are considered as minor groups, and can not be furthur classified due to the shortage of data attributes.

3. User interface and feedback

The user interface of our system differs from those of others. For each query, the system shows the clustering results and highly weighted terms at each phase as references to the query domain. The user can feedback interested keywords to refine furthur query (Fig. 5). The interface also shows that documents of a cluster is grouped because of some common terms appearing in the documents. This helps the user easily identify what the cluster is about without browsing through all the contents inside.

The feedback of the selected topics also help to customize the knowledge base, since there is a percentage of negligible words, like people names or mistyped words that are not common to other queries and are weighted high in the weighting. But with the user's feedback, we can identify what are the terms to be stressed and what are the ones that are irrelevant to the query through the re-weighting procedure in supervised mode: for each term t in the user's feedback, $DF(t, q)$ is doubled to enforce the significance of the term in the query q. And for each term t not in the feedback, $QF(t)$ is doubled by the same way to reduce its weight. In this way, feedback of the same keywords can be repeated to strengthen their significance.

While people may ask why not use a thesaurus like **Roget's**[9] as references for the user, they can soon discover that the information provided by the thesaurus are more like the synonyms and is not suitable for using here. And the extraction of information from the query results, a kind of post-processing, give better indication.

As for the elimination of mistyped words, the spell tool on UNIX is used to pick out the words not included in the dictionary. With only one exception that words spelled in uppercase are kept to avoid the case of proprietary keywords. Though not all errors

[9] http://www2.thesaurus.com/thesaurus/

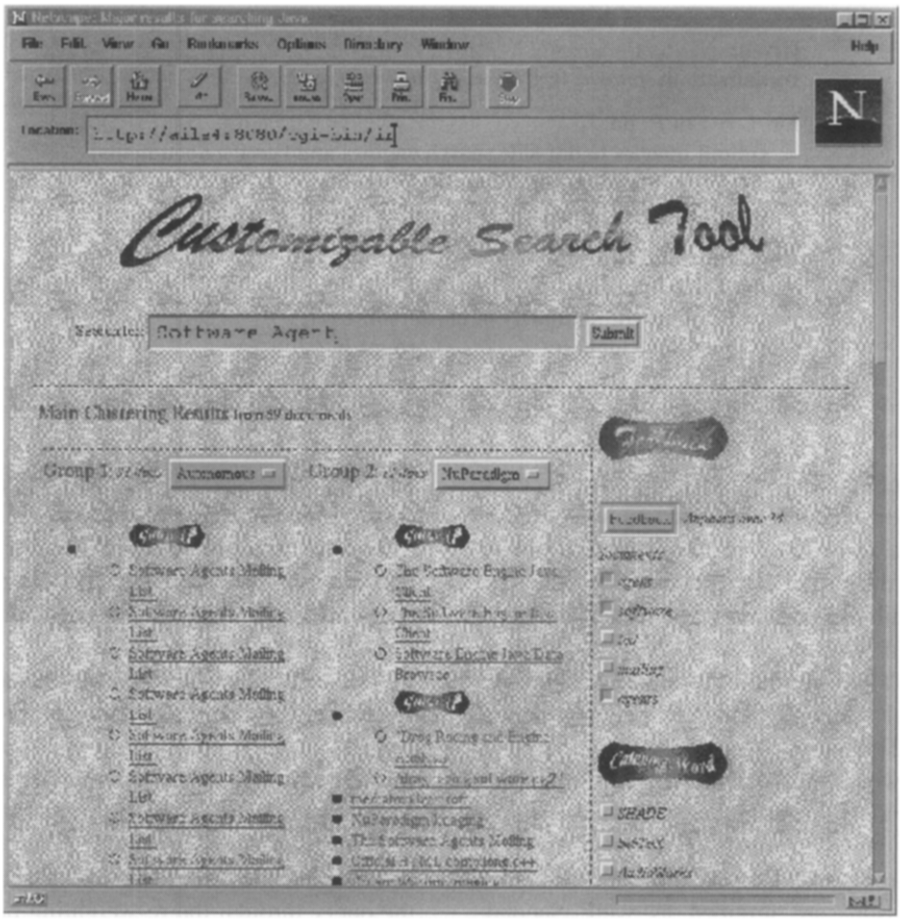

Fig. 5. **The user interface.**[10]

are deleted in this way, most mistyped words can be kicked out in the feedback phase of the user.

3.1. Concept space construction

To customize the search tool to the user, a concept space to the user's knowledge must be constructed through the queries and feedback of the user. The information must be kept in the storage subsystem such that the system can quickly show the related information about the queries that have been examined by the user.

In the current implementation of the system prototype, the storage subsystem is mainly on the recording of terms met in the searching process to form a simple dictionary, and also the related keywords to queries in the feedback. For the first part, the appearance frequency of each term t across queries are remembered as $QF(t)$ and can be updated as discussed above. And for the quick search of a keyword in the query history, each keyword that appears in the queries or feedback is given an identifier. And each query in the history is associated with the identifiers of keywords that appear in the query. Further more, as it has been used in most of the search engines, an inverted keyword database is constructed to accelerate the searching process.

4. Conclusions and future work

In this paper, we have proposed a new search process to overcome the issues in the current WWW

[10] The prototype of our system is also available at http://ails4.csie.ntu.edu.tw:8080/iragent.html

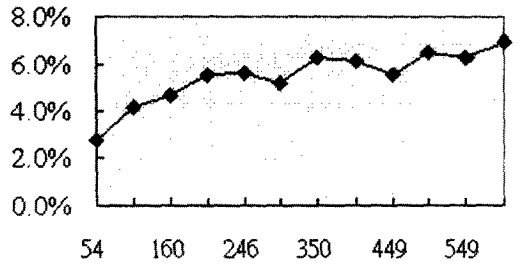

Fig. 6. Overlapping of URLs from 5 search engines: Alta Vista, Infoseek, Lycos, WebCrawler, Magellan.

search tools. This post-processing and clustering idea are the main contribution of our system. In this way, major topics in the query domain are highlighted to illustrate the general outlook of the query. And the construction of the concept space together with the user's feedback will gradually customize the system to fit the usage for the user.

The clustering idea, besides the application in the multi-engine search architecture, can be applied to any singular search tool as well. In fact we have applied it to AltaVista and Magellan respectively. However, the multi-engine search architecture has the advantage of increasing the availability and recall. As we can see from the experiments that the average overlapping of hypertexts from five search databases (i.e. AltaVista, Infoseek Lycos, WebCrawler, Magellan) is below 10% (Fig. 6). And the temporary denial or long delay of one search engine can be easily supplanted by other ones, showing the advantage of this architecture.

The current version of the system is implemented using the standard Common Gateway Interface (CGI) and can be accessed through any WWW browser. However, it is better to say this service working as an personal assistance for the user rather than a service to the public. Since the storage subsystem is trainable with the users feedback, it is best to have a client version of this system. Thus, we are working on an client version of the system as the job working on.

Acknowledgement

This project is partially supported by Institute for Information Industry, Taiwan.

References

[1] C.M. Bowman, P.B. Danzig, D.R. Hardy, M.F. Schwartz, Scalable Internet resource discovery: research problems and approaches, *Comm. of ACM*, Vol. 37(8), pp. 98-107, 1994

[2] C. M. Bowman, P.B. Danzig, D.R. Hardy, U. Manber and M.F. Schwartz, The Harvest information discovery and access system, Techical Report at U. Colorado

[3] R.M. Cormack, A review of classification, *Journal of the Royal Statistical Society, Ser. A*, 134, pp. 321-353

[4] D. Dreilinger, Integrating heterogeneous WWW search engines, May 1995, ftp://132.239.54.5/savvy/report.ps.gz

[5] Learning to understand information on the Internet (IJCAI-95)

[6] O. Etzioni, D. Weld, A Softbot-based interface to the Internet, *Comm. of ACM*, Vol. 37(7), July 1994, pp. 72-76, http://www.cs.washington.edu/research/softbots

[7] C. Faloutsos and D.W. Oard, A survey of information retrieval and filtering methods, University of Maryland, Technical Reports CS-TR-3514

[8] T.W.C. Huibers and B. van Linder, Formalising intelligent information retrieval agents, Tech. Rep., Utrecht University

[9] Q. Kong and J. Gottschalk, Information retrieval standard and applications, presented at BISCM'94, Beijing, http://www.dstc.edu.au/RDU/reports

[10] B. Pinkerton, Finding what people want: Experiences with the WebCrawler, *2nd Int. WWW Conference '94*, July 1994, Chicago, Oct. 1994, http://info.webcrawler.com/bp/WWW94.html

[11] C.J. van Rijsbergen, *Information Retrieval*, Butterworths, London, England, 1979, 2nd ed.

[12] H. C. Romesburg, *Cluster Analysis for Researchers*, Wadsworth, Belmont, CA, 1984

[13] E. Selberg and O. Etzioni, Multi-engine search and comparison using the MetaCrawler, in: *Proc. 4th World Wide Web Conference '95*, Boston, Dec. 1995

[14] B. Yuwono and D.L. Lee, WISE: a World Wide Web resource database system, to appear in *IEEE Trans. on Knowledge and Data Engineering, Special Issue on Digital Library*

Chia-Hui Chang is presently a Ph.D. candidate in Department of Computer Science and Information Engineering at the National Taiwan University. She received the B.S. degree in the same department in 1993.

Her research interests include distributed computing, intelligent agents, digital library, with a focus on information retrieval on WWW. Ms Chang is a student member of the EKE, the EKE Computer Society.

Ching-Chi Hsu was born in Taipei, Taiwan, on January 28, 1949. He received the B.S. degree in physics from National Tsing Hwa University in 1971, Hsishu, Taiwan, and both the M.S. and Ph.D. degrees in Computer Engineering from National Taiwan University, Taipei, Taiwan, in 1975 and 1982, respectively.

In 1977, he joined the faculty of the Department of Computer Science and Information Engineering at National Taiwan University and became an Associate Professor in 1982. Currently he is a professor and the department head of his department. His current research interests include distributed system, distributed processing of data and knowledge, information retrieval on Internet, intelligent systems.