Maximal Termsets as a Query Structuring Mechanism

Bruno Pôssas
Federal University of Minas Gerais
30161-970 Belo Horizonte-MG, Brazil
bavep@dcc.ufmg.br

Nivio Ziviani
Federal University of Minas Gerais
30161-970 Belo Horizonte-MG, Brazil
nivio@dcc.ufmg.br

Berthier Ribeiro-Neto
Federal University of Minas Gerais
30161-970 Belo Horizonte-MG, Brazil
berthier@dcc.ufmg.br

Wagner Meira Jr.
Federal University of Minas Gerais
30161-970 Belo Horizonte-MG, Brazil
meira@dcc.ufmg.br

ABSTRACT

Search engines process queries conjunctively to restrict the size of the answer set. Further, it is not rare to observe a mismatch between the vocabulary used in the text of Web pages and the terms used to compose the Web queries. The combination of these two features might lead to irrelevant query results, particularly in the case of more specific queries composed of three or more terms. To deal with this problem we propose a new technique for automatically structuring Web queries as a set of smaller subqueries. To select representative subqueries we use information on their distributions in the document collection. This can be adequately modeled using the concept of maximal termsets derived from the formalism of association rules theory. Experimentation shows that our technique leads to improved results. For the TREC-8 test collection, for instance, our technique led to gains in average precision of roughly 28% with regard to a BM25 ranking formula.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval—*Retrieval models*; H.3.3 [**Information Systems**]: Information Search and Retrieval—*Query formulation*; H.3.4 [**Information Systems**]: Systems and Software—*Performance evaluation (efficiency and effectiveness)*; H.2.8 [**Information Systems**]: Database Applications—*Data mining*

General Terms

Theory, Algorithms, Experimentation

Keywords

Information retrieval models, association rule mining, weighting index term co-occurrences, automatic query structuring

Technical Report TR012/2005
Department of Computer Science, Federal University of Minas Gerais, Belo Horizonte-MG, Brazil
.

1. INTRODUCTION

The huge volume of information now available on the Web has posed challenges to the users. Any short query presents the user with thousands of answers. If the first 10-20 answers are not satisfactory, the user has to sift the answers of his interest among dozens, even hundreds, of answers. Frequently, he gets impatient and takes one of two actions: he rewrites his query in a different form or he simply gives up.

If our user is more persistent, he will rewrite his query trying to make it more specific. It is our believe that, as users of Web search engines become more knowledgeable, they will write more specific, longer, and complex queries (we consider here that a *long* or *complex* query is one that contains two or more terms). Let us proceed with one example.

Consider the query *"tylenol drogaria belo horizonte"* formulated by a Brazilian student who has headache and seeks for a drugstore in the city of Belo Horizonte, where he lives. He wrote his query thinking of tylenol, the brand name of a popular pain killer. It is a simple request. Let us look at the results.

At *UOL Busca*, search engine of UOL[1], largest paid ISP in the Brazil, only one answer is returned: Tcafarma, a company that distributes medicines to hospitals and pharmacies. It does not solve the problem of our student. At Yahoo[2], three answers are returned: Tcafarma as before, Mercado Mineiro, an institute that runs market research, and Drogaria Pacheco, a small drugstore located far away from the student's home. Again, our student would have obtained frustrating answers. At Google[3], twenty-two answers are returned, which would seem better, but that none of them are relevant.

Curiously, had our user formulated his query as *"drogaria belo horizonte"*, instead, he would have gotten pointers to large chains of drugstores in Belo Horizonte, with stores close to his home, right in the first top ten answers in all three search engines. He did not obtain this information with his original query because all search engines process the user queries as a *conjunction* of the query terms.

The problem that we face can then be formulated as follows:

> *Given a conjunctive user query, is it reasonable to structure the query in smaller conjunctive components? When should this be attempted? Is it possible to improve precision through one such mechanism?*

[1] `http://www.uol.com.br`
[2] `http://www.yahoo.com.br`
[3] `http://www.google.com.br`

Our proposal to solve these questions is a new technique for automatically structuring queries based on *maximal termsets*, a concept directly derived from association rules theory [1, 5], which we will introduce later on. Our technique is referred to as *MAXTERM*, and the key idea is that information derived from the distributions of the query conjunctive components in the document collection can be used to guide the query structuring process. Given a user query, the effect is that, we can provide the user with the "best" set of answers that is possible to produce by directly matching the query against the documents in the collection. "Best" in the sense that the largest query components are used, not necessarily that they lead to higher precision figures. It is intuitive though that, if the user query makes sense, the query components best supported by the document collection are more likely to produce more relevant answers – a conjecture confirmed by our experimentation. Once this best set of answers has been produced, Web ranking techniques such as Page ranking [2] can be applied.

We instantiate the problem with a small fictitious example, as depicted in Figure 1. Our example collection C is composed of just 4 documents, none of them containing the 4 terms $\{a, b, x, y\}$ of the vocabulary. Thus, the user query $q = \{a, b, x, y\}$ returns the empty set. However, it is clear in this case that there are two answers that are far better than returning an empty set. These are documents d_3 and d_4. Notice that they could have been returned had we processed the user query as $q\prime = \{a, b, x\} \vee \{a, b, y\}$, which are two maximal termsets for the original user query in the context of our example document collection.

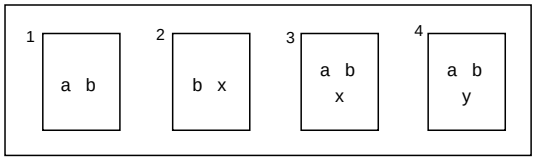

Figure 1: Example of a small collection of 4 documents.

Our approach allows to naturally produce answers to queries that otherwise would lead to empty result sets. This is accomplished by finding all maximal termsets derived from the user query that have support in the document collection. That is, maximal termsets provide a simple and elegant formalism for naturally structuring conjunctive user queries formed by an arbitrary number of terms.

The remaining of the paper is organized as follows. In the next section we discuss related work. In Section 3 we introduce the building blocks of our approach. Experimental results and the reference collections characteristics are presented in Section 4. Finally, in Section 5 we draw our conclusions and discuss briefly suggestions for future research.

2. RELATED WORK

Structured queries containing operators such as AND, OR, and proximity can be used to describe accurate representations of information needs. Although boolean query languages may be difficult for people to use, there is considerable evidence that trained users may achieve good retrieval effectiveness using them. Experimental results with the extended boolean model [15] and the network model [22, 23] showed that structured queries were more effective than simpler queries consisting of a set of weighted terms.

There are studies on the best translation of linguistic relationships from queries into boolean operators. Using both syntactic and semantic information, Das-Gupta [4] proposed an algorithm for deciding when the natural language conjunction "and" should be interpreted as a boolean AND or a boolean OR. Smith [19] presented a complex algorithm for translating a full syntactic parse of a natural language query into a boolean form.

The aforementioned studies all focused on the connections between linguistic relationships and boolean operators. Natural language processing models view a query as an expression of different concepts and their relationships that are of interest to the user. Correctly identifying these concepts in queries and in documents results in improved retrieval performance [21].

Phrases provide another means for structured queries to capture linguistic relationships, specially for natural language processing approaches. Both statistical (based on word co-occurrences) and syntactic (based on natural language processing techniques) methods have been successfully explored to identify indexing phrases. Mitra et al. [7] compared statistical and syntactic indexing phrases and observed a trade off between query accuracy and query coverage. Croft et al. [3] used phrases identified in natural language queries to build structured queries for a probabilistic model.

Instead of processing documents through a natural language processing system to identify phrases for indexing, there have been efforts to use linguistic processing to get a better understanding of the user information needs. Experiments by Smeaton and van Rijsbergen [18] implement a retrieval strategy that is based on syntactic analysis of queries. Narita and Ogawa [8] examined the utility of phrases as search terms in IR. They used single term selection and phrasal term selection in their query construction. Similar to Mitra et al. [7], they experimented with different representations for multi-word phrases (more than 2 words) and decided to use two word phrases for query construction.

The set-based model [10, 11] is an information retrieval model that exploits term correlations effectively, provides significant gains in precision, and has processing costs close to the costs of the vector space model, independently of the collection and query type considered. This model exploits the intuition that semantically related terms often occur close to each other in a document.

Our work differs from the aforementioned works in the following aspects. First, as in the set-based model, determination of index term relationships is derived directly from association rules theory, which naturally considers representative patterns of term co-occurrence. Our approach may be used for general document collections, and does not require a syntactic knowledge base, a linguistic processing of the user queries, or a limit in the number of terms in term correlations. Second, our approach does not correspond to a conjunctive normal form (CNF) query structuring mechanism. It treats the various conjunctive components differently, depending on their support in the document collection - a critical step not addressed by CNF. Third, experimental results show significant and consistent improvements in average precision curves in comparison to the vector space model, to the probabilistic model using the BM25 weighting scheme [14], and to the set-based model. Fourth, the implementation of our solution is practical and efficient, with processing times comparable to the times to process the vector space model. Fifth, our approach here is distinct from the set-based model that uses closed termsets instead of maximal termsets. Further, the ranking formula is distinct.

3. THE MAXTERM MECHANISM

In this section we present our approach for automatically structuring queries. It is composed by three components: decomposition, selection, and ranking criteria. This new approach is referred to as *MAXTERM*. The decomposition criterion defines how a query is divided into subqueries. The selection criterion defines which subqueries are used to compose the structured query. The ranking criterion defines how the documents that satisfy the subqueries are ranked.

Section 3.1 describes the decomposition criterion. Section 3.2 discusses the selection criterion. Finally, Section 3.3 presents the ranking criterion.

3.1 Decomposing Queries into Structured Queries

In this section we introduce the idea of structured queries.

Definition 1. Let $T = \{k_1, k_2, \ldots, k_t\}$ be the vocabulary of a collection C of documents (i.e., the set of t unique terms in C). There is a total ordering among the vocabulary terms, which is based on the lexicographical order of terms, so that $k_i < k_{i+1}$, for $1 \leq i \leq t-1$.

Definition 2. An n-termset S, $S \subseteq T$, is a set of n terms. When the number of terms is not important, we refer simply to the termset S.

Definition 3. A conjunctive query of n terms corresponds to an n-termset.

Definition 4. A conjunctive query q is satisfiable in a document collection C if at least a predefined number of documents from C contain all terms from q.

Definition 5. A structured query is a disjunction of conjunctive queries.

Definition 6. A structured query q_s is satisfiable in a document collection C if and only if all subqueries of q_s are satisfiable in C.

Given a query containing n terms, there are $2^n - 2$ subsets that can be derived from it (the power set minus the empty set and the n-termset). In practice, only the subsets that do occur in the text collection are computed. Further, the number of query terms is on average smaller than 10. The combination of those factors allow efficient computation, as we later discuss.

3.2 Selecting Structured Queries

In this section we discuss how to select the structured queries that are likely to produce more relevant answers. For that we need the concepts of termsets, termset rules, closed termsets, and maximal termsets.

3.2.1 Termsets

In this section we detail the concept of termsets as a basis for determining structured queries. The starting point of our discussion is the Set Based Model [10, 11], which targets correlations among query terms but does not handle queries that are not satisfiable.

Definition 7. Let $V = \{S_1, S_2, \ldots, S_{2^t}\}$ be the vocabulary-set of a collection C of documents, that is, the set of 2^t unique termsets that may appear in any document from C. The frequency dS_i of a termset S_i is the number of occurrences of S_i in C, that is, the number of documents where $S_i \subseteq d_j$ and $d_j \in C$, $1 \leq j \leq N$.

Definition 8. A termset S_i is a frequent termset if its frequency dS_i is greater than or equal to a given threshold, referred to as minimal frequency. An n-termset is frequent if and only if all of its $(n-1)$-termsets (i.e., subsets with one less term) are also frequent.

In terms of the selection criterion for structuring queries, the frequency requirement comes from the intuition that a termset should have a minimal relevance for being considered as a candidate to compose a structured query.

Example 1. Consider a vocabulary of four terms $T = \{a, b, x, y\}$, extracted from our sample collection C, as depicted in Figure 1. Consider also the user query $q = \{a, b, x, y\}$. There are 15 termsets associated with q and only 11 occur in our sample collection, as depicted in Table 1. The 1-termsets are S_a, S_b, S_x, and S_y, the 2-termsets are S_{ab}, S_{ax}, S_{ay}, S_{bx}, and S_{by}, and the 3-termsets are S_{abx} and S_{aby}.

Notice that termsets provide a natural formalism for considering index term correlations for ranking purposes. This allows computing answer sets that lead to higher precision figures.

Table 1: Vocabulary-set for the query $q = \{a, b, x, y\}$.

Termsets	Elements	Documents
S_a	$\{a\}$	$\{d_1, d_2, d_4\}$
S_b	$\{b\}$	$\{d_1, d_2, d_3, d_4\}$
S_x	$\{x\}$	$\{d_2, d_3\}$
S_y	$\{y\}$	$\{d_4\}$
S_{ab}	$\{a, b\}$	$\{d_1, d_3, d_4\}$
S_{ax}	$\{a, x\}$	$\{d_3\}$
S_{ay}	$\{a, y\}$	$\{d_1\}$
S_{bx}	$\{b, x\}$	$\{d_2, d_3\}$
S_{by}	$\{b, y\}$	$\{d_4\}$
S_{abx}	$\{a, b, x\}$	$\{d_3\}$
S_{aby}	$\{a, b, y\}$	$\{d_4\}$

3.2.2 Closed Termsets

In this section we revisit the idea of closed termsets, an extension to the concept of frequent termsets [10, 11].

Definition 9. The closure of a termset S_i is the set of all its j-termsets, $j < i$, that are frequent and co-occur with S_i in the same set of documents.

Definition 10. The closed termset CS_i is the largest termset in the closure of a termset S_i. More formally, given a set $\mathcal{D} \subseteq C$ of documents, the termset S_i, and the set $\mathcal{S}$ of its j-termsets, $j < i$, that occur in all documents from $\mathcal{D}$ and only in these, a closed termset CS_i satisfies the property that $\nexists S_j \in \mathcal{S} | (CS_i \subset S_j \wedge$ "S_j occurs in the same set of documents as CS_i".

Example 2. Consider the dataset of Example 1. Table 2 shows all frequent and closed termsets and their respective frequencies. If we define that a frequent termset must have a minimum frequency of 50%, the number of termsets is reduced from 11 to 5. Notice that the number of frequent termsets, although potentially very large, is usually small in natural language texts. Regarding the closed termsets, even in this small example, we see that the number of closed termsets is smaller than the number of frequent termsets.

Table 2: Frequent and closed termsets for the sample document collection of Example 1.

Frequency (ds)	Frequent Termsets	Closed Termsets
100% (4)	S_b	S_b
75% (3)	S_a, S_{ab}	S_{ab}
50% (2)	S_x, S_{bx}	S_{bx}
25% (1)	$S_y, S_{ay}, S_{by}, S_{aby}$	S_{aby}
25% (1)	S_{ax}, S_{abx}	S_{abx}

3.2.3 Maximal Termsets

In this section we introduce the maximal termsets and discuss its utilization for building structured queries.

Definition 11. A maximal termset MS_i is a frequent termset that is not a subset of any other frequent termset. That is, given the set $\mathcal{S}_\mathcal{D} \subseteq S$ of frequent termsets that occur in all documents from a document subcollection $\mathcal{D}$, a maximal termset MS_i satisfies the property that $\nexists S_j \in \mathcal{S}_\mathcal{D} | MS_i \subset S_j$.

Maximal termsets provide a natural formalism for structuring complex user queries into smaller components that find support in the document collection. This leads to improved results as we verify through experimentation.

Let FT be the set of all frequent termsets, CFT be the set of all closed termsets, and MFT be the set of all maximal termsets. It is straightforward to see that the following relationship holds: $MFT \subseteq CFT \subseteq FT$. The set MFT is much smaller than the set CFT, which itself is much smaller than the set FT.

Example 3. We use the same dataset of Example 1, where $q = \{a, b, x, y\}$ and C is the collection of documents. Table 3 shows all frequent and maximal termsets and their respective frequencies. Regarding the maximal termsets, we can see that the number of maximal termsets is significantly smaller than the number of frequent termsets. We have 11 frequent termsets and 2 maximal termsets.

Table 3: Frequent and maximal termsets for the sample document collection of Example 1.

Frequency (ds)	Frequent Termsets	Maximal Termsets
100% (4)	S_b	
75% (3)	S_a, S_{ab}	
50% (2)	S_x, S_{bx}	
25% (1)	$S_y, S_{ay}, S_{by}, S_{aby}$	S_{aby}
25% (1)	S_{ax}, S_{abx}	S_{abx}

Determining maximal termsets is an extension of the data mining problem of mining frequent itemsets [1]. Our maximal termsets enumeration algorithm is based on an efficient algorithm called GENMAX [5]. We adapted GENMAX to handle terms and documents instead of items and transactions, respectively.

3.3 Ranking Structured Queries

In the following subsections, we review the cosine ranking formula for the standard vector space model [16] and the BM25 ranking formula for the probabilistic model [14]. These models were used during experimentation to be compared with the proposed approach.

We use as a basis for ranking structured queries the *set-based vector model* proposed in [10] and further discussed in [9, 11, 12]. We also briefly review that model here.

3.3.1 The Vector Space Model Ranking Formula

In the vector space model the algebraic representation of the set of index terms for both documents and queries correspond to vectors in a t-dimensional Euclidean space, where t is equal to the number of different index terms in the document collection [16].

The term weights can be calculated in many different ways [17, 25, 20]. The best known term weighting schemes for the vector space model use weights that are given by (i) $tf_{i,j}$, the number of times that an index term i occurs in a document d_j and (ii) df_i, the number of documents that an index term i occurs in the whole document collection. Thus, the weight of an index term i in a document d_j is given by:

$$w_{i,j} = f\left(tf_{i,j}\right) \times idf_i = \left(1 + \log tf_{i,j}\right) \times \log\left(1 + \frac{N}{df_i}\right) \quad (1)$$

where N corresponds to the number of documents in the collection and idf_i corresponds to the inverse document frequency for term i. Such term-weighting strategy is called $tf \times idf$ (term frequency times inverse document frequency) scheme.

Similiarly, the weight of a term i in a query q is formally defined as:

$$w_{i,q} = f\left(tf_{i,q}\right) \times idf_i = \left(1 + \log tf_{i,q}\right) \times \log\left(1 + \frac{N}{df_i}\right) \quad (2)$$

where N is the number of documents in the collection, $tf_{i,q}$ is the number of occurrences of the term i in the query q and idf_i is the inverse frequency of occurrence of the term i in the collection, scaled down by a log function.

One of the most successful ranking formula for the vector space model is the cosine measure. It assigns a similarity measure to every document containing any of the query terms, defined as the scalar product between the set of document vectors $\vec{d_j}$, $1 \leq j \leq N$, and the query vector $\vec{q}$. This measure is equivalent to the angle between the query vector and any document vector. Thus, the similarity between a document d_j and a query q is given by:

$$sim(q, d_j) = \frac{\vec{d_j} \bullet \vec{q}}{|\vec{d_j}| \times |\vec{q}|} = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}}, \quad (3)$$

where $w_{i,q}$ corresponds to the weight of term i in query q, whose definition is equivalent to the weight of a term in a document, i.e., $w_{i,q} = tf_{i,q} \times idf_i$. The factors $|\vec{d_j}|$ and $|\vec{q}|$ correspond to the norm of document and query vectors, respectively. The ranking calculation is not affected by $|\vec{q}|$ because its value is the same for all documents. The factor $|\vec{d_j}|$ represents the length of document d_j.

3.3.2 The BM25 Ranking Formula

The probabilistic model [13] estimates the probability that the user will find a document d_j relevant for a user query q. The model assumes that this probability of relevance depends on the query and the document representations only. Further, it assumes that there is a subset of all documents that the user prefers as the answer set for the query q. Such an *ideal* answer set is labeled R and should maximize the overall probability of relevance to user. Documents in the set R are predicted to be relevant to the query, while documents not in this set are predicted to be non-relevant.

The BM25 measure [14] is one of the most successful ranking formula for the probabilistic model. The BM25 weighting scheme is a function of the number of occurrences of the term in a document, in the whole collection, and a function of the document length. Formally, the weight of a term i in a document d_j is defined as:

$$w_{i,j} = \frac{k_1 \times tf_{i,j}}{tf_{i,j} + k_1 \times \left(1 - b + b \times \frac{|\vec{d_j}|}{\overline{|\vec{d_j}|}}\right)} \times \log \frac{N - df_i + 0.5}{0.5}, \quad (4)$$

where $tf_{i,j}$ is the number of occurrences of the term i in the document d_j, k_1 and b are parameters, that depends on the collection and possibly on the nature of the user queries, $|\vec{d_j}|$ corresponds to a document length function, $\overline{|\vec{d_j}|}$ is the average document length, N is the number of documents in the collection, and df_i is the number of documents containing the term i.

The BM25 scheme also defines a weight of a term i in a query q. This weight is formally defined as:

$$w_{i,q} = \frac{(k_3 + 1) \times tf_{i,q}}{k_3 + tf_{i,q}}, \quad (5)$$

where $tf_{i,q}$ is the number of occurrences of the term i in the query q, and k_3 is a parameter, that depends on the collection and possibly on the nature of the user queries.

The probabilistic model computes the similarity between a document and the user query as the scalar product between the document vector $\vec{d_j}$, $1 \leq j \leq N$, and the query vector $\vec{q}$, as follows:

$$sim\,(q, d_j) = \vec{d_j} \bullet \vec{q} = \sum_{i \in q} w^{(1)} \times w_{i,j} \times w_{i,q} + k_2 \times \frac{\overline{|\vec{d_j}|} - |\vec{d_j}|}{\overline{|\vec{d_j}|} + |\vec{d_j}|}, \quad (6)$$

where $w_{i,j}$ is the weight associated with the term i in the document d_j, $w_{i,q}$ is the weight associated with the term i in the query q, $|\vec{d_j}|$ corresponds to a document length function, $\overline{|\vec{d_j}|}$ is the average document length, k_2 is another parameter, that also depends on the collection and possibly on the nature of the user queries, and $w^{(1)}$ is the Robertson-Sparck Jones weight [13], which is defined as:

$$w^{(1)} = \log \frac{(r + 0.5) / (R - r + 0.5)}{(df_i - r + 0.5) / (N - df_i - R + r + 0.5)}, \quad (7)$$

where N is the number of documents in the collection, df_i is the number of documents containing the term i, R is the number of relevant documents for the query q, and r is the number of relevant documents containing the term i.

3.3.3 *Modified Set-Based Model Ranking Formula*

In here we use a modified version of the ranking formula for the set-based model. We adopt the weighting schema of the BM25 ranking formula for computing termset weights.

The basic premise of the set-based model is that a document d_j and an user query q are represented by vectors of weighted *termsets* as follows:

$$\begin{aligned} \vec{d_j} &= (w_{S_1,j}, w_{S_2,j}, \ldots, w_{S_{2^t},j}) \\ \vec{q} &= (w_{S_1,q}, w_{S_2,q}, \ldots, w_{S_{2^t},q}) \end{aligned}$$

where t corresponds to the number of distinct terms in the collection, $w_{S_i,j}$ is the weight of termset S_i in document d_j, and $w_{S_i,q}$ is the weight of termset S_i in the query q.

In the set-based model, weights are associated with the termsets in a document or query, instead of terms. These weights are a function of the number of occurrences of the termset in a document and in the whole collection. In here, we have adapted the original set-based model to use the BM25 weighting scheme. Formally, the weight of a termset S_i in a document d_j is defined as:

$$w_{S_i,j} = \frac{k_1 \times Sf_{i,j}}{Sf_{i,j} + k_1 \times \left(1 - b + b \times \frac{|\vec{d_j}|}{\overline{|\vec{d_j}|}}\right)} \times \log \frac{N - dS_i + 0.5}{0.5}, \quad (8)$$

where $Sf_{i,j}$ is the number of occurrences of the termset S_i in the document d_j, k_1 and b are parameters, that depends on the collection and possibly on the nature of the user queries, $|\vec{d_j}|$ corresponds to a document length function, $\overline{|\vec{d_j}|}$ is the average document length, N is the number of documents in the collection, and dS_i is the number of documents containing the termset S_i.

The weight of a termset S_i in a query q is formally defined as:

$$w_{S_i,q} = \frac{(k_3 + 1) \times Sf_{i,q}}{k_3 + Sf_{i,q}} \quad (9)$$

where $Sf_{i,q}$ is the number of occurrences of the termset i in the query q, and k_3 is a parameter, which depends on the collection and possibly on the nature of the user queries.

The set-based model computes the similarity between a document and the user query as the scalar product between the document vector $\vec{d_j}$, $1 \leq j \leq N$, and the query vector $\vec{q}$, as follows:

$$sim(q, d_j) = \vec{d_j} \bullet \vec{q} = \sum_{S_i \in S_q} w_{S_i,j} \times w_{S_i,q}, \quad (10)$$

where $w_{S_i,j}$ is the weight associated with the termset S_i in the document d_j, $w_{S_i,q}$ is the weight associated with the termset S_i in the query q, and S_q is the set of all termsets generated from the query terms. That is, the ranking computation is restricted to the termsets generated by the query, and the number of termsets induced by the queries is usually small.

3.3.4 *The MAXTERM Ranking Formula*

In this work, we restrict the use of termsets to maximal termsets. Given a user query q, we enumerate its related maximal termsets and compute the partial similarities between each maximal termset S_i and a document d_j, according to the modified set-based model ranking formula.

To illustrate, let us consider our fictitious example depicted in Figure 1. For the query $q = \{a, b, x, y\}$, the related maximal termsets are S_{abx} and S_{aby}. We process the enumerated maximal termsets and rank the retrieved documents using Eq. (10).

4. EXPERIMENTAL RESULTS

In this section we describe the experimental results. First we present the reference collections we used, followed by a retrieval effectiveness evaluation and a performance evaluation of the proposed approach. Our experiments were performed in a Linux-based PC with an AMD-athlon 2600+ 2.0 GHz processor and 512 MBytes of RAM.

We quantify the retrieval effectiveness through standard measures of average recall and precision. We employed two aggregate metrics: (i) standard 11-point average precision figures and (ii) average precision over the retrieved documents. We use the "Wilcoxon's rank test" to establish whether the difference in precision and recall between the evaluated approaches is statistically significant [26]. Computational efficiency is evaluated through query response times.

4.1 The Reference Collections

In this evaluation we use two reference collections: WBR-04 and TREC-8 [24]. Table 4 presents the main features of these collections.

Table 4: Characteristics of the two reference collections TREC-8 and WBR-04.

Characteristics	Collection	
	TREC-8	WBR-04
Number of Documents	528,155	15,240,881
Number of Distinct Terms	737,833	4,217,897
Number of Available Topics	450	1,733,087
Number of Topics Used	50 (401-450)	100
Avg. Terms per Topic	10.80	5.95
Avg. Relevants per Topic	94.56	8.40
Size (GB)	2	80

The documents in the TREC-8 come from the following sources: The Financial Times (1991-1994), Federal Register (1994), Congressional Record (1993), Foreign Broadcast Information Service, and LA Times. The TREC-8 has a total of 450 requests, usually referred to as topics. Our experiments are performed with the 50 topics numbered 401–450. The mean number of keywords per query is 10.80. All queries were generated automatically using the topic title, description, and narrative.

The WBR-04 reference collection is composed of a database of Web pages, a set of example Web queries, and a set of relevant documents associated with each example query. The database is composed of 15,240,881 pages of the Brazilian Web, under the domain ".br".

A total of 100 example queries were selected from a log containing 1,733,087 queries submitted to the *UOL Busca* search engine[4]. Since we are interested in *complex* queries, we selected those composed of four or more terms, which corresponds to 23% of the processed log. Among all queries with four or more terms in our log, we selected two sets of queries: the 50 most frequent ones and 50 random queries, excluding those related to the topic sex. The mean number of keywords per query is 5.95.

For each of the 100 selected queries we composed a query pool formed by the top 10 ranked documents, as given by each of the ranking variants we considered, i.e., the MAXTERM approach, the set-based model, the probabilistic model using the BM25 weighting scheme, and the standard vector space model. Each query pool contained an average of 29.62 documents. We adopted the same pooling method used for the Web-based collection of TREC [6]. The average number of relevant documents per query pool is 8.40.

4.2 Retrieval Evaluation

We report results for our approach (MAXTERM), the set-based model (SBM), the probabilistic model using the BM25 weighting scheme (BM25), and the vector space model (VSM) for processing conjunctive queries. The BM25 parameters was set with the following values: $k_1 = 1.2$, $k_2 = 0$, $k_3 = 1000$, and $b = 0.75$.

One of the key features of both the set-based model and the MAXTERM technique is the possibility of controlling the minimal frequency threshold (i.e., the minimum value of frequency for a termset to be ruled as *frequent*). By varying the minimal frequency, we can exploit a trade-off between precision and the number of termsets taken into consideration. Higher the minimum frequency, smaller the number of termsets to consider, and faster the computation. Maximum precision is reached for a minimal frequency threshold equal to 10 documents for the TREC-8 collection, and equal to 150 documents for the WBR-04 collection. We have used the same parameters tuned for the set-based model in [11]. All of our results reported here for MAXTERM and for the set-based model are based on these values.

Figures 2 and 3 show the 11-point average precision for the evaluated ranking methods. We observe that MAXTERM and the set-based model yield better precision than the vector space and the probabilistic models, regardless of the collection and of the recall level. Our approach yields the largest improvements, what shows that structured queries outperform bag-of-words queries because they capture some of the relational structure normally expressed in natural language texts. Further, we observe that the improvements are larger for the TREC-8 collection, because its larger queries allow computing a more representative set of maximal termsets.

Overall average precision for the vector space model, the probabilistic model, the set-based model, and MAXTERM is presented in Table 5. For the TREC-8 test collection, the set-based model provided a nice gain of 27.44% relative to the vector model, and 15.86% relative to the probabilistic model in average precision, while our approach boosted this gain to 36.94% and to 28.03%, respectively. MAXTERM outperforms the set-based model because it takes into account just the co-occurrence patterns that represent meaningfully "entities" found in the document collections (maximal termsets). Those "entities" may correspond to linguistic constructs (e.g., noun phrases) or other valid relationship captured by statistical constructs.

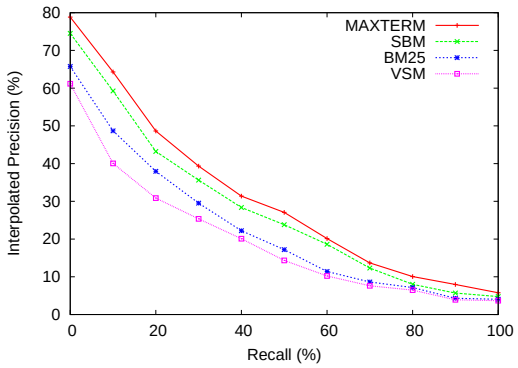

Figure 2: Precision-recall curves for the vector space model, the probabilistic model, the set-based model, and MAXTERM with the TREC-8 test collection.

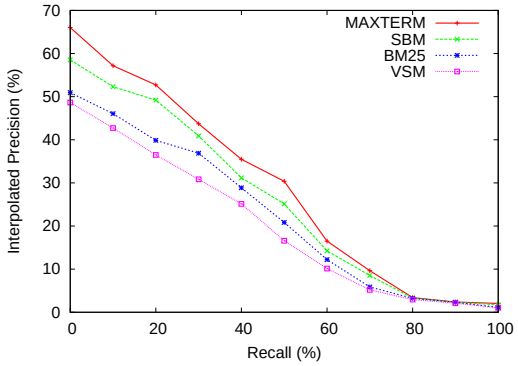

Figure 3: Precision-recall curves for the vector space model, the probabilistic model, the set-based model, and MAXTERM with the WBR-04 test collection.

For the WBR-04 test collection, while the set-based model yields a gain of 23.37% relative to the vector model, and 10.90% relative to the probabilistic model, our approach leads to a gain of 37.11% and of 23.25%, respectively. That is, our query structuring mechanism is also useful in the context of the Web. This might be important because complex queries tend to lead to more specific Web pages that are not well-known by the general public. This means that the number of links to these pages tends to be small, making ranking based on link analysis less effective. In this scenario, the gains provided by our approach might represent the difference between a good and a bad answer set.

Table 6 shows the statistical significance tests for the evaluated models. MAXTERM was found to be significantly better than the vector space model, the probabilistic model, and the set-based model for both test collections (TREC-8 and WBR-04) with a 95% confidence level.

[4] `http://busca.uol.com.br`

Table 5: Vector space model, probabilistic model, set-based model, and MAXTERM average precision figures for TREC-8 and WBR-04 collections.

Collection	Average Precision (%)			
	VSM	*BM25*	*SBM*	*MAXTERM*
TREC-8	22.41	24.65	28.56	30.69
WBR-04	20.18	23.56	26.13	29.04

Table 6: Comparison of average precision of the vector space model, the set-based model, with MAXTERM. Each entry has two numbers X and Y (that is, X/Y). X is the percentage of queries where MAXTERM is better that a technique B. Y is the percentage of queries where MAXTERM is worse than a technique B.

Collection	Statistical Significance		
	VSM	*BM25*	*SBM*
TREC-8	83/12	75/17	62/20
WBR-04	70/21	66/23	60/25

4.3 Performance Evaluation

We compare our approach to the set-based model, to the probabilistic model, and to the vector space model, when query response times are considered. Table 7 displays the response time for the evaluated approaches. We observe that the set-based model takes execution times 14.24% and 42.62% larger than the vector space model, 12.65% and 39.86% larger than the probabilistic model for the WBR-04 and the TREC-8 collections, respectively. The execution time increase for MAXTERM ranges from 9.94 to 20.97% relative to the vector space model, 8.41 to 18.62% relative to the probabilistic model for the WBR-04 and the TREC-8 collections, respectively. The results show that MAXTERM outperforms the set-based model considering both retrieval effectiveness and execution time.

Table 7: Average response times and response time increases for the vector space model, the probabilistic model, the set-based model, and MAXTERM with the TREC-8 and the WBR-04 test collections.

Collection	Average Response Time (s)			
	VSM	*BM25*	*SBM*	*MAXTERM*
TREC-8	0.6022	0.6141	0.8589	0.7285
WBR-04	0.5531	0.5609	0.6319	0.6081

The execution times of the set-based and MAXTERM models are proportional to the number of terms in the query for a given minimal frequency threshold. The number of termsets depends also on the query size and the minimal frequency threshold employed. Table 8 shows the average number of enumerated termsets for the set-based and MAXTERM models. The operations over the inverted lists of termsets dominate the execution time for these models. The execution times of our approach are lower when compared to the set-based model due to the smaller number of termsets it uses.

Table 8: Average number of termsets for the set-based model and MAXTERM with the TREC-8 and the WBR-04 test collections.

Collection	Average Number of Termsets	
	SBM	*MAXTERM*
TREC-8	285.90	5.54
WBR-04	124.11	1.92

5. CONCLUSIONS AND FUTURE WORK

We presented MAXTERM, a formalism for automatically structuring a user query into a disjunction of smaller conjunctive subqueries. Our approach analyses the document collection and determines the best conjunctive components based on their support in the document collection. We showed that MAXTERM allows significant improvements in retrieval effectiveness, with processing times close to the times to process the vector space model and the probabilistic model.

We evaluated and validated our proposed approach using two test collections. We showed through curves of recall versus precision that MAXTERM is superior for all the collections considered. Further, the low computational costs allow the use of our approach with general large collections.

Computation of termsets might be restricted by proximity information. This is useful because proximate termsets carry more semantic information than standard termsets. For future work we will investigate the behavior of the MAXTERM approach, when proximity information of query terms in the documents of the collection is taken into account.

Acknowledgments

This work was supported in part by CNPq scholarship 141.269/02-2 (Bruno Pôssas), by CNPq grant 520.916/94-8 (Nivio Ziviani), by CNPq grant 30.0188/95-1 (Berthier Ribeiro-Neto), by CNPq grant 30.9379/03-2 (Wagner Meira Jr.), and by the GERINDO project grant MCT/CNPq/CT-INFO 552.087/02-5.

6. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD International Conference Management of Data*, pages 207-216, Washington, D.C., May 1993.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the 7th International World Wide Web Conference*, pages 107–117, April 1998.

[3] W. B. Croft, H. R. Turtle, and D. D. Lewis. The use of phrases and structured queries in information retrieval. In *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 32–45, Chicago, Illinois, USA, 1991.

[4] P. Das-Gupta. Boolean interpretation of conjunctions for document retrieval. In *Journal of the American Society for Information Science*, volume 38, pages 349-368, 1987.

[5] K. Gouda and M. J. Zaki. Efficiently mining maximal frequent itemsets. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 163-170, 2001.

[6] D. Hawking, N. Craswell, P. B. Thistlewaite, and D. Harman. Results and challenges in web search evaluation. *Computer*

Networks, 31(11–16):1321–1330, May 1999. Also in Proceedings of the 8th International World Wide Web Conference.

[7] M. Mitra, C. Buckley, A. Singhal, and C. Cardie. An analysis of statistical and syntatic phrases. In *Proceedings of RIAO-97, 5th International Conference Recherche d'Information Assistee par Ordinateur*, pages 200–214, Montreal, CA, 1997.

[8] M. Narita and Y. Ogawa. The use of phrases from query texts in information retrieval. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 318–320, Athens, Greece, July 2000.

[9] B. Pôssas, N. Ziviani, and W. Meira Jr. Enhancing the set-based model using proximity information. In *The 9th International Symposium on String Processing and Information Retrieval*, pages 104–116, Lisbon, Portugal, September 2002.

[10] B. Pôssas, N. Ziviani, W. Meira Jr., and B. Ribeiro-Neto. Set-based model: A new approach for information retrieval. In *The 25th ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 230–237, Tampere, Finland, August 2002.

[11] B. Pôssas, N. Ziviani, B. Ribeiro-Neto, and W. Meira Jr. Set-based vector model: An efficient approach for correlation-based ranking. *ACM Transactions on Information Systems*. To appear.

[12] B. Pôssas, N. Ziviani, B. Ribeiro-Neto, and W. Meira Jr. Processing conjunctive and phrase queries with the set-based model. In *The 11th International Symposium on String Processing and Information Retrieval*, pages 171–183, Padova, Italy, October 2004.

[13] S. E. Robertson and K. S. Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, May-June 1976.

[14] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *Proceedings of the Third Text Retrieval Conference (TReC-3)*, pages 109–126. NIST Special Publication 500-225, April 1995.

[15] G. Salton, E. A. Fox, and H. Wu. Extended boolean information retrieval. In *Comunications of the ACM*, volume 26, pages 1022–1036, 1983.

[16] G. Salton and M. E. Lesk. Computer evaluation of indexing and text processing. In *Journal of the ACM*, volume 15(1), pages 8–36, January 1968.

[17] G. Salton and C. S. Yang. On the specification of term values in automatic indexing. In *Journal of Documentation*, volume 29, pages 351-372, 1973.

[18] A. F. Smeaton and C. J. van Rijsbergen. Experiments on incorporating syntatic processing of user queries into a document retrieval strategy. In *The 11th ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 31–51, Grenoble, France, 1988.

[19] M. E. Smith. *Aspects of the P-Norm Model of Information Retrieval: Syntatic Query Generation, Efficiency and Theoretical Properties*. PhD thesis, Computer Science Department, Cornell University, 1990.

[20] J. K. Sparck. A statistical interpretation of term specificity and its application in retrieval. In *Journal of Documentation*, volume 28(1), pages 11-21, 1972.

[21] M. Srikanth and R. Srihari. Exploiting syntactic structure of queries in a language modeling approach to ir. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 476–483. ACM Press, 2003.

[22] H. 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.

[23] H. Turtle and W. B. Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, 9(3):187–222, July 1991.

[24] E. Voorhees and D. Harman. Overview of the eighth text retrieval conference (trec 8). In *The Eighth Text Retrieval Conference*, pages 1–23. National Institute of Standards and Technology, 1999.

[25] C. T. Yu and G. Salton. Precision weighting – an effective automatic indexing method. In *Journal of the ACM*, volume 23(1), pages 76–88, January 1976.

[26] J. Zobel. How reliable are the results of large-scale information retrieval experiments? In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 307–314, Melbourne, Australia, August 1998. ACM Press.