

ULYSSES: A Lattice-based Multiple Interaction Strategy Retrieval Interface

Claudio Carpineto and Giovanni Romano
Fondazione Ugo Bordoni
Via Baldassarre Castiglione 59, 00142, Rome
Italy
{carpinet, romano}@fub.it

Abstract. We present a two-stage system for information exploration and retrieval. The first stage, named GALOIS, organizes the information contained into a database into a particular lattice structure; the second stage, named ULYSSES, is a visual interface to access the structure built in the earlier stage. In this paper we focus on the latter. ULYSSES is based on a tight integration of traditional and novel user interaction paradigms that can be seen as a search&bound approach to information retrieval. The user may search the retrieval space by browsing or querying, but he or she may also bound the retrieval space by specifying constraints that the information contained in it has to satisfy. These interaction modes can be naturally combined to produce a hybrid retrieval strategy that best reflects the user goals and his/her domain knowledge. The retrieval effectiveness of our system has been tested in an experiment on subject searching where it compared favourably with respect to a Boolean retrieval system.

1 Introduction

Two major search paradigms have been developed to retrieve information from a text database, querying and browsing. In query retrieval the user provides some description of the information being sought, and the system retrieves database items that match the description; this paradigm has a long tradition in information retrieval (van Rijsbergen, 1979; Salton & McGill, 1983) and is at the heart of most of the commercially available systems. The browsing paradigm, by contrast, has become popular more recently, especially after the advent of the Internet; in browsing retrieval the user explores some structure in which the database information has been organized to locate and retrieve items of interest (Marchionini & Shneiderman, 1988; Nielsen, 1990).

Each approach has strengths and limitations. Querying is flexible, general, and efficient, but it is harder for the user. Forming good queries can be a difficult task, especially if the user is not familiar with the domain; furthermore the amount of output produced in response to a query cannot be usually controlled. Browsing allows domain or system novices to explore an information space; also, it allows the user to exert complete control over the information to be displayed. The main disadvantages of browsing are that the user can get lost in the information space, that the performance for some retrieval measures (e.g., recall and search time) may be low, and that the organization of the information is predefined and is not easy to change.

Querying and browsing therefore seem to complement each other in a number of

ways. In fact, a good deal of research has been recently devoted to the development of hybrid approaches (Frei and Jauslin, 1983; Thompson and Croft, 1989, Maarek et al, 1991; Lucarella et al, 1993; Godin et al, 1993), while most resource discovery systems in use in the Internet, such as WAIS, Gopher and WWW, are based on some simple form of combination of the two retrieval strategies (see for instance Bowman et al, 1994). A principled integration of these techniques seems indeed to be one major factor involved in the construction of more general retrieval systems, although other knowledge sources and interaction strategies have been suggested that may be incorporated into a conventional framework, such as relevance feedback (Salton and Buckley, 1990), navigational aids (Furnas, 1985; Rivlin et al, 1994) user preferences (Kaplan et al, 1993), and distributed systems (Rao et al, 1993). Our research is in the same vein. In this paper we present an hybrid navigational approach to information retrieval that allows browsing, has query capabilities, and, in addition, provides a novel interaction method: bounding.

Bounding, like querying and browsing, 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 to 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 (Balas, 1968), artificial intelligence (Mitchell, 1982), and dynamic programming (Kumar, 1984), it has been little explored in the navigation retrieval context. We propose that bounding be applied to the structure being used to retrieve information. We introduce a simple and general framework whereby the user can employ the knowledge about the goal or the domain that s/he has or learns from the feedback obtained from the structure to bound the structure itself. In our view, bounding acts as a complementary strategy; browsing and querying are used to search the retrieval space, bounding may be used to restrict it. Unlike most other hybrid systems, these three strategies can be naturally combined into a personalized search strategy that reflects the user's goals and domain knowledge; most important, the proposed approach seems to combine browsing potentials with good retrieval performance.

The system we present has two main components. The first component, named GALOIS, is responsible for organizing the information of a database into a particular structure called Galois lattice. The second component, named ULYSSES, is a visual interface for accessing and retrieving the information contained in the structure built in the earlier stage using the interaction paradigms mentioned above. We will focus on the latter component.

The rest of the paper is organized in the following way. In section 2 we introduce the particular lattice built by GALOIS that acts as retrieval support structure. In section 3 we describe ULYSSES, the actual retrieval interface; we will discuss design and implementation of its four main functions, namely visualization, browsing, querying, and bounding. In section 4 we present some preliminary results which provide some evidence that this approach has good retrieval performance. In section 5 we discuss relation to other hybrid approaches, as well as directions for future work.

2 Galois Lattices as Retrieval Support Structures

The Galois lattice of a binary relation between a set of documents and a set of terms (sometimes referred to as *context*) is a particular clustering structure, in which each class is a couple, composed of a subset of documents (D), called extent, and a subset of terms (T), called intent. Each couple (D,T) must be a complete couple, 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 couples 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 couple. The resulting ordered set turns out to be a lattice (Davey and Priestley, 1990).

As an illustration, we will refer to a small bibliographic database consisting of 7 documents described by 8 terms (see Table 1).¹ The Galois lattice of the term by document matrix in Table 1 is shown in Figure 1. The ascending paths represent the subclass/superclass relation; the bottom class is defined by the set of all terms and contains no documents, the top class contain all documents and is defined by their common terms (none, in this case).

Table 1. A document-term relation.

	1	2	3	4	5	6	7
CATALOGUING		x	x				
EDUCATION					x	x	
EXPERT-SYSTEMS	x	x	x	x	x		x
INFORMATION-RETRIEVAL	x		x			x	
KNOWLEDGE-BASED-SYSTEMS				x			
LIBRARY-AUTOMATION					x	x	
NATURAL-LANGUAGES				x	x	x	x
USER-INTERFACES					x	x	x

Given the definition of Galois lattices, we have addressed the problem of their automatic determination. We have implemented in a system named GALOIS an algorithm that builds the lattice incrementally, where each update is substantially proportional or at most quadratic to the number of documents. A detailed explanation of Galois lattices, of their complexity and of the construction algorithm is contained in (Carpineto and Romano, 1994a). The physical Galois lattice structure has many applications, including machine learning and database theory (Carpineto and Romano, 1993); in particular, it can be used to support text browsing retrieval. In this case each class of the lattice is seen as a query (i.e., the intent) with its associated set of documents (i.e., the extent). This kind of structure presents many advantages for information retrieval. We have just mentioned that it can be built automatically from

¹ We should note that, although in this case the documents are described by single-valued properties, the framework can easily accomodate multi-valued properties such as author, year, and site.

the usual document term relation; also, it is possible to see (Carpineto and Romano, 1994a) that it can be generalized to incorporate thesaurus information over the terms describing the documents. More importantly, we will see in the rest of the paper that it can support an interaction paradigm involving multiple and integrated search strategies.

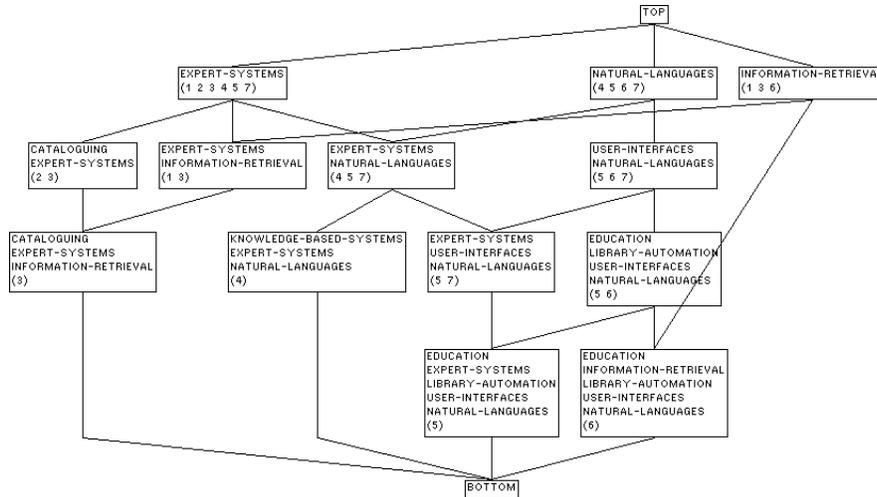


Figure 1. The Galois lattice of the context in Table 1.

3 Overview of ULYSSES

To enable the interaction between the user and the lattice structure we have realized a prototype visual interface on top of GALOIS named ULYSSES. It consists of four main components, namely one visualization module and three interaction modules: browsing, querying, and bounding. The interaction strategies are tightly integrated in that they work on the same search space (i.e., the lattice) and produce a combined result; the visualization module shows at any given one time the result of the actions taken by the user. In the next subsections we describe the four components in detail. ULYSSES is written in Lisp and runs on a Symbolics Lisp Machine.

3.1 Visualization

The first problem in the interface design is the visualization of the retrieval space. Unfortunately, the whole lattice is typically too large to fit on a screen, even for small databases; on the other hand, however, if the lattice is to be used for browsing retrieval this is not a real limitation, in that the information relative to global parameters, such as the lattice size or its full topology, is likely to 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. For this purpose, we have used a technique similar to fisheye view (Furnas, 1986), which has the advantage

of maximizing the size of the region that can be displayed without sacrificing local detail around the focus. 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 the use of parameters controlling the subgraph to visualize. The visualization module of ULYSSES consists of two procedures; the first computes the subgraph to draw, the second draws it. To compute the subgraph, 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 been already generated. Before adding another level to the subgraph to draw, the algorithm checks that the size of the resulting subgraph does not exceed a constant. The size is computed by a weighted sum of all the nodes in the subgraph, where the weight of each node is proportional to its size. Also, while computing each level the algorithm checks if the number of nodes adjacent to each single node to expand exceeds a 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 subgraph). 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 and Tamassia, 1989) and for manipulating the attributes controlling the graphical layouts (Sarkar and Brown, 1994).

In Figure 2 we show an example screen relative to the lattice in Figure 1. The current focus is the node: USER-INTERFACES, NATURAL-LANGUAGES, 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.

3.2 Browsing

Galois lattices present two useful properties for browsing. The first is that the lattice structure, in which there are many paths to a particular class, facilitates recovery from bad decision making while traversing the hierarchy in search of documents, as opposed to the tree structures that have been predominately been used in browsing retrieval, in which each class has exactly one parent. The second is that 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, Missaoui, & April, 1993).

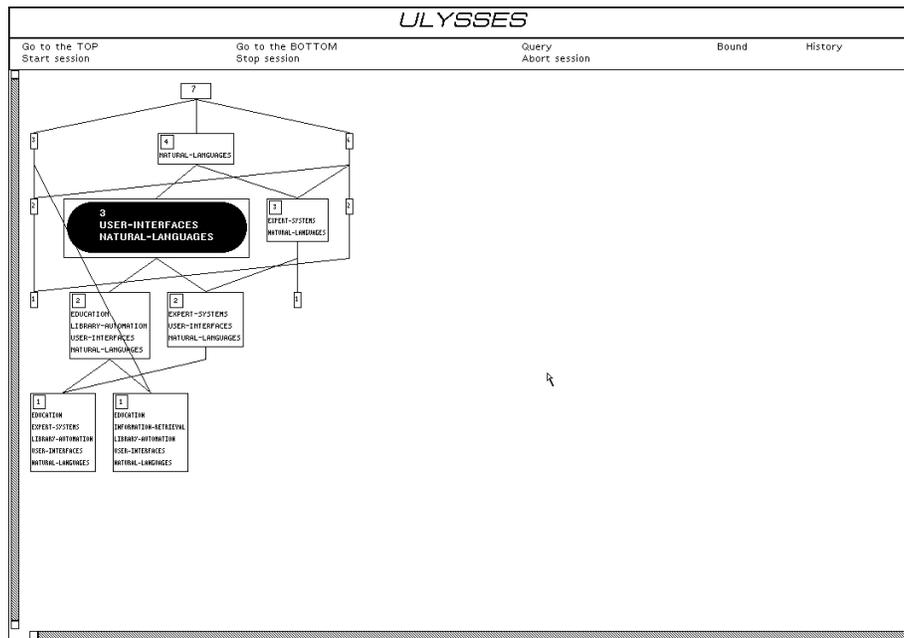


Figure 2. Display screen of ULYSSES, relative to the lattice in Figure 1, focusing on the node USER-INTERFACES, NATURAL-LANGUAGES.

In practice, to navigate through the lattice ULYSSES allows the user to select any node on the current screen by graphical direct 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 new focus. Selecting some node on the screen also allows the user to see the documents associated with it.

3.3 Querying

The background menu of ULYSSES allows selection of the other two basic interaction modes: query and bound. A query can be formulated in two manners: 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 some terms and/or add new terms. The result of a query is the class of the lattice equal to the query, if there is any, or one or more classes which partially match the query. The best partially matching queries are determined in the following way. The algorithm first computes the most general more specific nodes 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 (according to the lattice ordering relation) to the query, 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 those of them that contain the smallest number 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, for instance, in Williamson and Shneiderman (1992) and Ahlberg and Shneiderman (1994). This approach seems to be particularly suitable for ordinal attributes; if we considered also this kind of properties for the documents in the database, such as year and length of documents, we could use it for filtering the documents in the initial database prior to the application of ULYSSES.

3.4 Bounding

The bound interaction mode allows the user to change the lattice from which he or she is retrieving information during the search. The user may assert constraints which the sought documents have to comply with and the retrieval space is bounded accordingly. The constraints are expressed as inequality relations between the description of admissible classes and a particular conjunction of terms. For the sake of illustration we will assume that the 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 class and c_1 be a particular class of the search space. In our framework there are four possible kinds of constraints: $c \geq c_1$, $c \leq c_1$, $c \not\geq c_1$, $c \not\leq c_1$. 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 useful properties of the classes of documents from the point of view of their information retrieval performance (see Carpineto and Romano (1994b)).

Table 2 Pictorial representation of the user constraints

$c \geq c_1$	$c \leq c_1$	$c \not\geq c_1$	$c \not\leq c_1$

The application of this kind of bound framework to a Galois lattice has also other advantages, the most important of which is that the constraints may be used to retrieve documents that the strictly conjunctive query language of the basic Galois lattice retrieval approach would be unable to retrieve. Using the constraint $c \not\leq c_1$,

for instance, we can specify the target class(es) using negated terms, which is forbidden in an unconstrained search through Galois lattices.

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 and Romano, 1994b). For the scope of this paper it suffices 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 very compact and supports efficient update; when ULYSSES is presented with a new constraint, the elements of the two boundary sets are updated performing a local search through their adjacent nodes. As more and more constraints are seen 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 asserted constraints.

Bounding the search space has of course a direct effect on browsing and querying, in that the user will be only allowed 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, to reduce the user disorientation due to the change of screen stability, ULYSSES makes the node(s) of the nearest boundary set the new focus.

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

3.5 A Simple Interaction Session

Consider the database in Table 1, and suppose that the user wants to find the documents about 'knowledge-based systems for text storage and retrieval that do not employ linguistic techniques'. The user might begin the session submitting the query KNOWLEDGE-BASED SYSTEMS; in response, the system would show the screen in Figure 3. This display reveals many information relevant to the specific user question. On one hand, it shows that while there is only one document of the database indexed by KNOWLEDGE-BASED SYSTEMS, there are many documents indexed by a similar term (i.e., EXPERT-SYSTEMS); also, it suggests that the user could prune away a large part of the retrieval space that is not relevant to him/her by forbidding the term NATURAL-LANGUAGES. Therefore the user might now assert two constraints, namely \leq EXPERT-SYSTEMS and $\neg \leq$ NATURAL-LANGUAGES. Figure 4 shows how the user can actually do so. Consequently, the retrieval space is bounded as shown in figure 5. The new screen displays three nodes that are presumably of interest to the user, and that had not been fully visualized before.

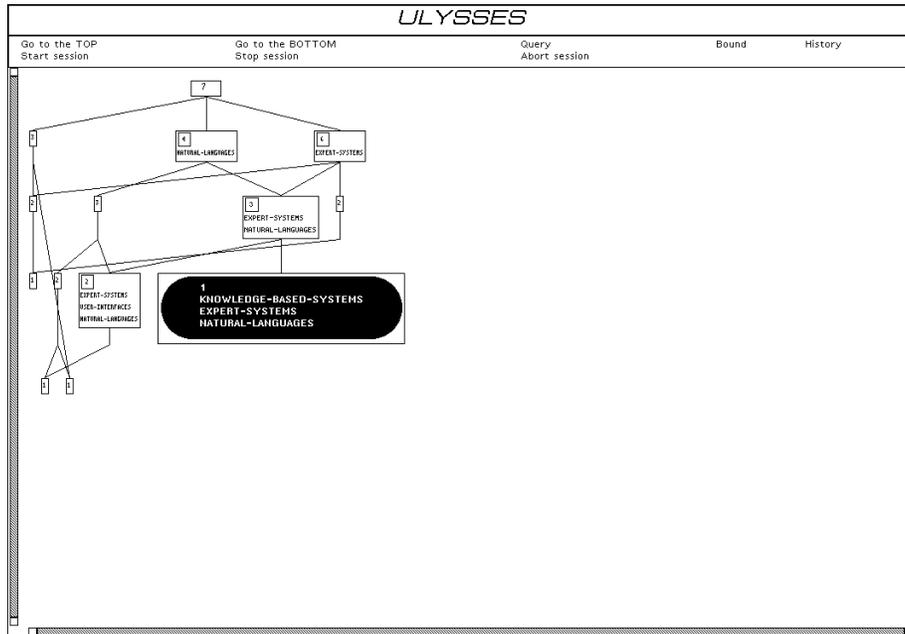


Figure 3. Display screen of ULYSSES in response to the query KNOWLEDGE-BASED-SYSTEMS.

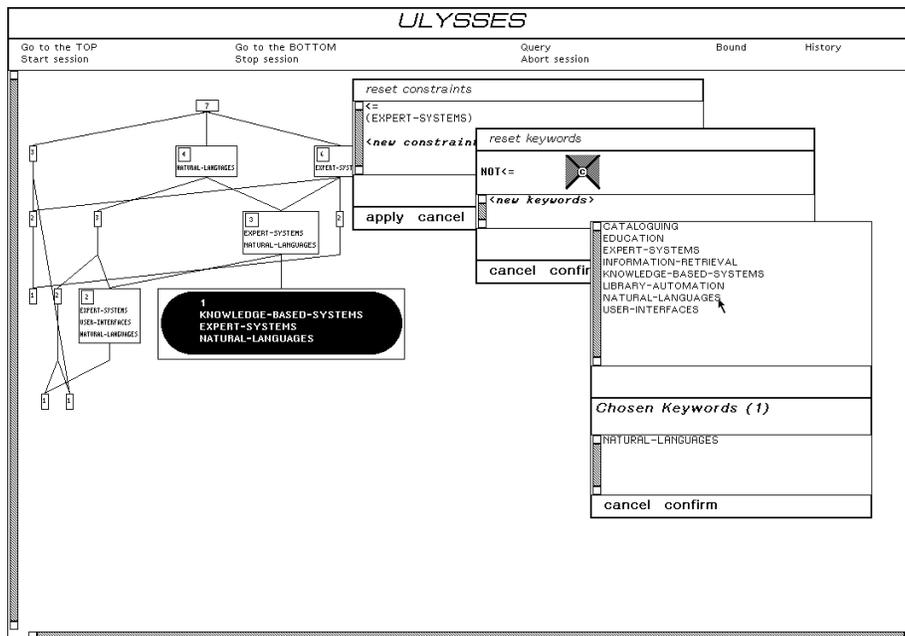


Figure 4. The user asserts the constraint \leq EXPERT-SYSTEMS (window in the background), then the constraint $\neg \leq$ NATURAL-LANGUAGES (windows in the foreground).

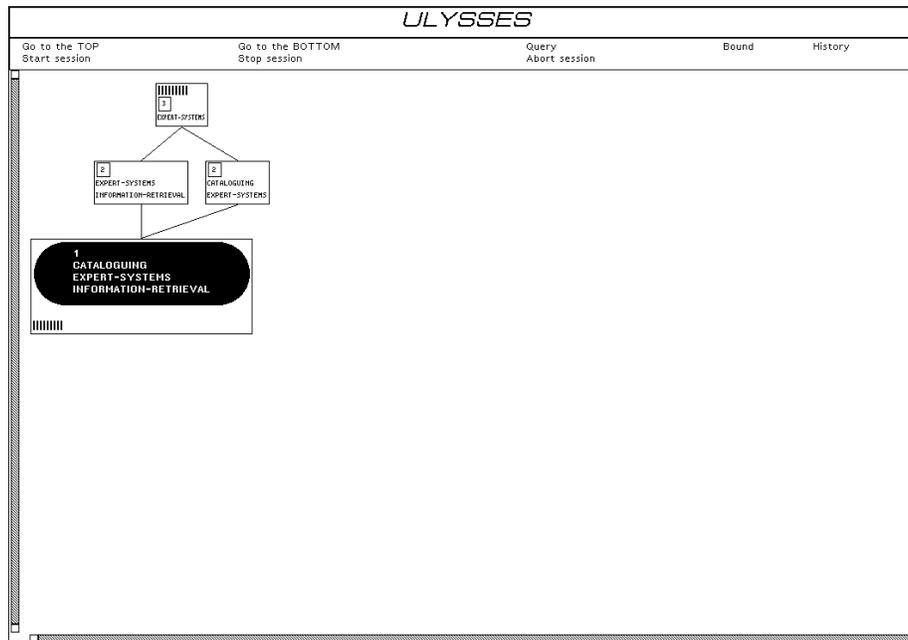


Figure 5. The bounded space after the introduction of the two constraints in Figure 4. The nodes marked with vertical bars belong to the boundary sets.

4 Evaluation

Our approach has both browsing potentials and more direct retrieval capabilities. In this section we concentrate 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 seen as unsatisfactory compared to more conventional query-based systems. We report the results of an experiment on subject searching where we compared our approach to a Boolean retrieval system. We chose a Boolean system because it is easy to implement and it is known to perform quite well on this task. The experiment was conducted on a collection of 1555 documents extracted from INSPEC, a commonly used large computer engineering collection. The documents were described by a title, an abstract and a set of controlled terms, with an average of 10.15 terms per document. This document by term matrix was used to generate both the Boolean and the lattice databases. The lattice contained 8769 nodes, with an average of 3.11 parents per node and a depth ranging from 2 to 15 edges.

The experimental protocol was as follows. The two retrieval methods were evaluated on 10 queries by 2 external users. As there is no standard methodology to determine a representative set of queries, we tried to mix different types of queries, such as queries containing general and specific subjects, known and unknown words, conjunctive and disjunctive concepts (e.g., "use of parallelism to improve efficiency of AI languages or algorithms"). Afterwards, for each query we manually produced its relevance judgements, i.e., the associated set of relevant documents. The average number of relevant documents for the 10 queries was 32.1. For assigning queries to

the two methods a repeated-measures design was used, in which each user searched each query using each method; also, to minimize sequence effects (Tague-Sutcliffe, 1992), we varied the order of the two methods, and, in addition, we tested one method a week after the other. During each search the user, who was not asked to finish within a certain time period, could see the abstract of the documents returned in response to Boolean queries - in the boolean method - or associated with the visited nodes - in the lattice method. The documents judged to be relevant by the user, as well as those scanned during the search, were noted as retrieved documents. For each search we considered three measures: recall, precision,² search time (i.e., the time taken by the user to perform his task). The results are displayed in Table 3.

Table 3. Average values of retrieval performance measures

Method	recall	precision	search time (sec)
Boolean retrieval	0.45 ($\sigma = 0.06$)	0.59 ($\sigma = 0.10$)	1666 ($\sigma = 197$)
Lattice-based retrieval	0.49 ($\sigma = 0.07$)	0.71 ($\sigma = 0.08$)	1867 ($\sigma = 201$)

The table shows that the lattice method obtained better evaluation scores for recall and precision, but not for search time. To see if these differences can be considered statistically significant (Hull, 1993) we performed a paired t-test for each measure. The test revealed no effect of the method on search time ($p = 0.26$) and recall ($p = 0.21$), and it did not reveal a significant difference in precision ($p = 0.08$). The fact that the two methods showed substantially similar retrieval performance is not surprising. In fact, they have relative advantages and disadvantages. Boolean queries have a greater expressive power than lattice queries, but the lattice method provides other retrieval strategies that may compensate for this limitation. In particular, browsing allows smooth query refinement, as opposed to Boolean retrieval where the user cannot control the amount of output obtained in response to a query; also, bounding permits the use of negated terms, and allows the user to exploit the feedback obtained from the structure to facilitate selection of relevant terms in the database.

While these results can only be seen as indicative, they provide some evidence that our approach has good retrieval performance; in particular, they seem to suggest that it may represent an alternative to more conventional systems even for typical retrieval tasks and for databases of non-trivial size.

5 Discussion and Future work

ULYSSES is characterized by two main novel features. The first is that among the

²These are the two standard measures to evaluate retrieval effectiveness. *Recall* is defined as the ratio of number of items retrieved and relevant to number of items relevant; *precision* is the ratio of number of items retrieved and relevant to number of items retrieved. Recall measures the ability to retrieve *all* relevant documents, while precision measures the ability to retrieve *only* relevant documents.

various interaction modes provided by the interface there is also the possibility for the user to exploit the information that he or she has about the goal or the domain to dynamically change the retrieval space, whereas in most other hybrid approaches to information retrieval and navigation the underlying support structure can be accessed in various ways but cannot be modified. In fact, ULYSSES can be seen as an attempt to relax one of the major impeding design factors of navigation system, namely the fact that the support structures are predefined and cannot be easily tailored to the user, by explicitly introducing additional knowledge about the goal or the domain into the structure in the form of constraints. We believe that although space bounding has been little explored in this context, it may be an effective complementary strategy to help avoid the lost in hyperspace problem and to improve performance of navigation-based retrieval systems. We want also to emphasize that although this interaction mode is very suitable for Galois lattices, it can also be applied to other retrieval spaces, as discussed in (Carpineto and Romano, 1994b).

The second main feature of ULYSSES is that it represents an approach to tightly integrating multiple interaction modes, including the bound mode, into a hybrid search strategy. For instance, the user may start with a query to locate the subspace of interest, and then browse through the subspace, while bounding may occur at any time during the search. If the user has enough information about the goal, he or she can prune the search space prior to the actual search. If this is not the case, the user may exploit the feedback obtained from the structure to dynamically bound the space during the search.

ULYSSES represents a major departure from many other hybrid approaches, in that the integration of different retrieval strategies is usually much looser. One typical choice is to maintain different retrieval methods in parallel, for instance a boolean query system and a hierarchical browsing system (Maarek, Berry, and 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 coupling more systems in cascade, for instance browsing a term space to determine the queries to submit to a direct retrieval system (Frei and Jauslin, 1983; Pedersen, 1993). However, this is still far from a tight and principled combination. ULYSSES takes one step further, similar to the systems described in Godin et al, 1993 and Bowman et al, 1994. The different retrieval methods work on a single retrieval space and share their intermediate results, so that the user does not have to commit himself to one particular method and does not have to worry about mapping different representations and results. The overall effect for the user is that ULYSSES exhibits behavior in a continuum of interactions, where each user may find the combination that best fits his/her own knowledge (ranging from novice to expert), goal (ranging from casual inspection to subject searching), and constraints (ranging from low to high search time).

In this paper, in addition to presenting a methodology and an implemented system, we also evaluated its retrieval effectiveness. We compared its performance on subject searching to that of a Boolean retrieval system, using a reasonably sized database. The results we obtained are quite promising, in that they suggest that a structure-based approach to information retrieval may combine browsing potentials for information exploration with good performance on more direct searches.

At the moment we are developing a module for automatic text indexing. Therefore

the next step of this research is to evaluate the retrieval performance of the full system on standard unindexed databases, to see if it can be seen as an alternative to more conventional and robust retrieval methods. Besides task performance, we are also interested in developing criteria to evaluate the user interface itself. Another research direction is to investigate the possibility of working with scalable support structures. The idea is to work with a concept lattice in which the set of index terms may be restricted or enlarged automatically, based on the user goal and other possible design constraints. We believe that this could provide a more flexible way to trade off multiple conflicting objectives, such as maximizing efficiency and effectiveness and minimizing the complexity of interaction between the user and the system.

Acknowledgments

We would like to thank an anonymous reviewer for many useful comments. This work has been carried out within the framework of the agreement between the Italian PT Administration and the Fondazione Ugo Bordoni.

References

- Ahlberg, C., Shneiderman, B. (1994). Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. In *Proceedings of CHI'94* (pp. 313-317), Boston, MA: Addison Wesley.
- Balas, E. (1968). "A note on the branch-and-bound principle", *Operations Research*, 16, 442-444.
- 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.
- Carpineto, C., & Romano, G. (1993). GALOIS: An order-theoretic approach to conceptual clustering. In *Proceedings of the 10th International Conference on Machine Learning* (pp. 33-40), Amherst, MA: Morgan Kaufmann.
- Carpineto, C., & Romano, G. (1994a). A lattice conceptual clustering system and its application to browsing retrieval. Submitted to *Machine Learning*.
- Carpineto, C., & Romano, G. (1994b). 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.
- 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. Tech. Rep. 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.
- Furnas, G. (1986). Generalized fisheye views. *Proceedings of the Human Factors in Computing Systems* (pp.16-23). North Holland.

- 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.
- Kumar, V. (1984). A General Bottom-up Procedure for Searching AND/OR Graphs. Proceedings of the Fourth National Conference on Artificial Intelligence, Austin, TX.
- Lucarella, D., Parisotto, S., Zanzi, A. (1993). MORE: Multimedia Object Retrieval Environment. *Proceedings of the Fifth ACM Conference on Hypertext* (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.
- Marchionini, G., & Shneiderman, B. (1988). Finding facts vs. browsing knowledge in hypertext systems. *IEEE Computer*, 21, 70-80.
- Mitchell, T. (1982). Generalization as search. *Artificial Intelligence*, 18, 203-226.
- Nielsen, J. (1990). *Hypertext & hypermedia..* San Diego, CA: Academic Press.
- Pedersen, G. (1993). A browser for bibliographic information retrieval based on an application of lattice theory. *Proceedings of SIGIR'93* (pp. 270-279), Pittsburgh, PA.
- Rao, R., Russel, D., & Mackinlay, J. (1993). System Components for Embedded Information Retrieval from Multiple Disparate Information Sources. *Proceedings of the ACM Symposium on User Interface Software and Technology* (pp. 23-33), Atlanta, Georgia.
- Rivlin, E., Botafogo, R., Shneiderman, B. (1994). Navigating in Hyperspace: Designing a Structure-based Toolbox. *Communications of the ACM*, 37, 2, 87-96.
- Salton, G., Buckley, C. (1990). Improving retrieval performance by relevance feedback. *JASIS*, 41, 288-297.
- Salton, G., & McGill, M. (1983). *Introduction to modern information retrieval*. New York: McGraw Hill.
- Sarkar, M., & Brown, M. (1994). Graphical Fisheye Views. *Communications of the ACM*, 37, 12, 73-84.
- Shneiderman, B. (1987). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Reading, MA: Addison Wesley.
- 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.
- van Rijsbergen (1975). *Information retrieval*. London: Butterworths.
- Williamson, C., Shneiderman, B. (1992). The Dynamic Homefinder: Evaluating Dynamic Queries in a Real-Estate Information Exploration System. In *Proceedings*

of SIGIR'92 (pp. 338-346), New York: ACM SIGIR Forum.