# Processing Conjunctive and Phrase Queries with the Set-Based Model *

Bruno Pôssas[1] Nivio Ziviani[1]
Berthier Ribeiro-Neto[1] Wagner Meira Jr.[1]

Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
30161-970 Belo Horizonte-MG, Brazil
{bavep,nivio,berthier,meira}@dcc.ufmg.br

**Abstract.** The objective of this paper is to present an extension to the set-based model (SBM), which is an effective technique for computing term weights based on co-occurrence patterns, for processing conjunctive and phrase queries. The intuition that semantically related term occurrences often occur closer to each other is taken into consideration. The novelty is that all known approaches that account for co-occurrence patterns was initially designed for processing disjunctive (OR) queries, and our extension provides a simple, effective and efficient way to process conjunctive (AND) and phrase queries. This technique is time efficient and yet yields nice improvements in retrieval effectiveness. Experimental results show that our extension improves the average precision of the answer set for all collection evaluated, keeping computational cost small. For the TReC-8 collection, our extension led to a gain, relative to the standard vector space model, of 23.32% and 18.98% in average precision curves for conjunctive and phrase queries, respectively.

## 1 Introduction

Users of the World Wide Web are not only confronted by an immense overabundance of information, but also by a plethora of tools for searching for the web pages that suit their information needs. Web search engines differ widely in interface, features, coverage of the web, ranking methods, delivery of advertising, and more. Different search engines and portals have different (default) semantics of handling a multi-word query. Although, all major search engines, such as Altavista, Google, Yahoo, Teoma, uses the AND semantics, i.e. conjunctive queries, (it is mandatory for all the query words to appear in a document for it to be considered).

In this paper we propose a extension to the set-based model [1, 2] to process conjunctive and phrase queries. The set-based model uses a term-weighting scheme based on association rules theory [3]. Association rules are interesting because they provide all the elements of the $tf \times idf$ scheme in an algorithmically efficient and parameterized approach. Also, they naturally provide for quantification of representative term co-occurrence patterns, something that is not present in the $tf \times idf$ scheme.

* This work was supported in part by the GERINDO project–grant MCT/CNPq/CT-INFO 552.087/02-5 and by CNPq grant 520.916/94-8 (Nivio Ziviani).

We evaluated and validated our extension of the set-based model (SBM) for processing conjunctive and phrase queries through experimentation using two reference collections. Our evaluation is based on a comparison to the standard vector space model (VSM) adapted to handle the these type of queries. Our experimental results show that the SBM yields higher retrieval performance, which is superior for all query types and collections considered. For the TReC-8 collection [4], containing 2 gigabytes of size and approximately 530,000 documents, the SBM yields, respectively, an average precision that are 23.32% and 18.98% higher than the VSM for conjunctive and phrase queries. The set-based model is also competitive in terms of computing performance. For the WBR99 collection, containing 16 gigabytes of size and approximately 6,000,000 web pages, using a log with 100,000 queries, the increase in the average response time is, respectively, 6.21% and 14.67% for conjunctive and phrase queries.

The paper is organized as follows. The next section describes the representation of co-occurrence patterns based on a variant of association rules. A review of the set-based model is presented in the Section 3. Section 4 presents our extension of the set-based model for processing conjunctive and phrase queries. In section 5 we describe the reference collections and the experimental results comparing the VSM and the SBM for processing disjunctive and phrase queries. Related works are discussed in Section 6. Finally, we present some conclusions and future work in Section 7.

## 2 Preliminaries

In this section we introduce the concept of termsets as a basis for computing term weights. In the set-based model a document is described by a set of termsets, where termset is simply an ordered set of terms extracted from the document itself.

Let $T = \{k_1, k_2, ..., k_t\}$ be the vocabulary of a collection of documents $D$, that is, the set of $t$ unique terms that may appear in a document from $D$. 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 1.** *An n-termset $s$ is an ordered set of $n$ unique terms, such that $s \subseteq T$. Notice that the order among terms in $s$ follows the aforementioned total ordering.*

Let $S = \{s_1, s_2, ..., s_{2^t}\}$ be the vocabulary-set of a collection of documents $D$, that is, the set of $2^t$ unique termsets that may appear in a document from $D$. Each document $j$ from $D$ is characterized by a vector in the space of the termsets. With each termset $s_i$, $1 \leq i \leq 2^t$, we associate an inverted list $ls_i$ composed of identifiers of the documents containing that termset. We also define the frequency $ds_i$ of a termset $s_i$ as the number of occurrences of $s_i$ in $D$, that is, the number of documents where $s_i \subseteq d_j$ and $d_j \in D$, $1 \leq j \leq N$. The frequency $ds_i$ of a termset $s_i$ is the length of its associated inverted list ($| ls_i |$).

**Definition 2.** *A termset $s_i$ is a frequent termset if its frequency $ds_i$ is greater than or equal to a given threshold, which is known as support in the scope of association rules [3], and referred as minimal frequency in this work. As presented in the original Apriori algorithm [5], an $n$-termset is frequent if and only if all of its $n-1$-termsets are also frequent.*

The proximity information is used as a pruning strategy to find only the termsets occurrences bounded by an specified proximity threshold (referred as minimal proximity), conforming with the assumption that semantically related term occurrences often occur closer to each other.

**Definition 3.** *A closed termset $cs_i$ is a frequent termset that is the largest termset among the termsets that are subsets of $cs_i$ and occur in the same set of documents. That is, given a set $\mathcal{D} \subseteq D$ of documents and the set $\mathcal{S}_{\mathcal{D}} \subseteq S$ of termsets that occur in all documents from $\mathcal{D}$ and only in these, a closed termset $cs_i$ satisfies the property that $\nexists s_j \in \mathcal{S}_{\mathcal{D}} | cs_j \subset s_j$.*

For sake of processing disjunctive queries (OR), closed termsets are interesting because they represent a reduction on the computational complexity and on the amount of data that has to be analyzed, without loosing information, since all frequent termsets in a closure are represented by the respective closed termset [1, 2].

**Definition 4.** *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 $\mathcal{D}$, a maximal termset $ms_i$ satisfies the property that $\nexists s_j \in \mathcal{S}_{\mathcal{D}} | ms_i \subset s_j$.*

Let $FT$ be the set of all frequent termsets, and $CFT$ 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: $MFT \subseteq CFT \subseteq CT$. The set $MFT$ is orders of magnitude smaller than the set $CFT$, which itself is orders of magnitude smaller than the set $FT$. It is proven that the set of maximal termsets associated with a document collection are the minimum amount of information necessary to derive all frequent termsets associated with that collection [6].

Generating maximal termsets is a problem very similar to mining association rules and the algorithms employed for the latter is our starting point [3]. Our approach is based on an efficient algorithm for association rule mining, called GENMAX [7], which has been adapted to handle terms and documents instead of items and transactions, respectively. GENMAX uses backtracking search to enumerate all $MFT$.

## 3 Review of the Set-Based Model (SBM)

In the set-based model, a document is described by a set of termsets, extracted from the document itself. With each termset we associate a pair of weights representing (a) its importance in each document and (b) its importance in the whole document collection. In a similar way, a query is described by a set of termsets with a weight representing the importance of each termset in the query. The algebraic representation of the set of termsets for both documents and queries correspond to vectors in a $2^t$-dimensional Euclidean space, where $t$ is equal to the number of unique index terms in the document collection.

### 3.1 Termset Weights

Term weights can be calculated in many different ways [8, 9]. The best known term weighting schemes use weights that are function of (i) $tf_{i,j}$, the number of times that an index term $i$ occurs in a document $j$ and (ii) $df_i$, the number of documents that an index term $i$ occurs in the whole document collection. Such term-weighting strategy is called $tf \times idf$ schemes. The expression for $idf_i$ represents the importance of term $i$ in the collection, it assigns a high weight to terms which are encountered in a small number of documents in the collection, supposing that rare terms have high discriminating value.

In the set-based model, the association rules scheme naturally provides for quantification of representative patterns of term co-occurrences, something that is not present in the $tf \times idf$ approach. To determine the weights associated with the termsets, we also use the number of occurrences of a termset in a document, in a query, and in the whole collection. Formally, the weight of a termset $i$ in a document $j$ is defined as:

$$w_{i,j} = (1 + \log sf_{i,j}) \times ids_i = (1 + \log sf_{i,j}) \times \log\left(1 + \frac{N}{ds_i}\right) \tag{1}$$

where $N$ is the number of documents in the collection, $sf_{i,j}$ is the number of occurrences of the termset $i$ in document $d_j$, and $ids_i$ is the inverted frequency of occurrence of the termset $s_i$ in the collection. $sf_{i,j}$ generalizes $tf_{i,j}$ in the sense that it counts the number of times that the termset $s_i$ appears in document $d_j$. The component $ids_i$ also carries the same semantics of $idf_i$, but accounting for the cardinality of the termsets as follows. High-order termsets usually have low frequency, resulting in large inverted frequencies. Thus, this strategy assigns large weights to termsets that appear in small number of documents, that is, rare termsets result in greater weights.

### 3.2 Similarity Calculation

Since documents and queries are represented as vectors, we assign a similarity measure to every document containing any of the query termsets, defined as the normalized scalar product between the set of document vectors $\boldsymbol{d_j}$, $1 \leq j \leq N$, and the query vector $\boldsymbol{q}$. This approach its equivalent to the cosine of the angle between these two vectors. The similarity between a document $d_j$ and a query $q$ is defined as:

$$sim(q, d_j) = \frac{\boldsymbol{d_j} \bullet \boldsymbol{q}}{|\boldsymbol{d_j}| \times |\boldsymbol{q}|} = \frac{\sum_{s \in S_q} w_{s,j} \times w_{s,q}}{|\boldsymbol{d_j}| \times |\boldsymbol{q}|}, \tag{2}$$

where $w_{s,j}$ is the weight associated with the termset $s$ in document $d_j$, $w_{s,q}$ is the weight associated with the termset $s$ in query $q$, $S_q$ is the set of all termsets such that all $s \subseteq q$. We observe that the normalization (i.e., the factors in the denominator) was not expanded, as usual. The normalization is done using only the 1-termsets that compose the query and document vectors. This is important to reduce computational costs because computing the norm of a document using termsets might be prohibitively costly.

### 3.3 Searching Algorithm

The steps performed by the set-based model to the calculation of the similarity metrics are equivalent to the standard vector space model. Figure 1 presents the searching algorithm. First we create the data structures (line 4) that are used for calculating the document similarities $A$ among all termsets $S_q$ of a document $d_j$. Then, for each query term, we retrieve its inverted list, and determine the first frequent termsets, i.e., the frequent termsets of size equal to 1, applying the minimal frequency threshold $mf$ (lines 5 to 10). The next step is the enumeration of all termsets based on the 1-termsets, filtered by the minimal frequency and proximity threshold (line 11). After enumerating all termsets, we evaluated its inverted lists, calculating the partial similarity of a termset $s_i \in S_q$ to a document $d_j$ (lines 12 to 17). After evaluating termsets, we normalize the document similarities $A$ by dividing each document similarity $A_j \in A$ by the norm of the document $d_j$ (line 18). The final step is to select the $k$ largest similarities and return the corresponding documents (line 19).

SBM(**Q**, **mf**, **mp**, **k**)<br>
  **Q** : *a set of query terms*<br>
  **mf** : *minimum frequency threshold*<br>
  **mp** : *minimum proximity threshold*<br>
  **k** : *number of documents to be returned*<br>
*1.* *Let* $\mathbf{A}$ *be a set of accumulators*<br>
*2.* *Let* $\mathbf{C_q}$ *be a set of 1-termsets*<br>
*3.* *Let* $\mathbf{S_q}$ *be a set of termsets*<br>
*4.* $A = \emptyset, \quad S = \emptyset$<br>
*5.* *for each query term* $t \in Q$ *do begin*<br>
*6.*   *if* $df_t \geq mf$ *then begin*<br>
*7.*     *Obtain the 1-termset* $s_t$ *from term* $t$<br>
*8.*     $C_q = C_q \cup \{s_t\}$<br>
*9.*   *end*<br>
*10.* *end*<br>
*11.* $S_q$ = *Termsets_Gen(*$C_q$*, mf, mp)*<br>
*12.* *for each termset* $s_i \in S_q$ *do begin*<br>
*13.*   *for each* $[d_j, sf_{i,j}]$ *in* $ls_i$ *do begin*<br>
*14.*     *if* $A_j \notin A$ *then* $A = A \cup \{A_j\}$<br>
*15.*     $A_j = A_j + w_{s_i,j} \times w_{s_i,q}$*, from Eq. (1).*<br>
*16.*   *end*<br>
*17.* *end*<br>
*18.* *for each accumulator* $A_j \in A$ *do* $A_j = A_j \div |\boldsymbol{d_j}|$<br>
*19.* *determine the* $k$ *largest* $A_j \in A$ *and return the corresponding documents*<br>
*20.* *end*

**Fig. 1.** The set-based model searching algorithm

### 3.4 Computational Complexity

The complexity of the standard vector space model and the set-based model is linear with respect to the number of documents in the collection. Formally, the upper bound on the number of operations performed for satisfying a query in the vector space model is $O(|q| \times N)$, where $|q|$ is the number of terms in the query and $N$ is the number of documents in the collection. The worst case scenario for the vector space model is a query comprising the whole vocabulary $t$ ($|q| = t$), which results in a merge of all inverted lists in the collection. The computational complexity for set-based model is $O(cN)$, where $c$ is the number of termsets, a value that is $O(2^{|q|})$, where $|q|$ is the number of terms in the query. These are worst case measures and the average measures for the constants involved are much smaller [1, 2].

## 4 Modeling Conjunctive and Phrase Queries in SBM

In this section we show how to model conjunctive and phrase queries using the framework provided by the original set-based model (see Section 3). Our approach does not modify its algebraic representation, and the changes to the original model are minimal.

### 4.1 Conjunctive Queries

The main modification of set-based model for the conjunctive and phrase query processing is related to the enumeration of termsets. In the original version of the set-based model, the enumeration algorithm determines all closed termsets for a given user query, and the minimal frequency and proximity thresholds. Each mined closed termset represents a valid co-occurrence pattern in the space of documents defined by the terms of the query. For disjunctive queries, each one of these patterns contributes for the similarity between a document and a query. The conjunctive query processing requires that only the co-occurrence pattern defined for the query can be found, i.e., the occurrence of all query terms in a given document must be valid. If so, this document can be added to the response set.

A maximal termset corresponds to a frequent termset that is not a subset of any other frequent termset (see definition 4). Based on this definition, we can extend the original set-based model enumerating the set of maximal termsets for a given user query instead of the set of closed termsets. To verify if the enumerated set is valid, we check the following conditions. First, the mined set of maximal termsets must be composed by an unique element. Second, this element must have all query terms. If all these conditions are true, we can evaluate the inverted list of maximal termset found, calculating its partial similarity to each document $d_j$ (lines 12 to 17 of the algorithm of Figure 1). The final steps are the normalization of document similarities and the selection of the $k$ largest similarities, returning the corresponding documents.

The proximity information is used as a pruning strategy to find only the maximal termset occurrences bounded by the minimal proximity threshold, conforming with the assumption that semantically related term occurrences often occur closer to each other. This pruning strategy is incorporated in the maximal termsets enumeration algorithm.

### 4.2 Phrase Queries

Search engines are used to find data in response to ad hoc queries. However, a significant fraction of the queries include phrases, where the user has indicated that some of the query terms must be adjacent, typically by enclosing them in quotation marks. Phrases have the advantage of being unambiguous concept markers and are therefore viewed as a valuable addition to ranked queries.

A standard way to evaluate phrase queries is to use an inverted index, in which for each index term there is a list of postings, and each posting includes a document identifier, an in-document frequency, and a list of ordinal word positions at which term occurs in the document. Given such a word-level inverted index and a phrase query, it is straightforward to combine the postings lists for the query to identify matching documents and to rank them using the standard vector space model.

The original set-based model can be easily adapted to handle phrase queries. To achieve this, we enumerate the set of maximal termsets instead of the set of closed termsets, using the same restrictions applied for conjunctive queries. To verify if the query terms are adjacent, we just check if its ordinal word positions are adjacents. The proximity threshold is then set to one and is used to evaluated this referential constraint.

We may expect that this extension to the set-based model is suitable for selecting just maximal termsets representing strong correlations, increasing the retrieval effectiveness of both type of the queries. Our experimental results(see Section 5.3) confirm such observations.

## 5 Experimental Evaluation

In this section we describe experimental results for the evaluation of the set-based model (SBM) for conjunctive and phrase queries in terms of both effectiveness and computational efficiency. Our evaluation is based on a comparison to the standard vector space model (VSM). We first present the experimental setup and the reference collections employed, and then discuss the retrieval performance and the computational efficiency.

### 5.1 Experimental Setup

In this evaluation we use two reference collections that comprise not only the documents, but also a set of example queries and the relevant responses for each query, as selected by experts. We quantify the retrieval effectiveness of the various approaches through standard measures of average recall and precision. The computational efficiency is evaluated through the query response time, that is, the processing time to select and rank the documents for each query.

The experiments were performed on a Linux-based PC with a AMD-athlon 2600+ 2.0 GHz processor and 512 MBytes RAM. Next we present the reference collections used, followed by the results obtained.

**Table 1.** Characteristics of the reference collections

| **Characteristics** | **Collection** | |
|---|---|---|
| | TReC-8 | WBR99 |
| Number of Documents | 528,155 | 5,939,061 |
| Number of Distinct Terms | 737,833 | 2,669,965 |
| Number of Topics | 450 | 100,000 |
| Number of Used Topics | 50 (401-450) | 50 |
| Average Terms per Query | 4.38 | 1.94 |
| Average Relevants per Query | 94.56 | 35.40 |
| Size (GB) | 2 | 16 |

### 5.2 The Reference Collections

In our evaluation we use two reference collections WBR99 and TReC-8 [4]. Table 1 presents the main features of these collections.

The WBR99 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 5,939,061 pages of the Brazilian Web, under the domain ".br". We decided to use this collection because it represents a highly connected subset of the Web, that is large enough to provide good prediction of the results if the whole Web was used, and, at the same time, is small enough to be handled by our available computational resources.

A total of 50 example queries were selected from a log of 100 000 queries submitted to the *TodoBR* search engine[1]. The queries selected were the 50 most frequent ones with more than two terms. Some frequent queries related to sex were not considered. The mean number of keywords per query is 3.78. The sets of relevant documents for each query were build using pooling method used for the Web-based TREC collection [10]. We compose a query pool formed by the top 15 documents generated by each evaluated model for both query types. Each query pool contained an average of 19.91 documents for conjunctive queries and 18.77 for phrase queries. All documents present in each query pool were submitted to a manual evaluation by a group of 9 users, all of them familiar with Web searching. The average number of relevant documents per query is 12.38 and 6.96 for conjunctive and phrase queries, respectively.

The TReC-8 collection [4] has been growing steadily over the years. At TREC-8, which is used in our experiments, the collection size was roughly 2 gigabytes. The documents presents in the TReC-8 collection are tagged with SGML to allow easy parsing, and come from the following sources: The Financial Times, Federal Register, Congressional Record, Foreign Broadcast Information Service and LA Times.

The TReC collection includes a set of example *information requests* (queries) which can be used for testing a new ranking algorithm. Each request is a description of an information need in natural language. The TReC-8 has a total of 450 queries, usually referred as a topic. Our experiments are performed with the 401-450 range of topics. This range of topics has 4.38 index terms per query.

[1] http://www.todobr.com.br

### 5.3 Retrieval Performance

We start our evaluation by verifying the precision-recall curves for each model when applied to the reference collection. Each curve quantifies the precision as a function of the percentage of documents retrieved (recall). The results presented for SBM for both query types were obtained by setting the minimal frequency threshold to a one document. The minimal proximity threshold was not used for the conjunctive queries evaluation, and set to one for the phrase queries.

As we can see in Table 2, SBM yields better precision than VSM, regardless of the recall level. Further, the gains increase with the size of queries, because large queries allow computing a more representative termset, and are consistently greater for the both query types. Furthermore, accounting for correlations among terms never degrades the quality of the response sets. We confirm such observations by verifying the overall average precision achieved for each model. The gains provided by SBM over the VSM in terms of overall precision was 7.82% and 23.32% for conjunctive queries and 9.73% and 18.98% for phrase queries for WBR99 and TReC-8 collections, respectively.

**Table 2.** Recall-precision curves for VSM and SBM

(a) Conjunctive queries results

| **Recall (%)** | **Precision(%)** | | | |
|---|---|---|---|---|
| | WBR99 | | TReC-8 | |
| | *VSM* | *SBM* | *VSM* | *SBM* |
| 0 | 44.14 | 48.81 | 63.17 | 74.41 |
| 10 | 43.03 | 48.81 | 44.06 | 53.63 |
| 20 | 42.23 | 46.13 | 33.87 | 38.56 |
| 30 | 38.40 | 41.93 | 26.36 | 31.81 |
| 40 | 37.41 | 39.66 | 20.11 | 25.69 |
| 50 | 37.41 | 39.32 | 15.35 | 21.24 |
| 60 | 36.71 | 37.74 | 10.22 | 16.66 |
| 70 | 33.97 | 36.53 | 7.63 | 10.88 |
| 80 | 29.62 | 32.55 | 6.48 | 7.24 |
| 90 | 27.27 | 28.25 | 3.90 | 5.13 |
| 100 | 25.00 | 26.29 | 3.67 | 4.32 |
| **Average** | 35.92 | 38.73 | 21.35 | 26.33 |
| **Improvement** | - | 7.82 | - | 23.32 |

(b) Phrase queries results

| **Recall (%)** | **Precision(%)** | | | |
|---|---|---|---|---|
| | WBR99 | | TReC-8 | |
| | *VSM* | *SBM* | *VSM* | *SBM* |
| 0 | 48.71 | 51.38 | 42.41 | 45.15 |
| 10 | 41.48 | 43.58 | 20.21 | 28.61 |
| 20 | 27.83 | 31.91 | 16.07 | 21.71 |
| 30 | 21.13 | 23.35 | 12.47 | 15.70 |
| 40 | 17.59 | 18.67 | 11.19 | 13.92 |
| 50 | 10.54 | 11.71 | 9.25 | 10.47 |
| 60 | 4.18 | 5.22 | 7.64 | 8.85 |
| 70 | 2.94 | 3.49 | 6.62 | 6.97 |
| 80 | 2.09 | 2.96 | 3.92 | 4.00 |
| 90 | 1.03 | 1.57 | 3.30 | 3.44 |
| 100 | 0.23 | 1.23 | 3.24 | 3.37 |
| **Average** | 16.16 | 17.73 | 12.39 | 14.75 |
| **Improvement** | - | 9.73 | - | 18.98 |

In summary, set-based model (SBM) is the first information retrieval model that exploits term correlations and term proximity effectively and provides significant gains in terms of precision, regardless of the query type. In the next section we discuss the computational costs associated with SBM.

### 5.4 Computational Efficiency

In this section we compare our model to the standard vector space model regarding the query response time, in order to evaluate its feasibility in terms of computational costs. This is important because one major limitation of existing models that account for term correlations is their computational cost. Several of these models cannot be applied to large or even mid-size collections since their costs increase exponentially with the vocabulary size.

We compared the response time for the models and collections considered, which are summarized in Table 3. We also calculated the increase of the response time of SBM when compared to VSM for both query types. All 100,000 queries submitted to the *TodoBR* search engine, excluding the unique term queries, was evaluated for the WBR99 collection. We observe that SBM results in a response time increase of 6.21% and 24.13% for conjunctive queries and 14.67% and 23.68% for phrase queries when compared to VSM for WBR99 and TReC-8, respectively.

**Table 3.** Average response time for VSM and SBM

| **Query Type** | **Avg. Response Time (s)** | | | |
|---|---|---|---|---|
| | WBR99 | | TReC-8 | |
| | *VSM* | *SBM* | *VSM* | *SBM* |
| conjunctive | 0.632 | 0.671 | 0.029 | 0.036 |
| phrase | 1.491 | 1.709 | 0.038 | 0.047 |

We identify one main reason for the relatively small increase in execution time for SBM. Determining maximal termsets and calculating their similarity do not increase significantly the cost associated with queries. This fact happens due to the small number of query related termsets in the reference collections, especially for WBR99. As a consequence, the inverted lists associated tend to be small and are usually manipulated in main memory in our implementation of SBM. Second, we employ pruning techniques that discard irrelevant termsets early in the computation, as described in [1].

## 6 Related Work

The vector space model was proposed by Salton [11, 12], and different weighting schemes were presented [8, 9]. In the vector space model, index terms are assumed to be mutually independent. The independence assumption leads to a linear weighting function which, although not necessarily realistic, is ease to compute.

Different approaches to account for co-occurrence among index terms during the information retrieval process have been proposed [13, 14, 15]. The work in [16] presents an interesting approach to compute index term correlations based on automatic indexing schemes, defining a new information retrieval model called generalized vector space model. Wong et al. [17] extended the generalized vector space model to handle queries specified as boolean expressions.

The set-based model was the first information retrieval model that exploits term correlations and term proximity effectively and provides significant gains in terms of precision, regardless of the size of the collection and of the size of the vocabulary [1, 2]. Experimental results showed significant and consistent improvements in average precision curves in comparison to the vector space model and to the generalized vector space model, keeping computational cost small.

Our work differs from that presented in the following way. First, all known approaches that account for co-occurrence patterns was initially designed for processing disjunctive (OR) queries. Our extension to the set-based model provides a simple, effective and efficient way to process conjunctive (AND) and phrase queries. Experimental results show significant and consistent improvements in average precision curves in comparison to the standard vector space model adapted to process these types of queries, keeping processing times close to the times to process the vector space model.

The work in [18] introduces a theoretical framework for the association rules mining based on a boolean model of information retrieval known as Boolean Retrieval Model. Our work differs from that presented in the following way. In our work we use association rules to define a new information retrieval model that provides not only the main term weights and assumptions of the $tf \times idf$ scheme, but also provides for quantification of term co-occurrence and can be successfully used in processing of disjunctive, conjunctive and phrase queries.

Ahonen-Myka [19] employs a versatile technique, based on maximal frequent sequences, for finding complex text phrases from full text for further processing and knowledge discovery. Our work also use the concept of maximal termsets/sequences to account for term co-occurrence patterns in documents, but the termsets are, successfully, used as a basis for an information retrieval model.

## 7 Conclusions and Future Work

We presented an extension for the set-based model to consider correlations among index terms in conjunctive and phrase queries. We show that it is possible to significantly improve retrieval effectiveness, while keeping extra computational costs small. The computation of correlations among index terms using maximal termsets enumerated by an algorithm to generate association rules leads to a direct extension of the set-based model. Our approach does not modify its algebraic representation, and the changes to the original model are minimal.

We evaluated and validated our proposed extension for the set-based model for conjunctive and phrase queries in terms of both effectiveness and computational efficiency using two test collections. We show through curves of recall versus precision that our extension presents results that are superior for all query types considered and the additional computational costs are acceptable. In addition to assessing document relevance, we also showed that the proximity information has application in identifying phrases with a greater degree of precision.

Web search engines are rapidly emerging into the most important application of the World Wide Web, and query segmentation is one of the most promising techniques to improve search precision. This technique reduce the query into a form that is more likely to express the topic(s) that are asked for, and in a suitable manner for a word-based or phrase-based inverse lookup, and thus improve precision of the search. We will use the set-based model for automatic query segmentation.

## References


1. Pôssas, B., Ziviani, N., Meira, W., Ribeiro-Neto, B.: Set-based model: A new approach for information retrieval. In: The 25th ACM-SIGIR Conference on Research and Development in Information Retrieval, Tampere, Finland (2002) 230–237
2. Pôssas, B., Ziviani, N., Meira, W.: Enhancing the set-based model using proximity information. In: The 9th International Symposium on String Processing and Information Retrieval, Lisbon, Portugal (2002)
3. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of the ACM SIGMOD International Conference Management of Data, Washington, D.C. (1993) 207-216
4. Voorhees, E., Harman, D.: Overview of the eighth text retrieval conference (trec 8). In: The Eighth Text Retrieval Conference, National Institute of Standards and Technology (1999) 1–23
5. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: The 20th International Conference on Very Large Data Bases, Santiago, Chile (1994) 487-499
6. Zaki, M.J.: Generating non-redundant association rules. In: 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, USA (2000) 34-43
7. Gouda, K., Zaki, M.J.: Efficiently mining maximal frequent itemsets. In: Proceedings of the 2001 IEEE International Conference on Data Mining. (2001) 163-170
8. Yu, C.T., Salton, G.: Precision weighting – an effective automatic indexing method. In: Journal of the ACM. Volume 23(1). (1976) 76–88
9. Salton, G., Buckley, C.: Term-weighting approaches in automatic retrieval. In: Information Processing and Management. Volume 24(5). (1988) 513–523
10. Hawking, D., Craswell, N.: Overview of TREC-2001 web track. In: The Tenth Text REtrieval Conference (TREC-2001), Gaithersburg, Maryland, USA (2001) 61–67
11. Salton, G., Lesk, M.E.: Computer evaluation of indexing and text processing. In: Journal of the ACM. Volume 15(1). (1968) 8–36
12. Salton, G.: The SMART retrieval system – Experiments in automatic document processing. Prentice Hall Inc., Englewood Cliffs, NJ (1971)
13. Raghavan, V.V., Yu, C.T.: Experiments on the determination of the relationships between terms. In: ACM Transactions on Databases Systems. Volume 4. (1979) 240–260
14. Harper, D.J., Rijsbergen, C.J.V.: An evaluation of feedback in document retrieval using co-occurrence data. In: Journal of Documentation. Volume 34. (1978) 189–216
15. Salton, G., Buckley, C., Yu, C.T.: An evaluation of term dependencies models in information retrieval. In: The 5th ACM-SIGIR Conference on Research and Development in Information Retrieval. (1982) 151-173
16. Wong, S.K.M., Ziarko, W., Raghavan, V.V., Wong, P.C.N.: On modeling of information retrieval concepts in vector spaces. In: The ACM Transactions on Databases Systems. Volume 12(2). (1987) 299–321
17. Wong, S.K.M., Ziarko, W., Raghavan, V.V., Wong, P.C.N.: On extending the vector space model for boolean query processing. In: Proceedings of the 9th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Pisa, Italy, September 8-10, 1986, ACM (1986) 175–185
18. Bollmann-Sdorra, P., Hafez, A., Raghavan, V.V.: A theoretical framework for association mining based on the boolean retrieval model. In: Data Warehousing and Knowledge Discovery: Third International Conference, Munich, Germany (2001) 21–30
19. Ahonen-Myka, H., Heinonen, O., Klemettinen, M., Verkamo, A.: Finding co-occurring text phrases by combining sequence and frequent set discovery. In Feldman, R., ed.: Proceedings of 16th International Joint Conference on Artificial Intelligence IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications, Stockholm, Sweden (1999) 1–9