The Effectiveness of Automatically Structured Queries in Digital Libraries

Marcos André Gonçalves[*] Edward A. Fox[*] Aaron Krowne[†]
Pável Calado[‡] Alberto H. F. Laender[‡] Altigran S. da Silva[§] Berthier Ribeiro-Neto[‡]

[*]Dept. of Computer Science
Virginia Tech
Blacksburg, VA, USA
{mgoncalv,fox}@vt.edu

[†]Library Systems
Emory University
General Libraries
Atlanta, GA, USA
akrowne@emory.edu

[‡]Dept. of Computer Science
Federal University of Minas Gerais
Belo Horizonte, MG, Brazil
{pavel,laender, berthier}@dcc.ufmg.br

[§]Dept. of Computer Science
Federal University of Amazonas
Manaus, AM, Brazil
alti@dcc.fua.br

ABSTRACT

Structured or fielded metadata is the basis for many digital library services, including searching and browsing. Yet, little is known about the impact of using structure in the effectiveness of such services. In this paper, we investigate a key research question: do structured queries improve effectiveness in DL searching? To answer this question, we empirically compared the use of unstructured queries to the use of structured queries. We then tested the capability of a simple Bayesian network system, built on top of a DL retrieval engine, to infer the best structured queries from the keywords entered by the user. Experiments performed with 20 users working with a DL containing a large collection of computer science literature clearly indicate that structured queries, either manually constructed or automatically generated, perform better than their unstructured counterparts, in the majority of cases. Also, automatic structuring of queries appears to be an effective and viable alternative to manual structuring that may significantly reduce the burden on users.

Categories and Subject Descriptors

H.3.7 [**Information Systems**]: Information Storage and Retrieval—*Digital Libraries*

General Terms

Experimentation, Human Factors

Keywords

digital libraries, structured queries, Bayesian networks

1. INTRODUCTION

Ensuring the high quality of Digital Library (DL) services is key to guaranteeing DL usefulness and patrons' satisfaction. Largely because of this concern for quality, metadata, and more specifically, structured or fielded metadata, has historically been the basis for many digital library services, including basic ones such as searching and browsing. Yet, regarding the effectiveness of such services, little is known about the impact of using structure. Moreover, while a few DL services try to utilize this information through the use of advanced interfaces[1], experience has shown that users rarely make use of these features, most probably due to the complexity of user interfaces and lack of knowledge of internal DL structures.

In this paper, we investigate a key research question: do structured queries improve search effectiveness in DLs? To answer this question, we empirically compared the use of unstructured queries to the use of structured queries. Since users are often unwilling, or unable, to manually structure their queries, we also provide a simple system that tries to close the gap between the user's information need and the DL content. This experimental engine, built on top of a Bayesian network model and a retrieval system optimized for DLs, tries to infer the best structured queries from the keywords entered by the user, based on knowledge of DL structures and collection statistics. A very simple text box user interface guarantees the simplicity of the process. To ensure proper treatment of their information need, users simply have to choose from an automatically produced ranked list of structured queries.

To test our hypotheses and methods, we performed a series of experiments with 20 users (graduate students and

[1]See, for example, `http://www.acm.org/dl`, or `http://www.informatik.uni-trier.de/~ley/db/indices/query.html`.

researchers) using CITIDEL (Computing and Information Technology Interactive Digital Educational Library)[2], a DL containing a large collection of computer science literature, including metadata from the ACM Digital Library, the DBLP collection, NDLTD-Computing (the computing subset of the Networked Digital Library of Theses and Dissertations)[3], and others sources. Results, using three different information retrieval measures, indicate that structured queries, either manually constructed or automatically generated, perform better than their unstructured counterparts in the majority of cases. Also, automatic structuring of queries appears to be a viable alternative to manual structuring, since it reduces work for users, while yielding superior effectiveness.

This paper is organized as follows. Section 2 explains the underlying models and context of the work. Section 3 describes ESSEX, a retrieval system optimized for DLs, which provides for basic retrieval capabilities and for the structuring process. Section 4 details the query structuring process, including the Bayesian network model and the query ranking schemes. Section 5 discusses experimental setup and results. Section 6 presents related work and Section 7 concludes the paper, also including plans for future work.

2. CONTEXT AND DEFINITIONS

In this work, we adopt a simplified view of the structured metadata that describes the contents of a DL. According to this view, each document or digital object do_i stored in the DL is described by, at least, one *metadata specification.* The j-th metadata specification for object do_i is defined as a set of pairs:

$$ms_{ji} = \{A_1 : v_{1_{ji}}, \ldots, A_n : v_{n_{ji}}\},\ n_{ji} \geq 1$$

where each A_k is an *attribute* or *metadata field* and each $v_{k_{ji}}$ is a value belonging to the domain of A_k. We note that the attributes do not need, necessarily, to be the same for all metadata specifications.

For some attributes, instead of a single value, we may have a set or list of values. For instance, in a metadata specification describing a paper, the attribute `author` might be a list of names. To represent this using our notation, we allow a same attribute to appear several times, here called a *value list.* Thus, if attribute A_p, in metadata specification ms_{ji}, has n different values, we can represent metadata specification ms_{ji} as:

$$ms_{ji} = \{\ldots, A_p : v_{1_p}, A_p : v_{2_p}, \ldots, A_p : v_{n_p}, \ldots\}$$

We define the *metadata schema* of a DL as the set of all attributes that compose any of the metadata specifications of that DL. Thus, the metadata schema of a DL D is defined as:

$$S_D = \{A | A \text{ is an attribute of some metadata specification in } D\} \quad (1)$$

We define an *unstructured query* U as a set of keywords (or *terms*):

$$U = \{t_1, t_2, \ldots, t_k\}$$

[2]See `http://www.citidel.org/`.
[3]See `http://www.ndltd.org/`.

As for a metadata specification, a *structured query* Q is defined as a set of pairs:

$$Q = \{A_1 : v_{1_q}, \ldots, A_n : v_{n_q}\},\ n_q \geq 1,$$

where each A_k is an attribute or metadata field and each v_{kq} a value belonging to the domain of A_k.

This simplified set of definitions allows us to ignore the details of how metadata is actually represented in the DL, since it can mapped from any actual representation format.

3. THE RETRIEVAL SYSTEM: ESSEX

ESSEX is a vector-space IR system optimized for the digital library setting. It is designed to be light and fast and to make few demands on the architecture of the rest of the DL system. It achieves these objectives by an optimized C++ implementation, an entirely in-memory index, and a background daemon model using socket communication with the DL application.

In addition to these architectural provisions, ESSEX has a number of query language features that make it well suited to digital libraries. Besides basic features such as force/forbid ("+" and "-") term operators, ESSEX supports field filters and adjustable field weightings[4].

Field filters have the syntax "field:term", where "field" is an indexed metadata field, and "term" is the query term. A field filter modifies the behavior of the search such that matches will only be made with term occurrences within the specified field.

ESSEX was developed primarily for CITIDEL and currently serves as the search engine for CITIDEL and PlanetMath[5]. Our familiarity with the code made it a natural choice as a test-bed for the experimental query structuring system discussed in this paper. In addition, ESSEX's field filtering capability served as the core of the query structuring engine. We also utilized ESSEX's support for the "+" operator, and may use its field weighting support in the future. Details on how some of these features were used are explained in the following sections.

4. RANKING QUERIES: THE BAYESIAN NETWORK MODEL

This section presents an overview of the automatic query structuring approach. We start by describing the general querying process and explain how user queries are structured automatically and ranked according to the likelihood that they will satisfy the user's needs.

4.1 The Query Structuring Process

In our ESSEX query structure inference system, query structuring consists of: (1) collecting the unstructured user query, (2) building a set of candidate structured queries, and (3) ranking the candidate queries according to the probability of best representing the user's needs, as proposed in [8,12].

[4]Field weightings allow the DL provider and user to change the contribution of the various metadata fields to the final results ranking. For more information on this and other features of ESSEX and how they make it a superlative choice for a digital library search engine, see `http://br.endernet.org/~akrowne/elaine/essex/index.html`.
[5]See `http://planetmath.org/`.

To explain these steps in detail, assume that the objects in our digital library have fields `author` and `title`. Let $U = \{t_1, t_2, t_3\}$ be the initial, unstructured query entered by the user, where t_1, t_2, and t_3 are three distinct terms. To create the candidate queries, ESSEX simply builds all possible combinations of field-term pairs, using the fields in the metadata schema of the DL and the terms entered by the user.

To illustrate, if term t_1 occurs both in the `title` and in the `author` fields of the objects in the DL, pairs $<$ `author` $: t_1 >$, and $<$ `title` $: t_1 >$ would be created. Similarly, if terms t_2 and t_3 occur only in the title of the objects in the DL, pairs $<$ `title` $: t_2 >$, and $<$ `title` $: t_3 >$, would be created. Given these field-term assignments, the candidate structured queries would be $Q_1 = (\texttt{author} : t_1, \texttt{title} : t_2, \texttt{title} : t_3)$ and $Q_2 = (\texttt{title} : t_1, \texttt{title} : t_2, \texttt{title} : t_3)$. Notice that these would be the only two possible queries, since we assume that one term cannot occur in two different fields of the same query. In this case, term t_1 cannot occur in the `title` field and `author` field of the same query.

The creation of the field-term pairs can be further restricted by considering a minimum frequency of occurrence of a term in the field values of the digital objects in the DL. Thus, if, say, term t_1 occurs less than N times in the `author` field, the pair `author` $: t_1$ would not be created. The value of N can be used both to increase efficiency, by reducing the number of candidate queries, as also to filter out spurious terms that may occur in a field due to errors in the data. In our experiments the value of N was set to 1, since this filtering process was proved unnecessary.

Once the set of candidate queries is created, each query is evaluated and ranked according to the probability of fitting the data in the DL. This is accomplished through the use of the Bayesian network model first proposed by Calado et al. in [8], as explained in the following section.

Figure 1 shows the architecture for the query structuring process in ESSEX. Evaluation of a structured query takes place in two phases. The first is the evaluation of the individual query terms (with field filters). For this phase, each term is sent to the search engine and a results set is received. The ranks of the results set documents are combined into a score for the query term. Because many structured query terms occur numerous times over the entire set of candidate structured queries, they are cached in a hash table which maps them to their corresponding fused scores. In our experiments, we found that this caching speeded up the entire structuring process by more than a factor of 3.

In the second phase, the scores of the structured query terms are combined into a final score for the entire structured query. This score is then used to generate the ranks for the set of all potential structured queries.

In cases were the full digital library content is not accessible, or in order to improve efficiency, not all of the DL objects are used in the query structuring process. Instead, only a subset of the DL is considered for building the candidate structured queries and for deriving the statistics necessary to the Bayesian network model. This subset is called the *sample database* and is generally built by taking a sample of objects from the DL that are representative of the whole DL content. A more detailed discussion on how a sample database is built can be found in [12].

We now present a brief explanation of the Bayesian network model used as a basis for this implementation, and emphasize the changes needed to adapt it to our experimental collection.

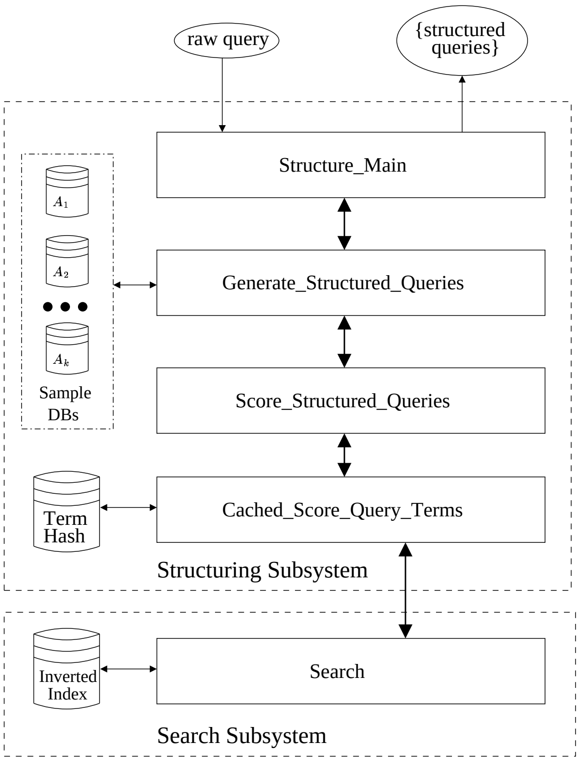

Figure 1: Architecture of the query structuring system.

4.2 Finding the Best Structured Queries

The ranking of candidate structured queries is accomplished through the use of the Bayesian network model proposed in [8]. For clarity, the model is presented in Figure 2. We note that although the network can be easily expanded to model any metadata schema, for simplicity, here we show only two fields, A_1 and A_2.

The network in Figure 2 consists of a set of nodes, each representing a piece of information. With each node in the network is associated a binary random variable, which takes the value 1 to indicate that the corresponding information will be accounted for in the ranking computation. In this case, we say that the information was *observed*. In the network, the DL is represented by node O, each node A_i represents a field, each node A_{ij} represents a value of field A_i, each node a_{ij} represents a term in the value of field A_i, each node Q_i represents a structured query to be ranked, and each node Q_{ij} represents the portion of the structured query Q_i that corresponds to the field A_j. Vectors $\vec{a_1}$ and $\vec{a_2}$ each represent a possible state of the variables associated with nodes a_{1i} and a_{2i}, respectively.

The likelihood of a candidate structured query Q_i fitting the DL O can be seen as the probability of observing Q_i, given that O was observed, i.e., $P(Q_i|O)$. By appropriately defining the conditional probabilities described by the net-

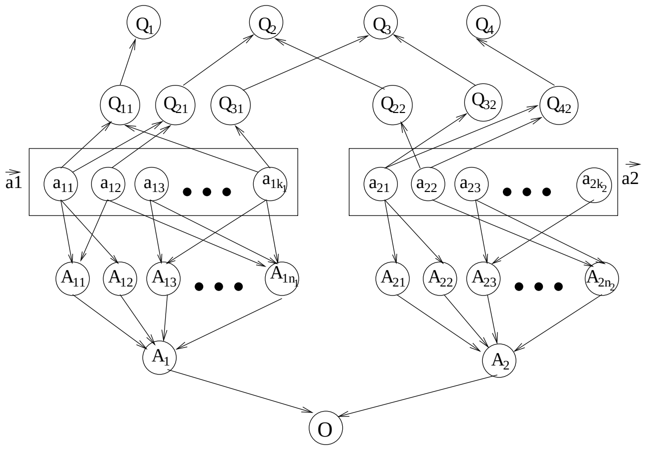

Figure 2: Bayesian network model for ranking structured queries.

work in Figure 2, we obtain the following equation:

$$P(Q_i|O) = \eta \times \frac{1}{2}\Big[1 - \prod_{j=1}^{n_1}\big(1 - \cos(A_{1j}, \vec{a}_1)\big) + 1 - \prod_{j=1}^{n_2}\big(1 - \cos(A_{2j}, \vec{a}_2)\big)\Big] \quad (2)$$

where $\vec{a}_1$ and $\vec{a}_2$ are the states in which only the query terms referring to fields A_1 and A_2, respectively, are observed; n_1 and n_2 are the total number of values for fields A_1 and A_2 in the DL; and η accounts for the constants α, $P(\vec{a}_1)$, and $P(\vec{a}_2)$. It is important to note that, although in [8] disjunctive and conjunctive operators were suggested for the final combination function, empirical tests with the collection used in our experiments indicated that using disjunctive operators for probability $P(A_i|A_{ij})$ and a mean for probability $P(O|A_i)$ yielded the best results. For further details on the derivation of Eq. (2), refer to [8, 22].

The function $\cos(A_{ij}, \vec{a}_i)$ represents the similarity between the field value A_{ij} and the terms in the candidate query being ranked. It is defined as the traditional vector space cosine similarity [25] between the vector of terms representing the field value A_{ij} and vector $\vec{a}_i$, which represents the terms in the query.

To compute this similarity, the value A_{ij} is seen as a vector of k_i terms. To each term t in A_{ij}, we assign a weight w_{it} that reflects the importance of the term for field A_i:

$$w_{it} = \mathrm{tf}_j(t) \cdot \mathrm{ftf}_i(t) \cdot \mathrm{fidf}(t) \quad (3)$$

where $\mathrm{tf}_j(t)$ is the term frequency of term t in document j, i.e., the number of times term t appears in document j; ftf_i is the field term frequency, i.e., the number of times term t occurs in field i; and fidf is the inverse field document frequency, i.e., the inverse of the number of fields term t appears in.

The first factor in Eq. (3), $\mathrm{tf}_j(t)$, is very common in vector-space IR. It indicates that the more times term t appears in document j, the more representative t is of document j. Although it is also common in IR to have an idf, or "inverse-document frequency" factor, we leave this out for reasons discussed below.

The second two factors are novel in our work. On the one hand, $\mathrm{ftf}_i(t)$, indicates that the more times term t appears in a field i, the more representative t is of field i. On the other hand, if term t appears in many fields, the factor $\mathrm{fidf}(t)$ indicates that it is probably too generic to be useful. We call these "field tf" and "field idf" respectively, as they are analogous to the standard tf and idf described previously. The difference is that they reflect term distribution relative to fields rather than documents. These term weighting functions differ from those used in [8] due to the fact that CITIDEL, and many other digital libraries, contain collections of scientific papers and, therefore, many textual metadata fields, being very different from the collections used in [8], which contained mostly information on commercial products extracted from Web databases. In the context of query structuring, one of the main differences in the CITIDEL case is that there is a large overlap between the vocabulary in the object's fields such as titles, abstracts, publications, and authors.

The net effect of these weightings is to value terms (1) strongly to the extent that they occur many times in the specified metadata field, (2) strongly to the extent that they occur in the most common field for the term, and (3) weakly to the extent that they are "diluted" by appearing in many metadata fields.

Let us consider an example of how this is useful. Assume that the unstructured query is given as "jones algorithm", and that the word "algorithm" appears evenly in the `title` and `abstract` fields, and a handful of times in the `publication` field. Also, assume the word "jones" appears a small amount in the field `abstract`, but much more frequently in the field `author`. With the weightings described above, occurrences of "algorithm" will have similar value in either the `title` or `abstract` fields. However, occurrences of "jones" in `author` will be worth much more than occurrences in `abstract`. Finally, occurrences of "algorithm" will be worth less than occurrences of "jones", because "algorithm" appears in three fields, while "jones" appears in only two.

Given this weighting scheme, the cosine of the angle between vector A_{ij} and vector $\vec{a}_i$ is defined as:

$$\cos(A_{ij}, \vec{a}_i) = \frac{\sum_{\forall t \in T_i} w_{it} g_t(\vec{a}_i)}{\sqrt{\sum_{\forall t \in T_i} w_{it}^2}} \quad (4)$$

where $g_t(\vec{a}_i)$ gives the value of the t-th variable of the vector $\vec{a}_i$, and T_i is the set of all terms in the values of field A_i.

We now can rank all the structured queries by computing $P(Q_i|O)$ for each of them. The user then can select one query for processing from among the top ranked ones, or the system can simply process the first query.

5. EXPERIMENTS

To test our research questions, namely, (1) if structured queries are better than unstructured ones and (2) if automatically structured queries can perform as well as (or better than) their manually constructed counterparts, we conducted a series of experiments with real users and the structuring Bayesian network, as described in Section 4, implemented on top of the ESSEX search engine.

5.1 Experimental Setup and Design

Experiments were performed on the CITIDEL collection which contains metadata from the ACM Digital Library, the DBLP collection, NDLTD-Computing, and other sources - totaling more than 440,000 metadata records. Only a subset of the ACM Digital Library, with approximately 98,000 metadata records, was used as a sample database for query structuring. This means that all information used by the Bayesian network model to rank the structured queries was taken only from this subset of CITIDEL. The ACM DL subset was chosen as the sample database since it contained the greatest breadth and depth of metadata, hence providing a comprehensive amount of metadata and content widely representative of the metadata and content of the whole collection. The set of metadata fields considered in the experiment was $S_{CITIDEL} = \{$`title`, `abstract`, `author`, `publication`$\}$, where `publication` means the name of the conference or journal where a paper was published.

Our experiments involved 20 subjects among researchers in the Virginia Tech Digital Library Research Lab and graduate students from a Digital Library graduate course. The process is illustrated in Figure 3. Each subject was instructed to issue five searches for items of their own interest in the CITIDEL collection and provide relevance judgments (as relevant or non-relevant) for the items returned. Subjects were divided in two groups: G1 and G2. Subjects in group G1 were not aware of the possibility of structuring queries with field information. They issued unstructured queries, which were then automatically structured using the Bayesian network model. Subjects in group G2 were required to issue manually structured queries. For comparison, an unstructured version of the manually structured query was created by removing all field structure information from these queries, which were again re-structured using the Bayesian network model.

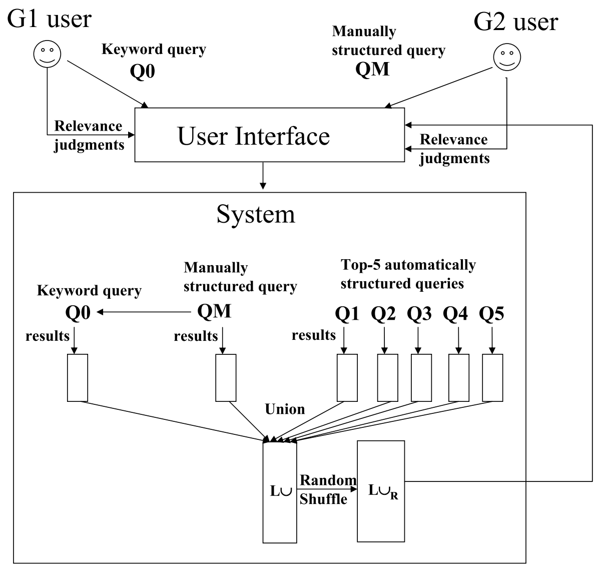

Figure 3: Experimental process and evaluation.

All queries, i.e., the unstructured query (Q0), the best of the top 5 structured queries (Q1–Q5), and the manually structured query (QM), were sent to the ESSEX search engine and the top 25 items returned by each were merged (with removal of duplicates). The resulting union set was completely shuffled and presented to the users for relevance judgments.

All relevant and non-relevant items returned for each query were used to compute *precision*, *recall*, and *F1* figures. Precision (P) is the percentage of retrieved items that are relevant. It is useful as an indication of how accurate the system was when retrieving the answers to the user's question. Recall (R) is the percentage of all the relevant items that were retrieved. It indicates if the system is able to retrieve all of the relevant items. High recall is especially useful when the user needs to be certain that all relevant information will be found. Finally, $F1$ combines precision and recall with equal weights and is defined as $F1 = 2PR/(P+R)$. The $F1$ measure combines precision and recall into a single value, providing a simple way of evaluating the system's overall performance.

To present the results, we also consider two forms of precision: 10-precision and R-precision. The 10-precision measure indicates the precision for the first 10 items retrieved by the system. This measure is important in practice since it is known that users tend to only look at the top results in a ranked answer set. The R-precision measure indicates the precision when all relevant documents were retrieved. It is a measure of how many spurious results the user has to look at before all relevant results are seen. Both measures are useful in determining not only if the system is able to show relevant results at the top of the list of retrieved items, but also if it can discover all relevant information while still keeping the noise level to a minimum.

5.2 Results

Before the experiments, all test subjects answered a short questionnaire regarding their background and knowledge in their area of interest in computer science. Among other questions, users were asked to cite five researchers and three publications that they would consider of importance in their selected research area. Surprisingly, most subjects did not know much about authors other than professors in the department, or about important publications (conferences and journals). Probably for this reason, queries were generally very short, averaging 2.59 terms per query, independently of being manually structured or not.

The average number of items indicated as relevant by subjects in group G1 was slightly higher than for group G2 (18.79 vs. 14.26 records), with a higher median (12 vs. 8) and standard deviation (19.46 vs. 14.33). This may be explained by the fact that when users are forced to use field structure information, queries tend to be more focused and so naturally tend to retrieve a smaller number of relevant items. This supports the assumption that structured queries are more precision-oriented.

Figure 4 shows the distribution of fields and combination of fields used by subjects in G2 in their manually structured queries. It is worth noticing that 63% of the queries contain only one field, there are no queries using only publication, only one using three fields, and none using all the four fields. This distribution may again be explained by the lack of user knowledge about publications and the difficulty of creating structured queries manually. We now examine the impact of query structuring, manually and automatically, on the quality of the retrieved results.

5.2.1 Unstructured vs. Structured Queries

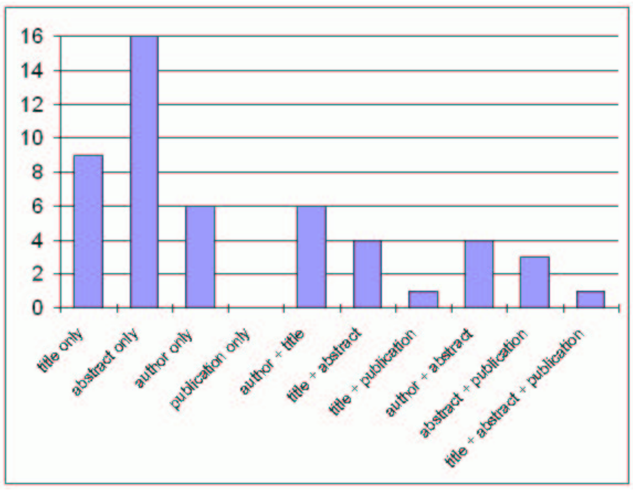

Figure 4: Distribution of fields and combination of fields in the manually structured queries.

Tables 1, 2, 3, and 4 show a comparison between the unstructured query (Q0), the top ranked automatically structured query (Q1), the best of the top 5 structured queries (Q1–Q5), and the manually structured query (QM).

Q1 vs. Q0	$F1$	10-precision	R-precision
G1	64.5%	83.3%	81.2%
G2	73.4%	89.3%	85.7%

Table 1: Percentage of times query Q1 is better or equal to query Q0 considering the $F1$, 10-precision, and R-precision measures, in groups G1 and G2.

Average	$F1$	10-precision	R-precision
Q0 (G1 + G2)	28.2	31.1	29.4
Q1 (G1 + G2)	36.4	51.4	49.7

Table 2: Average $F1$, 10-precision, and R-precision values for all the Q0 and Q1 queries, in groups G1 and G2 together.

For all 97 queries[6], the top ranked structured query Q1 had an overall better performance than the unstructured query Q0. Considering the $F1$ measure, results for Q1 were equal to or better than results for Q0 in an average of 69% of the searches, in both groups. In terms of 10-precision, Q1 was better or equal to Q0: 86.3%. In terms of R-precision, Q1 was better or equal to Q0 in 83.4% of the searches. We can conclude that, without the need of user intervention (except from entering the query keywords), the system is able to automatically find a structured query in the top of the ranked list that outperforms a simple keyword-based search in the majority of cases. In fact, as shown in Table 2, the average $F1$, 10-precision, and R-precision values for Q0 were 28.9, 31.9, and 29.4, while for Q1 the corresponding values were 35.9, 51.4, and 49.7.

[6]Raw data for three queries was lost.

best(Q1–Q5, QM) vs. Q0	$F1$	10-precision	R-precision
G1	81.2%	100%	100%
G2	85.1%	97.9%	97.9%

Table 3: Percentage of times the best (manual or automatically) structured query is better or equal to query Q0 considering the $F1$, 10-precision, and R-precision measures, in groups G1 and G2.

Average	$F1$	10-precision	R-precision
Q0 (G1+G2)	28.9	31.1	29.4
best(Q1–Q5,QM) (G1+G2)	62.2	84.5	84.7

Table 4: Average $F1$, 10-precision and R-precision values for the best (manual or automatically) structured query and query Q0, in groups G1 and G2 together.

When comparing the best of the Q1 through Q5 and QM queries with Q0, it is clear that using structured queries is better than a simple keyword-based search. The best structured query showed better or equal results than Q0 in 83.7% of the searches, considering the $F1$ measure and in 98.9% considering both 10-precision and R-precision. The average values were 62.3 for $F1$, 84.5 for 10-precision, and 84.7 for R-precision.

These results clearly indicate that structured queries, either manually constructed or automatically generated, perform better than their unstructured counterparts in the majority of cases. The situations in which the unstructured query Q0 performed better can be classified into four major cases:

1. Insufficient or outdated sampling data

   The most common reason for Q0 to outperform the structured queries was the absence of support in the sample database for very specific queries. This was especially evident in queries concerning new trends in computer science research (e.g. "peer-to-peer computing", "cognitive affordance", "multi-modal presentation", or "discourse processing"). One obvious solution would be to use the whole collection as the sample database, although this could have a negative impact on performance. Another possibility is to use better sampling strategies, which can guarantee high-quality coverage using the minimum possible data, and good policies for updates.

2. Very specific queries and strict separation between title and abstract

   It was noticed that, in almost all cases, subjects did not care in which field the relevant concepts in the query appeared. For instance, in very specific or short queries like "kerberos" users do not care if the word "kerberos" appears in the title or the abstract. Since the query structuring process must choose one field to insert the word "kerberos", say, the `title` field, the structured query becomes too specific, thus not retrieving all relevant items, despite having good precision. This suggests that some kind of combination (for instance, a Boolean OR) of fields with a large overlap in vocabulary, such as titles and abstracts, in the automatically structured query, may be beneficial.

3. The "+" constraint is too restrictive

   Another assumption in this work was that structured queries are, by nature, more focused and therefore more precision-oriented. This led to the design choice of enforcing the words in the structured queries to appear in the respective fields by using the "+" operator. In a few cases, this assumption proved too restrictive, mainly in long queries (e.g., "parallel all pairs shortest path") or in queries with two or more embedded concepts that never occur together in the collection (e.g., "peer-to-peer comparison systems"). In these cases the "+" constraint became too restrictive and returned none or few results. Subjects preferred to mark at least a few records as relevant, even if all relevant concepts in the query did not appear together in the document, rather than mark no result at all. One obvious solution is to relax the "+" constraint, but initial tests showed that performance would be degraded. A better choice could be to identify these extreme cases and only then relax the constraint or apply techniques of query splitting [21].

4. Failure of the model

   In very few cases the network model was unable to correctly rank the structured queries, even when there was support from the sample database. The main reasons for this problem are ties in the ranking and skewed keyword distributions. Ties occur when several of the structured queries get the same score and, thus, their ranking order becomes arbitrary. Keyword distributions in the collection are skewed because some fields tend to contain more words than others. For instance, the abstract field is larger than most others and, thus, contains the majority of words in the collection. For this reason, the Bayesian network model tends to assign higher probabilities to queries that contain abstracts. One possible solution to both problems is to assign different weights to each field, according to their relative importance in the collection. Preliminary experiments have shown that this strategy may be beneficial, but the proper choice of weights for all fields is hard to obtain and will require further experimentation.

5.2.2 *Manually Structured vs. Automatically Structured Queries*

We next compare the performance of the best automatically structured query to the manually structured query. As shown in Table 5, the best automatically generated query tied or outperformed query QM in 91.8% of the searches, considering the $F1$ measure. Considering 10-precision and R-precision, the best structured query equaled or outperformed QM in 97.9% of the searches. The average $F1$, 10-precision, and R-precision values for query QM are 55.6, 72.5, and 70.2, respectively, while for the best structured query the values are 59.8, 83.4, and 84.7, respectively, as shown in Table 6.

best (Q1–Q5) vs. QM	$F1$	10-precision	R-precision
G2	91.8%	97.9%	97.9%

Table 5: Percentage of times the best automatically structured query is better or equal to query QM considering the $F1$, 10-precision, and R-precision measures, in group G2.

Average	$F1$	10-precision	R-precision
Manual	55.6	72.5	70.2
best(Q1-Q5)	59.8	83.4	84.7

Table 6: Average $F1$, 10-precision and R-precision values for the best automatically structured query and query QM, in group G2.

We note that, in most cases, one of the top five automatically structured queries precisely matched the query manually structured by the user. Also, even when not completely correct, automatically structured queries generally outperformed manual structured queries.

These results can be explained. When judging the returned items' relevance, test subjects tended to focus mostly on the title field. Thus, in cases where the automatically structured query contains the relevant concepts in the title field, users almost always considered the returned items as relevant. On the other hand, items that contained relevant concepts in other fields, such as in the abstract or publication, were very often ignored. For this reason, the automatic structuring model was able to outperform the results of the manually structured queries, in which the abstract field was often the user's choice.

The few cases where QM performed better than one of the top five structured queries were due to one of the four cases described in the previous section, in particular Case 1, in which there was insufficient or outdated information in the sampling data. These results show that automatic structuring of queries is a viable alternative to substitute manual structuring, and one that significantly reduces the burden on the users while still yielding good performance.

6. RELATED WORK

Very few works have explored user interfaces that facilitate the search process in digital libraries. However, there are notable exceptions: the DLITE project [10], SenseMaker [5], and the query synthesizers described in [4]. In most cases, DL search services are limited to simple keyword-based query formulation, a rather common resource in all types of information retrieval systems [3]. More recently, keyword-based queries also have been introduced to structured databases [2, 13, 17]. Furthermore, there is a long history of work in the information retrieval community on (semi)automatic generation of queries [6,18,20,24,29] but it generally did not focus on structuring opportunities.

In this work, keyword-based queries formulated by the user are given structure by the use of a Bayesian network model. This is somehow similar to the work of Croft et al. [11], where Boolean queries are derived from a user given a natural language query, and then improved with automatically inferred phrases. Bayesian network models were first used in IR problems by Turtle and Croft [27] and later by Ribeiro-Neto and Muntz [23] (upon whose work our model is based). More recently, Acid et al. [1] further refined such models so that exact propagation algorithms can be used to efficiently compute probabilities. Bayesian networks also have been applied to other IR problems besides ranking as, for example, relevance feedback [19], automatic construction of hypertexts [26], query expansion [14], information filtering [9], ranking fusion [28], and document clustering and classification [7, 15]. Nevertheless, there have been few studies similar to this study.

7. CONCLUSIONS

In this paper, through a number of user experiments, we have shown that: (1) structured queries perform better than pure keyword-based queries in DL searching services based on fielded metadata; and (2) a system can be used to automatically add structure to the users' queries, thus providing a viable alternative to manual structuring that significantly reduces the burden on the users while still yielding good performance. The experiments performed confirm that, in the majority of cases, better results are achieved by structured queries than by unstructured queries. Also, using the Bayesian network model proposed in [8] and an appropriate term weighting scheme, automatically structured queries outperform not only the unstructured queries but also the query manually structured by the user.

We may conclude that a system such as the one used in this work can be effectively used to improve DL search services. We envision a search system that is able to suggest a few alternative structured queries to the user. Acccording to a semi-automatic scenario, these structured queries can be presented together with the results of the initial unstructured query. By clicking on one of these candidates, the user could get corresponding structured search results – a refinement on the initial list of items retrieved. Alternatively, according to a fully automatic scenario, the system can simply submit the highest ranked structured query, and provide corresponding results without user intervention.

Further improvements on the models used in this work are possible. For instance, if the list of candidate structured queries is too long, too much time would be spent by the user in selecting the most appropriate candidate. Thus, it would be important to guarantee that the top one or two candidates successfully contain the best structured query in the majority of cases. We believe that such a level of performance is ultimately attainable with minor adjustments to our model and implementation.

Besides testing the system in production mode in CITIDEL and other DLs hosted in the Digital Library Research Laboratory, future work will continue in a number of ways. First, we want to investigate different and more effective sampling strategies that minimize the discovered problems of incompleteness and outdated information, including good policies for updates. Second, we will investigate automatic field weighting based on the relative or perceived importance of the fields, in order to increase the accuracy of our model. Third, we intend to investigate new models that can combine fields with large vocabulary overlap (e.g., titles and abstracts) in the query, and to study possible ways to relax the "+" constraint without reducing effectiveness. Further, we plan to investigate the effect of the "+" constraint by itself in keyword-based queries. Fourth, we plan to incorporate relevance feedback and personalized ranking strategies to our belief network models [16]. Finally, we intend to work on new approaches to improve system performance that go beyond our simple caching strategy.

Acknowledgments

This research work was funded in part by the NSF—grants DUE0136690, DUE0121679 and IIS0086227, by the I3DL project—grant 680154/01-9, by the GERINDO project—grant MCT/CNPq/CT-INFO 552.087/02-5, by individual grants MCT/FCT SFRH/BD/4662/2001 (Pável Calado) and CNPq 304890/02-5 (Alberto H. F. Laender), by the SiteFix project, grant MCT-CNPQ-CT-INFO 55.2197/02-5, by a PHILIPS MDS Manaus R&D sponsorship (Altigran S. Da Silva), and by a fellowship from AOL (Marcos A. Gonçalves).

REFERENCES

[1] S. Acid, L. M. de Campos, J. M. Fernández-Luna, and J. F. Huete. An information retrieval model based on simple Bayesian networks. *International Journal of Intelligent Systems*, 18(2):251–265, January 2003.

[2] S. Agrawal, S. Chaudhuri, and G. Das. DBXplorer: A system for keyword-based search over relational databases. In *Proceedings of the 18th International Conference on Data Engineering*, pages 5–16, San Jose, CA, USA, February 2002.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, New York, NY, USA, 1999.

[4] M. Baldonado, S. Katz, A. Paepcke, C.-C. K. Chang, H. Garcia-Molina, and T. Winograd. An extensible constructor tool for the rapid, interactive design of query synthesizers. In *DL'98: Proceedings of the 3rd ACM International Conference on Digital Libraries*, pages 19–28, Pittsburgh, PA, USA, June 1998.

[5] M. Baldonado and T. Winograd. Sensemaker: An information-exploration interface supporting the contextual evolution of a user's interests. In *Proceedings of ACM CHI 97 Conference on Human Factors in Computing Systems*, pages 11–18, Altlata, GA, USA, March 1997.

[6] D. Cai, C. J. V. Rijsbergen, and J. M. Jose. Automatic query expansion based on divergence. In *Proceedings of the 10th International Conference on Information and Knowledge Management CIKM'01*, pages 419–426, New York, Novemeber 2001.

[7] P. Calado, M. Cristo, E. Moura, B. R.-N. Nivio Ziviani, and M. A. Gonçalves. Combining link-based and content-based methods for web document classification. In *Proceedings of the 12th International Conference on Information and Knowledge Management*, pages 394–401, New Orleans, LA, USA, 2003.

[8] P. Calado, A. S. da Silva, R. C. Vieira, A. H. F. Laender, and B. A. Ribeiro-Neto. Searching web databases by structuring keyword-based queries. In *Proceedings of the 11th International Conference on Information and Knowledge Management*, pages 26–33, McLean, VA, USA, 2002. ACM Press.

[9] J. P. Callan. Document filtering with inference

networks. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 262–269, Zurich, Switzerland, August 1996.

[10] S. B. Cousins, A. Paepcke, T. Winograd, E. A. Bier, and K. Pier. The digital library integrated task environment (DLITE). In *DL'97: Proceedings of the 2nd ACM International Conference on Digital Libraries*, pages 142–151, Philadelphia, PA, USA, July 1997.

[11] W. B. Croft, H. R. Turtle, and D. D. Lewis. The use of phrases and structured queries in information retrieval. In *Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 32–45, Chicago, IL, USA, October 1991.

[12] A. S. da Silva, P. Calado, R. C. Vieira, A. H. F. Laender, and B. A. Ribeiro-Neto. *Effective Databases for Text & Document Management*, chapter Keyword-based Queries over Web Databases, pages 74–92. Idea Group Publishing, 2003.

[13] S. Dar, G. Entin, S. Geva, and E. Palmon. DTL's DataSpot: Database exploration using plain language. In *Proceedings of 24th International Conference on Very Large Data Bases VLBD'98*, pages 645–649, New York, NY, USA, August 1998.

[14] L. M. de Campos, J. M. Fernández-Luna, and J. F. Huete. Query Expansion in Information Retrieval Systems Using a Bayesian Network-Based Thesaurus. In *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence (UAI–98)*, pages 53–60, San Francisco, CA, July 1998.

[15] S. T. Dumais, J. Platt, D. Hecherman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the 7th International Conference on Information and Knowledge Management CIKM'98*, pages 148–155, Bethesda, Maryland, USA, November 1998.

[16] W. Fan, M. D. Gordon, and P. Pathak. Discovery of context-specific ranking functions for effective information retrieval using genetic programming. *IEEE Transactions on Knowledge and Data Engineering*, 2003. In press.

[17] D. Florescu, D. Kossmann, and I. Manolescu. Integrating keyword search into XML query processing. *WWW9 / Computer Networks*, 33(1–6):119–135, 2000.

[18] E. A. Fox. *Relational Models of the Lexicon: Representing Knowledge in Semantic Networks*, chapter Improved Retrieval Using a Relational Thesaurus for Automatic Expansion of Boolean Logic Queries, pages 199–210. Cambridge University Press, 1988.

[19] D. Haines and W. B. Croft. Relevance feedback and inference networks. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2–11, Pittsburgh, PA, USA, June 1993.

[20] M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 206–214, Melbourne, Australia, August 1998.

[21] F. D. Neves and E. A. Fox. Extending retrieval with stepping stones and pathways - NSF proposal (funded), 2003.

[22] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, 2nd edition, 1988.

[23] B. Ribeiro-Neto and R. Muntz. A belief network model for IR. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 253–260, Zurich, Switzerland, August 1996.

[24] G. Salton, C. Buckley, and E. A. Fox. Automatic query formulations in information retrieval. *Journal of the American Society for Information Science*, 34(4):262–280, July 1983.

[25] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[26] D. Shin, S. Nam, and M. Kim. Hypertext construction using statistical and semantic similarity. In *DL'97: Proceedings of the 2nd ACM International Conference on Digital Libraries*, pages 57–63, Philadelphia, PA, USA, July 1997.

[27] H. R. Turtle and W. B. Croft. Inference networks for document retrieval. In *Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1–24, Brussels, Belgium, September 1990.

[28] R. F. Valle, B. A. Ribeiro-Neto, L. R. S. de Lima, A. H. F. Laender, and H. R. F. F. Junior. Improving text retrieval in medical collections through automatic categorization. In *Proceedings of the 10th International Symposium on String Processing and Information Retrieval SPIRE 2003*, pages 197–210, Manaus, Brazil, October 2003.

[29] E. M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 61–69, Dublin, Ireland, July 1994.