*Journal of Universal Computer Science, vol. 4, no. 9 (1998), 737-747*
*submitted: 1/9/98, accepted: 21/8/98, appeared: 28/9/98* © *Springer Pub. Co.*

# IMPROVING THE EFFECTIVENESS OF WEB SEARCH ENGINES USING SELECTABLE VIEWS OF RETRIEVAL RESULTS[1]

**Ezio Berenci**
(Fondazione Ugo Bordoni, Rome, Italy
berenci@fub.it)

**Claudio Carpineto**
(Fondazione Ugo Bordoni, Rome, Italy
carpinet@fub.it)

**Vittorio Giannini**
(Fondazione Ugo Bordoni, Rome, Italy
gvittori@fub.it)

**Abstract:** Perusal of textual displays of document surrogates produced by Web-based ranked-output retrieval services may require much user time, effort, and money. In this paper we present VIEWER, a graphical interface that allows visualization and manipulation of views of retrieval results, where a view is the subset of retrieved surrogates that contain a specified subset of query terms. We argue that VIEWER helps the user focus on relevant parts of the results and, in addition, it may facilitate query reformulation. We present the results of an experiment performed by six subjects on two medium size bibliographical test collections in which VIEWER, used as an interactive ranking systems, outperformed both best match ranking and coordination level-based ranking.

**Key Words**: Information retrieval, document ranking, visualization of retrieval results, user interface
**Categories**: H.3 [Information Systems]: Information Storage and Retrieval; H.5.2 [Information Interfaces and Presentation]: User Interfaces

## 1 Introduction

Search of document databases is an interactive process in which users submit a query, see the ranked documents returned in response to the query, and submit a new query, until either they are satisfied with the results or become frustrated and give up. While most research and commercial efforts have focused on producing effective systems for retrieving and ranking relevant documents in response to a query, little attention has been paid to ease the process of result inspection and query reformulation, especially in the Web domain.

Currently available search engines (e.g., Alta Vista, Excite) score queries against documents, compute the highest scoring documents, and present the user with a set of

---

[1] This is an extended version of a paper presented at the WebNet '98 conference in Orlando, Florida. The paper has received a "TOP Full Paper Award".

document surrogates in ranked order. While the display of surrogates should allow the user to make a quick and possibly accurate judgement about the relevance of retrieved documents without downloading the full documents, its utility is reduced by the lack of effectiveness of current retrieval engines' interfaces.

One main reason for users dissatisfaction is the use of standard similarity scores. Roughly, a document receives a higher score if the terms in the query are in the headline, if the terms appear many times, if the terms do not appear in other documents, or if phrases occur as they do in the query. These criteria are then combined in various ways and produce a final numerical coefficient. This numerical coefficient - the only global document characteristic provided by the system - is difficult for the user to evaluate and can be hardly used as an indication of whether the corresponding surrogate should be perused or not. Another related limitation of current Web retrieval interfaces is the lack of a concise representation of the content of all retrieved documents; conventional textual displays take much perusal time and screen space and does not enable inspection of more documents at a time. Given these characteristics of the interface, the inspection of Web retrieval results usually implies for the user to go through the document hitlist produced by the system, spending a considerable amount of time, effort, and money (for those systems that charge the user based on connect-time and the volume of downloaded data) for perusal of document surrogates.

One way to alleviate this problem is to develop a graphical interface that displays the characteristics of documents which are significant in supporting the decision to peruse or not, while giving the user more control over the set of document surrogates that can be selected for perusal. The need for concise display and user-oriented manipulation of retrieval results has been addressed by various systems. Among others, Bead [Chalmer and Chitsons 1992] and LyberWorld [Hemmje, Kunkel and Willet 1994] depict clustering patterns in a document space using three-dimensional visualization schemes, InfoCrystal [Spoerri 1994] uses a particular visual representation of a Venn diagram to suggest how to refine Boolean queries, TileBars [Hearst 1995] displays distribution of query terms within each document to locate its relevant parts, and Ulysses shows a lattice of terms and documents that can be searched in various and integrated ways [Carpineto and Romano 1996]. Most of the proposed approaches, however, cannot be applied to Web-based retrieval because they are either computationally expensive, or require sophisticated graphical facilities, or do not scale well, or rely on different underlying retrieval models, or, more often, present a combination of these features. One notable exception is [Veerasamy and Heikes 1997]'s system. Its main goal is to clarify the role played by query constituents in the result of ranked output systems, it is computationally efficient, and it uses a relatively simple graphical display. Our approach shares a similar concern but employs a radically different visualization and interaction scheme. Instead of visualizing the weights of the query terms of each retrieved document and let the user select those of interest, as in Veerasamy's system, we concentrate on all the possible subsets of query terms (i.e., subqueries) that can be generated from the user query, showing their distribution in the set of retrieved documents and letting the user to select the associated set of documents. We speak of view, because in this way the user may see parts of results without seeing the whole list. Views are defined in a precise way from the retrieved documents through a simple and comprehensible characteristic of their content, i.e., the subset of distinct query terms that they contain.

In the rest of this paper we present VIEWER (VIEws of WEb Results), a system for seeking information over the Web based on view manipulation. VIEWER copes with most computational constraints of Web-based retrieval (e.g., efficiency, portability, adaptability) that are not usually addressed in other document visualization systems and can be used as an interface to currently available search engines. We argue that VIEWER can be used both for focusing on relevant items of the document hitlist returned by the search engine and for driving the process of query reformulation. A major part of this work is then an experimental evaluation of the former aspect. We compare the effectiveness of VIEWER, seen as an interactive ranking system, and that of two automatic ranking systems, namely the search engine to which VIEWER is linked and a coordination level-based system. The results of the experiment are encouraging.

## 2 Visualization and manipulation of Web retrieval results with VIEWER

VIEWER is built around available «primary» Web search services, presenting users with a single unified interface [Fig. 1]. Users enter a query, which VIEWER forwards to a selected search engine (Alta Vista, in the current implementation).

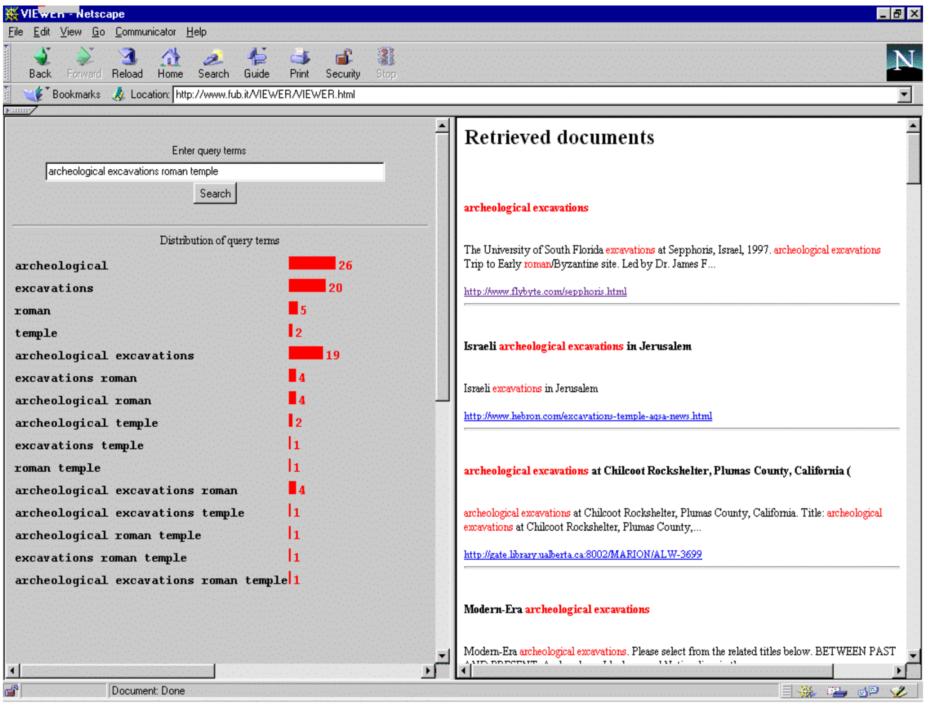


Figure 1: Visualization of retrieval results with VIEWER. The graphical distribution of subqueries in the first 40 retrieved document surrogates is displayed. Clicking a horizontal bar brings up the associated document surrogates in the document window.

VIEWER then collects the query results and shows, in a scrollable window, a subset of the document surrogates, in the same ranked order as returned by the search engine; in addition, it shows, in the rest of the screen, a graphical visualization of results. The visualization consists of an aligned sequence of horizontal bars, one for each of the $2^n-1$ nonempty subqueries that can be formed with the $n$ query terms. Subqueries are displayed in the order of increasing number of terms, with the longest subqueries at the bottom; the length of each bar is proportional to the number of document surrogates containing that subquery, which is also explicitly displayed next to the bar[2]. By clicking on a bar the user may select the corresponding view, bringing up the associated surrogates into the document window. As an example session with VIEWER, consider searching the following subject: «archeological excavations for Roman temple». [Fig. 1] shows the response of the interface to the user query: *archeological excavations roman temple*, as of February 20, 1998. The graphical display quickly shows that the results produced by Alta Vista are, in general, dissatisfying, because most retrieved documents do not apparently deal with Roman temples. In fact, there is only one truly relevant document - the one whose surrogate contains all query terms, which is, incidentally, the eight item returned by Alta Vista - with most of the first 40 retrieved documents concerning archeological excavations of different kinds. VIEWER allows the user to select those few surrogates that appear to or might be relevant, without perusing the others. In addition, it suggests that if the user is primarily interested in «Roman temples», he or she should submit a new query that does not contain the two words: archeological excavations.

## 3 Utility of view display and selection

It is often the case that some subsets of query terms are so important for a search topic that they will appear in all relevant documents (e.g., the subqueries «archeological excavations» and «Roman temple», in the example given above). In this situation, we would like to ask such questions as: how many documents contain the subquery *s*?, which documents contain subquery *s*? Also, if subquery *s* is contained in too many documents, we would like to ask: which terms can be added to *s* in such a way that the resulting set of documents is more manageable? If, on the contrary, subquery *s* is contained in few documents, we might ask: which terms can be deleted from *s* in such a way that the resulting set of documents is still manageable? We might also be interested in the relationships between different subqueries. So we might ask: what is the contribution of subquery *q* compared to subquery *s*?, does subquery *q* occur more frequently alone or in conjunction with *s*?, and so on.

With conventional ranked output systems such questions would remain unanswered, because the logic behind the retrieval mechanism of these systems does not allow the user to understand how the query constituents influenced the retrieval results. By contrast, VIEWER's graphical display of subquery distribution together

[2] It should be noted that some search engines, including Alta Vista, do full-text indexing. So it may happen that some query term is contained in the full document but not in its surrogate. However, given the ranking criteria described above, for best-ranked documents this is not usually the case.

with the possibility for the user to choose the relevant views seem to allow a quick answer for such questions. In addition to enriching inspection of retrieval results with facilities for selection, comparison and refinement involving groups of query terms, VIEWER has also potentials for facilitating query reformulation. It may help the user detect that some term (or subquery) is much more frequent than others in the first retrieved documents, in which case the user may formulate a new query using a narrower term or adding a term that helps to specify its intended sense, or, conversely, it may show that some query term is very rare, which may imply using a broader term or deleting some other frequent term from the query. In fact, the latter information may be very useful in Web-based retrieval, because it is often the case that while there are thousands of retrieved documents that match some query term, none or very few of them is contained in the first pages returned to the user (as the subquery «Roman temple» in the example given above). The graphical component of VIEWER may also help detect failure of intended senses of words, i.e., when two terms used in the query to identify one particular meaning do not occur together in the retrieval results, or, symmetrically, discover unwanted senses of words [Cooper and Byrd, 1997].

## 4 Architecture of VIEWER

VIEWER is a client-server system with two main components: the user interface and the search manager.

The interface program is implemented as a Java applet in a Web browser, which can be downloaded with the web page *http://www.fub.it/VIEWER/VIEWER.html.* The user interface sends the user query to the server machine, which forwards the query to a selected search engine and then collects and parses the first pages retrieved by the search engine in response to the query. The server, then, sends back to the client the parsed retrieved document surrogates, which are used by the interface component to produce the information displayed by VIEWER on the client's screen. VIEWER's code can be easily modified to work with different search engines or to adapt to their interface changes. The number of collected pages is a system design parameter, currently set at four.

## 5 Evaluation of VIEWER

### 5.1 Goal

The goal of the experiment was to evaluate the effectiveness of VIEWER in help the user focus on relevant items of the document hitlist produced by a ranked output retrieval system on a subject searching task. We compared the performance of VIEWER, used as an interactive ranking system, with that of two distinct automatic ranking systems: WAIS, a retrieval engine on top of which VIEWER was built to perform the experiment, and a coordination level-based ranking system, which uses a retrieval scheme similar to that of VIEWER.

### 5.2 Subjects

We tested six subjects in our experiment. The subjects were recruited in our institute; they had a computer science background with little knowledge about the subject searching domain. Each subject was provided with a short tutorial session on a training database to ensure that he or she could easily manipulate the interface used in the experiment.

### 5.3 Databases and queries

We did not perform subject searching over the Web because it would be difficult to assess the system's retrieval effectiveness and do comparative studies in this unrestricted domain. Rather, we used the CISI and the CACM collections, two electronically-available bibliographical collections widely used for laboratory tests. The CISI collection contains 1460 information science documents, the CACM collection consists of 3204 abstracts of the «Communications of ACM». For the experiment, we used a set of two to four term queries based on the topics taken from the two test collections, 35 for CISI and 52 for CACM. The queries were created by selecting terms from the topics and had an average length of 3.9 and 3.8 terms, for CISI and CACM, respectively.

### 5.4 Implementation of the two automatic ranking systems

As mentioned above, we considered two automatic ranking methods for comparative evaluation with VIEWER: WAIS and coordination level-based ranking.

WAIS is a classical ranked-output retrieval engine. We chose WAIS because it can be used for indexing and searching site specific information, as opposed to global Web search. We connected the WAIS server to the two test collections and executed their topics against the corresponding WAIS database.

Coordination level (CL) is a well known retrieval method that ranks the documents according to the number of distinct query terms that they contain, which is referred to as their coordination level (see for instance [Van Rijsbergen 1979]). Coordination levels therefore resemble of views, but should not be confused with them: a coordination level contains all views with a same number of distinct terms, regardless of the actual terms that describe each view. The CL method automatically returns a complete ranking. If the user query contains *n* terms, the documents that contains n query terms are ranked before those containing *n-1* terms, which, in turn, are ranked before those containing *n-2* terms, and so on. As CL produces a partly-ordered retrieval output, we further ranked the (equally-ranked) documents within each coordination level by using the ranking produced by WAIS for those documents. The topics of each test collection were then executed over their own database using overall ranking.

### 5.5 VIEWER as an interactive ranking system: test design

VIEWER cannot produce a ranking by itself, but it can be used to help users build their own document ranking. We designed an interactive procedure that was executed by the six subjects. The procedure works as follows. Each subject was shown the topic, the query extracted from it, and VIEWER's visualization of the distribution of query terms among the documents returned by WAIS in response to the query. Then the subject was asked to choose a sequence of views by repeatedly selecting one of the views offered by VIEWER until all views had been selected. As an illustration, [Fig. 2] shows an example screen produced during the search of the CACM documents relevant to the topic: «Optimization of intermediate and machine code». The topic was translated into the four term query: «optimization», «intermediate», «machine», and «code». The screen shows all possible views; the subject may select a view by clicking on its bar. After a view has been selected by the subject, it is shaded and numbered, according to the selection order. In [Fig. 2], four views have been selected, the last of which is: «optimization code».

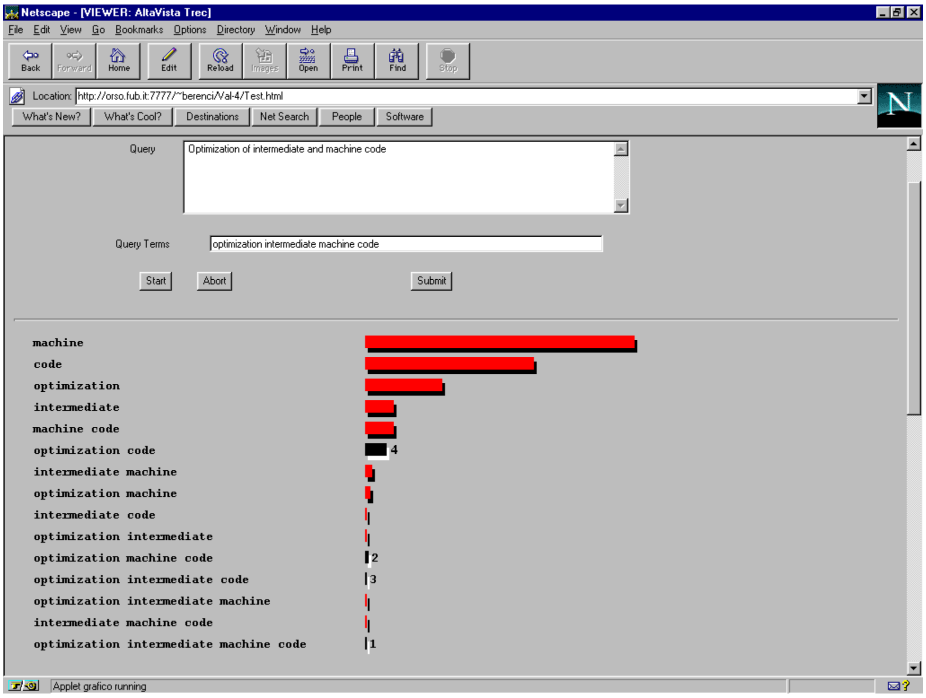


Figure 2. Ordering of the views associated with the CACM topic: *Optimization of intermediate and machine code.*

The documents were thus ranked according to the order chosen by the user to select the views. Documents contained in more views were ranked based on the earliest view in which they occurred. As with CL, we used the ranking produced by WAIS as a secondary ranking procedure to rank the documents within each view. As a result of this process, the final ranking built by the user corresponds to a particular sorting of the documents contained in the output returned by WAIS. Each subject took about an hour to execute the 35 queries of the CISI collection and about one and a half hour to execute the 52 queries of the CACM collection.

### 5.6 Results

The results, for each database, are displayed in [Fig. 3] and [Fig. 4].

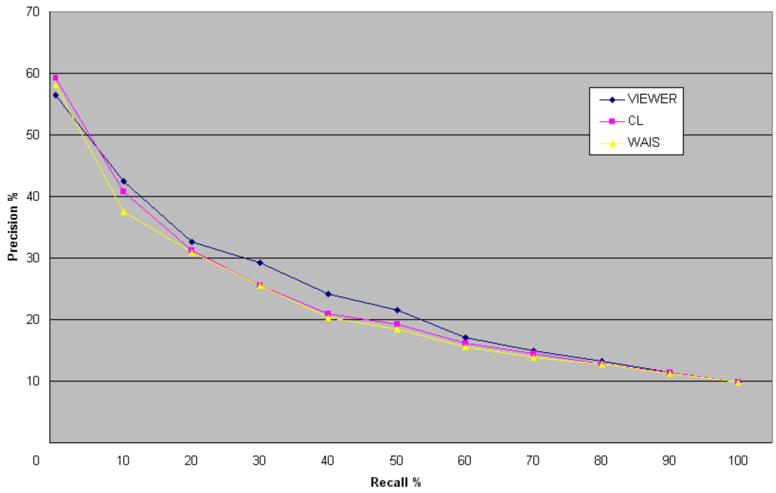


Figure 3: Precision and recall graph for the CISI test collection

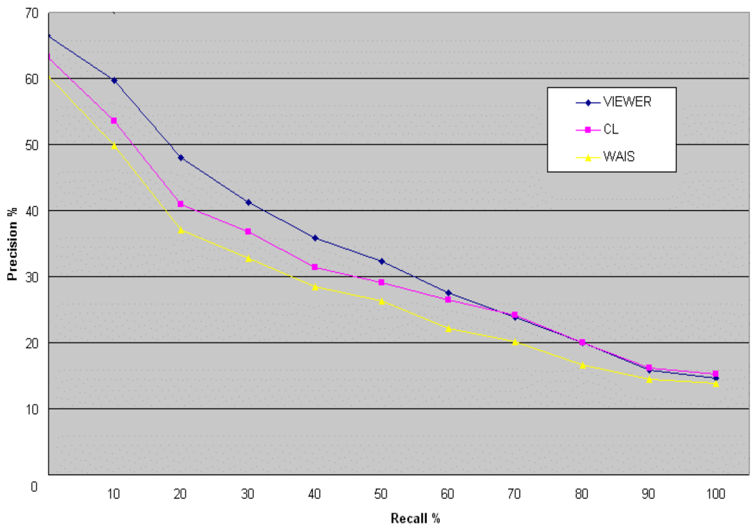


Figure 4: Precision and recall graph for the CACM test collection

The precision-recall curve is normalized considering, for each query, only the relevant documents that contain at least one query term; i.e, those that are actually retrieved and ranked by the three methods. The [Fig. 3] and [Fig. 4] curve reports interpolated precision at eleven recall levels, averaged over the query set; the results of VIEWER are averaged over the six subjects.

Both [Fig. 3] and [Fig. 4] show that the performance of VIEWER was better than WAIS and CL, which, in turn, was better than WAIS. The better precision of VIEWER over WAIS and CL, and of CL over WAIS is apparent at almost all the recall levels. A paired *t*-test performed, for each collection, over the whole set of data (i.e., values of precision for all queries at all recall levels) revealed that the difference between VIEWER and CL and between CL and WAIS were not statistically significant, but it did confirm the superiority of VIEWER over WAIS, at least for the CACM database ($p = 0{,}04$). These results therefore support our main claim that views selection allows the user to extract relevant documents from the output returned by best-match search engines. In addition, our results show another interesting and somewhat less expected phenomenon.

That the performance of CL was better than WAIS might, in effect, seem counter-intuitive, because best-match retrieval methods are generally considered to be more effective than exact-match retrieval methods, including CL. However, some evidence has been recently reported [Clarke, Cormack and Tudhope 1997] that shows that coordination level-based ranking performs better than best-match ranking when the user queries are short. In this respect, our results confirm these earlier finding on two different test collections; also, they suggest that interactive subquery ordering may be more effective than automatic ordering based on coordination level, although we should emphasize that [Clarke, Cormack and Tudhope 1997] employ a method for ranking the documents within a coordination level that is different from ours. The main reason for justifying the superiority of the interactive method over the automatic one is that the latter is a purely syntactic method. The automatic method therefore fails to recognize all the situations in which documents containing fewer query terms are more relevant than documents containing more query terms. This usually happens when there is a short subquery that corresponds to a concept that is relevant to the subject being searched while other longer subqueries are theoretically available that do not convey a precise meaning. Consider, for instance, the CACM topic shown in [Fig. 4], i.e., «optimization of intermediate and machine code», translated into the four term query: «optimization intermediate machine code». While «code optimization» is a relevant subquery for the topic, there are longer subqueries that are less relevant («intermediate machine code») or even meaningless («intermediate machine optimization»). In fact, the subject involved in [Fig. 4] consistently favoured the shorter, but more relevant, subquery «code optimization». Indeed, in our experiment, it was often the case that users favoured shorter subqueries. In particular, for a significant percentage (32%), the users applied a view which would have not been selected by the syntactic method. The results of the experiment, therefore, suggests that this situation occurs frequently and that it may significantly affect the overall performance. Our solution to this problem is semi-automatic; an automatic solution would be highly desiderable but it could be hardly achieved at the moment, even because there is no simple way of dealing with such lexical items as initials, acronyms, and proper nouns, which frequently occur in user queries.

## 6 Limitations and future work

The visualization component of VIEWER works with a limited number of query terms (in the current implementation, up to four), because the display of the subquery distribution in the retrieved documents would become cumbersome for longer queries. In practice, however, this may not be a serious limitations, for it has been often remarked that, at least in Web settings, queries to text retrieval systems are usually extremely short, often containing no more than two or three terms [Rose and Stevens 1996; Clarke, Cormack and Tudhope 1997]. Another important design parameter of VIEWER is the amount of textual information extracted from the retrieval results and used as input data to the interface. For each query, VIEWER processes only a limited subset of the document surrogates returned by the search engine, where each documents is usually described by a few tens of terms. Of course, it would be useful to work with more documents and more terms per document. However, increasing the number of terms per document is unfeasible unless we download the full documents referenced by the search engines, which would take an exceedingly long time, while increasing the number of document surrogates can only be done at cost of significantly slow down the response time, although such an additional time cannot be exactly estimated. Since the computation time required to build the visualization scheme is negligible, taking no more than a few seconds, the key parameter for VIEWER's efficiency is the search time. With a few tens of document surrogates, the total response time of VIEWER is usually fairly acceptable.

This research can be extended by further exploring the issue of the utility of graphical displays of Web retrieval results on more robust and realistic basis. The experiment reported in this paper has taken a first step into this somewhat overlooked direction, but our results can only be taken as indicative and much more work is needed. As suggested above, one factor that needs to be controlled for evaluating the efficiency and effectiveness of this kind of systems is the fraction of retrieval results used by the visual interface. In operational situations, however, there may be other important parameters that affect their overall utility. We plan to perform further experiments to evaluate how the performance results change when controlling a wider range of factors including database scale, query length, and possibility for the user to formulate more queries.

## References

[Carpineto and Romano 1996] Carpineto, C., & Romano, G. (1996). Information retrieval through hybrid navigation of lattice representations. *International Journal of Human-Computer Studies*, 45, 553-578.

[Chalmers and Chitson 1992] Chalmers, M., & Chitson, P. (1992). "Bead: explorations in information visualization", Proceedings of SIGIR'92, Copenhagen, Denmark, 330-337.

[Clarke, Cormack and Tudhope 1997] Clarke, C., Cormack, G., Tudhope, E. (1997) "Relevance ranking for one to three term queries". Proceedings of RIAO'97: Computer-assisted information searching on the Internet, Montreal, Canada, 388-400.

[Cooper and Byrd 1997] Cooper, J., & Byrd, R. (1997). "Lexical navigation: visually prompted query expansion and refinement". Proceedings of the 2nd ACM Digital Library Conference, 237-246.

[Hearst 1995] Hearst, M. (1995). "TileBars: Visualization of term distribution information in full text information access". Proceedings of CHI'95, Denver, Colorado, USA, 59-66.

[Hemmje, Kunkel and Willet 1994] Hemmje, M., Kunkel, C., & Willet, A. (1994). "LyberWorld - A visualization user interface supporting full text retrieval". Proceedings of SIGIR'94, Dublin, Ireland, 249-259.

[Rose and Stevens 1996] Rose, D., & Stevens, C. (1996). "V-Twin: A lightweight engine for interactive use". Proceedings of TREC-5, Gaithersburg, Maryland, USA.

[Spoerri 1994] Spoerri, A. (1994). "InfoCrystal: Integrating exact and partial matching approaches through visualization". Proceedings of RIAO'94: Intelligent Multimedia Information Retrieval Systems and Management, New York, New York, USA, 687-696.

[Van Rijsbergen 1979] Van Rijsbergen, C. (1979). "Information Retrieval" (second edition), Butterworths, London.

[Veerasamy and Heikes 1997] Veerasamy, A., Heikes, R. (1997). "Effectiveness of a graphical display of retrieval results". Proceedings of SIGIR'97, Philadelphia, USA, 236-245.

## Acknowledgments

We would like to thank Giovanni Romano for his help in preparing the experiment and evaluating the results. We would also like to thank Stefano Mizzaro for many useful comments on an earlier version of this paper. This work has been carried out within the framework of an agreement between Telecom Italia and the Fondazione Ugo Bordoni.