Effective Reformulation of Boolean Queries with Concept Lattices

Claudio Carpineto and Giovanni Romano

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

Abstract. In this paper we describe an approach, implemented in a system named REFINER, to combining Boolean information retrieval and content-based navigation with concept lattices. When REFINER is presented with a Boolean query, it builds and displays a portion of the concept lattice associated with the documents being searched centered around the user query. The cluster network displayed by REFINER shows the result of the query along with a set of minimal query refinements/enlargements. REFINER has two main advantages. The first is that it can be used to improve the effectiveness of Boolean retrieval, because it allows content-driven query reformulation with controlled amount of output. The second is that it has potentials for information exploration, because the displayed network is navigatable. We compared information retrieval using REFINER with conventional Boolean retrieval. The results of an experiment conducted on a medium-sized bibliographic database showed that the performance of REFINER was better than unrefined Boolean retrieval.

1 Introduction

Most large-scale retrieval systems and on-line information services are based on the Boolean model. The user queries the system using a set of terms connected by the Boolean operators and, or, and not, and the system returns the set of documents that match the query. Boolean retrieval systems have become popular in operational situations for two main reasons. The first is that they are easy to implement and computationally efficient. The second is that they allow high standards of performance, mainly due to the clarity and expressive power of their underlying model. Furthermore, the retrieval effectiveness of the Boolean model can be improved through various additional facilities usually provided by modern systems, such as truncation and proximity operators, co-occurrences information, and ranking heuristics.

It is well known, however, that Boolean retrieval systems have several limitations, some of which are addressed in this paper. One problem is that only documents that satisfy a query exactly are retrieved; in particular, the and operator is too strict because it fails even in the case when all its arguments except one are satisfied. Another problem is that users are often faced with the null-output or the information overload problem (e.g., [17], [14]) because they cannot control the number of documents produced in response to a query. A third limitation is that Boolean systems, like other query-based retrieval systems (e.g., vector-space and probabilistic), are not suitable for causal users and for exploration of new domains. As the user must specify a query perfectly (or partially) matching some description of the documents, this approach requires the user to have some specific goal in mind and to have some knowledge about the content of the database. This seems to be an important limitation of current query-based retrieval systems, especially in the light of the continuously growing amount of Internet accessible information resources.

In this paper we present an approach, implemented in a system called REFINER, that helps overcome these problems. Given a set of documents described by a set of terms, the approach is based on a mapping between the set of Boolean queries that can be defined over the terms and the nodes of a particular cluster lattice built from the terms/documents, called concept (or Galois) lattice. When the user formulates a query, the system computes a small portion of the lattice centered around the user query; it turns out that the nodes determined by REFINER are a complete set of minimal refinements/enlargements of the query with respect to the document/term relation. By using a visual interface, the user may exploit the information contained in the displayed region to refine a submitted query or may navigate through it; in the latter case, the displayed region is built dynamically as the user moves from one node to another. We argue that the integration of conventional Boolean querying with lattice-based refinement may result in better retrieval performance over unenhanced Boolean retrieval. In addition we show that REFINER is also suitable for information exploration, because the user may readily mix browsing and searching styles of interaction with the system.

The rest of the paper is organized in the following way. We first introduce the model underlying the clustered representation of a document/term relation. Then we present an algorithm for building the portion of lattice relevant to a given query, and describe REFINER, a retrieval interface for combining conventional Boolean querying with lattice-based query refinement and navigation. Next, we report the results of an experiment on subject searching in a reasonably-sized database where REFINER compared favourably with a conventional Boolean retrieval system. Finally, we compare our approach to related work and offer some conclusions.

2 Concept lattice

We first give an informal characterization of a concept lattice and then describe it more formally. As an illustration, Table 1 shows a very simple bibliographic database consisting of four documents described by four binary terms, while the concept lattice built from it is illustrated in Figure 1. Each node of the lattice is a pair, composed of a subset of the documents and a subset of the index terms; in each pair, the subset of terms contains just the terms shared by the subset of documents, and, similarly, the subset of documents contains just the documents sharing the subset of terms. The set of pairs is ordered by the standard set inclusion relation applied to the set of documents and terms that describe each pair. The partially ordered set is usually represented by a Hasse diagram, in which there is an edge between two nodes if and only if they are comparable and there is no other intermediate concept in the lattice (i.e., each node is linked to its maximally specific more general nodes and to its maximally general more specific nodes). The ascending paths represent the subclass/superclass relation; the bottom concept is defined by the set of all terms and contains no documents, the top concept contains all documents and is defined by their common terms (possibly none).

Table 1. A simple document/term relation with four terms (*a, b, c,* and *d*) and four documents (*1, 2, 3,* and *4*).

	1	2	3	4
a	x	x		x
b	x	x		
c			x	x
d	x		x	x

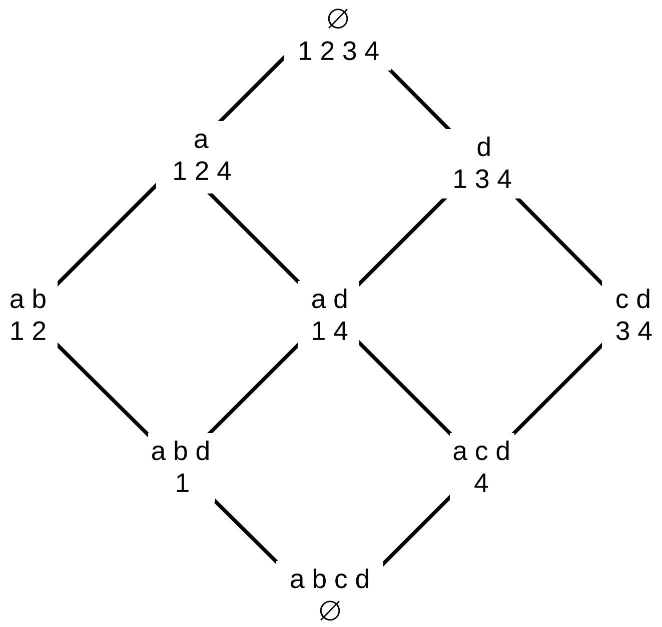


Fig. 1. The concept lattice of the database in Table 1.

More formally, consider a binary relation between a set of documents (D) and a set of terms (T), called *context*. Therefore a context is a triple (D, T, I) where $I \subseteq D x T$. We write *dIt*, meaning the document *d* has the term *t*. For $X \subseteq T$ and $Y \subseteq D$, define:

$X' = \{ d \in D \mid (\forall t \in X)\ dIt \}$, $Y' = \{t \in T \mid (\forall d \in Y)\ dIt \}$.

X' is therefore the set of all documents possessing all the terms in X and Y' is the set of all terms common to all documents in Y. Then a concept of the context (D,T,I) is defined to be a pair (X, Y) where

$X \subseteq T$, $Y \subseteq D$, and $X' = Y$, $Y' = X$;

X and Y are called the *intent* and the *extent* of the concept, respectively. Note that a subset A of D is the extent of some concept if and only if A"=A in which case the unique concept of which A is an extent is (A', A). Therefore only some pairs (X,Y), i.e., the pairs that are complete with respect to I according to the given definition, represent admissible concepts. For instance, in the lattice relative to the context in Table 1 there cannot be any pair having an intent equal to *b*, because all documents having *b* have also *a*. The set of all concepts of the context (D, T, I) is denoted by C(D, T, I). An ordering relation ($\leqq$) is easily defined on this set of concepts by

$(X_1, Y_1) \leq (X_2, Y_2) \ \leftrightarrow X_1 \supseteq X_2$ or, equivalently, by
$(X_1, Y_1) \leq (X_2, Y_2) \ \leftrightarrow Y_1 \subseteq Y_2$.

C(D, T, I) along with $\geq$ form a partially ordered set, that turns out to be a complete lattice [22].

A more thorough treatment of concept lattices and further theoretical and computational results are contained in [8] and [4]; in this paper we concentrate on their interpretation and application in the information retrieval domain. Each node can be seen as a query formed of a conjunction of terms (the intent) with the retrieved documents (the extent). The lattice allows gradual enlargement or refinement of a query. More precisely, following edges departing upward (downward) from a query produces all minimal conjunctive enlargements (refinements) of the query with respect to that particular database. In other terms, given a node it is not possible to delete (add) terms in such a way to obtain an intermediate concept between the node and its fathers (children). In the next section we show how this clustering model can be used to support Boolean querying refinement.

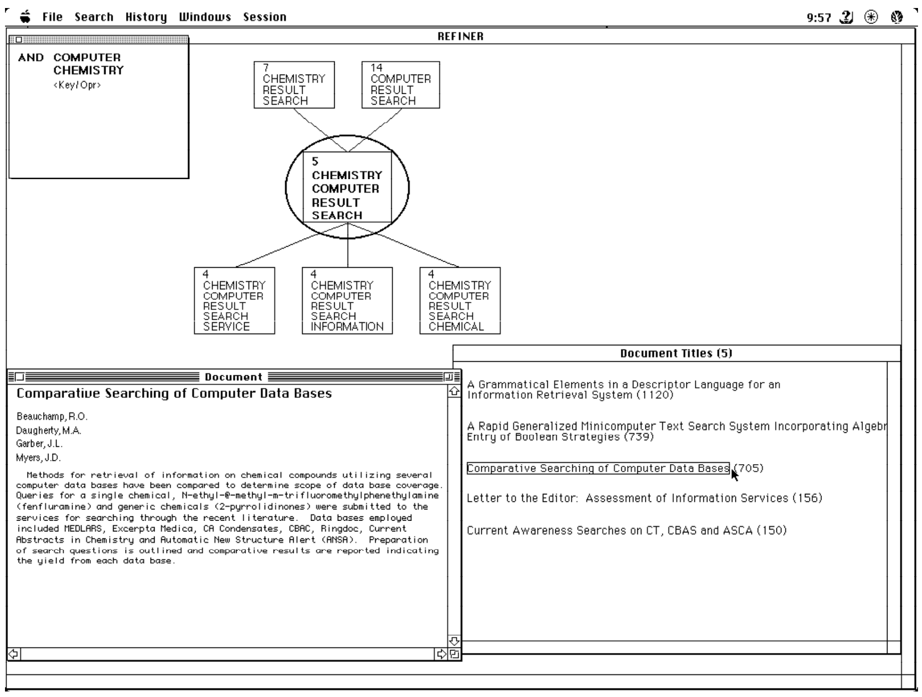
3 Description of REFINER

REFINER is based on a two-step procedure. In the first step it maps a Boolean query on some node in the lattice; in the second step it builds and display the set of parents and children of the node determined in the earlier step. The first step is carried out by computing the set of documents that satisfy the query (i.e., the extent of the lattice node corresponding to the query) and then by determining the set of terms possessed by all previously found documents (i.e., the intent of the lattice node corresponding to the query). For the simple and very common case when the query consists of a conjunction of terms, the lattice node associated with the query, assuming that V is the set of query terms, is given by (V", V'), where $V' = \{d \in D \mid (\forall t \in V)\ dIt \}$, $V'' = \{t \in T \mid (\forall d \in V')\ dIt\}$. For instance, if we take the context shown in Table 1, for the query *b* AND *d*, we get: V={b, d}, V'={1}, V"={a, b, d}. It may happen that in the lattice there are no nodes that contain all the terms specified in the query (i.e., there are no documents exactly matching the query). In this case REFINER adds a virtual node to the lattice, as if the query represented a new document, and applies the second step of the algorithm, described below, to it.

The second step is computationally more complicated. Here we consider only the problem of finding the parents of a given node, because the determination of its children is a dual problem. The candidate parents are all the nodes whose intent is more general than the intent of the node. However, it is not necessary to generate all such possible intents. Given a node (X , Y), it is sufficient to generate, for each document *d* that does not belong to Y, the set of documents $Z = Y \cup \{d\}$, then find $Z' = \{t \in T \mid (\forall d \in Z)\ dIt\}$, $Z'' = \{d \in D \mid (\forall t \in Z')\ dIt\}$. The set of parents is then obtained from the set of nodes (Z' , Z") generated in this way, by selecting all the most specific of them (i.e., those whose intent is not contained in the intent of some other node in the same set). Following the example introduced above, once the the query *b* AND *d* has been mapped on the lattice node < a b d, 1 >, REFINER computes three candidate parents, i.e., < a b, 1 2 >, < a d, 1 4 >, < d, 1 3 4 >, the third of which is removed because it is more general than the second one. The resulting two concepts are the parents of the given node (see Figure 1). The time complexity of the described algorithm is proportional to $\|D\| \cdot p$, where p is the average number of parents, which is usually proportional to the size of the query node's intent. In practice, the efficiency of the algorithm can be further improved by sorting the set of documents in the outer iteration in such a way that the documents having the largest intersections with X are examined first.

It is important to note that the method that we have just described can be applied not only to conjunctive queries with atomic arguments, as in the examples, but to any Boolean query. The procedure is the same, provided that there is some document that satisfies the query; the only difference is that the intent of the mapping node, which will be described by the (possibly empty) set of terms common to the extent's documents, may not contain some of the query terms while containing terms other than those present in the query.

REFINER, implemented in Common Lisp on a Power Macintosh, combines Boolean querying and lattice-based refinement with a visual interface. In Figure 2 we show an example screen of REFINER. The screen example was produced during an experiment described below, where a user was searching the documents of the CISI collection - a widely used, electronically-available bibliographical collection of 1460 information science documents described by a title and an abstract - relevant to the following question: *Computerized information systems in fields related to chemistry*. As shown in Figure 2, the user may formulate Boolean queries in a window placed in the left upper corner of the screen. REFINER allows full-text Boolean retrieval, with stop-wording and word-stemming; it also maintains a history of user queries and a history of the nodes visited for each query. In the example shown in Figure 2, the query input by the user was: *computer* AND *chemistry*. REFINER mapped the user query on to the lattice node with intent: {chemistry, computer, result, search}; i.e., all the documents in the database that had *computer* and *chemistry* had also *result* and *search*. The relevant portion of the lattice is displayed by REFINER on the main window. The node matching the query is shown in bold; for each node, it is shown its intent and the cardinality of its extent. Usually, the number of parents and children is limited; in any case, the number of adjacent nodes displayed on the screen cannot exceed a user-supplied threshold. Figure 2 shows also the list of documents associated with one of the nodes on the screen (encircled by REFINER) and the text of one of them.



Comparative Searching of Computer Data Bases

Beauchamp, R.O.
Daugherty, M.A.
Garber, J.L.
Myers, J.D.

Methods for retrieval of information on chemical compounds utilizing several computer data bases have been compared to determine scope of data base coverage. Queries for a single chemical, N-ethyl-@-methyl-m-trifluoromethylphenethylamine (fenfluramine) and generic chemicals (2-pyrrolidinones) were submitted to the services for searching through the recent literature. Data bases employed included MEDLARS, Excerpta Medica, CA Condensates, CBAC, Ringdoc, Current Abstracts in Chemistry and Automatic New Structure Alert (ANSA). Preparation of search questions is outlined and comparative results are reported indicating the yield from each data base.

A Grammatical Elements in a Descriptor Language for an Information Retrieval System (1120)

A Rapid Generalized Minicomputer Text Search System Incorporating Algebr Entry of Boolean Strategies (739)

Comparative Searching of Computer Data Bases (705)

Letter to the Editor: Assessment of Information Services (156)

Current Awareness Searches on CT, CBAS and ASCA (150)

Fig. 2. Display screen of REFINER relative to the search of the CISI documents relevant to the question: *Computerized information systems in fields related to chemistry*.

The display in Figure 2 reveals much information relevant to the specific user question that would have been otherwise difficult to acquire. On one hand, it shows two query refinements that would produce manageable outputs, whereas in the absence of this kind of information the user might be tempted to enlarge the initial query by deleting either computer or chemistry, which would result in an output list containing hundreds of documents. The display also reveals other terms in the database (e.g., *chemical*) that index documents that are likely to be of interest to the user.

One interesting feature of REFINER is that the procedure described above can be recursively applied to each of the nodes displayed on the screen. When the user clicks on a node, the system builds the region corresponding to the query contained in the intent of the selected node. In this way, it is as if the user were navigating through the lattice, with the advantage that we do not need to build the entire lattice in advance.

4 Evaluation

The goal of our experiment was to evaluate how the retrieval effectiveness of Boolean retrieval changes when it is enhanced with a lattice-based refinement facility. To obtain the basic Boolean retrieval system with which to compare REFINER, we simply turned off the query-refinement component of REFINER. In this way we minimized as much as possible the effect that having different interfaces has on performance: both systems ran on the same machine and used the same interaction devices. In our experiment we used the database CISI described above, where ranking methods usually perform poorly. Each document was automatically indexed; we excluded words on CACM's stop list of common words, and we mapped word variants into the same root by using by using a very large *trie*-structured morphological lexicon for English [13], 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). At the end of this treatment, each document was described by an average of 47.1 terms. Along with the CISI database comes a set of 35 linguistic 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 32. We tested four subjects in the experiment. The subjects were computer science students with little knowledge of the document domains and no prior knowledge about the systems. The two subjects were asked to retrieve the documents relevant to the 10 queries using the two retrieval methods. For assigning the queries to the methods we used a repeated-measures design, in which each subject searched each query using each method. To minimize sequence effects, we varied the order of the two methods. 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 or associated with the nodes displayed on the screen. The documents scanned during the search were noted as retrieved. We have to emphasize that in evaluating the effectiveness of interactive retrieval systems the definition of the retrieved set of documents is usually not obvious. Our choice is consistent with that of [20] and [21], where a document is rated as as a retrieved document as soon as its full description (the abstract, in our case) is recovered. For each search we considered four measures: recall, precision, number of Boolean queries, and search time (i.e., the time taken by the user to perform his task). The results are displayed in Table 2.

The table shows that searching with REFINER obtained better evaluation scores for recall, precision and search time. To see if these differences can be considered statistically significant we performed a paired t-test for each measure. The test revealed no effect of the method on precision ($p = 0.235$) and recall ($p = 0.289$).

However, it did reveal the superiority of REFINER with respect to search time ($p = 0.011$). These results seem to suggest that the use of content-based query refinement may reduce user search time without reducing retrieval effectiveness. These results are not surprising, because REFINER complements the basic capabilities of a Boolean retrieval system with other useful features. In particular, as explained above, REFINER allows smooth query refinement/enlargement, which is likely to be the key factor for obtaining the search time improvement. Another advantage of REFINER, as opposed to strict Boolean retrieval where this kind of information is not available, is that the user may exploit the feedback obtained from the structure to facilitate selection of relevant terms in the database or to discover unwanted senses of words [7], for instance when an intent contains two words that were not expected to appear together. Also, REFINER's AND operator would always produce a non-null output (unless none of the AND's arguments are satisfied), while the strict Boolean AND operator fails whenever there is at least one argument that is not satisfied. A further useful observation for justifying the marked superiority of REFINER over Boolean retrieval is that most (85%) of the queries submitted in our experiment were simple conjunctions of terms[1], whose refinements are more intuitive to the user.

Table 2. Average values of retrieval performance measures

Method	recall	precision	number of Boolean queries	search time (sec)
Boolean	0.366 (σ=0.098)	0.289 (σ=0.077)	13.2 (σ = 2.212)	1901 (σ =265)
REFINER	0.419 (σ=0.107)	0.330 (σ=0.085)	6.00 (σ = 2.44)	1467 (σ = 263)

5 Related work

This research is closely related to work done in four areas: reformulation of Boolean queries, hybrid approaches to information retrieval, applications of concept lattices to information browsing, and cooperative database systems. In this section we examine the relation to each of them in turn.

- Cleverdon [6] has proposed an approach to supporting interactive reformulation of Boolean queries named quorum-level searches. Starting from an AND clause containing *n* terms, the user may choose a new query from a n-level query hierarchy where each level is a disjunction of AND clauses obtained by removing *1, 2, ...n-1* terms from the original top AND clause. The query hierarchy may therefore drive the user from narrow to broader formulations of an initial conjunctive query[2]. In the same vein, Spoerri [19] has proposed a more systematic and powerful approach. He describes INFOCRYSTAL, a system that computes

[1] That users prefer simple conjunctive queries is also suggested by other studies [2]

[2] A similar technique has also been used in [1] with a different goal, namely to help assessors evaluate recall on large databases.

and display the result, in terms of number of documents, of all possible Boolean queries in a normal form that involve the terms present in a user query. This method has the advantage that it can be applied to any Boolean expressions, not only to conjunctive queries as quorum-level searches. However, in both of these approaches the meaning of the proposed reformulations may be difficult to understand for the user. While in REFINER each reformulation is obtained by deleting terms (query enlargement) or adding new terms (query refinement), each reformulation shown by INFOCRYSTAL (or quorum-level search) contains only and all of the terms present in the user query, connected with different operators and with different precedences. Thus, the queries displayed by the latter two systems may be semantically distant from the search query, and their utilization may be not obvious for the user. For instance, the set of reformulations produced by INFOCRYSTAL in response to the query "*a* AND *b* AND *c*" would also contain the query "(NOT *a*) AND (NOT*b*) AND (NOT *c*)". This raises the question of the system's retrieval effectiveness, for which no experimental evidence has been in fact reported. Another limitation of INFOCRYSTAL is that it does not scale up well, because if a query contains N terms it must build and display 2^{N-1} queries. Finally, unlike REFINER, both quorum-level search and INFOCRYSTAL cannot help user select other relevant terms present in the document collection.

- A third more recent approach to supporting users in Boolean retrieval is FIRE [3]. FIRE provides two main kinds of support: terminological help and strategic help. The terminological help consists of new system-generated terms for expanding or refining a user query. However, in FIRE the source of the terms proposed to the user is a (manually-constructed) thesaurus with no explicit relation to the content of the database being searched. This contrast with REFINER, where the new queries suggested by the systems are actual refinement/enlargements of the user query with respect to the given database. The strategic help provided by FIRE is based on a set of rules for modifying the constituents of a Boolean expression, including terms, operators, and precedence, when the users faces an adverse situation. Under this respect, FIRE is more powerful than REFINER, although certain kinds of strategic help can be easily deduced from REFINER's display screen. For instance, when a query produces an empty output, REFINER shows all the minimal query enlargements with a non empty answer set (if any). Finally, Brajnik et al. [3] compared the performance of automatic support for query reformulation versus non-automatic support (such as printed thesaurus or human expert), whereas we are more interested in evaluating whether an enhanced Boolean retrieval system can perform better than a basic Boolean system.
- A great deal of the research on hybrid approaches has concentrated on the combination of browsing and querying styles of interaction. Browsing is generally seen either as an aid for conventional query-based retrieval or as an alternative search method. The first approach is described by [9] and [16] where a search through a *term* space is used to improve query formulation in a distinct *document* space. An example of the second approach is [15] where a hierarchical browsing system supplements a Boolean query system. In these systems, the network that supports browsing is usually developed and maintained as a distinct

component. The main drawback of these retrieval architectures is that the user must map different concept representations and data spaces. REFINER, by contrast, integrates browsing and querying into a single term/document space. This has the advantage that the user may combine query refinement with direct inspection of the document database. Users, therefore, do not have to commit themselves to one specific retrieval mode; rather, they may exhibit a continuum of behaviors varying between querying and browsing that is characterized by the level of specificity of their information seeking goals.

- The application of concept lattices to information retrieval has been explored by [11] and [5]. These systems build the concept lattice associated with a document/term relation and then employ various methods to access the relevant information, including the possibility for the user to search only those terms that the user has specified [5]. These systems, however, are severely affected by their computational requirements: since they build an entire concept lattices, they can only be applied to databases of modest size, where each document is described by a small number of index terms. By contrast, REFINER builds only a relevant portion of the lattice, and therefore allows full-text indexing even for reasonably-sized databases. Furthermore, REFINER combines the potentials of lattice-based inspection with the great expressive power of Boolean retrieval, while the query mode of the systems described in [11] and [5] is inherently limited.
- Much work on improving cooperative behaviour of information systems has been done in the fields of database query answering systems, logic programming, and deductive databases (see [10] for a review). One common concern is to recover from a failing query, (i.e., a query producing an empty answer set) by extending relational and deductive database systems with facilities to find minimal failing, maximal succeeding, and minimal conflicting subqueries [12]. Compared to our approach, work done in this area relies on different assumptions (structured data with powerful query languages versus unstructured data with simple query languages) and has also a different scope. While for library searches it is convenient to consider both query enlargement and query refinement, for database querying it seems more useful to concentrate on ways to recover from failing queries (which can be seen as a special case of query enlargement), considering also implicit causes of failure, such as violation of integrity constraints.

6 Conclusion

We have presented REFINER, an approach to combining Boolean retrieval and content-based navigation with concept lattices. The main advantage of the system is that it suggests controlled ways for refining/enlarging a conjunctive Boolean query. An additional distinguishing feature of REFINER is that it combines querying and browsing styles of interaction: the user may choose a hybrid retrieval strategy from a continuum ranging from casual inspection to highly-specific information seeking goals. We compared REFINER's retrieval effectiveness with that of a conventional Boolean retrieval system on a subject searching task. The results suggest that the performance of query-refinement enhanced Boolean retrieval may be better than unrefined Boolean retrieval, especially with respect to search time, and for databases of non trivial size.

This research can be extended in several directions. One fundamental design choice in REFINER is the size and the topology of the lattice region that is built and displayed in response to a query. The current version of REFINER shows the set of parents and children of a focus node, but one could use advanced visualization techniques, such as fisheye views [18], to show more distant ancestors and descendants, as well as nodes that are not directly comparable with the focus (e.g., its siblings). It is clear that a small region is more efficient to build, while a large region shows more refinements/enlargements and contains more information on the content of the database. On the other hand, it is not clear whether using a larger region will automatically result in better retrieval performance, because this may also increase the cognitive overload and the user disorientation. We are currently exploring these fundamental trade-offs.

Another related research issue concerns the scope of this approach. The experiment that we performed provides some insigths, but its results can only be taken as indicative. As suggested above, one factor that needs be controlled to evaluate the system's efficiency and effectiveness is the characteristics of the lattice region displayed by the visual interface. However, in operational situations there may be other important parameters that need be controlled. 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, indexing strategies and characteristics of domain and users.

Acknowledgments

We thank three anonymous reviewers for pointing out relevant work in the field of cooperative databases and for many useful comments and suggestions. This work has been carried out within the framework of an agreement between the Italian PT Administration and the Fondazione Ugo Bordoni.

References

1. Blair, D. (1996). STAIRS Redux: Thoughts on the STAIRS evaluation, ten years after. *Journal of the American Society for Information Science*, 47 (1), 4-22.
2. 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*, 139-146.
3. Brajnik, G,, Mizzaro, S., Tasso, C. (1996). Evaluating User Interfaces to Information Retrieval Systems: A Case Study on User Support. *Proceedings of the 19th Annual*

International ACM SIGIR Conference on Research and Development in Information Retrieval, 128-136.

4. Carpineto, C., Romano, G. (1996a). A Lattice Conceptual Clustering System and Its Application to Browsing Retrieval. *Machine Learning*, 24, 1-28.
5. Carpineto, C., Romano, G. (1996b). Information retrieval through hybrid navigation of lattice representations. *International Journal of Human-Computer Studies*, 45, 553-578.
6. Cleverdon, C. (1974). Optimizing convenient on-line access to bibliogrphic databases. Information Services and Use, 4 (1), 37-47.
7. Cooper, J., Byrd, R. (1997). Lexical navigation: visually prompted query expansion and refinement. *Proceedings of the Second ACM Digital Library Conference,* 237-246.
8. Davey, B., Priestley, H. (1990). *Introduction to Lattices and Order*. Cambridge, Great Britain: Cambridge University Press.
9. Frei, H., Jauslin, J. (1983). Graphical presentation of information and services: a user oriented interface. *Information Technology: Research and Development*, 2, 23-42.
10. Gaasterland, T., Godfrey, P., Minker, J. (1992). An Overview of Cooperative Answering. *Journal of Intelligent Information Systems*, 1(2), 123-157.
11. 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.
12. Godfrey, P. (1997). Minimization in Cooperative Response to Failing Database Queries. *International Journal of Cooperative Information Systems*, 6(2), 95-149.
13. 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.
14. Lesk, M. (1989). What to do when there is too much information. *Proceedings ACM-Hypertext'89*. Pittsburgh, PA, USA, 305-318.
15. Maarek, Y., Berry, Kaiser, G. (1991). An Information Retrieval Approach For Automatically Constructing Software Libraries. *IEEE Transactions on Software Engineering*, 17(8), 800-813.
16. 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,* Pittsburgh, PA, USA, 270-279.
17. Salton, G., McGill. (1983). *Introduction to Modern Information Retrieval.* New York: McGraw Hill.
18. Sarkar, M,, Brown, M. (1994). Graphical Fisheye Views. *Communications of the ACM*, 37(12), 73-84.
19. Spoerri, A. (1994). InfoCrystal: Integrating exact and partial matching approaches through visualization. In *Proceedings of RIAO 94: Intelligent Multimedia Information Retrieval Systems and Management*, New York, USA, 687-696.
20. Tague-Sutcliffe, J. (1992). The pragmatics of information retrieval experimentation, revisited. *Information Processing & Management*, 28(4), 467-490.
21. 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*., Dublin, Ireland, 212-220.
22. Wille, R. (1984). Line diagrams of hierarchical concept systems. *International Classification*, 2, 77-86.