Information retrieval through hybrid navigation of lattice representations

CLAUDIO CARPINETO AND GIOVANNI ROMANO

Fondazione Ugo Bordoni, Via Baldassarre Castiglione 59, I-00142 Rome, Italy
email: {carpinet, romano}@fub.it

RUNNING TITLE: Navigation of Lattice Representations

Summary. In this paper we present a comprehensive approach to automatic organization and hybrid navigation of text databases. An organizing stage first builds a particular lattice representation of the data, through text indexing followed by lattice clustering of the indexed texts. The lattice representation, then, supports the navigation stage of the system, a visual retrieval interface that combines three main retrieval strategies: browsing, querying, and bounding. Browsing and querying are used to search the retrieval space, bounding is used to restrict it based on the information that users have, or get during their interaction with the system. We show that such a hybrid paradigm permits high flexibility in trading off information exploration and retrieval and, in addition, has good retrieval performance. We compared information retrieval using lattice-based hybrid navigation with conventional Boolean querying. The results of an experiment conducted on two medium-sized bibliographic databases showed that the performance of lattice retrieval was comparable to or better than Boolean retrieval.

1. Introduction

1.1 THE ORGANIZING/NAVIGATING PARADIGM

In most information retrieval systems, the user provides some description of the information being sought, and the system retrieves database items that match the description. The query-based paradigm has a long and predominant tradition in information retrieval (van Rijsbergen, 1975; Salton & McGill, 1983): various models have been proposed, e.g., boolean, vector space (Salton & McGill, 1983), probabilistic (Turtle & Croft, 1991), some of which have been incorporated into commercially available systems. However, more recently, another approach to information retrieval that explores a different assumption - that humans seem to prefer recognition tasks to description tasks (Marchionini, 1992) - has gained widespread acceptance. It consists of organizing the information into some sort of structure, and letting the user navigate through it, therefore it is sometimes called the organizing/navigating paradigm (Bowman, Danzig, Manber, & Schwartz, 1994). Hypertexts (Nielsen, 1990), cluster browsers (Crouch, Crouch & Andreas, 1989), and object-oriented retrieval interfaces (Lucarella, Parisotto & Zanzi, 1993) can all be seen as variations of this basic theme. The chief advantage of this kind of approach is flexibility. As the user may retrieve a document of interest without specifying a query perfectly or partially matching some description of the document, this approach does not require prior knowledge about the query language or the content of the data base, nor does it require the user to have some specific goal in mind. Therefore it is also suitable for casual users and for exploration of new domains (Marchionini & Shneidermann, 1988), two features that have become more and more important with the continuously growing amount of Internet accessible information resources.

However, the organizing/navigating paradigm also presents some serious limitations. One problem is the construction of the supporting structure. Although the possibility of building the structure automatically has been investigated (e.g., Thompson & Croft, 1989; Maarek, Berry & Kaiser, 1991; Savoy, 1993; Agosti, Melucci & Crestani, 1995), this often requires a large amount of manual work as well as the possession of an initial set of index terms representing each document (e.g., Frei & Jauslin, 1983; Frisse & Cousins, 1989; Pedersen, 1993). Another, perhaps more important, limitation of navigational approaches is their retrieval ineffectiveness, even for databases of modest size. The difficulty of finding specific information is related, in part, to the well known difficulty of orientation of the user in the information space, for which sophisticated navigational aids have been developed (e.g., Furnas, 1986; Mackinlay, Robertson, & Card, 1991). But there seems to be at least two other fundamental questions to address. The first is that the user should be given the opportunity to locate the sought information without having to navigate through the structure. The second is that the organization of the information should be adapted to the specific user’s needs and knowledge, while in most systems it is predefined and cannot be easily changed.

Thus, one of the major challenges for navigation system design is to find new ways of exploiting the flexibility of the basic approach while reducing the accompanying disadvantages. To attack this problem, as we will see in the next section, several hybrid approaches have been proposed that try to incorporate other knowledge sources and interaction strategies into a conventional navigation system.

1.2 HYBRID NAVIGATIONAL APPROACHES

A great deal of the research on hybrid navigational approaches to information retrieval has concentrated on the combination of browsing and querying. One typical choice is to maintain different retrieval methods in parallel, for instance a hierarchical browsing system and a boolean query system (Maarek, Berry & Kaiser, 1991). In this case, the integrated system is, in practice, like a switch whereby the user may select either strategy. A tighter form of integration is achieved by cascading the two strategies. A well known approach is that of browsing through a term space to refine the queries to submit to a retrieval system operating in a distinct document space (Frei & Jauslin, 1983; Pedersen, 1993). Alternatively, one can use query prior to browsing. For instance, Lucarella (1993) presents a graph-based object model in which browsing is applied to pattern instances that have been formerly extracted from the database schema using query-patterns. In this kind of combined schema the output of the first module is used as an input to the next module to focus its search.

A more active approach is presented by Thompson and Croft (1989), where several knowledge sources, including browsing through document association maps and cluster-based searches, are combined through a blackboard architecture. The system provides guidance as to which action to perform next based on an evaluation of the system user interaction. In particular, browsing becomes available after the other search methods have failed to achieve acceptable results, and subject to the system's assessment of a user profile.

One practical major disadvantage of most such integrated approaches is that the user must map different representations and results, while the system has to maintain different structures. An approach that helps overcome these problems is presented in the works of Godin, Missaoui, and April (1993) and Agosti, Melucci and Crestani (1995), where browsing and query formulation can be combined to search a unique retrieval space including both terms and documents. This is a deep form of integration, because the two strategies share the same data space and exchange their search results. A similar integrated paradigm is also adopted for networked information retrieval by many systems available on the Internet, in which a basic navigational paradigm, such as Gopher (based on hierarchical file structure) or WWW (based on hypertext), has been complemented with gateways to other retrieval-oriented tools working on a homogeneous information space. Veronica, for example, indexes the file and menu names of Gopher servers, LYCOS indexes the title, content and URL of WWW documents.

The development of hybrid strategies can also be seen as an attempt to bias the search towards the user needs. This is consistent with arguments that contextual sources should play a larger role in information retrieval (Salton & Buckley, 1990) as well as in interface design (Totterdel et al, 1990). This question has been addressed in some recent works aimed at developing user model based navigational aids. For instance, in the system described by Kaplan, Fenwick and Chen (1993), users' specific information needs and preferences are explicitly taken into account - via simple associative matrices - to help them find their way through a browsable network. Another example is the work of Lokuge and Ishizaki (1995), where the authors advocate the use of a learning mechanism to direct the search towards the information space regions chosen by the user in past interactions.

Adding query-based search mechanisms and introducing user information into link-based search are seen, therefore, as two primary means for building more general and effective navigation systems. Our approach, described in the next section, shares the same perspective.

1.3 OUR APPROACH

The system that we present consists of two main stages. The first stage is responsible for building a lattice representation of text databases. Given any collection of texts, we first automatically extract a set of indices representing each text; then we build a particular network of clustered texts - the Galois lattice of the text-index relation - characterizing the whole database. The resulting structure supports the second stage of our system, a visual interface for exploring and retrieving the information contained in the lattice representation. In this paper the major emphasis will be on the latter stage, named ULYSSES.

ULYSSES is based on a hybrid navigational paradigm that combines the two main interaction methods seen above, namely browsing and querying, and a third interaction method, based on *bounding*. Bounding, like organizing/browsing and querying, is a general search technique to find the sought elements of a set without exhaustively enumerating all its elements. It is based on the utilization of the available knowledge about the problem and works towards progressively reducing the original set into one or smaller subsets until their elements can be easily evaluated. Although the central idea of bounding is at the heart of many search algorithms developed in such diverse fields as operations research, artificial intelligence, and dynamic programming (e.g., Balas, 1968; Mitchell, 1982; Kumar, 1984), it has been little explored in the navigational retrieval context. We have applied bounding to the structure being used to retrieve information, with the aim of complementing the other two main interaction strategies. In our approach, browsing and querying are used to search the current retrieval space by focusing on some regions of it, while bounding may be incrementally used to restrict the current retrieval space, based on the knowledge about the goal or the domain that the user has or learns from the system feedback. The three interaction modes work on the same retrieval space, using a single representation of results; therefore, as will be seen more fully later, they can be combined in various ways by the user to form multiple retrieval strategy interaction sequences.

To summarize, our approach presents three main features:

- The construction of the network supporting information retrieval is fully automatic, and it does not require any preconstructed knowledge structures, or any unrealistic reliance on the availability of a set of index terms.

- The retrieval interface provides a novel interaction mode (i.e., bounding) that facilitates the exploitation of personalized knowledge into the search process. The concern is the same as that expressed by Kaplan, Fenwick and Chen (1993) and Lokuge and Ishizaki (1995), but rather than recommending to users the best paths through an underlying fixed network, ULYSSES provides users with the ability to restrict the network itself.
- ULYSSES takes the multiple interaction paradigm one step further, for it combines browsing, querying, and bounding into a single search framework. As a result, users may dynamically choose a hybrid search strategy that best reflects their goals, their domain knowledge, and the results of the interaction. The advantages include greater flexibility - the system may be used for a spectrum of tasks ranging from casual inspection to subject searching - and good retrieval capabilities.

We have to emphasize that using Galois lattices for supporting browsing retrieval is not a new idea (Godin et al, 1986; Godin et al, 1989; Godin et al, 1993). This research takes a step forward, extending previous work in many significant ways. The main enhancements are the introduction of the bound facility and the integration of this and several other useful features, such as automatic indexing, fisheye view browser for lattice, and use of thesaurus, into a basic lattice framework.

A major part of this paper is, then, an empirical evaluation that supports our previous claim that the system has good retrieval capabilities. We report the results of an experiment on subject searching in two reasonably-sized databases, where our system compared favourably with a conventional Boolean retrieval system. In the next two sections we describe each stage of our system in detail; we start with the construction of the supporting network.

2. Building lattice representations of text databases

2.1 AUTOMATIC TEXT INDEXING

The first task for building a global representation of a text database is to identify the content of each text. Basically, there are two kinds of approaches. In the AI-based approach, natural language processing or machine learning techniques (e.g., Sowa, 1984; Srihari & Burhans, 1994; Baudin, Pell & Kedar, 1994) can be used to build or refine an internal representation of each text. Although most of these methods can produce *deep* conceptual indices, they can only work in restricted environments and usually require extensive knowledge about the semantics of the application domain. An alternative approach to content extraction that is domain-independent, knowledge-free, and mostly effective is adopted, with some variants, in most information retrieval systems. Our indexing procedure is inspired by the latter approach. It consists of the following steps.

1. *Text segmentation***.** Our system first identifies the individual words occurring in a text collection, ignoring punctuation and case.

2. *Word stemming*. We reduce each word to word-stem form. This is done by using a very large *trie*-structured morphological lexicon for English (Karp et al, 1992), that contains the standard inflections for nouns (singular, plural, singular genitive, plural genitive), verbs (infinitive, third person singular, past tense, past participle, progressive form), adjectives (base, comparative, superlative).

3. *Stop wording*. We use a stop list to delete from the texts the (root) words that are insufficiently specific to represent content. Our stop list, included in the CACM dataset, contains 428 common function words, such as *the, of, this, on*, etc. and some verbs, e.g., *have, can, indicate*, etc.

4. *Word weighting*. For each document, we derive a measure of the usefulness of each remaining word for indexing purposes. The goal is to identify words that characterize the document to which they are assigned, while also discriminating it from the remainder of the collection. In fact, most of the weighting functions that have been proposed tend to favor terms relatively frequent in the document, relatively rare outside of the document. To compute the weight of term k in document i (w_{ik}) for a collection of n documents we use the following function, which has an information theory basis (Salton and McGill,1983):

$$w_{ik} = \mathrm{FREQ}_{ik} \cdot SIGNAL_k,$$

where FREQ_{ik} is the frequency of term k in document i,

$$SIGNAL_k = \log_2(TOTFREQ_k) - NOISE_k \quad \text{, and}$$

$$NOISE_k = \sum_{i=1}^{n} \frac{FREQ_{ik}}{TOTFREQ_k} \log_2 \frac{TOTFREQ_k}{FREQ_{ik}} \quad .$$

5. *Word selection*. For each document we select as indices those words whose weight value is one standard deviation above the mean; i.e., such that $w_{ik} > \overline{w}_i + \sigma_i$, where $\overline{w}_i$ represents the mean and σ_i the standard deviation of the distribution of w_{ik} values within document i. This is a classical threshold that has already been used for index selection by another automatic indexing system producing a different type of indices (Maarek, Berry & Kaiser, 1991). Of course, restricting the index set by some heuristic thresholds may affect the

subsequent retrieval process; however, it should be noted that this is probably not a real limitation, for it has been often remarked that working with a larger set of indices does not necessarily improve retrieval performance (e.g., Voorhes, 1994; Salton, 1989).

This indexing method combines simple and well known linguistic and statistical concepts, nevertheless it is sufficient to accomplish our goal, i.e., assigning to each document an informative but restricted set of indices suitable for the subsequent processes of cluster formation and visualization. We must also emphasize that the basic method could be easily extended to incorporate more advanced features without harming its generality and efficiency: for instance, we could add vocabulary normalization tools such as synonym list and thesaurus associations, or use adjacent words to form term-phrases (Maarek, Berry & Kaiser, 1991; Chen, Hsu, Orwig, Hoopes & Nunamaker, 1994). On the other hand we should note that there is some evidence that, in many practical situations, the use of more elaborate systems than single-word term extraction did not result in significant performance improvement (Salton, 1989).

After the system has determined a set of indices describing each text, it builds a cluster network characterizing the whole database. This issue is discussed in the next section.

2.2 LATTICE CONCEPTUAL CLUSTERING OF INDEXED TEXTS

Our approach is based on a clustering structure called concept (or Galois) lattice. Given a binary relation between a set of documents and a set of terms, whether automatically or manually built, the Galois lattice is a particular set of clusters, in which each class is a pair, composed of a subset of documents (D), called extent, and a subset of terms (T), called intent. Each pair (D,T) must be a complete pair, meaning that T must contain just those terms shared by all the documents in D, and, similarly, the documents in D must be precisely those sharing all the terms in T. The set of pairs can then be ordered by applying the standard set inclusion relation to the set of terms (or, dually, to the set of documents) that describe each pair. The resulting ordered set, usually represented by a Hasse diagram (i.e., a diagram in which there is an edge between two nodes if and only if they are comparable and there is no other intermediate cluster in the lattice), turns out to be a lattice (Davey & Priestley, 1990). To illustrate, consider the simple database of Table 1, containing nine documents described by nine index terms. The corresponding Galois lattice is shown in Figure 1.

Given the definition of Galois lattices, we addressed the problem of their automatic determination. We implemented in a system named GALOIS an algorithm that builds the lattice incrementally, where each update takes time proportional to the number of documents to be clustered. We also studied the space complexity of Galois lattices, and found empirical and theoretical evidence that, when the number of index terms per document is bounded, the size of the lattice grows linearly with respect to the number of documents. A detailed explanation of Galois lattices, of their complexity and of the construction algorithm is contained in (Carpineto & Romano, 1996a); in the same paper we describe also a generalization of Galois lattices that can take into account semantic information over the terms describing the documents.[1]

A Galois lattice presents many useful properties for supporting information retrieval. In particular, compared to the statistical clustering methods that have been predominately used in information retrieval (e.g., Willet, 1988; Crouch, Crouch & Andreas, 1989; Maarek, Berry & Kaiser, 1991),[2] this approach presents two main features. The first is that each cluster is described by an intensional (or conceptual) description, rather than by just the set of documents covered by the cluster; the second is the possession of a clear semantics by which to characterize the whole structure in terms of the document description. The retrieval interface that we describe in the next section is largely based on these properties.

TABLE 1

FIGURE 1

3. Lattice-based hybrid navigation

To enable the interaction between the user and the lattice we have implemented a prototype visual interface on top of GALOIS, named ULYSSES.[3] It consists of four main components, namely one visualization module and three interaction modules, for browsing, querying, and bounding. In the next subsections we describe each facility in detail, and show how they can be combined to form hybrid search strategies.

3.1. VISUALIZATION

The first problem in the interface design is the visualization of the retrieval space. In general, the lattice representation of a document collection is too large to fit on a screen, even for small databases. To visualize large structures, one simple approach is to allow for multiple views, such as local and global views (e.g., Crouch, Crouch & Andreas, 1989;

Wille, 1989), but this has the disadvantage that the user must map different graphical representations. To make the transition from local to global information more smooth and continuous, several *focus+context* techniques have been recently developed (e.g., Mackinlay, Robertson, & Card, 1991; Lamping, Rao, & Pirolli, 1995), the best known of which is probably the generalized fisheye view (Furnas, 1986; Sarkar & Brown, 1994). ULYSSES employs a variant of this latter approach. In the basic fisheye view approach there is a current focus of interest in the graph to be displayed and the information is shown in varying levels of detail depending on the distance from the focus. In ULYSSES these properties have a simple and natural counterpart. The current focus is represented by the last node selected by the user through one of the three interaction modes, as will be better explained below. The distance between a given node and the current focus is the length of the shortest path between the two nodes. As for the levels of detail, we have defined four types of display involving different styles, sizes, fonts and types of information: we start with the focus node (large size, large font, bold, full information) and progressively reduce the node size, the font size and the amount of information displayed, until we reach the less detailed nodes, painted as small boxes containing only the number of associated documents.

The main difference between our approach and a basic fish eye view approach is that ULYSSES does not have to display the whole structure. As the lattice is used for information retrieval, one can argue that it is probably not so important to see the entire lattice while focusing on some particular node. Indeed, the information relative to global parameters, such as the lattice size or its full topology, may be of little help in the retrieval process. Rather, the user will want to focus on some selected node and browse through the nodes in the adjacent region, containing similar information. Therefore, in trading off focus and context information ULYSSES emphasizes the former; only a part of the region around the focus is usually displayed, subject to two parameters controlling its size and topology.

The detailed working of the visualization module of ULYSSES is the following. There are two procedures: the first computes the graph to draw, the second draws it. To compute the graph, one level at a time is generated; the first level is just the focus node, the n-th level is obtained by the (n-1)th level considering for each node in the level n-1 all the adjacent nodes that have not already been generated. Before adding another level to the graph to draw, the algorithm checks that the size of the resulting graph does not exceed a constant. The size is computed by a weighted sum of all the nodes in the graph, where the weight of each node is proportional to its size. Also, while computing each level the algorithm checks whether the number of nodes adjacent to each single node to expand exceeds another constant, in which case they are not included in the set of nodes to draw (instead, the node to expand will be simply labelled with the number of its successors when drawing the graph). Once we have decided which nodes to draw using which format (in practice each level is assigned a format, as explained above), the actual layout is produced by a standard graphical routine available on the Symbolics Lisp Machine. Although the routine does not produce particularly informative layouts (for instance it allows the user to control sizing but not positioning of the elements of the graph), it is very simple and efficient to use. In order to produce a more aesthetic and fisheye-oriented layout we should resort to more sophisticated algorithms for painting graphs (Eades & Tamassia, 1989), or for manipulating the attributes controlling the graphical layouts (Sarkar & Brown, 1994).

In Figure 2 we show an example screen of the lattice in Figure 1. The current focus is the node: DIAGNOSIS, EXPERT-SYSTEMS, KNOWLEDGE-BASED-SYSTEMS, MEDICINE. The two constants have been set respectively to 30 and 5; with this choice, given the small lattice at hand, ULYSSES displays the whole retrieval space.

FIGURE 2

3.2. BROWSING

ULYSSES' browsing mode takes advantage of the lattice properties. First, the fact that each node can be seen as a query (the intent) with its associated set of documents (the extent) may improve the retrieval of specific information. In particular, this may facilitate the recognition of useful nodes during browsing, as well as improving the efficiency of the query-cluster matching process for automatic search. Another useful property is that the lattice structure, in which there are many paths to a particular node, facilitates recovery from bad decision making while traversing the hierarchy in search of documents, as opposed to the strict hierarchical structures used in most cluster-based browsers, in which each class has exactly one parent. Third, the lattice allows gradual enlargement or refinement of a query. More precisely, following edges departing upward (downward) from a query produces all minimal conjunctive refinements (enlargements) of the query with respect to that particular database (Godin, Gecsei & Pichet, 1989).

In practice, to navigate through the lattice ULYSSES allows the user to select any node on the current screen by direct graphical manipulation (Shneiderman, 1987), i.e. by pointing and clicking with the mouse on the desired node. The selection of a certain node makes it the new focus; as a consequence, the retrieval space is redrawn around the node selected. The user may also see the documents associated with *any* node on the current screen, without changing the current focus.

3.3. QUERYING

The background menu of ULYSSES allows selection of the other two basic interaction modes: querying and bounding. A new query can be formulated in two ways: either the user specifies the new terms from scratch, or the user modifies the current query (i.e., the intent of the current focus). In the latter case the user can remove a term, add a new term, or specialise/generalise a term using the information contained in the thesaurus. The result of a query is the node of the lattice (whose intent is) equal to the query, if there is any, or one or more nodes that partially match the query. The best partially matching queries are determined in the following way. The algorithm first computes the most general nodes that are more specific than the query. If none is found, it finds the most specific more general nodes. If neither case occurs, which means that all the nodes in the lattice are incomparable to the query (according to the lattice ordering relation), the algorithm seeks *similar* nodes. It finds all the nodes that have the maximum number of terms in common with the query and then returns the most general of them. Thanks to the intensional description of the nodes and to their ordering, the algorithm is very efficient: it requires matching the query against a limited number of nodes in the lattice, and each matching usually involves short conjunctions of terms.

The query mode allows the user to make large jumps to regions of interest; also, it can be seen as a way to explore the relationships between the properties of the data set. An alternative approach to query-driven exploration that does not require the construction of the entire set of relationships in advance, as in ULYSSES, is described by Williamson and Shneiderman (1992) and Ahlberg and Shneiderman (1994). One interesting thing about this approach is that it is particularly suitable for ordinal attributes, while ordinal attributes cannot be easily dealt with by ULYSSES (see Wille (1992)). Therefore it could be employed to filter the documents in the initial database - using such information as year and length of documents - prior to the application of ULYSSES.

3.4. BOUNDING

Bounding allows users to change the space from which they are retrieving information during the search. The user may apply constraints with which the sought documents have to comply and the retrieval space is bounded accordingly. The constraints are expressed as inequality relations between the description of admissible clusters and a particular conjunction of terms. For the sake of illustration we will assume that such conjunction of terms coincides with the intent of some node in the lattice, but we have to emphasize that it can be *any* conjunction of terms. Let c be an admissible cluster and c_1 be a particular cluster of the search space. In our framework there are four possible kinds of constraints: $c \geq c_1$, $c \leq c_1$, $c \neg\geq c_1$, $c \neg\leq c_1$. These constraints have an immediate graphical interpretation in terms of the partitions they induce over the search space, as shown in Table 2; they cause the white regions to be pruned away from the space, thus restricting the search to the gray regions. When the search space is a Galois lattice, the four constraints seem to express also interesting properties of the clusters of documents from the point of view of their information retrieval performance. Two constraints, namely $c \leq c_1$ and $c \neg\leq c_1$, seem to be particularly useful, while the other two have less obviuos interpretations (see Carpineto and Romano (1994). The constraint $c \leq c_1$ can be used to restrict the search to the documents having c_1; the constraint $c \neg \leq c_1$ can be used to prune away from the search space all the documents having c_1. It should be noted that the application of a constraint, in addition to pruning some nodes from the lattice, may also change the extent of the remaining admissible nodes. In particular, the application of a constraint of type $c \geq c_1$ or $c \neg\leq c_1$ causes not only the elimination of the nodes that are, respectively, not above or below c_1, but it also eliminates the documents containing c_1 from the extent of the remaining nodes.

TABLE 2

Of course this is only an abstract definition of the bound framework. The next step is to find an algorithm which, given a set of constraints, is able to incrementally represent and update the constrained space. In fact, we developed such an algorithm and described it elsewhere (Carpineto & Romano, 1994). For the scope of this paper, suffice it to say that the algorithm employs a particular representation of the constrained space realized by two boundary sets, one containing the most specific elements of the space (i.e., the lower boundary set) and the other containing the most general elements of the space (i.e., the upper boundary set). This representation is compact and supports efficient update (i.e., proportional to the square of the cardinality of the boundary sets). As more and more constraints are added the admissible space shrinks, and the two boundary sets may eventually converge to the target class. It may also be the case that the asserted constraints turn out to be too strong given a particular lattice, thus causing the admissible space to contain too few documents, or even making it empty. To recover from this situation ULYSSES allows the user to retract previously specified constraints.

Bounding the search space has of course a direct effect on browsing and querying, in that it only allows the user to jump to nodes that are within the admissible region, but it may also change the space visualization. This happens whenever the current focus, as an effect of the new constraint(s), is no longer admissible; in this case, ULYSSES makes the node(s) of the nearest boundary set the new focus.

Before closing this section, it is useful to clarify the relation between querying and bounding, and their relative scopes, in the search process. In both interaction modes the user provides some information about the goal, and the system focuses the search on some relevant space region; but there are two main differences. The first is that the description languages are usually different, and therefore some information may be expressed in one mode but not in the other. For instance, by applying the constraint $c \neg \leq c_1$ we can specify the target nodes using negated terms, which is forbidden in the strictly conjunctive language of ULYSSES' query mode. The second difference is that the two strategies take advantage in different ways of the same piece of information. Suppose that users are interested in the documents containing c_1. Users may query the system by questioning c_1, or they may apply the constraint $c \leq c_1$. In both cases the likely result is a jump to some nodes containing c_1, but in the bounding mode the change will also affect later retrieval. The advantage of bounding is that the whole search becomes more focused, the disadvantage is that the user is no longer able to retrieve relevant documents in the region that has been pruned away. Therefore bounding is more useful when the goal is very precise or when the user becomes more aware of the terminology/structure of the database, later on during the search. An additional advantage of bounding in ULYSSES is that documents violating the constraints are removed from all the nodes in which they are contained, thus reducing the time necessary to scan the documents associated with each admissible node.

In the next section, we illustrate by a simple interaction session how the different strategies of ULYSSES can complement each other.

3.5 FORMING HYBRID RETRIEVAL STRATEGIES

At this point, it should be clear that at any given time ULYSSES is in a certain state, characterized by a current retrieval space and by a focus within it. In each state, the user may select an operator (browsing, query, or bounding) and apply it. As a result, ULYSSES makes a transition to a new state, possibly characterized by a new retrieval space and/or a new focus. At this point the user evaluates the new state for document retrieval, and iterates this basic cycle. In fact, such an approach has many similarities with the GOMS user's cognitive model described by Card, Moran & Newell (1983), and with the theory of user activity presented by Norman (1986). Therefore each interaction sequence may be composed of several operators, connected in various orders. For instance, users may initially bound the search space exploiting their knowledge about the goal, then query the system to locate a region of interest within the bounded space, then browse through the region; also, at any time during this process, they may further bound the retrieval space based on the feedback obtained during the interaction.

As an example of the user forming hybrid retrieval strategies, consider the following scenario. Suppose that the user wants to find the documents contained in the lattice database in Figure 1 that concern 'applications of expert systems to new domains'. This kind of request cannot be easily translated into a query matching the given indexing language; in fact, it seems to require some form of direct inspection of the database. The user might begin the session submitting the query EXPERT-SYSTEMS; in response, the system would return the screen shown in Figure 2, whose focus is the most general node containing EXPERT-SYSTEMS. This display reveals much information relevant to the specific user question that would have been otherwise difficult to acquire. On one hand, it shows that there are only two documents of the database indexed by EXPERT-SYSTEMS, while there are many more documents (eight) indexed by a similar term (i.e., KNOWLEDGE-BASED SYSTEMS); also, it suggests that a large percentage of the documents in the database deal with an application domain that is not relevant to the user, i.e., medicine. At this point the user might exploit this information by inputting two constraints, namely $\leq$ KNOWLEDGE-BASED SYSTEMS and $\neg \leq$ MEDICINE. Figure 3 shows how the user can actually do so. Consequently, the retrieval space is bounded as shown in Figure 4, and the new focus, consisting of two nodes, is the lower boundary set of the bounded representation. The new screen displays one node that is presumably of interest to the user (i.e., the one with the document about knowledge-based systems for cooking), and that had not been fully visualized before.

FIGURE 3

FIGURE 4

4. Experimental evaluation

4.1 GOAL

The objective of the experiment was to evaluate the effectiveness of ULYSSES on a typical retrieval task: subject searching. Although our system has both browsing potentials and direct retrieval capabilities we concentrated on the latter, partly because pure browsing is difficult to evaluate, partly because the retrieval effectiveness of most structure-based approaches to information retrieval is generally seen as unsatisfactory compared to more conventional query-based systems. In the experiment we compared ULYSSES with a Boolean retrieval system; we chose a Boolean system because it is easy to implement, and it is known to perform reasonably well on this task.

4.2 SUBJECTS

We tested four subjects in the experiment. The subjects were graduate students in computer science with little knowledge of the document domains and no prior knowledge about the systems. The subjects were provided with a tutorial session of about an hour for each system on a training database, and were offered more training if they desired it. Great attention was paid to ensure that at the end of the training they could easily manipulate the interface for specifying Boolean queries and that they could make full use of the lattice system facilities, includind the bound mode. Fifty dollars was paid to each subject for his participation.

4.3 DATABASE AND QUERIES

We used two databases in our experiment, INSPEC-AI and CISI.
INSPEC-AI is a collection of 1555 documents extracted from INSPEC, a commonly used large computer-engineering collection. We queried INSPEC by questioning "artificial intelligence", and selected the 1555 most recent (as of January 1994) elements out of some 10,000 documents retrieved. We chose this size because it is large enough for the test to be considered significant, and small enough for the relative lattice to be computed, stored, and accessed easily. The documents were described by a title, an abstract and a set of terms. In order to deal with a controlled and compact vocabulary we used only the terms labelled as *preferred*. In addition, we expanded each term using its broader terms according to the broader/narrower relation among preferred terms given in the 1991 INSPEC thesaurus.[4] With this enlarged set of keywords each document was described by 10.15 terms on average. The corresponding document by term matrix was used to generate both the Boolean and the lattice search spaces. The lattice contained 8769 nodes, with an average of 3.11 parents per node

and path lengths ranging from 2 to 15 edges from the lattice's top to the bottom node; each node was described by an average of 3.74 terms (7.42 if we consider also the broader terms) and 6.07 documents. In order to perform subject searching on the INSPEC-AI database, we manually produced a set of 20 queries (see Carpineto and Romano (1996a) for their list). We also manually computed for each query its relevance judgements, i.e., the associated set of relevant documents. The average number of relevant documents for the 20 queries was 30.5.

The database CISI is a widely used, electronically-available bibliographical collection of 1460 information science documents, described by a title and an abstract, but without index terms. The CISI database was first automatically indexed as specified in section 2.1. It initially contained 184,584 words, 9706 of which were distinct. After stemming and stop wording the database contained 90,154 words, with 7598 distinct words. By word weighting and word selection we finally assigned to each document an average of 6.2 terms. The lattice built from this document by term matrix contained 3740 nodes, with an average of 2.75 parents per node and a depth ranging from 2 to 7 edges; each node was described by 3.14 terms and 4.47 documents on average. Along with the CISI database comes a set of 35 queries with their relevance judgements. For the experiment we randomly selected 20 queries among them; the average number of relevant documents for the 20 queries was 39.

4.4 SOFTWARE

In the choice of software we considered both experimental constraints and interface issues. We implemented a Boolean retrieval system that employs the same indexing strategy as the lattice system (i.e., the one described in Section 2.1), and that allows the user to formulate Boolean queries and select documents in an iterative fashion. We minimized as much as possible the effect that having different interfaces has on performance: the Boolean system runs on the same machine as the lattice system, and both systems use many identical interaction devices, such as the vocabulary window, the document display-and-selection facility, the thesaurus window, and the history of user queries. In Figure 5 we show an example screen of the Boolean retrieval interface during the search of the CISI documents relevant to the following query: "*How much do information retrieval and dissemination systems, as well as automated libraries, cost? Are they worth it to the researcher and to industry?*". Figure 5 shows in the left upper corner the Boolean query input by the user (i.e., "*cost* AND (*retrieval* OR *library*)"), whose terms were selected from a vocabulary window similar to that shown in Figure 3; Figure 5 also shows a window containing the titles of the documents retrieved by the system in response to the query, and a window displaying the content of one of such documents.

We implemented the Boolean retrieval system ourselves, rather than choosing some commercially available Boolean system, because this would have made it much more difficult to meet our experimental constraints and control the interface effect. For the same reasons did we not choose a more advanced prototype retrieval system, such as SMART (Salton & McGill, 1983) or INQUERY (Turtle & Croft, 1991).

FIGURE 5

4.5 TEST DESIGN

The four subjects were asked to retrieve the documents relevant to the 20 queries using the Boolean method and the lattice method. For assigning the queries to the two retrieval methods we used a repeated-measures design, in which each subject searched each query using each method. To minimize sequence effects (Preece, Rogers, Sharp, Benyon, Holland & Carey, 1994), we divided the four subjects into two pairs of two subjects. One pair then did Boolean retrieval followed by lattice retrieval and the other pair did lattice retrieval followed by Boolean retrieval. We decided which pair did which task first randomly.

During each search the user, who was not asked to finish within a certain time period, could see the abstracts of the documents associated with the visited nodes. The documents judged to be relevant by the user, as well as those scanned during the search, were noted as retrieved. This choice deserves some explanation, because in evaluating the effectiveness of browsing retrieval systems the definition of the retrieved set of documents is usually not obvious. One approach (Godin, Missaoui & April, 1993) is to consider as retrieved documents only the documents that have been seen and judged to be relevant by the user. The disadvantage of this approach is that we might be measuring the users judgement more than the effect of the retrieval method. A more typical choice (e.g., Tague-Sutcliffe, 1992; Turtle, 1994) is to rate a document as a retrieved document as soon as its full description (the abstract, in our case) is recovered, without considering the user’s judgement. We have taken the latter approach.

Since no single evaluation tool provides results which, especially for interactive methods, are truly unbiased, we used different evaluation scenarios and measures. For each search, we measured both the performance in the course of the interaction (i.e., dynamic performance) and the performance at the end of the search (i.e., raw performance).

For the dynamic performance, we measured precision[5] at each retrieved document point; in this case, recall,[6] the other most widely used measure in laboratory experiments, can be simply derived from precision by multiplying the latter by a constant. To average over the set of queries we used the following procedure. For each database and for each retrieval method, we discarded 16 of the 80 available queries, namely the eight queries with the smallest number of retrieved documents and the eight queries with the largest number of retrieved documents. At this point, for each database, we chose as cutoff value the number of documents retrieved by the query with the least number of retrieved documents in one of the two methods. In this way, we obtained a cutoff value of 25 for INSPEC-AI and 30 for CISI.

The measure of the dynamic performance is a direct indication of how the retrieval ability of the two methods varies with the number of retrieved documents, but it says little about the final raw performance and it does not give any indication of the time taken to retrieve the documents themselves, which is an important measure of the user's effort. To account for these aspects, at the end of each search, we measured four variables: precision, recall, search time (i.e., the time taken by the user to perform his task) and number of documents retrieved.

4.6 RESULTS

The results are displayed in Figure 6, Table 3, and Table 4.

Figure 6 shows that the dynamic performance of lattice retrieval was, in general, better than Boolean retrieval. For the INSPEC-AI database, lattice retrieval obtained better average precision at all but four retrieved documents points. The superiority of lattice over Boolean is much more apparent for the CISI database, where lattice markedly outperformed Boolean at each of 31 retrieved documents values. A paired t-test performed over the whole set of data (i.e., values of precision for all queries at all retrieved documents points) confirmed that the difference can be considered statistically significant for CISI ($p = 0.0002$), and, to a lesser extent, for INSPEC-AI ($p = 0.041$). These results are not surprising, because although Boolean queries have a greater expressive power than lattice queries, the lattice method provides other retrieval strategies that may compensate for this limitation. In particular, browsing allows smooth query refinement, while it is well known that in Boolean retrieval the user cannot control the amount of output obtained in response to a query (e.g., Salton & McGill, 1983; Lesk, 1989). Another advantage of lattice retrieval is that the user may exploit the feedback obtained from the structure to facilitate selection of relevant terms in the database, as opposed to Boolean retrieval where this kind of information is not available. Finally, there are two other reasons two explain why the greater expressive power of Boolean querying resulted in worse retrieval performance. On one hand, we observed that, consistently with the evidence shown by Borgman and Meadow (1985) that in Boolean retrieval users prefer conjunctive queries, most (82%) of the Boolean queries submitted by the users during the experiments were conjuntions of few atomic arguments (i.e., simple terms); on the other hand, it should be noted that the possibility of expressing negated terms in the bound mode of lattice retrieval may compensate for the Boolean operator NOT, while performing more lattice searches is equivalent to using the Boolean operator OR.

Table 3 shows that, for the CISI database, lattice retrieval obtained better evaluation scores for each of the raw retrieval performance variables. A paired t-test revealed that the method had no effect on time ($p = 0.17$) or retrieved documents ($p = 0.37$), but it did reveal the superiority of lattice retrieval with respect to recall ($p = 0.038$) and precision ($p = 0.008$). The results shown in Table 4, relative to the raw performance on the INSPEC-AI database, are slightly different. Lattice retrieval obtained better recall and precision values, but performed worse than Boolean retrieval with respect to retrieved documents and search time. In this case, the differences were not statistically significant: $p = 0.30$ for recall, $p = 0.11$ for precision, $p = 0.15$ for retrieved documents, $p = 0.14$ for search time.[7] The raw precision of lattice was greater than Boolean, although by a smaller margin than the CISI database, as was also apparent in the dynamic performance scenario. The values of raw recall were very similar, because Boolean was less precise but it retrieved more documents. The values of retrieved documents and search time were similar across both databases. Roughly, the two methods took a similar time to retrieve a similar number of documents. These results can be better explained by looking at the operations performed by the users during the experiments. In using the Boolean retrieval system, they performed, for each search on one of the two databases, an average of 10.22 queries ($\sigma = 4.41$); on the other hand, in using the lattice retrieval system, they on average performed 6.12 queries ($\sigma = 3.01$), visited 16.13 nodes ($\sigma = 7.58$), and bounded the search space 3.14 times ($\sigma = 1.90$). Therefore, in lattice retrieval the users, on the whole, saw the result of a larger number of queries than in Boolean retrieval while performing a smaller number of direct queries, because most of the queries were explored as a result of the feedback obtained from the structure. On the other hand, the users retrieved a larger number of documents per query when using the Boolean system. The observed behavior seems to indicate that in lattice retrieval the users concentrated on finding good queries, whereas in Boolean retrieval they devoted a larger effort to the analysis of the query results. To sum up, in Boolean retrieval the users were primarily engaged in direct query formulation (which entailed a time consuming scanning of the vocabulary window) and scanning of the query results, while in lattice retrieval they also seemed to spend much time in browse-driven query searches and, to a smaller extent, in constraint specification/retraction.

It is useful to compare these results with those obtained by Godin, Missaoui and April (1993) using a Galois lattice approach with more limited retrieval capabilities. Godin, Missaoui and April (1993) reported the result of an experiment conducted on a small-scale (113 documents) manually-indexed database that showed that lattice retrieval was slightly worse than Boolean retrieval. Therefore, there is a clear improvement in performance when using our system; such improvement is especially important considering that our databases were one order of magnitude larger than Godin et al.'s. In fact, their approach is less powerful than ours for a number of reasons. Godin et al.'s system allows browsing and query formulation, but it has a very simple graphical interface and it does not provide any bound facility. Even its query mode is severely limited, compared to ours, because the terms contained in a query must be a subset of (the intent of) some node in the lattice; the query EXPERT-SYSTEMS(ANALOGY, for instance, would return no answer for the lattice in Figure 1. In addition, Godin et al.'s system cannot use a thesaurus, and it has no indexing mechanism.

FIGURE 6

TABLE 3

TABLE 4

5. Scaling considerations

Although in the experiments we used reasonably-sized databases, their scale is still not large relative to operational environments. In this section we address this issue; i.e., what happens as scale is increased? First of all, it should be noted that there are two sources of limitations that might affect how well our approach could scale up: computational limitations and effectiveness limitations. We will examine each of them in turn.

With the kind of databases that we used in our experiment the construction of the supporting lattice is fast (a few minutes) and the response time of the interface is good (at most a few seconds in the bound mode), but the issue remains as to whether the two main components of the system would scale up to large databases. The interface is less critical. In fact, the complexity of the bound facility, which is the most costly operation in the interaction with the user, depends on the cardinality of the two boundary sets defined in Section 3.4, and this parameter may become exceedingly large only in certain cases (e.g., when a database produces a very large and shallow lattice), while in general it is not directly related to the size of the database. The key limiting factor seems to be the lattice construction. If the number of terms per document is bounded, as usually happens, the lattice can be constructed even from databases containing a significantly larger number of documents than used in the experiments. In contrast, as the number of terms per document increases, there may arise computational problems even for databases of a size similar to those used in the experiments, because the size of the lattice may grow quadratically with respect to the number of objects (Carpineto & Romano, 1996a). However, for the case when the system would run into computational barriers, we could take an alternative approach that may alleviate the problem. Rather than build an entire concept lattice in advance, we could let the user formulate a query first, and then compute, dynamically, a very limited portion of the lattice, centered around the node in the lattice that would match the query (e.g., its parents and children). In fact, such a dynamic approach, that has been recently explored by Carpineto and Romano (1996b), seems to adequately address the scaling issue while retaining the main advantages of the static approach, with the significant exception of the bound facility.

The second source of concern for scalability is effectiveness limitations, because there is some evidence that passing from small-scale to large-scale databases may sharply affect performance results, both in automatic and interactive information retrieval systems (e.g., Blair and Maron, 1985, Turtle, 1994). As the effectiveness of our approach crucially depends on the operations performed by the user, it is useful to see if there are any aspects in the human-system interaction that might be affected by scaling. In particular, what happens to ULYSSES' visual browsing tool in a large environment? Fortunately, the complexity of the displayed graph is not a function of the lattice size, as seen in section 3.1; furthermore, the complexity of the information shown in each node would probably remain low, because it can be seen that even when the set of indices describing each document is large, the mean number of terms per node is usually small (Carpineto & Romano, 1996b). Therefore, the willingness and the ability of the user to read and manipulate the information shown in the browsing interface should not change as a consequence of the shift to a larger environment. This is encouraging, of course, but it is not enough to guarantee that the retrieval performance of the system would be equally good. Working with a large lattice would probably reduce the utility of the bound operations, and it would make it more difficult to find nodes of interest through link-based navigation. As a consequence, the overall retrieval performance might still be badly affected.

6. Conclusion

This paper was roughly split into two parts. In the first part we presented a system for information exploration and retrieval that is based on the organizing/navigating paradigm. The system allows the user to navigate through a particular lattice representation built from a text database. During the navigation, the user may use multiple interaction techniques, which can be combined into a search&bound approach to information retrieval. We discussed the merits of the system both from a computational and a usability point of view. The system works in unrestricted environments and is fully automatic; in other words, it can be applied to collections of texts that vary in subject matter and extent, and it does not require domain-specific preconstructed knowledge structures. One of its most distinguishing features is flexibility. Users can use the system to browse or retrieve information; but this is not like a switch, because they can combine the two types of search and they have more ways of using their knowledge of the goal or domain to influence the information finding.

In the second part of our paper we evaluated the retrieval effectiveness of the system. We compared its performance with that of a Boolean retrieval system on a typical task: subject searching. Our results confirm the intuition of Godin, Missaoui and April (1993) that lattice retrieval can be seen as an alternative to more conventional methods even for pure retrieval, and, in addition, they extend previous results in many significant ways. In particular, we showed that the performance of lattice retrieval may be better than Boolean retrieval with respect to precision and recall, and that this result can be obtained even for databases of non trivial size.

This research can be extended in several directions. In section 5 we discussed the influence of database scale on the system's efficiency and effectiveness, which is a fundamental aspect to evaluate the scope of this approach. In operational situations, however, there may be other important parameters that need be controlled, such as indexing strategies, query characteristics, subject domain, thesaurus availability, interface details, and user profile. While the experiments that we performed may also provide some insigths into the sensitivity of the results to these factors, it is clear that this can only be taken as indicative; one direction for future work is to perform further experiments to evaluate how the performance results change when controlling a wider range of factors including database scale, and characteristics of system, domain, and users.

Another planned research avenue, which also has the advantage of addressing the scalability issue, is to combine our system with the resources available on the Internet. The idea is to use the powerful tools available in World Wide Web for searching large and distributed databases, such as WebCrawler or Lycos, as a pre-processing step, and then apply our system to the set of elements retrieved in this way, which are usually returned in an unindexed and unstructured fashion. In this view, our system will act as an organizer/navigator of the information retrieved after a Web search, with the aim of improving its presentation and facilitating its access.

Acknowledgements

We would like to thank three anonymous reviewers for many useful comments and suggestions. This work was carried out under the auspices of an agreement between Telecom Italia and the Fondazione Ugo Bordoni.

References

Agosti, M., Melucci, M. & Crestani, F. (1995). Automatic Authoring and Construction of Hypertexts for Information Retrieval. *ACM Multimedia Systems*, 3, 15-24.

Ahlberg, C. & Shneiderman, B. (1994). Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. *Proceedings of ACM CHI'94 Conference on Human Factors in Computing Systems*. 313-317. Boston, MA: ACM

Balas, E. (1968). "A Note on the Branch-and-Bound Principle", *Operations Research*, 16, 442-444.

Baudin, C., Pell, B., Kedar, S. (1994). Incremental Acquisition of Conceptual Indices for Multimedia Design Documentation. *Proceedings of the AAAI-94 Workshop on Indexing and Reuse in Multimedia Systems*, pp. 157-163, Seattle, Washington.

Blair, D. C., Maron, M. E. (1985). An Evaluation of Retrieval Effectiveness for a Full-Text Document-Retrieval System. *Communications of the ACM*, 28, 289-299.

Borgman, C. L. & Meadow, C. T. (1985). Designing an Information Retrieval Interface Based on User Characteristics. *Proceedings of the Eight Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 139-146.

Bowman, M., Danzig, P., Manber, U., & Schwartz, F. (1994). Scalable Internet Resource Discovery: Research Problems and Approaches. *Communications of the ACM*, 37, 8, pp. 98-114.

Card, S., Moran, T & Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates: London.

Carpineto, C., & Romano, G. (1994). Dynamically Bounding Browsable Retrieval Spaces: an Application to Galois Lattices. In *Proceedings of RIAO 94: Intelligent Multimedia Information Retrieval Systems and Management*, pp. 520-533, New York.

Carpineto, C., & Romano, G. (1995). ULYSSES: A Lattice-Based Multiple Interaction Strategy Retrieval Interface. In Blumenthal, Gornostaev & Unger (Eds.), *Human-Computer Interaction, 5th International Conference, EWHCI'95, Moscow, Russia, July 1995, Selected Papers*, pp. 91-104, Lecture Notes in Computer Science 1015, Springer-Verlag.

Carpineto, C., & Romano, G. (1996a). A Lattice Conceptual Clustering System and Its Application to Browsing Retrieval. *Machine Learning*, 24, 1-28.

Carpineto, C., & Romano, G. (1996b). Enhancing Boolean Retrieval with Content-Based Query Refinement. Technical Report 5T02696, Rome, Italy: Fondazione Ugo Bordoni.

Chen, H., Hsu, P., Orwig, R., Hoopes, L. & Nunamaker, J. (1994). Automatic Concept Classification of Text from Electronic Meeting. *Communications of the ACM*, 37, 10, 57-73.

Crouch, D., Crouch, C., & Andreas, G. (1989). The Use of Cluster Hierarchies in Hypertext Information Retrieval. *Proceedings of the ACM Hypertext '89 Conference*, pp. 225-237, Pittsburgh, PA: ACM.

Davey, B., & Priestley, H. (1990). *Introduction to Lattices and Order*. Cambridge, Great Britain: Cambridge University Press.

Eades, P., & Tamassia, R. (1989). Algorithms for Drawing Graphs: an Annotated Bibliography. Technical Report CS-89-90, Dept. of Computer Science, Brown Univ., Providence, R.I.

Frei, H. & Jauslin, J. (1983). Graphical Presentation of Information and Services: A User Oriented Interface. *Information Technology: Research and Development*, 2, 23-42.

Frisse, M. E. & Cousins, S. B. (1989). Information Retrieval from Hypertext: Update on the Dynamic Medical Handbook Project. *Proceedings ACM-Hypertext’89*, pp.199-212, Pittsburgh, PA, USA.

Furnas, G. (1986). Generalized Fisheye Views. *Proceedings of ACM CHI'86 Conference on Human Factors in Computing Systems*. pp.16-23. Boston, MA: ACM.

Godin, R., Saunders, E., & Gecsei, J. (1986). Lattice Model of Browsable Data Spaces. *Information Sciences*, 40, 89-116.

Godin, R., Gecsei, J., & Pichet, C. (1989). Design of a browsing interface for information retrieval. *Proceedings of the Twelfth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 32-39. Cambridge, MA: ACM SIGIR Forum.

Godin, R., Missaoui, R., & April, A. (1993). Experimental Comparison of Navigation in a Galois Lattice with Conventional Information Retrieval Methods. *International Journal of Man-machine Studies*, 38, 747-767.

Kaplan, C., Fenwick, J., & Chen, J. (1993). Adaptive Hypertext Navigation Based on User Goals and Context. *User Modeling and User-Adapted Interaction*, 3, 193-220.

Karp, D., Schabes, Y., Zaidel, M., Egedi, D. (1992). A Freely Available Wide Coverage Morphological Analyzer for English. *Proceedings of the 14th International Conference on Computational Linguistics (COLING '92)*, Nantes, France.

Kumar, V. (1984). A General Bottom-up Procedure for Searching AND/OR Graphs. *Proceedings of the Fourth National Conference on Artificial Intelligence*, Austin, TX.

Lamping, J., Rao, R. & Pirolli, P. (1995). A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*. pp.401-408. Denver, Colorado, USA.

Lesk, M. (1989). What To Do When There Is Too Much Information. *Proceedings ACM-Hypertext’89*. pp. 305-318, Pittsburgh, PA, USA.

Lokuge, I. & Ishizaki, S. (1995). GeoSpace: An Interactive Visualization System for Exploring Complex Information Spaces. *Proceedings of the ACM CHI'95 Conference on Human Factors in Computing Systems*. pp.450-455. Denver, Colorado, USA.

Lucarella, D., Parisotto, S. & Zanzi, A. (1993). MORE: Multimedia Object Retrieval Evironment. *Proceedings ACM-Hypertext’93*. pp. 39-50. Seattle, WA.

Maarek, Y., Berry, D., & Kaiser, G. (1991). An Information Retrieval Approach for Automatically Constructing Software Libraries. *IEEE Transactions on Software Engineering*, 17, 8, 800-813.

Mackinlay, J. D., Robertson, G. G. & Kard, S. K. (1991). The Perspective Wall: Detail and Context Smoothly Integrated. *Proceedings of the ACM CHI’91 Conference on Human Factors in Computing Systems*. pp. 173-179. New Orleans, LA, USA.

Marchionini, G. (1992). Interfaces for End-User Information Seeking. *Journal of the American Society for Information Sciences*, 43(2), 156-163.

Marchionini, G., & Shneiderman, B. (1988). Finding Facts vs. Browsing Knowledge in Hypertext Systems. *IEEE Computer*, 21, 70-80.

Mitchell, T. M. (1982). Generalization As Search. *Artificial Intelligence*, 18, 203-226.

Nielsen, J. (1990). *Hypertext & Hypermedia*. San Diego, CA: Academic Press.

Norman, D. (1986). Cognitive Engineering. In Norman, D., Draper, S. (Eds.*): User Centered System Design*, pp. 31-61. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Pedersen, G. (1993). A Browser for Bibliographic Information Retrieval Based on an Application of Lattice Theory. *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 270-279. Pittsburgh, PA.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S. & Carey, T. (1994). *Human-Computer Interaction*. Addison Wesley.

Salton, G., & McGill, M. (1983). *Introduction to Modern Information Retrieval*. New York: McGraw Hill.

Salton, G. (1989). *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison Wesley.

Salton, G. & Buckley, C. (1990). Improving Retrieval Performance by Relevance Feedback. *Journal of the American Society for Information Sciences*, 41, 288-297.

Sarkar, M, & Brown, M. (1994). Graphical Fisheye Views. *Communications of the ACM*, 37, 12, 73-84.

Savoy, J. (1993). Searching Information in Hypertext Systems Using Multiple Sources of Evidence. *International Journal of Man-Machine Studies*, 38, 1017-1030.

Shneiderman, B. (1987). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Reading, MA: Addison Wesley.

Sowa, J. (1984). *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley.

Srihari, R. & Burhans, D. (1994). Visual Semantics: Extracting Visual Information from Text Accompanying Pictures. *Proceedings of the twelfth National Conference on Artificial Intelligence*. pp. 793-798. Seattle, Washington, AAAI Press.

Tague-Sutcliffe, J. (1992). The Pragmatics of Information Retrieval Experimentation, Revisited. *Information Processing & Management*, 28, 4, 467-490.

Thompson, R., & Croft, B. (1989). Support for Browsing in an Intelligent Text Retrieval System. *International Journal of Man-machine Studies*, 30, 639-668.

Totterdel, P., Rautenbach, A., Wilkinson, A. & Anderson, S. (1990). Adaptive Interface Techniques. In Browne, D., Totterdel, P., Norman, M. (Eds.): *Adaptive User Interfaces*. London: Harcourt Brace Jovanovich.

Turtle, H.,& Croft, B. (1991). Evaluation of an Inference Network-Based Retrieval Model. *ACM Transactions on Information Systems*, 9(3), 187-222.

Turtle, H. (1994). Natural Language vs. Boolean Query Evaluation: A Comparison of Retrieval Performance. *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. pp. 212-220, Dublin, Ireland.

van Rijsbergen (1975). *Information retrieval*. London: Butterworths.

Voorhes, E. M. (1994). Query Expansion Using Lexical-Semantic Relations. *Proceedings of the Seventeenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. pp. 61-69, Dublin, Ireland.

Wille, R. (1989). Lattices and Data Analysis: How to Draw Them Using a Computer. In Rival, I. (Eds.), *Algorithms and Order*, pp. 33-58. Boston: Kluwer.

Wille, R. (1992). Concept Lattices and Conceptual Knowledge Systems. *Computers & Mathematics with Applications*, 23 (6-9), 493-515.

Willet, P. (1988). Recent Trends in Hierarchic Document Clustering: a Critical Review. *Information Processing & Management*, 24, 5, 577-597.

Williamson, C. & Shneiderman, B. (1992). The Dynamic HomeFinder: Evaluating Dynamic Queries in a Real-Estate Information Exploration System. *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 339-346, Copenhagen, Denmark.

Footnotes

[1] The basic Galois lattice is a purely syntactic structure, in which the order over the clusters is independent of possible semantic relationships between the terms; however, in the presence of auxiliary information expressed as a broader/narrower relationship between the terms, the lattice can be adapted so that broader terms index more general clusters. The essence of this generalization is that when we compute the terms shared by sets of documents we have to take into account also the terms that are implicitly possessed by each document according to the thesaurus.

[2] The potentials of clustering for information retrieval have long been known, the main justification for this being what van Rijsbergen (1975) termed the *cluster hypothesis*, namely the fact that documents associated in the same clusters tend to be relevant to the same questions.

[3] The main features of ULYSSES were first introduced by Carpineto and Romano 1995. ULYSSES, like the indexing module and GALOIS, has been implemented in Common Lisp, and runs on a Symbolics Lisp Machine. More recently, we have re-implemented the whole system, that consists of about 300 K-bytes of code, on Apple Macintosh.

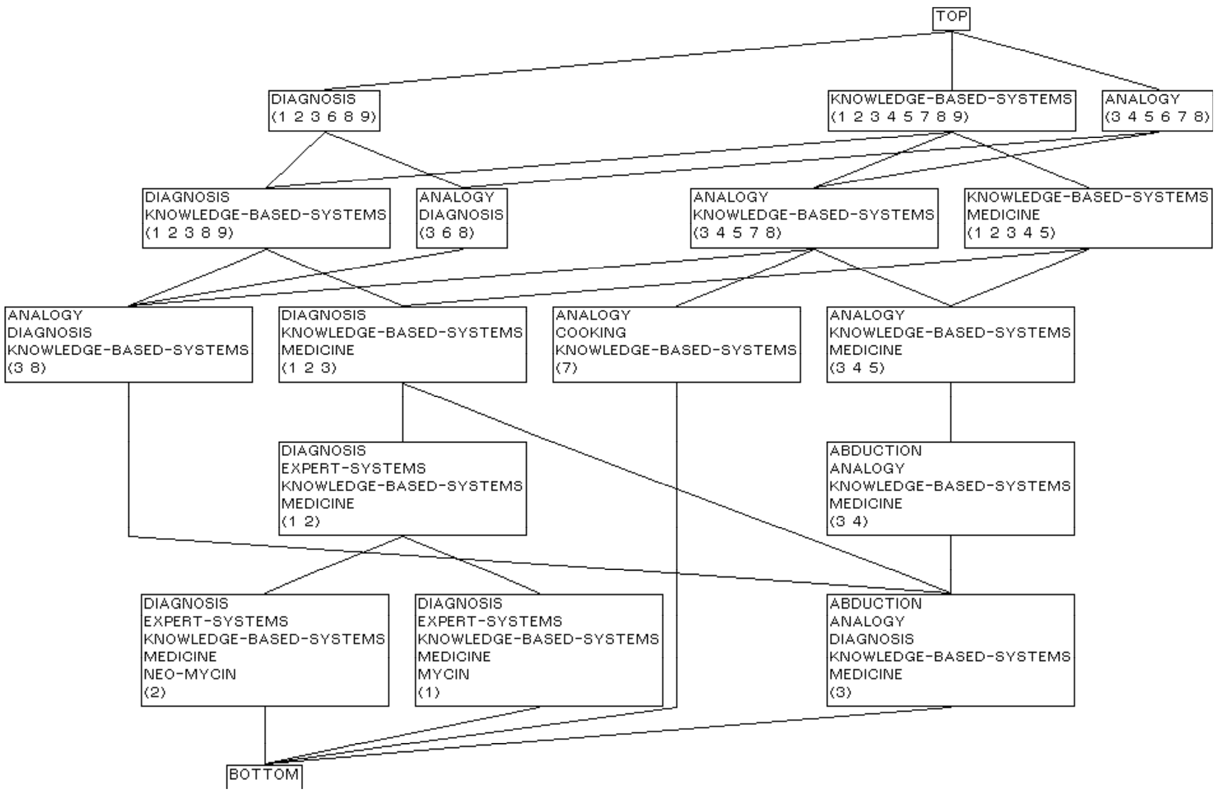
[4] We showed elsewhere (Carpineto & Romano, 1996a) that the utilization of this kind of information may improve effectiveness of lattice retrieval.

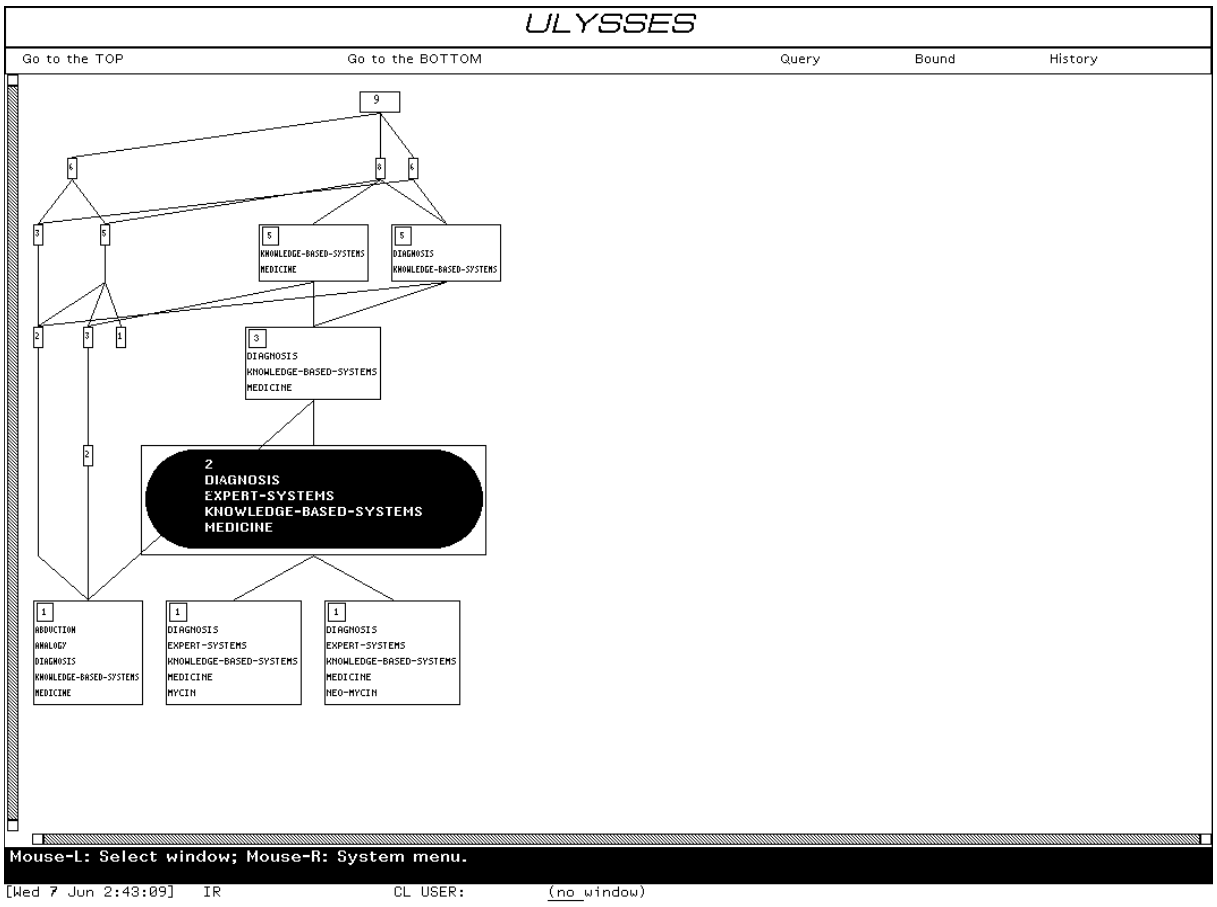
[5]*Precision* is defined as the ratio of number of items retrieved and relevant to the number of items retrieved; precision measures the ability to retrieve *only* relevant documents.

[6] *Recall* is defined as the ratio of number of items retrieved and relevant to the number of items relevant; recall measures the ability to retrieve *all* relevant documents.

[7] These results are different than, but consistent with, the results of an earlier experiment on the same database (Carpineto & Romano, 1995), where we used a smaller set of queries with a somewhat larger average number of relevant documents per query.

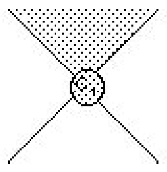
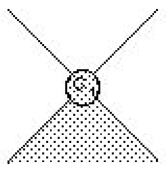
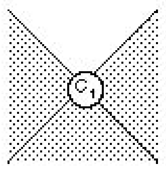
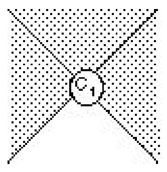
	1	2	3	4	5	6	7	8	9
ABDUCTION			x	x					
ANALOGY			x	x	x	x	x	x	
COOKING							x		
DIAGNOSIS	x	x	x			x		x	x
EXPERT-SYSTEMS	x	x							
KNOWLEDGE-BASED-SYSTEMS	x	x	x	x	x		x	x	x
MEDICINE	x	x	x	x	x				
MYCIN	x								
NEO-MYCIN		x							

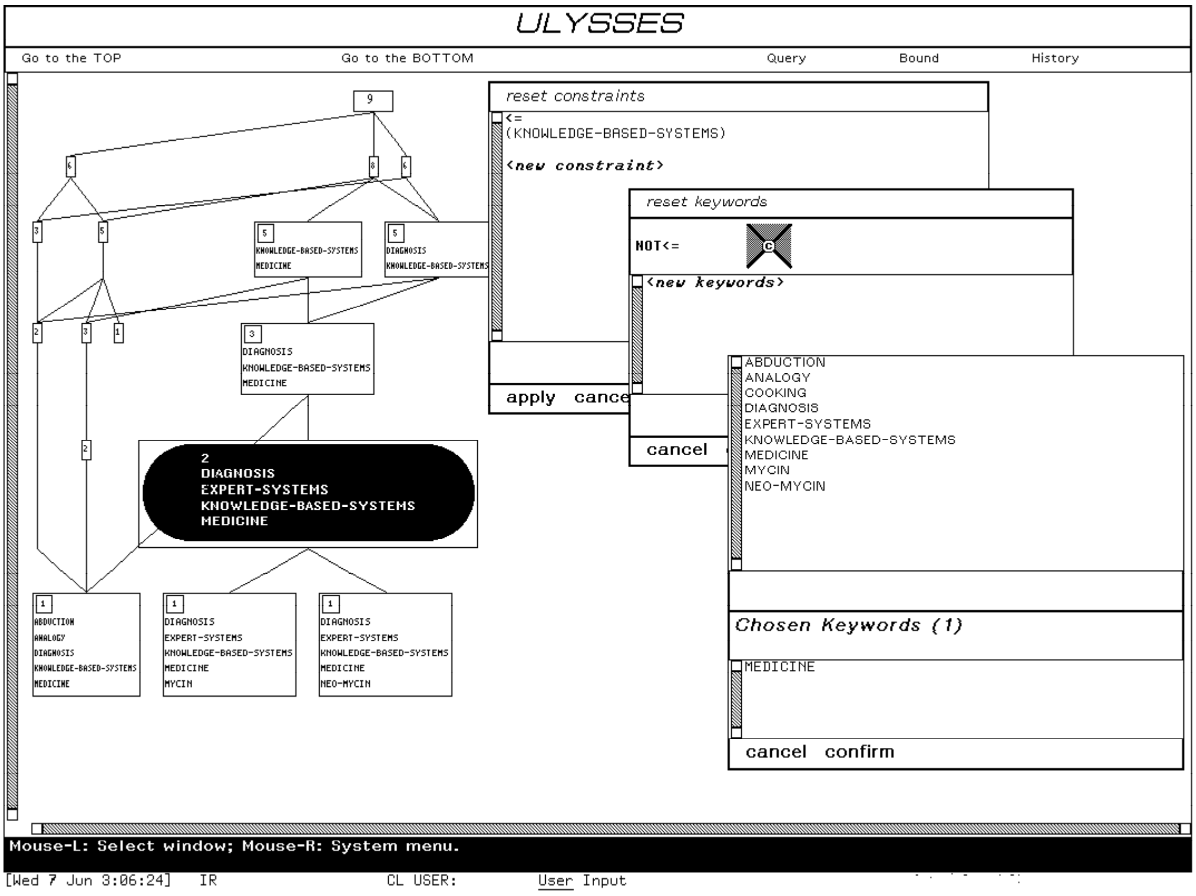

TOP
DIAGNOSIS (1 2 3 6 8 9)
KNOWLEDGE-BASED-SYSTEMS (1 2 3 4 5 7 8 9)
ANALOGY (3 4 5 6 7 8)
DIAGNOSIS KNOWLEDGE-BASED-SYSTEMS (1 2 3 8 9)
ANALOGY DIAGNOSIS (3 6 8)
ANALOGY KNOWLEDGE-BASED-SYSTEMS (3 4 5 7 8)
KNOWLEDGE-BASED-SYSTEMS MEDICINE (1 2 3 4 5)
ANALOGY DIAGNOSIS KNOWLEDGE-BASED-SYSTEMS (3 8)
DIAGNOSIS KNOWLEDGE-BASED-SYSTEMS MEDICINE (1 2 3)
ANALOGY COOKING KNOWLEDGE-BASED-SYSTEMS (7)
ANALOGY KNOWLEDGE-BASED-SYSTEMS MEDICINE (3 4 5)
DIAGNOSIS EXPERT-SYSTEMS KNOWLEDGE-BASED-SYSTEMS MEDICINE (1 2)
ABDUCTION ANALOGY KNOWLEDGE-BASED-SYSTEMS MEDICINE (3 4)
DIAGNOSIS EXPERT-SYSTEMS KNOWLEDGE-BASED-SYSTEMS MEDICINE NEO-MYCIN (2)
DIAGNOSIS EXPERT-SYSTEMS KNOWLEDGE-BASED-SYSTEMS MEDICINE MYCIN (1)
ABDUCTION ANALOGY DIAGNOSIS KNOWLEDGE-BASED-SYSTEMS MEDICINE (3)
BOTTOM



Mouse-L: Select window; Mouse-R: System menu.

```
[Wed 7 Jun 2:43:09]   IR                    CL USER:          (no window)
```

$C \geqslant C_1$	$C \leqslant C_1$	$C \neg \geqslant C_1$	$C \neg \leqslant C_1$
			



reset constraints

```
<=
(KNOWLEDGE-BASED-SYSTEMS)

<new constraint>
```

reset keywords

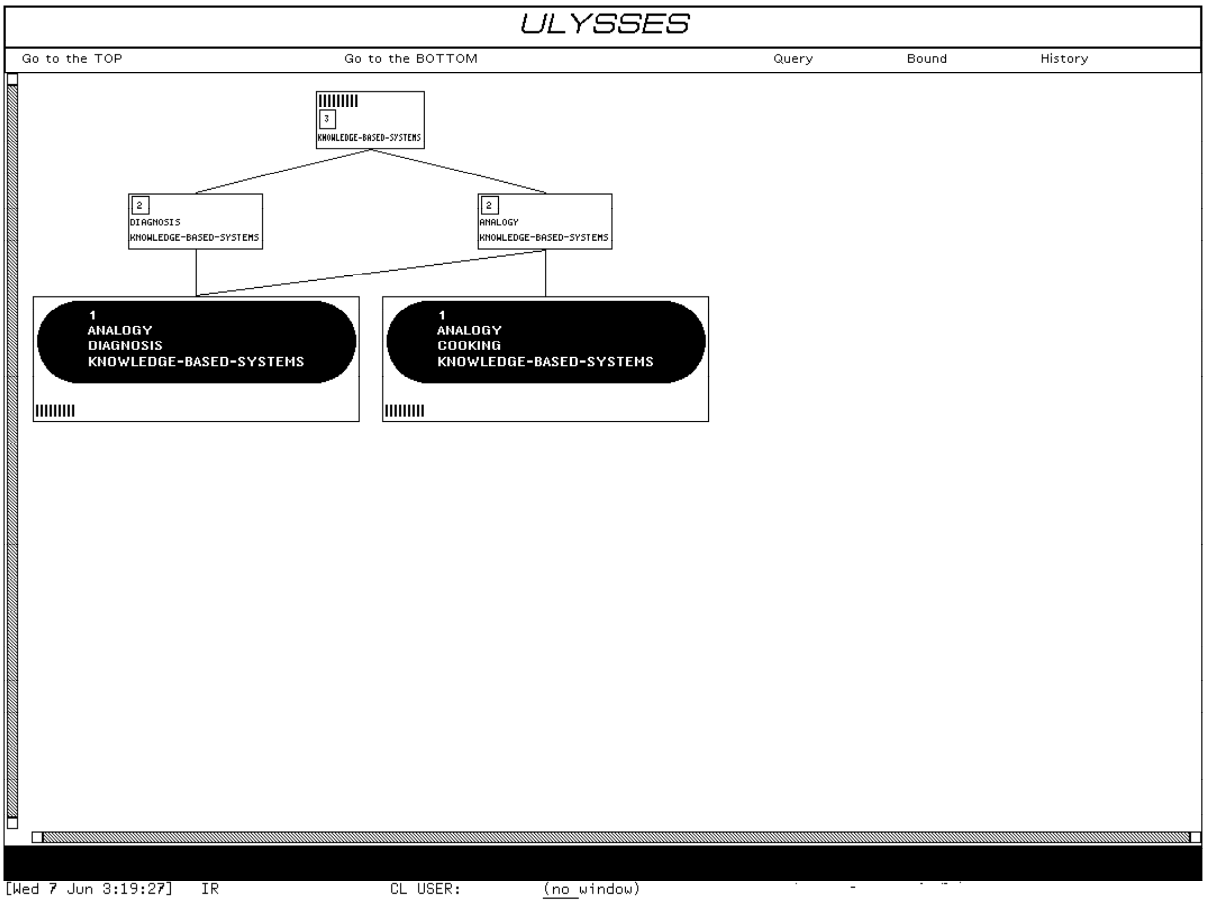
```
NOT<=

<new keywords>
```

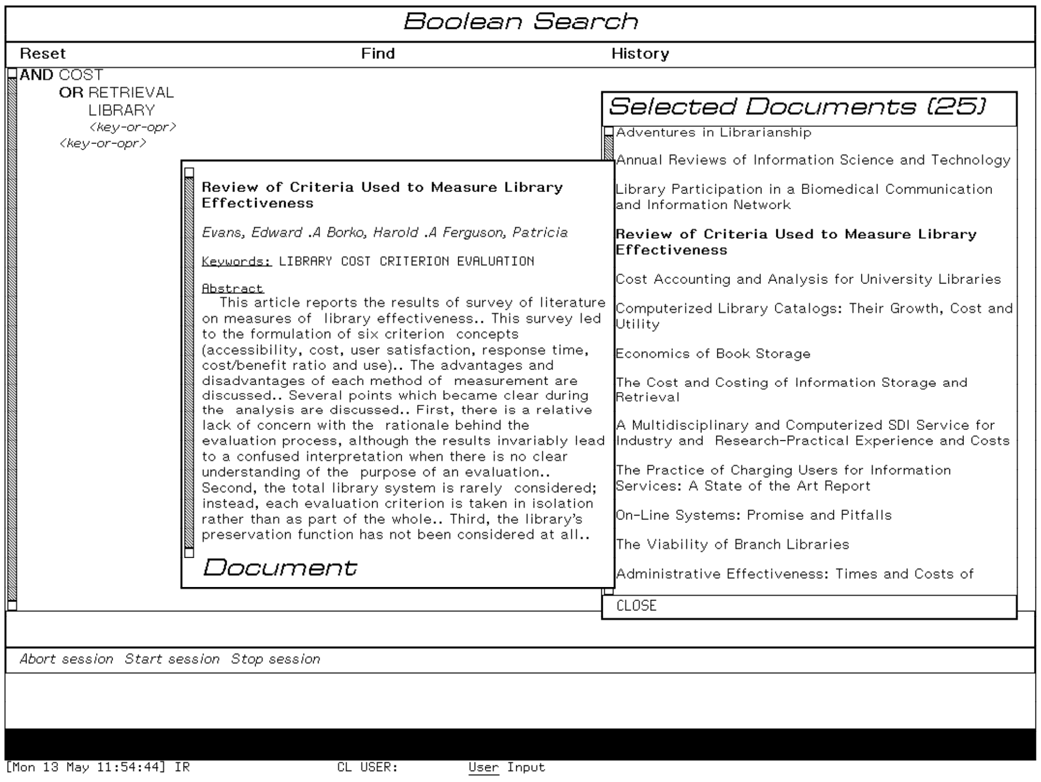
```
ABDUCTION
ANALOGY
COOKING
DIAGNOSIS
EXPERT-SYSTEMS
KNOWLEDGE-BASED-SYSTEMS
MEDICINE
MYCIN
NEO-MYCIN
```

Chosen Keywords (1)

```
MEDICINE
```

```
Mouse-L: Select window; Mouse-R: System menu.
[Wed 7 Jun 3:06:24]   IR                  CL USER:        User Input
```


ULYSSES
Go to the TOP
Go to the BOTTOM
Query
Bound
History
3
KNOWLEDGE-BASED-SYSTEMS
2
DIAGNOSIS
KNOWLEDGE-BASED-SYSTEMS
2
ANALOGY
KNOWLEDGE-BASED-SYSTEMS
1
ANALOGY
DIAGNOSIS
KNOWLEDGE-BASED-SYSTEMS
1
ANALOGY
COOKING
KNOWLEDGE-BASED-SYSTEMS
[Wed 7 Jun 3:19:27]
IR
CL USER:
(no window)



AND COST
    OR RETRIEVAL
        LIBRARY
        <key-or-opr>
    <key-or-opr>

Review of Criteria Used to Measure Library Effectiveness

Evans, Edward .A Borko, Harold .A Ferguson, Patricia

Keywords: LIBRARY COST CRITERION EVALUATION

Abstract
This article reports the results of survey of literature on measures of library effectiveness.. This survey led to the formulation of six criterion concepts (accessibility, cost, user satisfaction, response time, cost/benefit ratio and use).. The advantages and disadvantages of each method of measurement are discussed.. Several points which became clear during the analysis are discussed.. First, there is a relative lack of concern with the rationale behind the evaluation process, although the results invariably lead to a confused interpretation when there is no clear understanding of the purpose of an evaluation.. Second, the total library system is rarely considered; instead, each evaluation criterion is taken in isolation rather than as part of the whole.. Third, the library's preservation function has not been considered at all..

Adventures in Librarianship

Annual Reviews of Information Science and Technology

Library Participation in a Biomedical Communication and Information Network

Review of Criteria Used to Measure Library Effectiveness

Cost Accounting and Analysis for University Libraries

Computerized Library Catalogs: Their Growth, Cost and Utility

Economics of Book Storage

The Cost and Costing of Information Storage and Retrieval

A Multidisciplinary and Computerized SDI Service for Industry and Research-Practical Experience and Costs

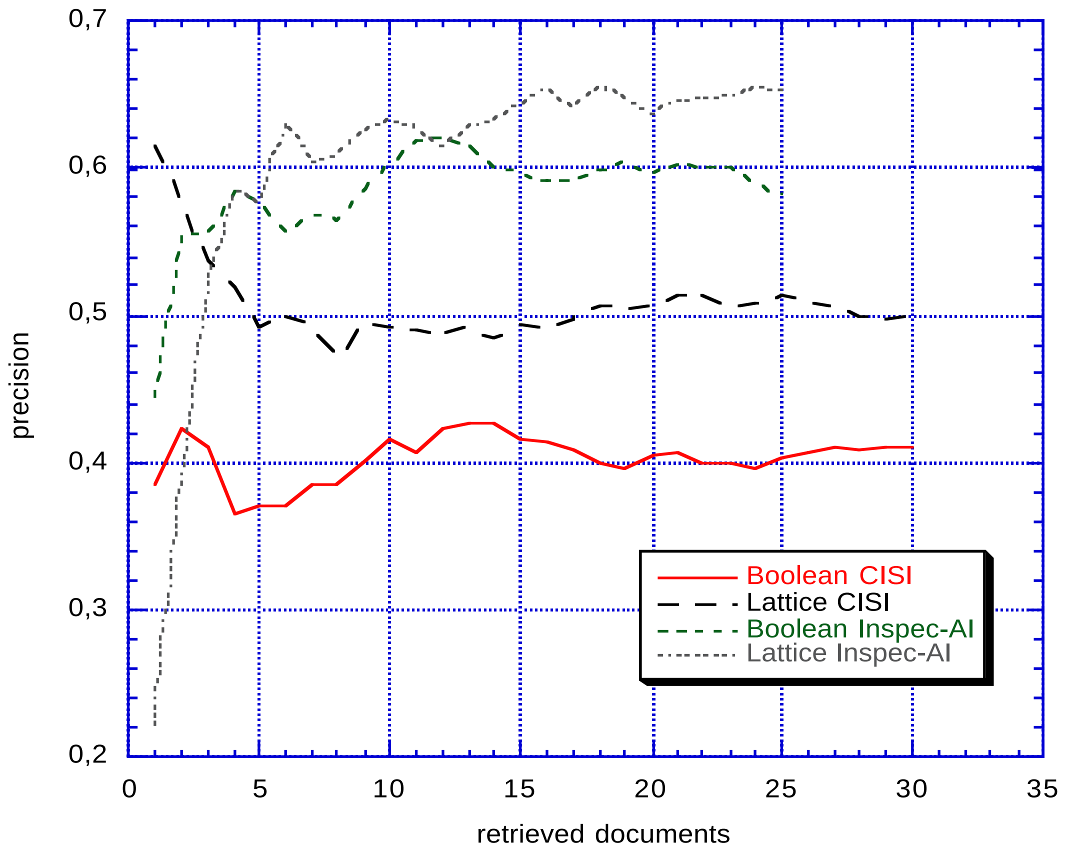
The Practice of Charging Users for Information Services: A State of the Art Report

On-Line Systems: Promise and Pitfalls

The Viability of Branch Libraries

Administrative Effectiveness: Times and Costs of

[Mon 13 May 11:54:44] IR   CL USER: User Input


0,7
0,6
0,5
0,4
0,3
0,2
precision
0
5
10
15
20
25
30
35
retrieved documents
Boolean CISI
Lattice CISI
Boolean Inspec-AI
Lattice Inspec-AI

Method	recall	precision	retrieved documents	search time (sec)
Boolean	0.326 (σ = 0.076)	0.371 (σ = 0.058)	37.7 (σ = 8.5)	2022 (σ = 523)
Lattice	0.403 (σ = 0.082)	0.493 (σ = 0.066)	38.0 (σ = 9.2)	1734 (σ = 417)

Method	recall	precision	retrieved documents	search time (sec)
Boolean	0.523 (σ = 0.107)	0.598 (σ = 0.161)	32.1 (σ = 7.1)	1607 (σ = 531)
Lattice	0.538 (σ = 0.076)	0.671 (σ = 0.094)	30.2 (σ = 7.5)	1670 (σ = 470)

Legends

Table 1. A simple database containing nine documents described by nine index terms.

Figure 1. The Galois lattice of the database in table 1.

Figure 2. Display screen of ULYSSES, relative to the lattice in Figure 1, focusing on the node *DIAGNOSIS, EXPERT-SYSTEMS, KNOWLEDGE-BASED-SYSTEMS, MEDICINE*.

Table 2. Pictorial representation of the user constraints.

Figure 3. After the user has specified the constraint $\leq$ *KNOWLEDGE-BASED-SYSTEMS* (window in the background), he or she specifies a new constraint of type $\neg \leq$ (window in the middle), and his argument: *MEDICINE* (windows in the foreground). The windows are automatically arranged into a cascade (Preece et al, 1994), so that many constraints can be asserted at the sime time, without hiding the lattice.

Figure 4. Display screen of ULYSSES in response to the actions taken in Figure 3. The nodes marked with vertical bars belong to the boundary sets of the new search space, that contains only five nodes. It should be noted that the constraint $\neg \leq$ *MEDICINE* has also drastically reduced the set of documents associated with the remaining admissible nodes.

Figure 5. Example screen of the Boolean retrieval interface during the search of the CISI documents relevant to the query: "*How much do information retrieval and dissemination systems, as well as automated libraries, cost? Are they worth it to the researcher and to industry?*".

Figure 6. Precision of lattice and Boolean retrieval for the two databases as a function of retrieved documents.

Table 3. Average values of raw retrieval performance measures for the CISI database.

Table 4. Average values of raw retrieval performance measures for the INSPEC-AI database.