WISE: A World Wide Web Resource Database System

Budi Yuwono and Dik Lun Lee, *Member, IEEE*

Abstract—This paper describes the World Wide Web Index and Search Engine (WISE) for Internet resource discovery. The system is designed around a resource database containing meta-information about WWW resources and is automatically built using an indexer robot, a special WWW client agent. The resource database allows users to search for resources based on keywords, and to learn about potentially relevant resources without having to directly access them. Such capabilities can significantly reduce the amount of time that a user needs to spend in order to find the information of his/her interest. We discuss WISE's main components: the resource database, the indexer robot, the search engine, and the user interface, and through the technical discussions, we highlight the research issues involved in the design, the implementation and the evaluation of such a system.

Index Terms— Resource discovery, information retrieval, world wide web, meta-information indexing.

1 Introduction

ONE of the most pressing issues with today's explosive growth of the Internet is the so-called resource discovery problem [15]. That is, how to find information of interest among the vast and ever growing amount of information available online. Research in this area has been gaining popularity among online information providers and researchers in information retrieval, database, artificial intelligence, and distributed computing areas. This research leads to the development of systems which allow users to locate resources by specifying keywords, i.e., searching on databases or catalogs which index the online resources.

In this paper, we discuss the WWW Index and Search Engine[1] (WISE), which is an integrated World Wide Web (WWW [4]) resource discovery system developed at the Department of Computer Science, the Hong Kong University of Science and Technology.[2] As the name implies, WISE is based entirely on WWW and is mainly designed for locating information on WWW as well as other online resources advertised on WWW. Through the discussion on the technical aspects of WISE, we try to highlight the issues involved in the design, implementation, and evaluation of such a system.

Our approach to Internet resource discovery is similar to other robot-based WWW indexing systems such as Lycos,[3] WebCrawler,[4] and the World Wide Web Worm[5] (WWWW), in that index data is collected by running an automated client program, known as a Web robot, to access WWW servers on the Internet. The system serves as a broker or directory service which supports keyword search using the index database. The search capability complements the WWW's powerful navigational access method by finding the potentially relevant starting points for subsequent browsing. It also supports browsing on meta-information for users to decide which resources to access. The meta-information is a compact representation of the resources and therefore takes less time to browse through. WISE supports this by providing users with a list of potentially relevant WWW addresses or Uniform Resource Locators (URLs [1]) and their descriptions in a hierarchical format.

The robot-based approach is by far the only workable interim solution before a more suitable communication protocol, such as the one used by Archie FTP directory service [6], for meta-information exchange between Web servers and information brokers is available. So far, ALIWEB [7] is the only known system to employ a server-to-server meta-information exchange protocol. However, this scheme demands a highly cooperative environment, which is hard to come by on the Internet (at least for now) where servers are largely distributed and autonomous.

WISE currently covers sites in Hong Kong and some other sites carrying information about Hong Kong. As of this writing,[6] there are 70,618 URLs spread over 276 sites in our database, and these numbers are expected to grow. We consider it infeasible for a WWW index broker which collects detailed meta-information like WISE does to cover WWW servers all around the world. In fact, we advocate the deployment of specialized brokers with minimal coverage overlaps (e.g., specialization based on the geographic locations, types of institution of the sites, and specialization on the domain of the information content). Ideally, these specialized brokers should be able to communicate with one another such that user requests can be routed to the appropriate broker(s) which can potentially provide the best answers. As far as geographical specialization is concerned, we have done our share by covering Hong Kong. We are currently working on a distributed multi-broker version of WISE, conveniently called Distributed WISE or D-WISE, and a scheme which incorporates existing heterogeneous and autonomous brokers into a single large virtual meta-information broker.

Fig. 1 shows the architecture of WISE. In this paper, we describe the system's main components, namely the index database, the indexer robot, the search engine, and the user interface. Section 2 discusses the design of the WISE resource database starting from the system's functional requirements to the specification and the implementation of the database. Section 3 discusses the index database creation and maintenance using WISE's indexer robot. Section 4 discusses the design of WISE's search engine which involves the development of ranking strategies for keyword-based searching in the WWW environment. Section 5 discusses the user interface to the search engine which supports hypertext mapping, relevance feedback mechanism and query sharing mechanisms. Finally, in Section 6, we close the paper with a brief comparison with other WWW index brokers, conclusions, and an overview of our future work.

2 Resource Database

An Internet resource database is an organized catalog or directory which contains meta-information describing the online resources. The term *resource* here refers to an online data object of any format. The design of such databases is dictated by the functionalities that the system is required to support.

2.1 Data Entities

The basic functional requirements for WISE as a resource discovery tool are:

1) to support keyword-based searching,

[1] Formerly known as the CS-HKUST WWW Index Server.
[2] WISE can be accessed at ⟨http://www.cs.ust.hk/IndexServer/⟩.
[3] ⟨http://lycos.cs.cmu.edu/⟩.
[4] Then at ⟨http://webcrawler.cs.washington.edu/WebCrawler/Home.html⟩.
[5] ⟨http://www.cs.colorado.edu/home/mcbryan/WWWW.html⟩.
[6] April 16, 1996.

- *B. Yuwono is with the Department of Computer and Information Science, Ohio State University, Columbus, Ohio.*
- *D.L. Lee is with the Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, KLN., Hong Kong. Email: dlee@cs.ust.hk.*

Manuscript accepted April 29, 1996.
For information on obtaining reprints of this article, please send e-mail to: transkde@computer.org, and reference IEEECS Log Number K96048.

1041-4347/96$05.00 ©1996 IEEE

2) to support meta-information browsing, and
3) to provide an up-to-date resource database.

The first requirement entails the need to identify and extract the important keywords of a data object as its descriptor. The extraction of keywords from textual data is described in Section 3.2. However, not all data objects contain texts. Fortunately, nontextual objects such as binary code, images (noninline ones), audio and video files are referenced on WWW by hypertext anchors or textual references, which can be used as the object descriptors. This method also applies to interactive resources such as gopher, FTP (file transfer protocol) and telnet-based services, and dynamically generated WWW pages such as those created on-the-fly by server-side programs.

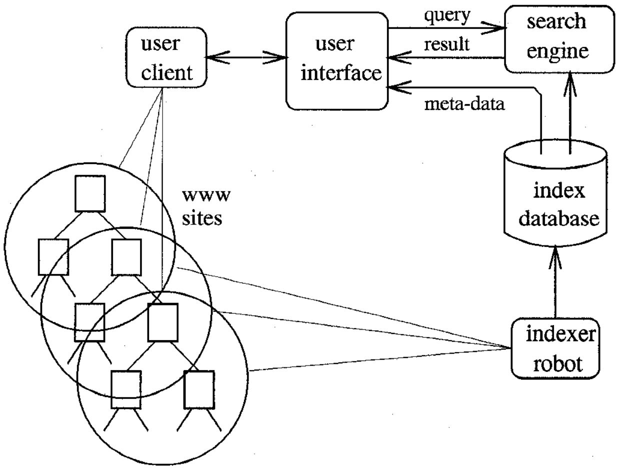


Fig. 1. The architecture of WISE.

The second requirement entails the need to provide compact representations of data objects to make browsing efficient yet informative enough for the user to judge whether a given resource is worth accessing or not. We opted to use the title of a WWW page as the main representation of the page. This *page title* is a short natural language description of a WWW page provided by its author, thus, capturing the topic of a page from the author's point of view. Although page titles do not always reflect the contents of the corresponding pages, as often is the case for a group of related pages with the same title, page titles are relatively easy for users to understand. In the case of interactive and nontextual resources, the associated hypertext anchors which are nearly as descriptive as page titles are used instead.

Other useful meta-information associated with a data object or resource is its data format, the protocol for accessing it, and its physical location which is represented by a network address, port number, and a file path. Fortunately, all this meta-information is readily available from the resource's URL [1].

Another potentially useful meta-information is the size of a data object. This information allows the user to estimate the time and space it will take to retrieve the object. However, size is obviously not a characteristic common to all types of resources, e.g., size is not relevant to interactive resources such as FTP, Gopher, and Telnet-based services. Also, in the case of dynamically generated objects such as those created on-the-fly by server-side scripts or programs, the size of such objects cannot be predetermined. We opted not to include this meta-information mainly because it has little relevance to the task of resource discovery.

Still, another useful type of meta-information is the logical relationships among a group of related resources. Such relationships can help users understand the context in which a given resource is situated. Ideally, such meta-information should be based on conceptual relationship. However discovering such relationships requires a considerable amount of real-world, and often domain-specific, knowledge and expensive intelligent processing. For practical and robustness reasons we opted to use hyperlinks explicitly defined by WWW page authors. The semantics of such hyperlinks is arbitrary in that WWW page authors use them for various reasons and sometimes without consideration of information coherence.

The third requirement entails the need to keep track of the state information associated with every resource in the database. By keeping the most recent date on which a resource is entered into the resource database, we can easily check the validity of an entry in the resource database by comparing the date of last update with the resource's date of last modification. For interactive resources and dynamically generated WWW pages, the validity of their records is based only on their availability or inavailability. Fig. 2 shows the entity-relationship schema of the resource database.

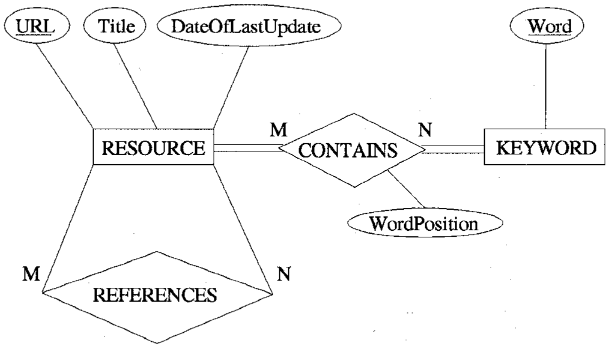


Fig. 2. The entity relationship schema of WISE's resource database.

2.2 Data Organization

An index is maintained for WISE to efficiently support keyword searching. The index is a table which maps a keyword or term to a set of URLs (resource identifiers) containing the term, including nontextual URLs whose descriptors contain the term. For record deletion and updating, a forward index which maps a URL to a list of terms that the URL contains, is also required. The terms on the list are in the same order as they appear in the text. The word position information is useful for phrase searching. The rest of the data attributes (i.e., title, date of last update, and list of hyperlinks) are stored in a set of tables in a relational style.

In order to minimize the storage requirement and the disk access time, URLs and terms in the above tables are represented by unique URL-ID numbers and unique term-ID numbers, respectively. Two additional tables are used for mapping between URL-ID and URL, and between term-ID and term. We implemented the above tables using the public domain GNU Database Manager (GDBM [10]) library package.

3 META-DATA COLLECTION

WISE is a robot-based WWW indexing system; its index database is built by an autonomous client agent, known as WWW robot, which accesses resources from various WWW sites or servers to collect the meta-information.

3.1 Indexer Robot

Our indexer robot is basically a WWW client which communicates with WWW servers using the Hypertext Transfer Protocol

(HTTP [2]). It only retrieves objects of text format (including non-HTML texts) from remote servers. In other words, it avoids interactive and dynamically generated resources, as well as nontextual and application-specific objects on which it is not capable of processing. From the retrieved text, it extracts titles and keywords, and collects hyperlink data (URL references and their associated anchor texts). It then follows the hyperlinks pointing to text-based URLs in a breadth-first manner. The robot practically retrieves all reachable HTML files from a site. It, however, is not capable of finding hyperlinks through image-maps (click-able images). Such capability would require sophisticated intelligence for recognizing image regions, at the least. URLs which are already in the database are retrieved only if their last modification dates are later than the dates the corresponding records were last updated. This type of conditional access requests is supported by HTTP.[7] The newly generated data will override the old data.

WISE's indexer robot honors the proposed standard for robot exclusion[8] which prevents the robot from accessing a site, or some portion of its collection, where the robot is not welcome. In order to avoid tying up a server's resources, the robot does not retry on failed accesses. It also recognizes loops caused by circular symbolic links by detecting the presence of repetitive substring patterns in a URL. This loop detection scheme is somewhat over-restrictive, in that a URL containing intentional repetitions of a path segment is mistaken for a URL with a looping-path. However, our observation showed that such a URL is very rare, as most people would think that multiple occurrences of the same directory name within a directory structure are not useful. In order to prevent the robot from getting into a loop which is caused by dynamically generated URLs referencing to the generating URL itself, the robot is designed not to access resources identified as scripts or executable programs. The indexer robot is entirely written in the C language.

A list of target sites is used to startup the indexer robot. The URLs of the sites to be included on the list can be keyed in manually by the robot's administrator or submitted by any user through a special HTML form. Since we are limiting our coverage to sites in Hong Kong only, we run some ad hoc checking on unknown site domains on the list to find out whether the sites actually reside in Hong Kong. After this manual inspection, the robot is then run to access one site at a time from the list. Upon finding a hyperlink referencing to a new site, the robot stores the URL of the site into a special list. The contents of this special list are then appended to the target list for the next processing batch. This indexing process is performed anytime a new URL(s) is registered with the system or discovered in the previous indexing process.

All records in the resource database are validated once every two weeks. This validation process is performed by sending out a probe request, the HTTP HEAD request, to check whether a resource is still accessible and whether the resource has been modified since the corresponding record was last updated. If the remote server's response indicates that the resource is no longer accessible, then the corresponding record is deleted and its URL–ID is freed up for reuse. If the resource has been recently modified, then the corresponding URL is appended onto the target list for later batch processing.

3.2 Keyword Extraction

WISE indexer robot builds the system's inverted file index by extracting keywords from HTML texts. Unlike the majority of similar systems built before WISE, which extract words in page titles only, WISE indexer robot extracts words from page titles, all levels of headings, anchor hypertexts, the first sentence of every list item, and words or sentences in italic and bold face. These words are explicitly marked as HTML tokens [3]. It is reasonable to assume that the author of a WWW page will use these token markups only for words or sentences he/she considers important. Therefore, we can assume that these words make good representations of the page's content. Among the extracted words, stop words such as function words and other common words are removed. Word stemming is then performed on the remaining words to remove suffices.

4 Information Retrieval

In this section, we discuss WISE's search engine which performs keyword search and resource ranking.

4.1 Query Preprocessing

The simplest form of a user's query is a list of one or more keywords. Optionally, users can use hyphens for search on phrases. WISE's query preprocessing involves stop-word removal, word stemming, and phrase identification. The resulting keywords are then looked up on the inverted file index to create a pool of potentially relevant URLs (resource identifiers). If there are phrases specified in the query, URLs which do not contain the phrases are removed from the pool. The pool is then handed over to the ranking algorithm.

4.2 Ranking Algorithm

The current version of WISE employs a ranking algorithm called TFxIDF which is based on the well-known vector space model [13]. The decision to use TFxIDF was made based on the result of an experiment which compares the algorithm with a number of alternative algorithms. This point is discussed later in this section after the descriptions of the algorithms.

In the following discussion, the word *term* refers to *keyword* and the word *document* refers to *resource*. These terms are more commonly used to describe information retrieval concepts. We also use the following notations:

M : the number of query terms.

Q_j : the jth query term, for $(1 < j < M)$.

N : the number of documents in the database.

D_i : the ith document in the database, for $1 < i < N$.

$R_{i,q}$: the relevance score of D_i with respect to query q.

$Li_{i,k}$: the occurrence of an incoming hyperlink from D_k to D_i, where $Li_{i,k} = 1$ if such a hyperlink exists, or 0 otherwise.

$Lo_{i,k}$: the occurrence of an outgoing hyperlink from D_i to D_k, where $Lo_{i,k} = 1$ if such a hyperlink exists, or 0 otherwise.

$C_{i,j}$: occurrence of Q_j in D_i, where $C_{i,j} = 1$ if D_i contains Q_j, or 0 otherwise.

4.2.1 *TFxIDF*

TFxIDF algorithm is based on the vector space model [13]. The similarity between a document and a query is measured by the cosine of the angle between their vector representations in a multidimensional space. The similarity value is taken as the relevance score of the document with respect to the query.

Generally speaking, the relevance score of a document is the sum of the weights of the query terms that appear in the document, normalized by the Euclidean vector length of the document. The weight of a term is a function of the term's occurrence frequency in the document (the term frequency,

[7] The If-Modified-Since request header is set.

[8] The proposal can be accessed at http://web.nexor.co.uk/users/mak/doc/robots/norobots.html.

or simply TF) and the number of documents containing the term in the collection (the inverse document frequency, or IDF). This weighting function gives higher weights to terms which occur frequently in a small set of the documents. However, as we have shown empirically in [17], the vector-length normalization causes a drop in the average retrieval precision when applied to the WWW document environment. A similar phenomenon is reported in [12] for collections of short documents. Therefore, the vector-length normalization factor is not used in our TFxIDF algorithm. That is, the relevance score of document D_i with respect to query q is computed by:

$$R_{i,q} = \sum_{term_j \in q} \left(0.5 + 0.5 \frac{TF_{i,j}}{TF_{i,max}} \right) IDF_j \tag{1}$$

where:

$TF_{i,j}$: the term frequency of Q_j in D_i

$TF_{i,max}$: the maximum term frequency of a keyword in D_i

IDF_j : $\log\left(N / \sum_{i=1}^{N} C_{i,j} \right)$

This term weighting formula is known as the *nfx* formula [12]. As shown in the equation, the TF component is normalized by the maximum TF in the vector, and further normalized to lie between 0.5 to 1.0.

4.2.2 Other Algorithms

We have considered a number of alternative ranking algorithms. These are our original algorithms which are specifically designed to exploit the hyperlink data in hypertext environments. They are: most-cited, binary-vector spread activation,[9] and weight-vector spread activation algorithms.

1) Most-cited:

This algorithm takes advantage of information about hyperlinks connecting WWW pages. Each document is assigned a relevance score which is the sum of the number of query terms contained in other documents citing the document in question. More formally, the relevance score of document D_i with respect to query q is computed by:

$$R_{i,q} = \sum_{k=1,\,k \neq i}^{N} \left(Li_{i,k} \sum_{j=1}^{M} C_{k,j} \right) \tag{2}$$

This algorithm assigns document weight which is a function of the number of query words that a document contains (we refer to this weighting scheme as a binary vector model), and assigns, among documents with nonzero weights, larger relevance scores to the cited documents than the citing documents.

2) Binary-vector spread activation:

As in the previous algorithm, documents are assigned weights whose values are a function of the number of query terms they contain. This scheme can be considered as an extension to the standard Boolean model whose binary set membership criterion does not allow document ranking.

More formally, document D_i is assigned a relevance score with respect to query q as follows:

$$R_{i,q} = \sum_{j=1}^{M} C_{i,j} \tag{3}$$

The binary-vector spread activation algorithm extends this document weighting scheme by propagating the occurrence of a query term in a document to the neighboring documents connected by hyperlinks. This is based on the assumption that if two documents are linked to one another then there is some conceptual relationship between their contents. The additional weight induced by a query term propagated from a neighboring document is set to be less than the weight induced by a local query word. For document D_i, the algorithm assigns a relevance score with respect to query q as follows:

$$R_{i,q} = \sum_{j=1}^{M} I_{i,j} \tag{4}$$

where $I_{i,j}$ is defined as:

$$I_{i,j} = \begin{cases} c_1 & \text{if } C_{i,j} = 1 \\ c_2 & \text{if there exists } k \text{ such that } C_{k,j} = 1 \text{ and } Li_{i,k} + Lo_{i,k} > 0 \\ 0 & \text{otherwise} \end{cases}$$

c_1 and c_2 are constants (c_1, $c_2 > 0$) where $c_1 > c_2$. In our implementation, we use $c_1 = 10$ and $c_2 = 1$. It has been shown in [17] that the algorithm is not sensitive to the values of these two constants.

3) Weight-vector spread activation:

This algorithm combines the vector space model and the spread activation model. In this algorithm, each document is first assigned a relevance score using TFxIDF algorithm, then the score of a document is propagated to the neighboring documents through hyperlinks. More formally, the algorithm assigns a relevance score to document D_i with respect to query q as follows:

$$R_{i,q} = S_{i,q} + \sum_{j=1,\,j \neq i}^{N} \alpha\, Li_{i,j} \cdot S_{j,q} \tag{5}$$

where $S_{i,q}$ is the TFxIDF score of D_i as defined in (1). α ($0 < \alpha < 1$) is a constant link weight. Through an experiment, we found that 0.2 is the optimal value for α [17].

Among the retrieved documents returned by the ranking algorithm, only the top H documents are selected. H is called the *maximum number of hits*. WISE's default value of H is set to 40. Some users have suggested the use of a threshold value to limit the number of hits where the cut-off boundary is more meaningful. However, our observation showed that, from the user's point of view, setting the maximum number of hits is more intuitive than setting a threshold value, particularly since the user does not know in advance what the range of the scores is going to be. We are currently investigating the effectiveness of using percentile as the cut-off criterion.

4.3 Retrieval Effectiveness

We evaluated the four algorithms above on a text database which covers WWW resources at the Chinese University of Hong Kong[10] (CUHK). CUHK site was chosen because of its reasonable size, and because it has the most diverse collection of information compared to

[9] The algorithm was formerly called *Boolean spread activation* [17] which we feel can be misleading.

[10] WWW servers in cuhk.hk domain.

other Hong Kong sites at the time of the experiment.[11] We froze the entire WWW collection by copying all of the WWW pages from the site to a local disk. We recorded 2,393 URLs including 1,139 non-HTML URLs.

Fifty-six test queries were used. The test queries were generated as follows. First, we selected at random 100 URLs from the collection. Of these 100 URLs, we removed URLs of directory type (indices, catalogs, hotlists, tables of contents, etc.) and non-HTML URLs, resulting in 56 URLs. Next, we manually extracted keywords from each of these 56 URLs by selecting keywords which can be used to construct one or more phrase that represents the content of the URL and finally, constructed a query from that phrase after adding some synonyms where appropriate.

Given a query, the judgment on whether a WWW page is relevant or not is somewhat ambiguous, as it is subjective in nature. For this evaluation, we define relevance in the context of resource discovery, i.e., a URL is considered relevant to a query if the URL is pointing to a resource containing information pertinent to the query. Pertinent in the sense that the resource contains the desired information or the resource contains information about where the desired information can be found. Any text of one sentence long or longer, mentioning one or more of the query words in the same context as the context implied by the query, is considered. We manually examined the entire collection to identify the relevant URLs for each query. The maximum number of hits for this experiment was set to 500 URLs.

We used the standard evaluation procedure to compute the recall/precision of the algorithms [13]. Fig. 3 shows the recall/precision curve for each of the algorithms. The high average precision obtained in this experiment is attributed to the query construction procedure which guarantees that each query has at least one highly relevant document. Although the vector spread-activation shows the best average precision, the difference between it and TFxIDF is practically negligible. Since TFxIDF is a much simpler method to implement and is computationally less expensive, it is chosen for WISE.

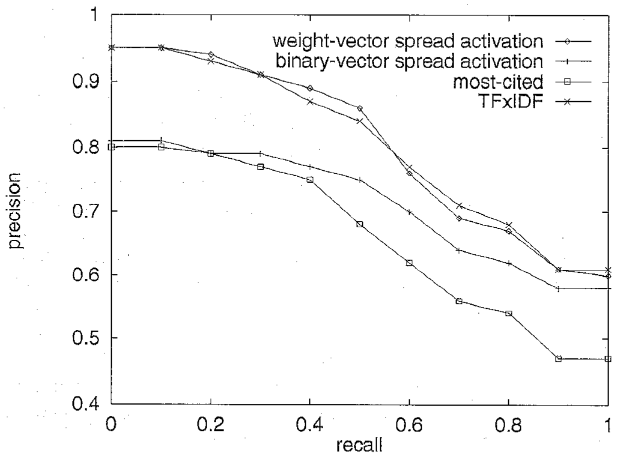


Fig. 3. The recall/precision curves of the four document ranking algorithms obtained by averaging the curves over the 56 test queries.

The result of the experiment provides us with some hints on the nature of the WWW information retrieval environment. The relatively superior retrieval effectiveness of TFxIDF and weight-vector spread activation algorithms shows that the concentration or distribution of terms in a WWW page and across WWW pages is a good indicator of the page's contents or portions thereof. Algorithms which rely on hyperlink meta-data, while intuitive, do not perform satisfactorily. Our qualitative analysis shows that the interconnectivity between WWW pages does not always indicate conceptual relationships between the contents of these pages. Very often this lack of coherence is caused by the fact that many WWW pages, particularly the so-called home pages, put many different and unrelated topics in a single page.

As far as usability is concerned, we admit that with the vector space retrieval model, it is harder for a user to fine tune his/her query by adding and removing query words than it is with a pure Boolean search. The reason is that it is not easy for an average user to get the feel of the impact of the collection-wide distribution of a query word, which largely determines the weight of the word in a document, on the search results. On the other hand, it is unreasonable to expect an average user, given the opportunity, to express his/her information need by assigning a weight value to each query word. In general, however, the vector space model provides better retrieval recalls for a wider range of query precision, i.e., from a poorly constructed query to a highly precise query, than the standard Boolean model.

5 User Interface

WISE's user interface to the search engine is an HTML form which can be invoked using standard WWW client programs such as Mosaic, Netscape, etc. The interface allows the user to, among other things, type in a query, set the maximum number of hits, initiate a search, and invoke another HTML forms to register a URL. The user interface mechanism is implemented using the standard Common Gateway Interface (CGI) and is run by an NCSA HTTPD WWW Server program. A more detailed description of the user interface can be found in [16].

After typing in a query, the user sends it to the search engine by clicking on the *Search the Web* submit button. Upon receiving the result from the search engine, the user interface displays a list of URLs (represented by their respective titles) ordered in descending relevance scores. The user can directly access these URLs by clicking on the titles. In addition to the above ordered list, the result page also shows other information and buttons for various functionalities, which are described in the following paragraphs.

5.1 Hierarchical Page Map

The concept of meta-information browsing as mentioned in Section 2 is applied here. In order to help the user identify the context of a particular URL shown on the result page, the hit URLs are displayed in a hierarchical organization. This organization, shown by the use of line indentations, depicts the parent-child or referrer-referee hyperlink relationships among the URLs (see Fig. 4). Accompanying each of the URLs titles is an icon indicating the URLs data format (text, image, sound, video, etc.) or one of two special icons representing an expandable item (closed suitcase) and a shrinkable item (open suitcase) for HTML files. When the user clicks on a closed suitcase icon the URLs referenced by the URL associated with the suitcase icon are shown as a nested list of titles under the referencing URL (see Fig. 4). At this point the closed suitcase icon is replaced by an open suitcase icon. Clicking on the open suitcase icon will revert the list to its original state, i.e., the child URLs under the referencing URL are hidden. This expand/shrink operation is performed by the user interface based on the hyperlink data available in the resource database. In addition to the page icons, the number of pages refer-

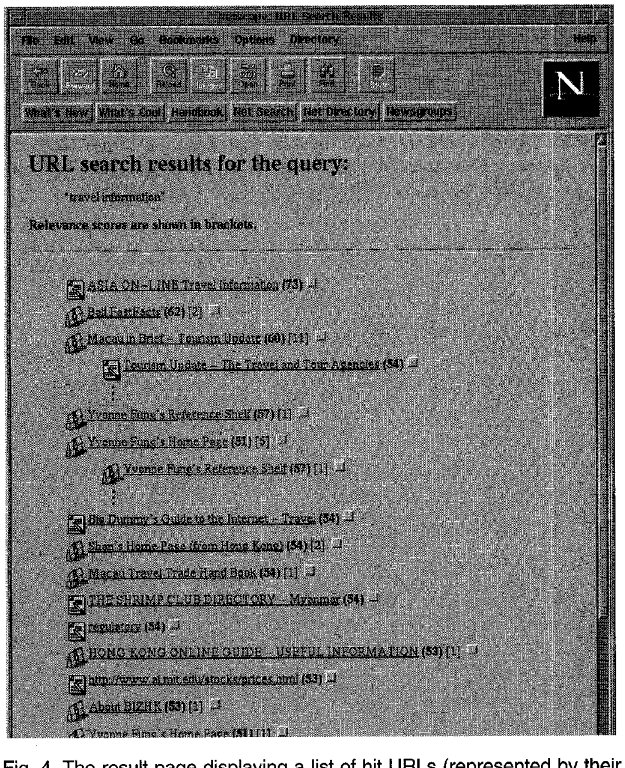
[11] April 26, 1995.

enced by each hit URL or child URL is shown in square brackets. Because of the use of the hierarchical organization, where hit URLs are grouped together based on links between them, the ordering of the hit URLs on the result page is based on the maximum relevance score within each group (see Fig. 4). The effect of this method is that URLs of low relevance scores may be ranked high by their association with URLs of high relevance scores.

Lastly, the system keeps track of the state of the user session by assigning a unique page-ID to every result page it generates. Every request resulted from the user's clicking on an icon on a result page is tagged with the page's ID. The user interface maintains the information associated with every page-ID for a period of time.

5.2 Relevance Feedback

The remaining part of the result page is an HTML form used for the relevance feedback mechanism. First, the user is asked to mark URLs judged as relevant to his/her query by clicking on a check box button next to each of the URLs titles (see Fig. 4), we refer to these URLs as *feedback URLs*. The relevance feedback mechanism expands the query by adding a number of most frequently occurring keywords from the feedback URLs. The keywords and their occurrence frequencies in the feedback URLs are obtained from the forward index described in Section 2. The relevance feedback algorithm is based on the algorithms proposed in [8] which are designed to refine a query by capturing the context in which the original query words appear in the feedback URLs. Finally, the reformulated query is submitted to the system for processing when the user clicks on the *Feedback* button, and another result page is then shown.



URL search results for the query:

"travel information"

Relevance scores are shown in brackets.

ASIA ON-LINE Travel Information (73) □
Bali FastFacts (62) [2] □
Macau in Brief - Tourism Update (60) [11] □
  Tourism Update - The Travel and Tour Agencies (54) □
  ⋮
Yvonne Fung's Reference Shelf (57) [1] □
Yvonne Fung's Home Page (51) [5] □
  Yvonne Fung's Reference Shelf (57) [1] □
  ⋮
Big Dummy's Guide to the Internet - Travel (54) □
Shan's Home Page (from Hong Kong) (54) [2] □
Macau Travel Trade Hand Book (54) [1] □
THE SHRIMP CLUB DIRECTORY - Myanmar (54) □
regulatory (54) □
HONG KONG ONLINE GUIDE - USEFUL INFORMATION (53) [1] □
http://www.ai.mit.edu/stocks/prices.html (53) □
About BIZHK (53) [1] □
Yvonne Fung's Home Page (51) [1] □

Fig. 4. The result page displaying a list of hit URLs (represented by their titles) ordered in descending group-scores. Line indentation is used to indicate hierarchical dependence relationships among the URLs.

The user interface component consists of two separate modules, one for handling the page expansion/shrinking operations and the other for handling the relevance feedback mechanisms. These modules are written in the C language and run as CGI scripts.

5.3 Usability

We have not conducted a formal usability study on our system. However, from our access log file, we learned that the average query length is 1.5 words. While there is nothing wrong with very short queries, as users often just want to browse around without any specific information needs in mind, such a query often results in more URLs than the maximum number of hits allows, particularly if the keywords used are the very common ones. In such a case, the user may not be aware of the existence of URLs which may be of his/her interest but were cut off by the maximum number of hits.

We also learned from the access log that most users rarely, if at all, refine their queries, e.g., by adding or dropping keywords to narrow down or broaden up the scope of their searches when we expected them to do so. However, we cannot tell what happened without any verbal protocol data. It may be that the user was more tempted to explore the WWW by following links shown on the result page rather than reformulating the query. It may also be that the user was not familiar enough with the keyword searching process to be able to fine-tune the query. User interface techniques, such as single-line help technique which shows a single-line description of the item pointed by the cursor, can help improve the system's usability. However, implementing these techniques on the server's side, as opposed to on the client's side, is not practical. Ideally, functionalities such as hypertext mapping and relevance feedback mechanism should be implemented on the client's side, where the client down loads the necessary data from the server and performs the computation locally. This scheme will also make better display techniques, such as graphical hypertext maps with fish-eye view [14] or zooming capabilities, possible.

6 Conclusions

In the past two years, the population of robot-based WWW index servers has been growing rapidly.[12] Among the well known robots before WISE, only a few employ full-text indexing, e.g., WebCrawler [11], the Repository Based Software Engineering Project Spider[13] (RBSE [5]), and Lycos, while others index only page titles and anchor hypertexts. Today, all major robot-based index servers such as Alta Vista[14] employ full-text indexing. WISE's scheme lies between full-text and title-only schemes by taking advantage of HTML-based meta-information as much as possible. This scheme is efficient in terms of storage requirement yet still allows sophisticated information retrieval models such as the vector space model to work effectively. Our entire resource database which currently covers 70,618 URLs takes up only 78 MBytes of disk space. This is an important aspect in our effort to promote an architecture consisting of distributed index servers with local coverages. With a relatively small storage requirement more sites can be expected to participate in such a system. We believe that localized index coverage is more responsive to the rapidly changing nature of the WWW, i.e., it is more cost efficient to update hundreds of small distributed resource databases frequently than a single large centralized one. Moreover, we learned that even though WISE indexes only WWW pages in Hong Kong, it can help users in locating a wide variety of resources around the world, at least indirectly via

[12] A list of active WWW robots can be found at http://web.nexor.co.uk/mak/doc/robots/active.html.
[13] http://rbse.jsc.nasa.gov/eichmann/urlsearch.html.
[14] http://www.altavista.digital.com/.

some pages belonging to Hong Kong users. This constitutes a form of resource discovery sharing among members of the user community. WISE's other strong point is its use of graphical user interface to present meta-information, including grouping and ranking of related URLs.

As the WWW grows at an increasing rate, methods to make information available online more manageable are highly in demand. Despite the potential benefit of advanced information retrieval techniques in improving the effectiveness of access to online information, little research has been done on applying these techniques to the WWW environment. The work presented in this paper is but an early stage of such research. We believe that there is still room for improvements and many strategies yet to be explored. For instance, search strategies which take advantage of WWW-specific meta information such as the semantics of hyperlinks and highly user-configurable query processors may offer a better retrieval performance. Furthermore, better user interface designs are crucial in making these techniques more intuitive for the average user to use.

Our current work in progress includes developing a client-based user interface to WISE. We are studying the possibility of using an architecture which is based on embedded portable applications such as the Java[15] applets. Research in the distributed version of WISE (D-WISE) is also underway. Still, another related area of research that we are pursuing is the development of methods for Chinese text retrieval on the World Wide Web.

REFERENCES

[1] T. Berners-Lee, "Uniform Resource Locators," *Internet Working Draft*, Jan. 1, 1994.

[2] T. Berners-Lee, "Hypertext Transfer Protocol," *Internet Working Draft*, Nov. 5, 1993.

[3] T. Berners-Lee, and D. Connolly, "Hypertext Markup Language," *Internet Working Draft*, July 13, 1993.

[4] T. Berners-Lee, R. Cailliau, J. Groff, and B. Pollermann, "World Wide Web: The Information Universe," *Electronic Networking: Research, Applications, and Policy*, vol. 1, no. 2, 1992.

[5] D. Eichmann, "The RBSE Spider-Balancing Effective Search Against Web Load," *Proc. First Int'l Conf. World Wide Web*, pp. 113-120, Geneva, 1994.

[6] A. Emtage and P. Deutsch, "Archie: An Electronic Directory Service for the Internet," *Proc. USENIX Winter Conf.*, pp. 93-110, Berkeley, Calif., 1992.

[7] M. Koster, "ALIWEB: Archie-Like Indexing in the Web," *Computer Networks and ISDN Systems*, vol. 27, no. 2, pp. 175-182, 1994.

[8] D. Lee and A. Chuang, "Performance of Document Ranking and Relevance Feedback," to appear in *IEEE Software*.

[9] O. McBryan, "GENVL and WWWW: Tools for Taming the Web," *Proc. First Int'l Conf. World Wide Web*, pp. 79-90, Geneva, 1994.

[10] P. Nelson, "GDBM—The GNU Database Manager," Online Manual Pages, version 1.7.3, 1990.

[11] B. Pinkerton, "Finding What People Want: Experiences with the WebCrawler," *Proc. First Int'l Conf. World Wide Web*, Geneva, 1994.

[12] G. Salton and C. Buckley, "Term-Weighting Approaches in Automatic Text Retrieval," *Information Processing and Management*, vol. 24, no. 5, pp. 513-523, 1988.

[13] G. Salton and M. McGill, *Introduction to Modern Information Retrieval*. New York N.Y.: McGraw-Hill, 1983.

[14] M. Sarkar and M. Brown, "Graphical Fish-Eye Views of Graphs," *Proc. CHI '92: Human Factors in Computing Systems*. New York N.Y.: ACM Press, pp. 83-92, 1992.

[15] M. Schwartz, A. Emtage, B. Kahle, and B. Neumann, "A Comparison of Internet Resource Discovery Approaches," *Computer Systems*, vol. 5, no. 4, pp. 461-493, 1992.

[16] B. Yuwono, S. Lam, J. Ying, and D. Lee, "A World Wide Web Resource Discovery System," *Proc. Fourth Int'l World Wide Web Conf.*, Boston, pp. 145-158, Dec. 1995.

[17] B. Yuwono and D. Lee, "Search and Ranking Algorithms for Locating Resources on the World Wide Web," *Proc. 12th Int'l Conf. Data Engineering*, New Orleans, pp. 164-171, Feb. 1996.

[15] Information on Java can be found at http://java.sun.com/.