Efficient algorithms for ranking documents represented as DNF formulas

David E. Losada and Alvaro Barreiro
AILab, Department of Computer Science, University of A Corunna
A Corunna, Spain
{losada,barreiro}@dc.fi.udc.es

Abstract

This paper describes efficient procedures to compute similarity within a logical framework. The theory that underlies our approach is based on the use of Belief Revision techniques to get a measure of the uncertainty of $d \rightarrow q$, where d and q are logical representations of a document and a query respectively. However, a direct implementation of those techniques would have to deal with logical interpretations and, as a consequence, its complexity would be exponential. We propose a syntactic characterization of the logical formulas involved that allows us to design polynomial-time algorithms for computing similarity. The algorithms are presented in increasing order of expressiveness. An important aspect is that efficient IR systems realizing the theory and managing representations more expressive than classical ones can be built.

1 Introduction

One of the major problems when using logic as a representational tool is the complexity of the resulting system. Usually, the more expressive the representations are, the more complex the algorithms become. As a matter of fact, in the Artificial Intelligence field many efforts have been focused on the analysis of the tradeoff between the expressiveness of the systems and the complexity of the reasoning tasks. In Information Retrieval (IR), systems' complexity arises as a critical issue. Systems dealing with large amounts of data require a minimum response time and, therefore, very efficient procedures. As a result, logical models of IR have to take great care of their complexity results. In this work we implement document ranking within the Belief Revision (BR) framework. We follow the theoretical development of [13], where a ranking of documents given a query is obtained within Dalal's revision [5]. A document and a query are represented in Propositional Logic by logical formulas d and q respectively and Belief Revision procedures are suitable for quantifying the uncertainty of $d \rightarrow q$. This research goes in the line of some works [4, 15, 2, 18, 14, 8] that have followed different techniques in order to implement Van Rijsbergen's uncertainty principle [17]. However the model proposed in [13] needs to handle logical interpretations and, thus, its implementation would take exponential time to decide relevance. To overcome this problem, we propose a syntactic characterization for the formulas representing documents and queries. This way, we are able to develop efficient algorithms computing the similarity between a document and a query. An important advantage of the model is that documents' representations are more expressive than classical ones. Using propositional logic to model documents allows us to represent classical binary vectors (as conjunctions of terms) but more complex representations, involving conjunctions and disjunctions, can also be handled. Therefore, the availability of efficient procedures for computing similarity along with the handiness of expressive representations seem to be promising properties for a future logic-based IR system.

The rest of the paper is organized as follows. The second section presents some basic concepts of BR and the way BR was used to rank documents given a query. The algorithms are depicted in the third section in increasing order of expressiveness. The paper finishes with some conclusions and future lines of work.

2 Belief Revision

Belief Revision is a kind of non-monotonic reasoning that has been widely used in several domains [19]. Belief Revision theory deals with the change that has to be done to a knowledge base after the arrival of new information. The most interesting case arises when there is contradiction between the old and the new knowledge. A mandatory behaviour for BR methods is that the result of the revision has to be consistent. Therefore, in case of contradiction, a BR operator has to select some pieces of information to be eliminated. The properties that a *reasonable* BR operator should fulfil have been studied thoroughly and were formalized as a set of postulates. These postulates were originally defined for deductively closed set of sentences in some unspecified language [1] and equivalent postulates for knowledge bases were later proposed [12]. A basic property is that the knowledge base and the new information have different status: after the revision the new information has to be accepted no matter what happens with the old knowledge. However, a policy of information economy governs the deletion of information from the knowledge base, i.e. as much old information as possible should be preserved. We will focus on propositional languages and model-based approaches to BR. In these approaches the selection of information is based on orders among logical interpretations. The next paragraph introduces some preliminaries that are needed to understand the rest of the section.

The propositional alphabet is denoted by $\mathcal{P}$. An interpretation is a function from $\mathcal{P}$ to the set $\{true, false\}$. Interpretations will be represented by the set of propositional letters that are mapped into true. A model of a formula is an interpretation that makes the formula true and $Mod(\psi)$ denotes the set containing all the models of the formula ψ. The expression $\psi \circ_D \mu$ denotes the result of the revision of the theory ψ with the new information μ using Dalal's revision operator. The symmetric difference between two sets A and B, $A \triangle B$, is defined as $A \triangle B = (A \cup B) \setminus (A \cap B)$, where $\setminus$ is the regular difference between sets.

The following paragraphs present an overview of Dalal's revision operator [5]. This operator is appropriate to model document ranking [13]. In order to impose an order over the set of logical interpretations, Dalal defined the difference between two interpretations as the set of *differing* propositional letters and the distance between two interpretations as the number of those differing letters.

$Diff(I, J) = \{p \in \mathcal{P} | I \models p \text{ iff } J \models \neg p\}$, $Dist(I, J) = |Diff(I, J)|$.

As interpretations are represented by the set of letters mapped into true, the difference between two interpretations can be computed using the symmetric difference between their respective sets. The distance between the set of models of a formula ψ and a given interpretation I is defined as the distance from I to its closest interpretation in $Mod(\psi)$, i.e. $Dist(Mod(\psi), I) = min_{M \in Mod(\psi)} Dist(M, I)$

Given a formula ψ, an order between interpretations can be extracted from the nearness of each interpretation to the set of models of the formula ψ. This way, Dalal's total pre-order $\leq_\psi$ is defined as:

$I \leq_\psi J$ iff $Dist(Mod(\psi), I) \leq Dist(Mod(\psi), J)$

When revising a theory ψ with a new information μ, the models of the new information that are the closest to the theory are selected to be the models of the revised theory, i.e. $Mod(\psi \circ_D \mu) = Min(Mod(\mu), \leq_\psi)$

2.1 Document Ranking

This section explains briefly how BR techniques were used in [13] to get a similarity measure between a document and a query. The propositional alphabet models the vocabulary of index terms. Let us consider a document and a query represented as propositional formulas d and q respectively. Within the BR process that revises q with d using Dalal's revision operator, $q \circ_D d$, we can get a notion of closeness from the document to the query. The revision process $q \circ_D d$ uses a measure of closeness from each model of the document to the set of models of the query. We can consider the closeness to the set of models of the query as the closeness to the query itself. If a document is represented by a formula that has only one model, each document can be identified by its only model and the measure of distance from the model of the document to the set of models of a query can be regarded as a measure of distance from the document to the query. When a document has several models, the measure of distance from the document to the query is the average of the distances from each model of the document to the set of models of the query.

$distance(d, q) = \frac{\sum_{m \in Mod(d)} dist(Mod(q), m)}{|Mod(d)|}$

A similarity measure in the interval [0,1] was defined from $distance$ by normalization. This similarity measure, $BRsim$, subsumes the inner product query-document similarity measure for vectors with binary weights [13]. However the model is more general because representations more expressive than classical vectors can be handled. In the next section the reader can find an example of the computation of $BRsim$ from symmetric differences between interpretations.

3 Algorithms

The direct implementation of a model-based approach produces an algorithm whose complexity analysis is exponential. This is because logical interpretations are needed and the number of them grows exponentially with the size of the propositional alphabet. In these cases, some kind of syntactic method can be very helpful to reduce complexity. For instance, del Val [6, 7] proposed a syntactic characterization in order to be able to compute efficiently different revision operations. We are interested in Dalal's revision, which is an NP-Complete problem [5]. In fact, following the development shown in previous section, we have to compute all the models of both the theory and the new information and, afterwards, compute symmetric diferences between them. Along this work we will use the name of *models method* to refer to the direct application of Dalal's theory.

We propose to use a syntactic characterization of the formulas involved that allow us to avoid many operations between models. The main constraint is that the formulas have to be in disjunctive normal form (DNF) [1]. A DNF formula can be represented as a set of clauses, $\psi = \{\psi_1, \psi_2, \ldots\}$, where each clause is a set of literals representing their conjunction. The set ψ represents the disjunction of all the clauses. For instance, the formula $a \vee (b \wedge \neg c)$ is represented as $\{\{a\}, \{b, \neg c\}\}$. A conjunction of literals can be regarded as a partial model, representing the set of models resulting from combining the truth value of the atoms non appearing in the conjunction. For instance, given the propositional alphabet $\{a, b, c\}$, the conjunction $a \wedge \neg b$ represents the models $\{a\}$ and $\{a, c\}$. We suggest to use a distance between clauses instead of a distance between models. We can define a measure of distance, $CDist$, between two clauses ψ_i and μ_j as follows,

$CDiff(\psi_i, \mu_j) = \{l \in \psi_i | \neg l \in \mu_j\}, CDist(\psi_i, \mu_j) = |CDiff(\psi_i, \mu_j)|$

The difference is given by is the set of literals in ψ_i whose negation is in μ_j and the distance is the cardinality of the difference. The rest of the section shows several algorithms following this syntactic characterization in order to compute the similarity measure $BRsim$.

3.1 Representations as conjunctive formulas

This section presents the algorithm that computes similarity for a document and a query both stored as conjunctions of literals. In this case, query and document are already in DNF form and can be both represented as a set with one clause, i.e. $\psi = \{\psi_1\}$ and $\mu = \{\mu_1\}$. Although the representation is restricted, it is expressive enough to represent classical vectors with binary weigths. Moreover, *partial* vectors can be expressed because it is not mandatory that all the letters of the alphabet appear, either positive or negative. The following lines show the algorithm for conjunctive formulas.

Algorithm 1:
Procedure Similarity(ψ,μ)
Input:query $\psi = \{\psi_1\}$, document $\mu = \{\mu_1\}$
Output:BRsim(ψ,μ)

1. Compute $CDist(\psi_1, \mu_1)$
2. $distance = CDist(\psi_1, \mu_1) + \frac{|\psi_1 \backslash \psi_1 \cap \mu_1| - CDist(\psi_1, \mu_1)}{2}$
3. Return $(1 - \frac{distance}{|\psi_1|})$

Let us recall that the aim of the algorithm is to get the value of $BRsim$, which is a normalization of a distance obtained from the average of the distances from each model of the document to the set of models of

[1] A DNF formula has the form: $c_1 \vee c_2 \vee \ldots$ where each c_j is a conjunction of literals: $l_1 \wedge l_2 \wedge \ldots$. A literal is a propositional letter or its negation.

the query. The value of $CDist(\psi_1, \mu_1)$ is the number of literals in ψ_1 that appear in μ_1 with opposite value. Then, the interpretation for these terms will be different for any two models m_1 and m_2, where m_1 stands for any model of μ and m_2 stands for any model of ψ. As a result, all the models of the document have to fare at least $CDist(\psi_1, \mu_1)$ to any model of the query. So, the value of $CDist(\psi_1, \mu_1)$ is directly accummulated to $distance$. The elements of the set $\psi_1 \setminus \psi_1 \cap \mu_1$ are the literals in ψ_1 not belonging to μ_1. Therefore, the value $|\psi_1 \setminus \psi_1 \cap \mu_1| - CDist(\psi_1, \mu_1)$ is the number of literals in ψ_1 whose letter does not appear in μ_1, either positive or negative. As these letters do not appear in the representation of μ, half of the models of μ will map the letter into true and the other half will map it into false. On the other hand, and as a consequence of the presence of the literal in ψ_1, all the models of ψ have to map that letters into the same truth value. Whatever this fixed truth value is, half of the models of μ will have the opposite one, increasing $\frac{|\psi_1 \setminus \psi_1 \cap \mu_1| - CDist(\psi_1, \mu_1)}{2}$ in distance. Finally, the distance is transformed into a similarity value in the interval [0,1], given that the greatest value of $distance$ is $|\psi_1|$.

The complexity analysis is as follows. Step 1 can be done extracting the literals of ψ_1 and checking whether the opposite literal belongs to μ_1. It can also be done with the reciprocal process, that is, extracting in μ_1 and checking in ψ_1. Each check can be done in unit time because an array can be used to store which literals belong to a clause. Then, step 1 can be done in linear time respect to the size of either ψ_1 or μ_1. In an analogous way, step 2 can be accomplished in linear time respect to the size of either of the clauses involved. To sum up, the algorithm can be executed linearly with respect to the size of either the document or the query. As the query has usually less literals, we can conclude that the algorithm has a complexity of $\mathcal{O}(|\psi_1|)$.

The value $CDist(\psi_1, \mu_1)$ represents the number of differing terms between the query and the document. In fact, it represents the result of a matching equivalent to the inner product query-document matching function for binary vectors. When there are query terms not appearing in the document representation (this is not a regular assumption in classical systems) the model does not assume that the document is (or is not) actually about that index term. This behaviour is captured in the approach by the fact that there will be half of the models with that query term positive and half of them with the term negative.

The next example shows the computation of $BRsim$ using Algorithm 1 and using the models method. Notice the amount of models and distances between them that are computed by the models methods. On the contrary, Algorithm 1 requires very few computations. With a realistic size of the propositional alphabet the difference between both methods would be enormous. As a matter of fact, models method's run time grows exponentially with the size of the alphabet, while Algorithm 1 grows linearly w.r.t the size of the query.

Example 1: Consider the alphabet, the document and the query represented as:

$\mathcal{P} = \{a, b, c, d, e\}$
$q = a \wedge e$
$d = a \wedge b \wedge \neg d$

Algorithm 1:

Input: Query in DNF form: $\psi = \{\psi_1\}, \psi_1 = \{a, e\}$
Document in DNF form: $\mu = \{\mu_1\}, \mu_1 = \{a, b, \neg d\}$

1. $CDist(\psi_1, \mu_1) = |\{l \in \psi_1 | \neg l \in \mu_1\}| = |\emptyset| = 0$
2. $distance = 0 + \frac{|\{a,e\} \setminus \{a\}| - 0}{2} = 0.5$
3. $1 - \frac{distance}{|\psi_1|} = 1 - \frac{0.5}{2} = 0.75$. Return(0.75).

Models method:

Document models → Query models ↓	d $\{a, b\}$	 $\{a, b, c\}$	 $\{a, b, e\}$	 $\{a, b, c, e\}$
$\{a, e\}$	$\{b, e\}$	$\{b, c, e\}$	$\{b\}$	$\{b, c\}$
$\{a, b, e\}$	$\{e\}$	$\{c, e\}$	$\emptyset$	$\{c\}$
$\{a, c, e\}$	$\{b, c, e\}$	$\{b, e\}$	$\{b, c\}$	$\{b\}$
$\{a, d, e\}$	$\{b, d, e\}$	$\{b, c, d, e\}$	$\{b, d\}$	$\{b, c, d\}$
$\{a, b, c, e\}$	$\{c, e\}$	$\{e\}$	$\{c\}$	$\emptyset$
$\{a, b, d, e\}$	$\{d, e\}$	$\{c, d, e\}$	$\{d\}$	$\{c, d\}$
$\{a, c, d, e\}$	$\{b, c, d, e\}$	$\{b, d, e\}$	$\{b, c, d\}$	$\{b, d\}$
$\{a, b, c, d, e\}$	$\{c, d, e\}$	$\{d, e\}$	$\{c, d\}$	$\{d\}$

Table 1. Symmetric differences between models of the query and models of the document

Document models $\rightarrow$ Query models $\downarrow$	d $\{a,b\}$	$\{a,b,c\}$	$\{a,b,e\}$	$\{a,b,c,e\}$
$\{a,e\}$	2	3	1	2
$\{a,b,e\}$	1	2	0	1
$\{a,c,e\}$	3	2	2	1
$\{a,d,e\}$	3	4	2	3
$\{a,b,c,e\}$	2	1	1	0
$\{a,b,d,e\}$	2	3	1	2
$\{a,c,d,e\}$	4	3	3	2
$\{a,b,c,d,e\}$	3	2	2	1
$dist(Mod(q),m_i) = min_{m \in Mod(q)} dist(m,m_i)$	1	1	0	0
$distance(d,q) = \frac{\sum_{m \in Mod(d)} dist(Mod(q),m)}{\|Mod(d)\|}$	$\frac{2}{4} = 0.5$			

Table 2. Cardinalities and computation of the distance

Given k, the number of literals appearing in the query, $BRsim(d,q) = 1 - \frac{distance(d,q)}{k}$
For the present case: $BRsim(d,q) = 1 - \frac{0.5}{2} = 0.75$

3.2 Representations as general DNF formulas

In this section we consider representations for queries and documents that have conjunctions and disjunctions. Specifically, documents and queries are now represented as generic DNF formulas. This means that the indexing process has to store documents in this form. However, users do not have to articulate their information needs in DNF, but the traslation can be done automatically by the system. This is possible because any propositional formula can be translated into its DNF equivalent. In this case the similarity measure is not equivalent to a classical measure because classical systems do not consider DNF representations for documents. As the measure subsumed the inner product query-document matching function, we think $BRsim$ keeps being appropriate for IR when applied to this general case. Intuitively, the distance from a model of the document to the query is the distance from the model of the document to the nearest clause of the query. This corresponds with the semantics of the disjunction. In fact, given a disjunctive query, the satisfaction of one of the choices expressed in the disjunction is enough to consider a document as relevant, no matter how far to the document are the rest of the choices.

This paragraph explains some symbols used in the algorithm. The size of the alphabet is denoted by S, $S = |\mathcal{P}|$. Given an interpretation m, $LIT(m)$ is the transformation of m into a set of literals, i.e. $LIT(m) = m \cup \{\neg l | l \in \mathcal{P} \setminus m\}$. The symbols ψ_{min} and ψ_{max} (μ_{max}) are the size of the smallest and the greatest clause in ψ (μ), respectively.

Algorithm 2:
Procedure Similarity(ψ,μ)
Input:query $\psi = \{\psi_1, \psi_2, \ldots\}$
document $\mu = \{\mu_1, \mu_2, \ldots\}$
Output:BRsim(ψ,μ)

1. $Distance = 0; Total\_Models = 0;$
2. Compute the set of models of μ
3. Extract a new m, model of μ
4. $Distance\_to\_\psi = S$
5. Extract a new $\psi_i \in \psi$
6. Compute $CDist(\psi_i, LIT(m))$
7. If $CDist(\psi_i, LIT(m)) < Distance\_to\_\psi$ then $Distance\_to\_\psi = CDist(\psi_i, LIT(m))$
8. Go to step 5 until no more ψ_is remain
9. $Total\_Models++$; $Distance+ = Distance\_to\_\psi$
10. Go to step 3 until no more μ models remain
11. $distance(d,q) = \frac{Distance}{Total\_Models}$
12. Return($1 - \frac{distance(d,q)}{\psi_{min}}$)

Roughly speaking, the algorithm extracts each model of the document, computes the distance to the query ψ, the distances are accumulated and, in the end, the average of the distances to the query is obtained dividing by the total number of models of the document. In order to compute the distance from each model of the document to the query, the model is firstly transformed into a set of literals and then, this set is used to match against each conjunction of the query. The least distance is the distance from the model to the query and is stored in $Distance\_to\_\psi$. Note that $Distance\_to\_\psi$ is initialized to the size of the propositional alphabet, which is an upper bound for distances to the query. Last step produces the similarity measure in the interval $[0, 1]$. The upper bound of the distance is ψ_{min}, the size of the smallest ψ_i, because the distance from a model m of the document to the set of models of the query ψ takes the distance from m to the closest query model. For any clause ψ_i of ψ, its closest model to m fares at most the size of the clause ψ_i.

The algorithm needs at most $2^S|\mu|\mu_{max}$ iterations in order to compute the set of models of the document (step 2). The loop from step 3 to 10 is executed at most 2^S times. The loop from step 5 to 8 takes $|\psi|$ iterations. The computation of $CDist$ in step 6 has a worst case complexity of ψ_{max}. As a result, in the worst case the algorithm takes $2^S|\mu|\mu_{max} + 2^S|\psi|\psi_{max}$ steps. Although exponential in S, an important point is that the size of the alphabet does not depend on the input to the algorithm. Thus, S is fixed and the value 2^S is bounded by a constant.

This algorithm constitutes a substantial improvement respect to the models method because it does not compute any model of the query. Queries use to have very few terms and, therefore, a lot of models. Then, not computing query models seems to be a great benefit. Even though models of the document are needed, documents are expected to have few models.

3.3 Experiments

Despite having improved the complexity results with respect to the models method, we had the intuition that, in particular for Algorithm 2, the result was not enough. Therefore, we decided to accomplish some informal experiments in order to clear up how far we could apply our algorithms when dealing with realistic vocabulary sizes. We wrote C code implementing Algorithm 1, Algorithm 2 and the models method. Our tests on models method clearly showed that it cannot work in actual systems. With alphabet sizes of more than 15 terms the algorithm took an extremely long time to compute similarity. On the contrary, Algorithm 1 presented a great performance. We tried out sizes up to 500 terms and the response time was of the order of microseconds. In fact, the response time grew with the size of the query but it kept on the same levels when changing the alphabet size. Then, Algorithm 1 could be used with alphabet sizes much greater than 500 and, as a result, it could stand on the basis of a realistic system. However the experiments we ran with Algorithm 2 were rather disappointing. Despite improving models method's results, Algorithm 2 could not be used for large-scale systems. The response time was reasonably low only for alphabet sizes less than 50 terms. In conclusion, some kind of adjustment is needed in order to be able to compute similarity between general DNF formulas. Next section presents our proposal to overcome this problem.

3.4 Reducing Complexity

The main source of complexity stands on the definition of the similarity measure $BRsim$ itself. The similarity measure is based on a distance that is an average of distances from models:

$$distance(d, q) = \frac{\sum_{m \in Mod(d)} dist(Mod(q), m)}{|Mod(d)|}$$

When both document and query have not disjunctions, their representations have only one clause that can be used to represent the whole set of models. This is what Algorithm 1 does. It manages to obtain the average of the set of models of the document without computing any model. This way exponentiality is avoided. However this technique cannot be used when DNF formulas have more than one clause. In that case, we cannot get the average of the whole set of models of the document from computations between clauses. The main reason is that clauses can have common models and, thus, working only with the clauses of the document we would count more than once the common models. As a result, Algorithm 2 needs to compute the models of the document in order to obtain $BRsim$.

We propose to define a clause-based similarity measure, instead of a model-based one. Given $\psi = \{\psi_1, \psi_2, \ldots\}$ and $\mu = \{\mu_1, \mu_2, \ldots\}$ the DNF representations of a query and a document respectively, we propose to use the similarity measure $Csim$, which is defined as follows:

$$Cdistance(\mu,\psi) = \frac{\sum_{\mu_j \in \mu} min_{\psi_i \in \psi}(CDist(\psi_i,\mu_j) + \frac{|\psi_i \setminus \psi_i \cap \mu_j| - CDist(\psi_i,\mu_j)}{2})}{|\mu|}$$

$$Csim(\mu,\psi) = 1 - \frac{Cdistance(\mu,\psi)}{\psi_{min}}$$

Last step is the regular normalization that was already done for $BRsim$. In order to obtain the distance from the document to the query we use an average over the clauses of the document. The distance from an individual clause to the query is the distance to the nearest query clause. The distance from a clause to another clause is computed as in Algorithm 1, i.e. it is the number of differing terms ($CDist(\psi_j,\mu_i)$) plus half the query terms not mentioned by the clause of the document. Note that we have tried to keep the measure as close as possible to Dalal's semantics (the minimum of the distances to the clauses of the query is taken). In fact, $Csim(d,q) = BRsim(d,q)$ when the formulas involved are DNF with only one clause. We have proposed to use an average over the document's clauses because we wanted to maintain $BRsim$'s fairness ($BRsim$ makes the average of the models). However, other choices could have been taken. For instance, we could select the distance from the closest clause. Anyway, we think these alternatives measures have to be tested against a document collection in order to see how they behave with respect to precision and recall. Our aim in this work is to show that a measure over DNF formulas can be efficiently computed. Next lines show the algorithm we have designed to compute $Csim$. Algorithm 1 is a particular case of this new algorithm.

Algorithm 3:

Procedure Similarity(ψ,μ)
  Input:query $\psi = \{\psi_1, \psi_2, \ldots\}$
    document $\mu = \{\mu_1, \mu_2, \ldots\}$
    Output:Csim(ψ,μ)

1. $Distance = 0$;
2. For each μ_j in μ
3. $Distance\_to\_\psi = S$
4. Extract a new $\psi_i \in \psi$
5. $d = CDist(\psi_i,\mu_j) + \frac{|\psi_i \setminus \psi_i \cap \mu_j| - CDist(\psi_i,\mu_j)}{2}$
6. If $d < Distance\_to\_\psi$ then $Distance\_to\_\psi = d$
7. Go to step 4 until no more ψ_is remain
8. $Distance{+} = Distance\_to\_\psi$
9. Go to step 2 until no more μ_js remain
10. $distance(d,q) = \frac{Distance}{|\mu|}$
11. Return($1 - \frac{distance(d,q)}{\psi_{min}}$)

Given the previous formulas, the algorithm is straightforward. Each clause μ_j of the document is extracted and the distance from μ_j to the query is computed as the distance to the closest query clause (and stored in $Distance\_to\_\psi$). Final steps do the average and the normalization. The loop from step 2 to step 9 takes $|\mu|$ iterations and the loop from step 4 to step 7 takes $|\psi|$ iterations. The computation of step 5 can be done in linear time with respect to the size of any of the clauses involved. As queries are expected to have less terms, the most efficient implementation of step 5 has a worst case of ψ_{max} (size of the greatest clause in ψ) iterations. As a result, the algorithm has a complexity of $\mathcal{O}\ (|\mu||\psi|\psi_{max})$. The good complexity result of this algorithm favours its use for large amounts of data. Moreover, we think the measure is appropriate for IR as it subsumes the inner product query-document similarity measure.

3.5 More experiments

Additional experiments were carried out with Algorithm 3. We developed procedures written in the C programming language implementing Algorithm 3 and the tests were run in the same conditions as in the previous experiments. We tried out vocabulary sizes up to 500 terms and the response time was of the order of microseconds. Consequently, the performance of Algorithm 3 (and Algorithm 1, which is a case of Algorithm 3) is promising for applying it within actual systems.

4 Conclusions and Future Work

In this work we have developed efficient procedures for computing similarity between documents and queries expressed as DNF formulas. The theory that underlies is the Belief Revision framework, where a similarity measure, $BRsim$, can be obtained. However the direct application of BR techniques leads up to exponential algorithms. Therefore, we have proposed to use a syntactic characterization of the formulas involved to be able to build efficient algorithms. The algorithms have been presented in increasing order of expressiveness. Algorithm 1, which deals with conjunctive representations, is very efficient and, despite its simplicity, it is expressive enough to represent classical vectors with binary weights. Moreover, it can express partial vectors. The second algorithm accepts generic DNF formulas but, although it is better than the direct application of the BR techniques, its complexity would still not allow us to use it within actual systems. Therefore, we propose to use a different similarity measure, $Csim$, that is clause-based instead of model-based. We have tried to keep $Csim$ as close as possible to $BRsim$ and, in fact, for conjunctions of terms $Csim$ also subsumes the inner product query-document matching function. Then, we have developed an efficient algorithm for computing $Csim$ that ensures its applicability in realistic environments.

An important issue stands on the fact that document representations are more expressive than classical ones and, thus, a system with such an expressive capability seems to have chance to improve performance. In fact, more expressive document representations have been recently claimed for advanced applications [9] and for image retrieval [10]. Representing a document with a DNF formula gives us the possibility of expressing several views of the document. For instance, the title and the abstract could be the basis for building two different clauses of the document's representation. This representation would be more accurate because title and abstract are independent sources of the semantics of a document and to mix the keywords from both items within a vector seems to be less precise. For instance, the structure of TREC topics, containing a title, a short description and a narrative prompts to store a split representation. An important line of future work is the search for methods that get DNF representations from plain text. This goes in the line of a recent work [16] that tackles the problem of automatically build a taxonomy of concepts from a set of plain text documents. Having DNF representations for plain text documents, we will be able to test our algorithms with standard collections and investigate the behaviour of $Csim$ and other alternative measures. In this sense it is important to note that some research conducted in Passage Retrieval [11, 3] has concluded that to divide a plain text into parts improves retrieval.

Acknowledgements: This work was supported in part by project PGIDT99XI10201B from the government of Galicia, *Xunta de Galicia*, (Spain) and by project PB97-0228 from the government of Spain. We would also like to thank the Glasgow IR group for its support during the stay of the first author at the Department of Computing Science of the University of Glasgow. The first author thanks the *Ministerio of Educación y Cultura* (Spain) for its finantial support via a research stay grant which made possible his stay at Glasgow.

References

[1] C. E. Alchourron, P. Gärdenfors, and D. Makinson. On the logic of theory change: Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50:510–530, 1985.

[2] P. Bruza. Preferential models of query by navigation. In F. Crestani, M. Lalmas, and C. J. Van Rijsbergen, editors, *Information Retrieval, Uncertainty and Logics:advanced models for the representation and retrieval of information*, Norwell, MA, 1998. Kluwer Academic Publishers.

[3] J.P. Callan. Passage-level evidence in document retrieval. In *Proc. of SIGIR-94, the 17th ACM Conference on Research and Development in Information Retrieval*, pages 302–310, Dublin, July 1994.

[4] F. Crestani and C. J. Van Rijsbergen. Information retrieval by logical imaging. *Journal of Documentation*, 51(1):3–17, 1995.

[5] M. Dalal. Investigations into a theory of knowledge base revision: Preliminary report. In *Proc. of AAAI-88, the 7th National Conference on Artificial Intelligence*, pages 475–479, 1988.

[6] A. del Val. *Belief Revision and Update*. PhD thesis, Stanford University, 1993.

[7] A. del Val. Syntactic characterizations of belief change operators. In *Proc. of IJCAI'93, the Thirteenth International Joint Conference on Artificial Intelligence*, pages 540–545, Chambery, France, 1993.

[8] N. Fuhr. Probabilistic Datalog - a logic for powerful retrieval methods. In *Proc. of SIGIR-95, the 18th ACM Conference on Research and Development in Information Retrieval*, pages 282–290, Seattle, Washington, 1995.

[9] N. Fuhr. Probabilistic Datalog: Implementing logical information retrieval for advanced applications. *Journal of the American Society for Information Science*, 51(2):95–110, 2000.

[10] C. Globe and S. Bechhofer. "Fetch me a picture representing triumph or similar": Classification based navigation and retrieval for picture archives. In *Proc. of Seminar on Searching for Information: Artificial Intelligence and Information Retrieval Approaches*, pages 4/1–4/4, Glasgow, UK, 1999.

[11] M. Hearst and C. Plaunt. Subtopic structuring for full-length document access. In *Proc. of SIGIR-93, the 16th ACM Conference on Research and Development in Information Retrieval*, pages 59–68, Pittsburgh, June 1993.

[12] H. Katsuno and A. O. Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52:263–294, 1991.

[13] D. E. Losada and A. Barreiro. Using a belief revision operator for document ranking in extended boolean models. In *Proc. of SIGIR-99, the 22th ACM Conference on Research and Development in Information Retrieval*, pages 66–73, Berkeley, California, August 1999.

[14] C. Meghini, F. Sebastiani, U. Straccia, and C. Thanos. A model of information retrieval based on a terminological logic. In *Proc. of SIGIR-93, the 16th ACM Conference on Research and Development in Information Retrieval*, pages 298–307, Pittsburgh, PA, 1993.

[15] J. Y. Nie. An information retrieval model based on modal logic. *Information Processing & Management*, 25(5):477–491, 1989.

[16] M. Sanderson and B. Croft. Deriving concept hierarchies from text. In *Proc. of SIGIR-99, the 22nd ACM Conference on Research and Development in Information Retrieval*, pages 206–213, Berkeley, August 1999.

[17] C.J. Van Rijsbergen. A non-classical logic for information retrieval. *The Computer Journal*, 29:481–485, 1986.

[18] C.J. Van Rijsbergen and M. Lalmas. An information calculus for information retrieval. *Journal of the American Society for Information Science*, 47(5):385–398, 1996.

[19] M-A. Williams. Aplications of belief revision. In *Lecture Notes in Artificial Intelligence 1472*, pages 285–314. Springer Verlag, 1998.