The Use of Cluster Hierarchies in Hypertext Information Retrieval*

Donald B. Crouch, Carolyn J. Crouch and Glenn Andreas

Department of Computer Science
University of Minnesota - Duluth
320 Heller Hall
Duluth, Minnesota 55812

ABSTRACT

The *graph-traversal approach* to hypertext information retrieval is a conceptualization of hypertext in which the structural aspects of the nodes are emphasized. A user navigates through such hypertext systems by evaluating the semantics associated with links between nodes as well as the information contained in nodes. [Fris88] In this paper we describe an hierarchical structure which effectively supports the graphical traversal of a document collection in a hypertext system. We provide an overview of an interactive browser based on cluster hierarchies. Initial results obtained from the use of the browser in an experimental hypertext retrieval system are presented.

INTRODUCTION

Information retrieval is concerned with the representation, storage and retrieval of documents or document surrogates. Information retrieval activities are routinely conducted on-line under the control of search intermediaries or end users who have been trained to use somewhat complex user-system interfaces. However, poor query formulations and inadequate user-system interaction still occur even with skilled users. For example, Cleverdon has noted that "if two search intermediaries search the same question on the same database on the same host, only 40 percent of the output may be common to both searches." [Clev84]

What is being done to aid users of information retrieval systems? The most common approaches are generally directed either toward the development of aids based on sophisticated user interfaces or toward the development of expert system techniques for the more complex operations of text retrieval systems. [Crou89] Research involving sophisticated user interfaces is primarily concerned with system functioning and convenience as it relates to the user; its goal is to facilitate the use of the system by providing computerized aids previously available only to the search intermediary in non-computerized forms. Among the facilities normally included in systems of this type are vocabulary displays, thesaurus expansion of vocabulary items designed to add related terms to already existing search words, the construction and storage of search protocols, operations with previously formulated queries, etc. While this type of research is warranted and its results encouraging, it has not necessarily produced more effective retrieval but instead has generated tools for *effortless* learning and use of an information retrieval system.

The other major area of research in the development of user aids for information retrieval is concerned with the design of expert systems that facilitate access to the stored

*This work was supported by the National Science Foundation under grant IRI 87-02735.

collections. The goal of such research is to capture the expertise of search intermediaries in formulating Boolean queries and in dealing with other types of retrieval services. The expert approach is based on the use of domain-specific knowledge that covers the topic areas represented by the collection, a language analyzer that can understand natural language queries and translate them into appropriate internal forms, and rules for search formulation and search strategy designed to choose search methods based on user criteria. It may eventually become feasible to generate search formulation criteria in the form of rules that do in fact reflect the expert knowledge of trained search intermediaries. However, for the time being, one has reason to be careful in accepting many of the currently unevaluated design proposals for expert system approaches as effective solutions to the retrieval problem.

We submit that a viable alternative to using either very sophisticated user interfaces or expert systems as a solution to the retrieval problem consists of using only simple user-system interactions which enhance the effectiveness of retrieval operations through the addition of properly designed user friendly features. These features allow the user to function in an active role, replacing the full natural language comprehension which is desirable yet currently unavailable in an automatic search expert.

This approach to interface design is inherent in the concept of hypertext information retrieval. Hypertext supports a user's exploration of informational data items by representing data as a network of nodes containing text, graphics and other forms of information. [Smit88] A user may navigate through the hypertext system by following the links between nodes. The path a user follows is determined by his/her analysis of the information contained within the nodes and the semantics associated with links between the nodes. [Fris88]

In hypertext information retrieval, each node is generally assumed to be a single document. Links exist which connect each document to other documents having keywords in common with it; the semantics of the links between nodes are keywords (document index terms) or some descriptive information representing the connected documents. In this paper we introduce an hierarchical structure which provides additional semantic information within and between nodes. This structure seems particularly well suited to the user's exploration of a document collection in a visual context. The user may browse among the data items by analyzing a graphical display of the structure itself as well as the semantic links between nodes.

In the next two sections, we briefly describe the retrieval model and the characteristics of the hierarchy on which our structure is based. We then describe a prototype of a hypertext retrieval system utilizing the cluster hierarchy and present the initial results of an experiment comparing retrieval performance of the hypertext system with that of an automatic retrieval system.

INFORMATION RETRIEVAL MODELS

The most common information retrieval models are the Boolean retrieval model and the vector space model. These two models are briefly described and their use in conventional information retrieval systems examined.

Boolean Retrieval Model

Most retrieval systems are based on the *Boolean model*. Queries are expressed as a set of terms connected by the Boolean operators *and*, *or* and *not*. Such systems retrieve information by performing the Boolean operations on the corresponding sets of documents containing the query terms. Although the Boolean model can be used effectively in automatic text retrieval (in fact, a query can be formulated to retrieve any particular subset of items), imprecise or broad requests utilizing the *or* relation can result in the retrieval of large numbers of irrelevant texts while narrow or overly precise queries

utilizing the *and* relation can exclude many relevant items. In practice a compromise is often obtained by the use of a query formulation that is neither too broad nor too narrow. [Salt86]

Although the Boolean model has been widely accepted, it does have its problems:

- Boolean queries are difficult to construct; intermediaries are generally required to add terms not originally included, provide synonyms or alternate spellings, drop high-frequency terms, etc. [Fox86],
- Boolean systems generally do not provide for the assignment of term weights,
- the size of the subset of documents to be returned is difficult to control, and
- the retrieved documents are usually presented in a random order (no ranking based on an estimate of the query-document relevance is provided).

The difficulties associated with the construction of Boolean queries are well known. One author recently commented that "research and development in information retrieval since the 1950's has concentrated on methods which can provide better retrieval without the need for Boolean queries." [Colv86]

Vector Space Model

The *vector space model* is conceptually the simplest retrieval model and is better suited for use in hypertext retrieval systems than the Boolean model. In the vector space model, the content of each document or query is represented by a set of possibly weighted content terms (i.e., some form of content identifier, such as a word extracted from the document text, a word phrase, or concept class chosen from a thesaurus). A term's weight reflects its importance in relation to the meaning of the document or query. Each informational item (document) may then be considered a term vector, and the complete document collection becomes a vector space whose dimension is equal to the number of distinct terms used to identify the documents in the collection. [Rijs79, Salt83]

In the vector space model, it is assumed that similar or related documents or similar documents and queries are represented by similar multidimensional term vectors. Similarity is then generally defined as a function of the magnitudes of the matching terms in the respective vectors.

A vector representation of documents and queries facilitates certain retrieval operations, namely:

- The construction of a clustered document file (consisting of classes of documents such that documents within a given class are substantially similar to each other). In clustered collections, an automatic search can be limited to the documents within those clusters whose class vector representations are similar to the query vector.
- The ranking of retrieved documents in decreasing order of their similarity with the query.
- The automatic reformulation of the query based on relevance assessments supplied by the user for previously retrieved documents. The intent of *relevance feedback* is to produce a modified query whose similarity to the relevant documents is greater than that of the original query while its similarity to the nonrelevant items is smaller.

The vector processing model also exhibits certain disadvantages, namely:

- Some model parameters, such as the query-document similarity function, are not derivable within the system but instead are chosen a priori by the system designer.

- Terms are assumed to be independent of one another.
- Term relationships are not expressible within the model.

A recent characterization of the vector space model is contained in [Wong84].

CLUSTERED DOCUMENT ENVIRONMENTS

A principal advantage of the vector space model for use in hypertext information retrieval is that algorithms exist for structuring a document collection in such a manner that similar documents are grouped together. A cluster hierarchy is represented by a tree structure in which terminal nodes correspond to single documents and interior nodes to groups of documents. In a hypertext system based on a clustered environment, the user can readily focus his/her search on those groups (clusters) that are likely to contain documents which are highly similar to the query. Additionally, the cluster hierarchy is beneficial as a browsing tool in that it makes it possible easily to locate neighboring items with related subject descriptions.

Agglomerative Cluster Hierarchy

Voorhees [Voor85] has shown that retrieval effectiveness may be enhanced in automatic retrieval systems when a type of clustering, known as *agglomerative hierarchic clustering*, is used to generate a cluster structure. In such a clustering method, each document in the collection is considered initially to be a singleton cluster. The two *closest* clusters are successively merged until only one cluster remains. The definition of *closest* depends on the actual clustering method being used.

Fig. 1 contains an example of a hierarchy for the *single link* agglomerative clustering method. In the single link method the similarity between two clusters is the maximum of the similarities between all pairs of documents such that one document of the pair is in one cluster and the other document is in the other cluster. It may be noted that in the hierarchy documents may appear at any level and that clusters overlap only in the sense that smaller clusters are nested within larger clusters.

Each cluster in Fig. 1 is labelled with the *level of association* between the items under it. The clustering level determines the association strength of the corresponding items. Thus the similarity between items B, C and D in Fig. 4 is 0.9. On the other hand, the similarity between item A and the cluster containing items B, C and D is only 0.7. The level of association is a useful link semantic in a hypertext system.

Searching a Clustered Environment

To retrieve documents automatically in a clustered environment, comparisons are generally made between the query vector and document vectors using one of the standard measures of similarity. A cluster search simplifies the search process by limiting the search to subsets of documents. For example, with an agglomeratively clustered tree such as that shown in Fig. 1, a straightforward, narrow, *depth-first* search starts at the top of the tree and calculates the similarity between the query and each of its children. The child most similar to the query is selected, and the similarity between the query and each of the non-document children of that node is calculated. The process is repeated until either all the similarities between the query and the non-document children of some node are less than that between the query and the node itself, or all the children of that node are document nodes. The documents comprising the cluster represented by that node are returned. The search may be broadened by considering more than one path at each level. The broadest search considers all paths and abandons them as they fail certain criteria.

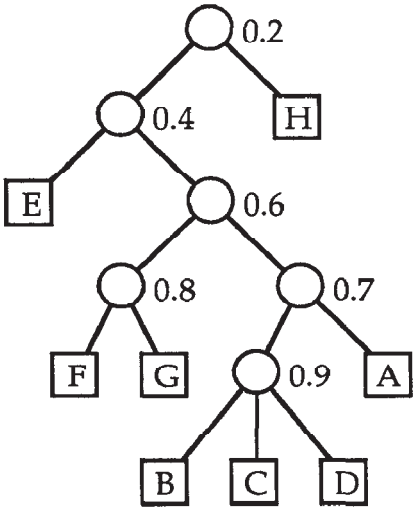

Fig. 1 A sample single link hierarchy

A *bottom-up* search may also be performed on such a tree. The cluster at the lowest level of the tree whose centroid is most similar to the query is chosen as the node at which the search will start. The search continues up the tree until the similarity between the query and the parent of the current node is smaller than the similarity between the query and the current node. The documents contained in the cluster corresponding to the current node are returned. The bottom-up search is often more effective due to the uncertainty involved at high levels of the hierarchy. [Crof80]

Cluster hierarchies have been used effectively in automatic searches. Such hierarchies are also useful in performing searches based on browsing operations. These types of operations, we believe, can produce significant improvement in retrieval performance. Automatic cluster searches are highly structured; the next link in the search path is determined solely on the basis of the similarity between the query vector and the vector representation of the node being evaluated. By displaying suitable portions of the hierarchy during the course of the search operations and letting the user choose appropriate search paths at each point, the output obtained should be superior to that obtained by automatic cluster searching. For example, in a hypertext system with an interactive browser, following evaluation of items B, C, and D in the sample tree of Fig. 1, the user has the choice of exploring either a tightly clustered structure containing items F and G (which are very similar to each other with a similarity value of 0.8) or of staying in the same cluster and evaluating item A (at a lower similarity level of 0.7). In contrast, the control mechanism of the automatic search procedure may terminate the search at the node labelled 0.7 and never evaluate the cluster containing items F and G. The effectiveness of this type of user-directed, interactive browsing is determined by comparing the results of such interactive searches to those obtained by automatic cluster searches.

THE INTERACTIVE BROWSER

A browser incorporating the cluster hierarchy as its primary network structure was implemented on a Macintosh IIx computer using HyperCard. The Macintosh is connected via a local area network to a SUN System on which the SMART information retrieval system [Salt71] resides. The SMART system provides packages for textual analysis, clustering, performance evaluation, etc.

To conduct a search using the browser, a user initially specifies a natural language query which is subsequently transformed into a term vector representation via the SMART retrieval system. The hypertext system then displays a window containing the original query and its corresponding term vector (Fig. 2). As suggested by the annotated display of

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.