Information Processing and Management 46 (2010) 514–532

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Managing structured queries in probabilistic XML retrieval systems

Luis M. de Campos *, Juan M. Fernández-Luna, Juan F. Huete, Carlos Martín-Dancausa

Departamento de Ciencias de la Computación e Inteligencia Artificial, E.T.S.I. Informática y de Telecomunicación, Universidad de Granada, 18071 Granada, Spain

ARTICLE INFO

Article history:
Received 29 October 2008
Received in revised form 23 November 2009
Accepted 27 November 2009
Available online 29 December 2009

Keywords:
Structured queries
XML
Structured document retrieval

ABSTRACT

Focusing on the context of XML retrieval, in this paper we propose a general methodology for managing structured queries (involving both content and structure) within any given structured probabilistic information retrieval system which is able to compute posterior probabilities of relevance for structural components given a non-structured query (involving only query terms but not structural restrictions). We have tested our proposal using two specific information retrieval systems (Garnata and PF/Tijah), and the structured document collections from the last six editions of the INitiative for the Evaluation of XML Retrieval (INEX).

1. Introduction

New standards in document representation have caused Information Retrieval (IR) to design and implement models and tools to index, retrieve and present documents according to the given document structure. While standard IR treats documents as if they were atomic entities, modern IR needs to be able to deal with more elaborate document representations, such as for example documents written in SGML, HTML or XML. These document representation formalisms enable *structured* documents (i.e. documents whose content is organised around a well-defined structure) to be represented and described. Examples of these documents are books and textbooks, scientific articles, technical manuals, etc. This means that documents should no longer be considered as atomic entities, but as aggregates of interrelated units that need to be indexed, retrieved, and presented both as a whole and separately, in relation to the user's needs. In other words, given a query, an Information Retrieval System (IRS) must retrieve the set of document components or structural units that are most relevant to this query, not just entire documents.

Within IR, the area of research dealing with structured documents is known as *structured document retrieval*. A good overview of structured document retrieval can be found in Chiaramella (2001) and Lalmas (2009). The inclusion of the structure of a document in the indexing and retrieval process affects the design and implementation of the IR system in many ways. First of all, the indexing process must consider the structure in an appropriate way so that users can search the collection both by content and structure. Secondly, the retrieval process should use both structure and content when estimating the relevance of documents. Finally, the interface and the whole interaction must enable the user to make full use of the document structure. In fact, querying by content and structure can only be achieved if the user can specify in the query *what* he or she is looking for, and *where* this should be located in the required documents. The "what" involves the specification of the content, while the "where" is related to the structure of the documents.

There are already many IR systems which are able to deal with structured documents and the series of INEX (INitiative for the Evaluation of XML Retrieval) Workshop proceedings (Fuhr, Lalmas, & Trotman, 2008; Fuhr, Lalmas, & Trotman, 2007;

* Corresponding author. Fax: +34 958243317.
E-mail addresses: lci@decsai.ugr.es (L.M. de Campos), jmfluna@decsai.ugr.es (J.M. Fernández-Luna), jhg@decsai.ugr.es (J.F. Huete), cmdanca@decsai.ugr.es (C. Martín-Dancausa).

0306-4573/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ipm.2009.11.006

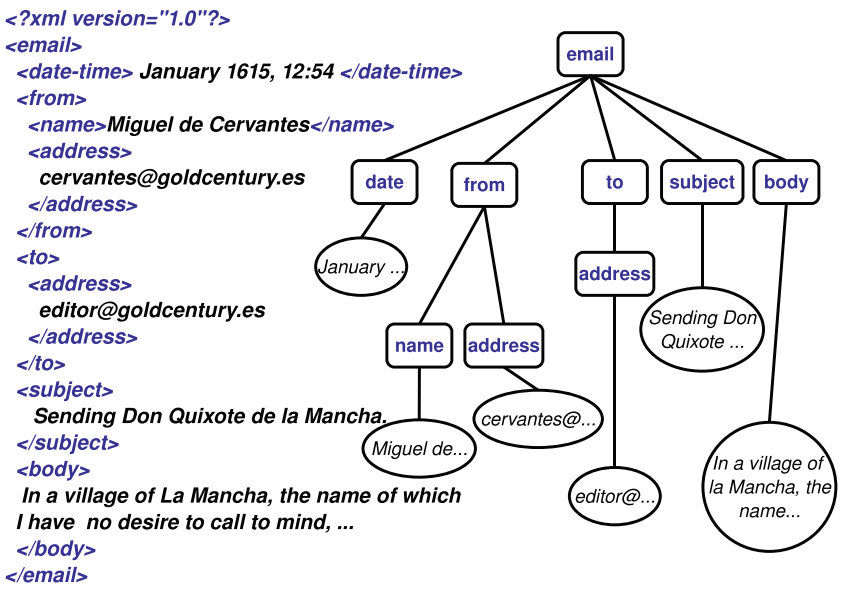

Fig. 1. Example of XML document and its associated tree.

Fuhr, Lalmas, Malik, & Kazai, 2006; Fuhr, Lalmas, Malik, & Szlávik, 2005; Fuhr, Malik, & Lalmas, 2003; Fuhr, Gövert, Kazai, & Lalmas, 2002) is an excellent source of information. It is, however, also true that in many cases these systems take as input non-structured queries involving only content (the so-called *content-only*, CO, queries),[1] although their output may be any kind of document component.

What we propose in this paper is a general methodology to convert some of these systems into fully structured IR systems which are able to process structured queries involving both content and structure (the so-called CAS queries). Our starting point, therefore, is any information retrieval system which is capable of computing (given a non-structured query) a relevance value for each structural unit in a document, representing the posterior probability of relevance of that unit given the query.[2]

In Section 2 of this paper, we explain the kind of structured query being dealt with. Section 3 describes our proposal for transforming a partially structured IRS into a fully structured one, by allowing the system to manage structured queries. Section 4 reviews some of the existing approaches for managing content and structure queries, whereas the experimental evaluation of our proposal is explained in Section 5. Finally, Section 6 contains our concluding remarks and various proposals for future research.

2. XML and structured queries

The *eXtensible Markup Language*, known as *XML* (W3C, 2006), is a simple language which was initially designed to meet the challenges of electronic publishing. Nowadays, it also plays an increasingly relevant role in the exchange of a wide range of data via Internet. XML is a simplified version of the *Standard Generalized Markup Language, SGML*, which was originally designed for document definition, but too complex for the Web.

XML is in fact a meta-language as its main objective is to facilitate the creation of markup languages to represent structured data[3] (this is one of the main differences with HTML, as XML is used to describe rather than show data, as HTML does).

The sets of rules which define a new language are usually expressed in a DTD (Document Type Definition) or XML Schema file. These files describe the elements comprising the documents derived from that DTD or Schema as well as their relationships, i.e. the internal structure of the documents.

Broadly speaking, an XML document could be viewed as an ordered, labeled tree. Each internal node of the tree is an element, including the root node, which may contain other element nodes or text nodes. The leaf nodes contains text associated to other elements. Each internal node in the conceptual tree is represented in the XML file with an opening tag plus a closing tag. Fig. 1 shows an e-mail represented in XML and its associated tree (where rectangles are tags and ellipses are texts). This tree-type representation is called *Document Object Model, DOM*, and offers a way of accessing and processing XML documents.

[1] Queries made of terms or words and not containing any reference or restriction relative to structural elements in the document collection.

[2] The restriction to probabilistic systems is more formal than practical, because the proposed method will work with any system returning relevance values in the [0,1] interval. Otherwise, any kind of normalization could be used.

[3] Some examples of languages generated from XML are DocBook and MathML, among others.

In addition to being a markup language, XML is also a technological framework where different technologies are developed with the aim of offering useful services for the most frequent user's requests. This is the case of *XML Path Language, XPath* (W3C, 2007): a language aimed to access parts of an XML document. XPath allows the selection and reference of texts and elements in an XML file.

For the purpose of this paper, we will review only the features of XPath relevant to our work. Therefore, a name of an element, (*A*), selects all elements with that name (for example, *address*, in Fig. 1, would select two nodes of the tree). A slash is used to select child nodes in the tree (*A*/*B*, where *B* is a direct descendant of *A*): *from/address* in the example. A double slash means that any number of elements could be included in the path (*A*//*B*, where *B* is a descendant, though not necessarily a direct one of *A*). For instance, *email//address* would select those units *address* which are directly or indirectly contained in an *email* element, i.e. with an arbitrary number of elements in between: *email/from/address* and *email/to/address*. A slash at the beginning of the expression means that the path starts at the root element (/*A*). An asterisk * selects all the elements placed in the path after it (/*A*//*B*//*) (for example, */email//** would select all the descendants of *email*). Finally, a pair of opening and closing brackets and a number between them, after an element, establishes the order of the element as a child from left to right: //*A*/*B* [3] (Azevedo, Pantuza, & Ziviani, 2005) means that the XPath expression engine would select the third *B* element child of *A*.

At this point, we must distinguish between the two views of the XML content (Manning, Raghavan, & Schütze, 2008): data- and text-centric XML. On the one hand, a data-centric XML document is seen as a container for data, usually non-text data (worker profiles, invoices, flight schedules, etc.). The document meaning depends on the structured data contained, and presents a regular and complex structure and homogeneous content. When the user formulates queries for such a type of XML document, he or she is interested in an exact match, such as a database-type style. On the other hand, text-centric XML documents usually represent text documents (books, e-mails, etc.), and the structure is more irregular and the data heterogeneous. The queries, although taking the structure into account, focus on the text and require ranking. The aim of text-centric XML retrieval is therefore to develop methods to find correspondence between the text of the query and the text of the XML documents (but also considering structural restrictions) and this is the context in which this paper is set.

In order to allow queries combining content and structure (Content And Structure Queries, CAS, in INEX terminology) to be specified, the NEXI language (Trotman & Sigurbjörnsson, 2005) was designed. It is a simplified XPath containing only the descendant operator (//) in a tag path and also an extended XPath containing the *about* function. NEXI has been used by INEX since 2004.

The kind of structured CAS query considered by NEXI can take two possible forms:

- //*C*[*D*]: Returns *C* units that fulfill the condition *D*.
- //*A*[*B*]//*C*[*D*]: Returns *C* descendants of *A* where *A* fulfills the condition *B* and *C* fulfills the condition *D*.

A and *C* are *paths* (sequences of elements or structural units), specifying structural restrictions, whereas *B* and *D* are *filters*, which specify content restrictions, and // is the descendant operator. *C* is the *target* path (the last structural unit in *C* is the one that we want to retrieve) and path *A* is the *context*. Each content restriction will include one or several *about* clauses, connected by either *and* or *or* operators; each *about* clause contains a text (a sequence of words or terms) together with a relative path, from the structural unit which is the container of the clause to the structural unit contained in it where this text should be located. The *about* clause is the IR counterpart of the classical *contains* clause used in XPath (which requires an exact matching between the textual content of the clause and a part of the text in the structural element being evaluated). However, *about* does not demand such a strict matching but states, vaguely, that a relevant element should satisfy the information need expressed by means of the text contained in the clause.

Example 1. Let us suppose that the hierarchical structure (e.g. the XML tree) of a document collection is the one displayed in Fig. 2. An example of a NEXI-structured query is the following:

$$//\mathrm{A}[\mathrm{about}(.,\mathrm{text2})\ \mathrm{and}\ \mathrm{about}(.//F,\mathrm{text3})\ \mathrm{and}\ \mathrm{about}(.//\mathrm{J},\mathrm{text4})]//\mathrm{D}[\mathrm{about}(.,\mathrm{text1})\ \mathrm{and}\ \mathrm{about}(.//\mathrm{N},\mathrm{text5})]$$

What we want to retrieve with this query are D units which are contained within A units. The target D units should speak about *text*1 and contain an N unit speaking about *text*5; the context A units should be about *text*2 and also contain F and J units dealing with *text*3 and *text*4, respectively (see Fig. 3).

3. Managing structured queries

In this section, we will explain how a structured CAS query of the type considered in Section 2 can be managed using an IR system able to process only CO queries.

Each about clause which is part of the structured query will give rise to a (non-structured) subquery. This subquery will be used to compute the posterior probabilities of the structural units specified within the clause, by postprocessing the output of the base system (removing the units which do not satisfy the structural conditions specified in the about clause and in the path).[4]

[4] This could also be done by modifying the base system, in order to compute *only* the probabilities of the structural units satisfying these structural conditions. This requires a more complex interaction with the base system but would possibly result in greater efficiency.

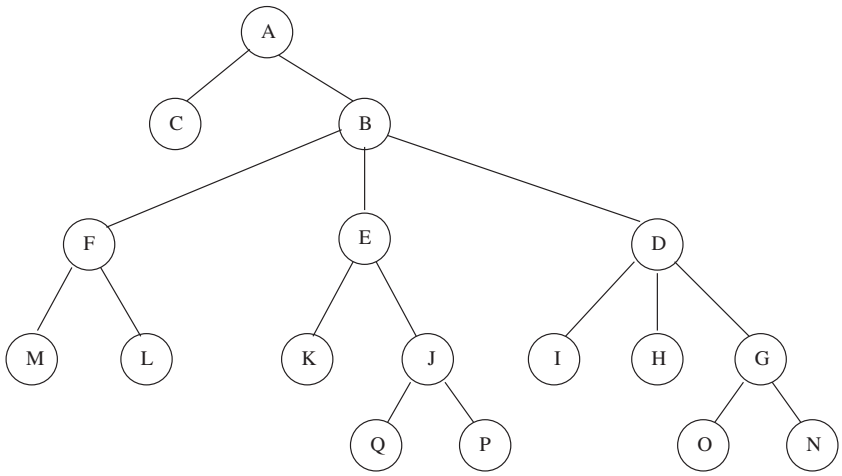

Fig. 2. Tree structure of the documents for Example 1.

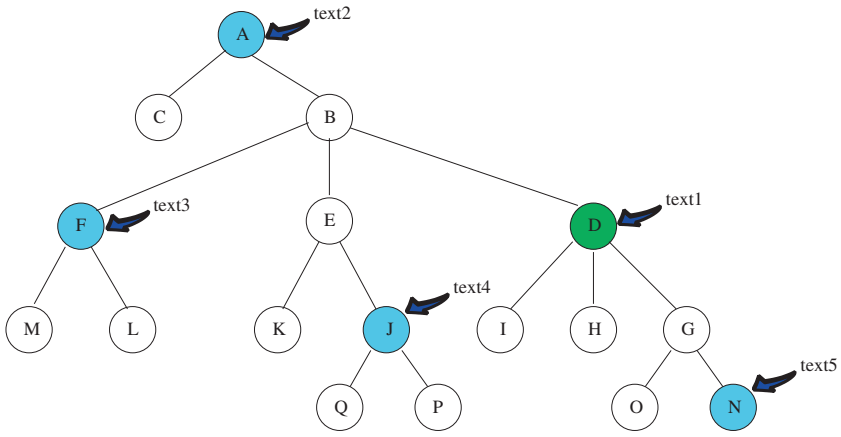

Fig. 3. Structural units in the hierarchy where some pieces of text have been specified for Example 1.

Fig. 4 shows the modules required for a probabilistic retrieval system in order to deal with structured queries. A first module, a *NEXI Query Processor*, is shown and this extracts the content subqueries from the NEXI query (those which occur in the about clauses). These are passed to the XML retrieval system. It then runs a retrieval and obtains a ranking of relevant elements for each subquery. With these rankings plus the original NEXI query, the last module, the *Output Processor*, filters the results that do not satisfy the structural restrictions for each subquery, selects the objective elements and computes an RSV (relevance status value) for each element, returning a sorted list of units satisfying the initial query.

Let us consider the following example (simple query of type //*A*[*B*]):

Example 2.

$$Q = //\text{chapter}[\text{about}(.//\text{title}, \text{text1}) \text{ and } \text{about}(.//\text{section}, \text{text2})]$$

This query, in the context of a book collection, searches for chapter units with a title about *text*1 and containing a section about *text*2. This is an example of the first, simpler type of query supported by NEXI.

In this example, the posterior probabilities computed for the subquery *text1* only focus on the title units which are contained in chapter units; the posterior probabilities for the subquery *text2* are also restricted to those section units contained within chapter units. In this way, structural units associated to paths such as, for example, */book/chapter/section*, */book/chapter/section/paragraph* or */book/chapter/author* would be discarded for the first subquery, whereas structural units associated to paths such as */book/chapter/section/paragraph*, */book/chapter/title* or */book/chapter/author* would also be eliminated for the second subquery.

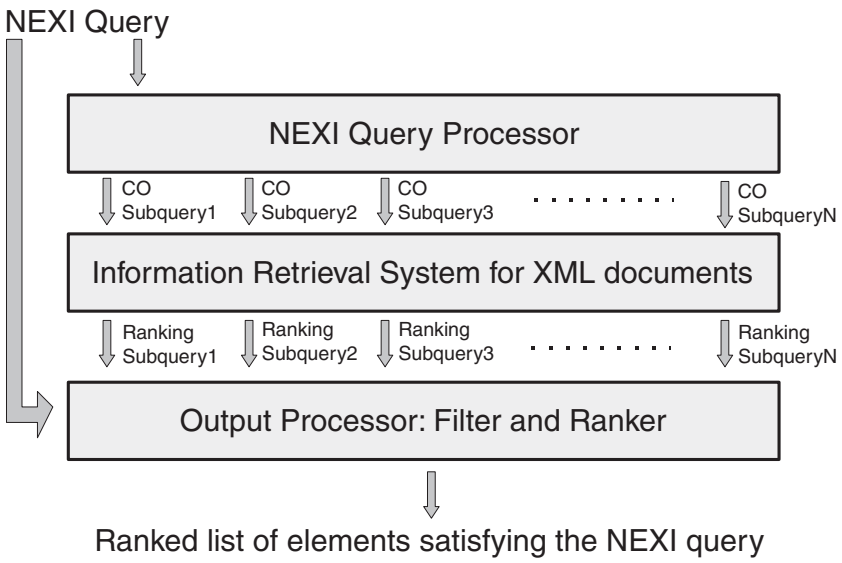

Fig. 4. Architecture of a system for managing CAS queries.

Once we have discarded these units (thus keeping only the probabilities $p((//chapter//title)_l|text1)$[5] and $p((//chapter//section)_l|text2)$ in the example), we must compute the probability of the structural unit containing the about clause(s) (in the example this is the probability of the chapter units, $p((//chapter)_l|Q)$). This requires:

1. computing the aggregated probability of all the structural units of the same type associated to the corresponding unit (in the case of the container unit having more than one of these units) and then;
2. combining these aggregated probabilities for the different structural units specified in the different about clauses.

In the example, we need to compute the aggregated probability of the section units (step 1), $p((//chapter)_l//section|text2)$ (because, in this case, a chapter may contain more than one section), whereas the probability of the title unit, $p((//chapter)_l//title|text1)$, does not change, because a chapter will only have a single title.[6] The combined probability of section units and title units will then be associated to their container chapter units (step 2), $p((//chapter)_l|Q)$.

In order to satisfy the about clauses, it is sufficient for one section to be relevant (and also the corresponding title of the chapter), and it is not necessary for many or all of the sections of a chapter to be relevant. Therefore, to aggregate the probabilities of the structural units of the same type (when the container unit can have more than one of these units), it is appropriate to use a combination model representing a disjunction. However, to combine the probabilities of different types of units inside the container unit, this will depend on the type of connective being considered, either conjunction ("and") or disjunction ("or") (it will be a conjunction in the example). If a given chapter, *i*, contains exactly three sections and one title, the probability of the chapter would be obtained as

$$\begin{aligned} p((//chapter)_i|Q) &= p((//chapter)_i//title|text1) \text{ AND } p((//chapter)_i//section|text2) \\ &= p(((//chapter)_i//title)_1|text1) \text{ AND} \\ &\quad (p(((//chapter)_i//section)_1|text2) \text{ OR} \\ &\quad p(((//chapter)_i//section)_2|text2) \text{ OR} \\ &\quad p(((//chapter)_i//section)_3|text2)) \end{aligned}$$

where AND and OR should be understood as operators for aggregating probabilities (instead of the boolean operators).

This process will compute the posterior probabilities of the structural units containing the about clauses specified in the query, for both the context (*A*) and the target (*C*), but these probabilities must also be combined to obtain the final probabilities of the target structural units. In the previous example this is not necessary because we only specified a target unit and not any context for this unit. Let us, however, examine another, more complex example (query of type //*A*[*B*]//*C*[*D*]):

Example 3.

$$Q = //\text{chapter}[\text{about}(.//\text{title}, \text{text1}) \text{ and } \text{about}(.//\text{section}, \text{text2})]//\text{bibliography}[\text{about}(., \text{text3})]$$

[5] In this context, an XPath reference such as *//chapter//title* represents the set of all the structural units compatible with this reference, and $(//chapter//title)_l$ denotes an element in this set.

[6] Although we would also need to aggregate the probabilities if section units may also contain title units, because the path/ chapter/section/title also matches //chapter//title.

This query attempts to retrieve bibliography units containing *text*3, which must be in chapter units with a *text*1-related title and contain a section about *text*2. The bibliography units are the target and the chapter units are the context.

In this case, in addition to computing the posterior probabilities of chapter units, $p((//chapter)_l|B)$ as in the previous example, we would similarly compute the posterior probabilities of bibliography units in chapter units given the query $D = text3$, $p((//chapter//bibliography)_l|D)$. It is then necessary to combine $p((//chapter)_l|B)$ with $p((//chapter//bibliography)_l|D)$ to obtain the final relevance degree of each bibliography unit in relation to the structured query, $p((//chapter//bibliography)_l|Q)$. This combination clearly must also be conjunctive.

3.1. Combining probabilities

In the general case, let us consider a query Q = //A[B]//C[D], i.e. $Q = Q_A$ AND Q_C, where $Q_A = //A[B],\ Q_C = //A//C[D]$, and

$$B = [\text{about}(.//A_1, \text{text}_{A1}) \text{ and/or} \ldots \text{and/or about}(.//A_n, \text{text}_{An})]$$
$$D = [\text{about}(.//C_1, \text{text}_{C1}) \text{ and/or} \ldots \text{and/or about}(.//C_m, \text{text}_{Cm})]$$

Since this query gives rise to $n + m$ subqueries, we use the base IRS to compute the probabilities $p(.|\text{text}_{Ai})$ and $p(.|\text{text}_{Cj}),\ i = 1,\ldots,n,\ j = 1,\ldots,m$, for all the structural units.

First, we filter the results of these queries to focus on the structural units of interest, namely $p((//A//A_i)_l|\text{text}_{Ai})$ for each about clause in the context and $p((//A//C//C_j)_l|\text{text}_{Cj})$ for each about clause in the target.

Secondly, these probabilities are used to obtain the aggregated probabilities of the structural units of the same type, $p((//A)_l//A_i|\text{text}_{Ai})$ for the context and $p((((//A)_l//C)_h//C_j|\text{text}_{Cj})$ for the target, which are computed as

$$\begin{aligned} p((//A)_l//A_i|\text{text}_{Ai}) &= OR_k(p((((//A)_l//A_i)_k|\text{text}_{Ai})) \\ p((((//A)_l//C)_h//C_j|\text{text}_{Cj}) &= OR_k(p((((((//A)_l//C)_h//C_j)_k|\text{text}_{Cj})) \end{aligned} \tag{1}$$

Thirdly, we combine these aggregated probabilities for the different structural units specified in the different about clauses, in order to obtain the probabilities for the content and the target, $p((//A)_l|Q_A)$ and $p((((//A)_l//C)_h|Q_C)$, respectively:

$$\begin{aligned} p((//A)_l|Q_A) &= \text{OP}_{i=1}^{n}(p((//A)_l//A_i|\text{text}_{Ai})) \\ p((((//A)_l//C)_h|Q_C) &= \text{OP}_{j=1}^{m}(p((((//A)_l//C)_h//C_j|\text{text}_{Cj})) \end{aligned} \tag{2}$$

where the operator OP may be either conjunction (AND) or disjunction (OR).

The final probability of the desired structural units, $p((((//A)_l//C)_h|Q)$, will be obtained as

$$p((((//A)_l//C)_h|Q) = \text{AND}(p((//A)_l|Q_A), p((((//A)_l//C)_h|Q_C)) \tag{3}$$

3.2. Probabilities of disjunctions and conjunctions

In order to manage disjunctive and conjunctive aggregations in a probabilistic setting, a simple and reasonable option is to use (noisy) OR and (noisy) AND gates. Let $U_1,\ldots,U_n$ be a set of structural units, and we want to compute the disjunction $\text{OR}_{i=1}^{n}(p(U_i))$ and the conjunction $\text{AND}_{i=1}^{n}(p(U_i))$.

3.2.1. Noisy-OR gates

A noisy OR gate is defined by means of (Pearl, 1988):

$$\text{OR}_{i=1}^{n}(p(U_i)) = 1 - \prod_{i=1}^{n}(1 - w_o(U_i)p(U_i)), \tag{4}$$

where $w_o(U_i)$, with $0 \leqslant w_o(U_i) \leqslant 1$, is a weight representing the probability that the disjunction is true if U_i alone is true, the other units $U_j,\ j \neq i$, being false. If $\forall\ i\ w_o(U_i) = 1$, we have a pure (non-noisy) OR gate, which in this case is equivalent to using the probabilistic sum *t*-conorm (Schweizer & Sklar, 1983).

3.2.2. Noisy-AND gates

A noisy AND gate is defined by means of (Pearl, 1988):

$$\text{AND}_{i=1}^{n}(p(U_i)) = \prod_{i=1}^{n}(1 - w_a(U_i)(1 - p(U_i))) \tag{5}$$

where $w_a(U_i)$, with $0 \leqslant w_a(U_i) \leqslant 1$, in this case is the probability that the conjunction is false if U_i is false and all the other units $U_j,\ j \neq i$, are true. As before, if $\forall\ i\ w_a(U_i) = 1$, we have a pure (non-noisy) AND gate and the probability of the conjunction may be reduced to the product of the individual probabilities (which is equivalent to using the product *t*-norm, Schweizer & Sklar, 1983).

3.3. Implementation details

The previous proposed methodology has been implemented by means of several programs which are external to the base XML information retrieval system being considered.

The first program, the query processor, parses the NEXI query and extracts the different content-only subqueries. These are passed to the XML retrieval system, which runs a retrieval for each subquery and obtains a ranking of relevant elements. These ranked outputs are stored in (text) files, containing the document identifier, the XPath and the RSV of each element.

These files are then processed by another program, the output processor, which in turn calls (according to the structure of the original query) three filters implemented as perl scripts.

The first script removes the results that do not satisfy the structural restrictions for each subquery and computes the aggregated probabilities of the structural units of the same type, using an OR gate in Eq. (1); this filter is executed once for each subquery. A second script then takes the output files of the first script and computes the probabilities of the context and the target, using either an AND gate or an OR gate in Eq. (2); this filter is executed twice, once for the target and once for the context. The third script computes the final probabilities by combining the probabilities of the context and the target by using an AND gate in Eq. (3).

4. Related work

In this section, we would like to review various techniques for answering CAS queries found in the specific literature, mainly publications derived from the INEX workshops. We do not wish to make an exhaustive analysis but merely to show the main types of CAS resolution methods and, for those techniques more similar to our proposal, to present the main differences with them. We should also mention that this brief study starts in 2004, the year when INEX made the decision to use NEXI as the language to specify CAS queries. In INEX 2002 and 2003, although this type of query existed in the official tasks, another specification was used.

We have observed two types of approaches in the specialized bibliography for solving structured queries: those in which the retrieval of XML elements given a CAS query is integrated in the XML retrieval model, and those in which CAS queries are managed as an additional, differentiated layer on top of the XML retrieval model. Let us start with the second group as our approach also belongs to it.

Studying several different papers describing methods where CAS queries resolution was made on top of a retrieval model, we can easily appreciate how CAS queries are solved in a similar way: content queries of the about clauses are launched to the retrieval systems. Once the rankings are generated, their elements are filtered and aggregated in some way, according to the structural restrictions. The selection of the final elements is made from those which, while satisfying the target path, are also in the ranking of elements fulfilling the context path. In some cases, the resolution method is also supported by the existence and use of several indexes containing information about different XML elements.

Some examples following the previously mentioned CAS queries resolution method are the following.

In Vittaut, Piwowarski, and Gallinari (2005), the authors design a general algebra to express NEXI queries as probabilistic events, which can then be evaluated using a Bayesian network model as the underlying XML retrieval model.

The base of the paper (Larson, 2005) is a Logistic Regression and Okapi BM-25 algorithm implementation for XML retrieval (plus a combination technique) under several indexes. CAS queries are solved by analyzing the NEXI expression and deciding which indexes to use in the search as well as restricting elements that do not fit the structural restrictions. Each content query contained in the about clauses is run and the results are combined.

Crouch, Mahajan, and Bellamkonda (2005) present an extension of the Vector Space Model (VSM) for XML documents. In this case, each document is seen as a set of subvectors, each corresponding to some specific type of element. The content queries of each about clause are run through the targeted subvector. Later, results are merged and the target elements are returned.

In Geva (2005), the content of the leaf elements is indexed and additional information is stored (paths from the roots, etc.) in a database. The score of each leaf element is computed with a very simple formula, that takes into account the terms in each about clause, and propagated through the remaining nodes in the XML documents. Elements that satisfy the context and target parts of the NEXI query are selected and their scores added. In this line of relevance propagation, and representing the XML documents as trees, in Sauvagnat and Boughanem (2005), XML retrieval is seen as a relevance propagation process in the tree starting from leaf nodes. A NEXI query is decomposed into subqueries which are then further decomposed into elementary subqueries, and these are propagated first in the tree. The relevance values obtained are aggregated to obtain scores for the subqueries, and in turn these are aggregated to obtain the score of the original query.

Based on the system presented in Geva (2005) as the underlying XML retrieval engine, in van Zwol (2006), a CAS evaluation method is shown whereby the original NEXI query is divided into the context and target subqueries: rankings are obtained for each content query from an about clause; each ranking is filtered to match the path constraints from each about clause; projections of score values to the elements at the end of the paths are computed, and the different rankings within a filter are then combined. At the end of these processes, there are two sets of elements: those satisfying the context query and those satisfying the target query. The output are only those nodes of the target query with a matching ancestor in the context query.

With a multinomial language model as a retrieval model and three indexes (structure, article level, and element level) as the physical representation, in the approach presented in Sigurbjörnsson, Kamps, and de Rijke (2004), Sigurbjörnsson,

Kamps, and de Rijke (2005), the NEXI query is decomposed into pairs (path, content query). Each subquery, constraining different parts of a document, is run and a ranking of suitable elements obtained. Finally, the rankings must be mixed in order to obtain a single sorted list of elements fulfilling the target path, with an associated score.

Our approach could be easily classified in this category since it is completely independent of the underlying XML retrieval model. We could say that the overall process of the evaluation of NEXI queries presented in this paper is similar to those presented in Sigurbjörnsson et al. (2004) and van Zwol (2006). Broadly speaking, the main differences are the combination technique and the selection of the final elements, which in our case are based on a probabilistic approach based on noisy gates. More precisely, the method considered in van Zwol (2006) uses combinatory logic and min–max normalization to combine the different rankings, and the final elements selected are only those target elements that have a matching ancestor in the set of elements obtained for the context query (whereas in our case all the preselected target elements are returned, although those ones whose ancestors do not match with elements in the context query become penalized). In the method proposed in Sigurbjörnsson et al. (2004), the mixture of the results of the subqueries also uses different operators (maximum and sum). Moreover, the context subqueries contribute to the final RSV of a target element in the same way that the target subqueries. Finally, different ways of decomposing the CAS query into a set of CO subqueries are also proposed in Sigurbjörnsson et al. (2004).

Now, let us focus on the CAS queries resolution approach where the mechanism of resolution is totally integrated in the XML retrieval model, i.e. given a query the score function usually incorporates some XML features to deal with CAS queries. Broadly speaking, in the specialized literature, we observe how modifications of the Vector Space Model and integration of XML in relational databases are the predominant research lines.

With respect to the first line, the main changes in the VSM are the incorporation of XML features to the classic cosine ranking function, the use of different indexes to store the information provided by the different XML elements and the use of additional tree representations, combining scores obtained with the VSM with tree matching. Examples of this category are the following.

In Azevedo et al. (2005), the authors modify the VSM including new XML specific features: the nesting nature of the XML elements and the fact that each element could be retrieved. The authors design a new ranking function similar to the cosine measure for flat documents. To deal with CAS queries, a factor is included to measure how the query structural constraints are satisfied by the paths of the elements. The paper (Mass & Mandelbrod, 2005) presents a model based on indexing the different types of elements in separate indexes. Based on the VSM, the query is run on each index and the different rankings are merged. For CAS queries, the content query is run against the article index to locate candidates that fulfill the query constraints. In a second step, each part of the query is submitted in parallel to each one of the remaining indexes. A relevance value is computed only for those valid elements from the first step. A third extension is found in Weigel, Schulz, and Meuss (2005). This model is adapted to represent the document structure and enhanced with a tree representation of the queries and documents. NEXI queries are solved by computing content scores using the VSM and making tree matching operations to satisfy the structural restrictions.

Various examples joining XML retrieval with relational databases may also be found. The definition of new query languages adapted to cope with structural restrictions or the mapping of structural queries to SQL are the most common approaches to solve CAS queries by means of DBMSs. A couple of representatives of these two methodologies are (Mohan & Sengupta, 2005; Shimizu, Terada, & Yoshikawa, 2007). In the former, a DBMS is adapted to work with XML documents, defining a query language, based on SQL, called DSQL to work with structured documents. Structural restrictions are very easy to deal with as they are naturally included in the DSQL language itself. In the latter (Shimizu et al., 2007), CAS queries are directly mapped to SQL queries. In addition, a score function is introduced in order to produce a ranking of relevant elements.

Other methods where relational databases are used are (Theobald, Broschart, Schenkel, Solomon, & Weikum, 2007), where index structures are implemented in a relational database, and query resolution is based on a combination of computing content and structural scores; and (Pehcevski, Thom, Tahaghoghi, & Vercoustre, 2005), as a case combining a text search engine and a native XML database.

A third main line to resolve CAS queries is based on representing the XML documents by means of trees: Arvola, Kekäläinen, and Junkkari (2006) show a method based on structural indexes (represented as trees) and element weight computations which takes into account the query and the context of each element. Element weights are calculated according to the content query from each about clause and those with weights greater than 0 are selected. Elements that do not fulfill the corresponding structural restriction are then disregarded. These two steps are repeated until the whole NEXI query is processed, applying the final target structural restrictions to the remaining elements. Another tree-related approach is that presented in Ali et al. (2007) which uses a combination of structural summaries, labeled trees representing the XML hierarchical structure of documents and queries, and the BM25 retrieval model. Broadly speaking, the former is used to select document components satisfying the structural restrictions of the NEXI query, and the latter to assign a relevance score and therefore to rank them.

5. Experimental evaluation

In this paper, firstly we will use *Garnata* as our base system in the experiments. This is a structured information retrieval system based on probabilistic graphical models, namely Bayesian networks and influence diagrams, but so far it is only able

Table 1
Comparison between the augmented CAS and the base-CO systems.

	Augmented CAS	Base-CO	% Improvement	*p*-Value
iP[0.00]	0.670683	0.467583	43.44	5.27E−10**
iP[0.01]	0.609759	0.406762	49.91	2.98E−11**
iP[0.05]	0.475207	0.332509	42.92	1.00E−06**
iP[0.10]	0.377938	0.294003	28.55	7.71E−04**
AiP	0.143025	0.123672	15.65	9.18E−02

to process non-structured queries. A description of this system can be found in de Campos, Fernández-Luna, Huete, and Romero (2006), whereas its theoretical basis is explained in more detail in de Campos, Fernández-Luna, and Huete (2004), de Campos, Fernández-Luna, and Huete (2005). Garnata has been tested at three editions of the INEX Workshop (de Campos, Fernández-Luna, Huete, Martín-Dancausa, & Romero, 2008; de Campos, Fernández-Luna, Huete, Martín-Dancausa, & Romero, 2009; de Campos, Fernández-Luna, Huete, & Romero, 2007) and has also been applied to build a real IRS for parliamentary documents (de Campos, Fernández-Luna, Huete, Martín, & Romero, 2008; de Campos et al., 2009).

The first XML document collection considered is the one used in the last three editions of the INEX Workshop, namely *Wikipedia* (an XML version of the English Wikipedia), at the beginning of 2006 (Denoyer & Gallinari, 2006) with its 659,388 articles (and around 4,600 megabytes in size). In terms of the queries (and the corresponding relevance judgments) used to test our proposal, we have selected the set of queries developed for INEX'2006,[7] INEX'2007[8] and INEX'2008.[9] We have not, however, employed all the original queries which have relevance judgments used in these editions of INEX, but a subset containing 90 queries (34 from 2006, 36 from 2007 and 20 from 2008). For the sake of completeness, the list of these CAS queries is displayed in Tables 8–10 in the Appendix.

We have not used either queries which are formally equivalent to content-only queries (i.e. queries of type //*[about(.,-text)]) or the majority of queries that while not strictly equivalent to content-only queries are in fact equivalent if we consider the narrative[10] of these queries (typical examples of this situation are queries such as //article[about(.,text)]). In general, we have used only those queries where the relevance judgments are coherent with the structural restrictions imposed by the CAS queries. In a few cases, we have modified the original CAS query to reach greater coherence between structural restrictions and the narrative of the query. For example, the query //section[about(.,pyramids of egypt)//image[about(.,pyramids)], looks for image elements contained in section elements, whereas the narrative of this query establishes that section elements containing images are looked for. We have then reformulated the query as //section[about(.,pyramids of egypt) and about(.//(figure|image),pyramids)].

In order to obtain the corresponding CO query for each CAS query, we removed all the structural components of the CAS query and kept only the content words. We then ran the Garnata retrieval system using the CO queries (base-CO) and the augmented system using the CAS queries, and compared the results. The weights of the noisy OR and noisy AND gates were fixed (without previous tuning) to constant values,[11] $w_o(U_i) = 1$ and $w_a(U_i) = 0.999$.

The measures of retrieval effectiveness are those used in the *focused* task of the INEX'2007 and 2008 ad hoc track (Kamps, Pehcevski, Kazai, Lalmas, & Robertson, 2008), namely the interpolated precision (iP) at selected recall levels (iP[0.0], iP[0.01], iP[0.05] and iP[0.10]) and the average interpolated precision (AiP), all of them averaged across the 90 queries. In the focused task the system must return a ranked list of the most focused document components and the resulting document parts should not overlap. The criterion used to decide, when we find two overlapping components in the raw output generated by Garnata, which to preserve in the final output, is to keep the component having the greatest relevance value and, in case of tie, we keep the more general component (the one containing a larger amount of text) (de Campos et al., 2008).

The results of our experiments are summarized in Table 1, which displays the corresponding performance measures for the base-CO and the augmented system, as well as the percentage of improvement achieved by the proposed system and the *p*-value of the statistical paired-*t* test used to detect significant differences. If the *p*-value is inferior to either 0.05 or 0.01 (significant or very significant difference) we denote this by using "$*$" or "$**$", respectively. In Fig. 5 we also display the recall-precision curves for the 101 recall levels.

The results are quite conclusive: the proposed method for managing CAS queries systematically improves the results of the system using CO queries in relation to all the performance measures. Moreover, the improvements achieved are statistically significant, ranging from a minimum of 15% to a maximum of 50%. We can also observe that the results are much better for lower recall levels. In fact, in Fig. 5 we can observe that the precision values of the augmented system are always better than those of the base-CO system for all the recall levels until the value 0.4; next, they become worse from 0.41 to 0.76 and again become better from 0.77 to 1.0. This means that adequately processing structured queries seems to concentrate

[7] http://www.inex.is.informatik.uni-duisburg.de/2006/.
[8] http://www.inex.is.informatik.uni-duisburg.de/2007/.
[9] http://www.inex.otago.ac.nz/.
[10] The narrative is a brief text associated to each query which represents the most authoritative description of the user's information need, and therefore serves as the main point of reference against which relevance should be assessed.
[11] We thought that very high values, close to 1, would be better than lower ones.

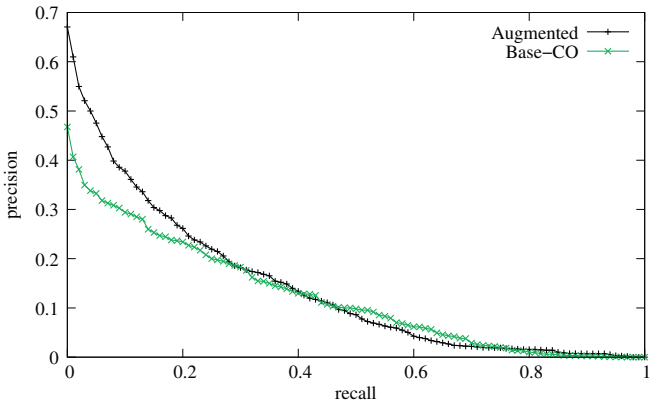

Fig. 5. Recall-precision curves for the augmented CAS and the base-CO systems.

Table 2
Comparison between the augmented CAS and the base-CO systems broken down by year.

	2006			2007			2008		
	CAS	CO	%	CAS	CO	%	CAS	CO	%
iP[0.00]	0.667	0.432	54.25	0.633	0.437	44.80	0.745	0.582	27.95
iP[0.01]	0.614	0.413	48.78	0.562	0.377	49.05	0.688	0.450	52.95
iP[0.05]	0.513	0.358	43.35	0.444	0.316	40.53	0.466	0.319	46.34
iP[0.10]	0.406	0.323	25.71	0.359	0.281	27.71	0.364	0.268	35.96
AiP	0.156	0.163	−4.50	0.146	0.107	36.01	0.116	0.086	34.74

highly relevant document parts in the first positions in the ranking of the results. Consequently, this technique greatly improves the precision and does not significantly deteriorate the recall (which is the initial intuition motivating the use of CAS queries).

In order to see whether the improvement obtained is consistent across the three different sets of queries considered, we display in Table 2 the performance measures broken down by year. We can observe a behaviour more or less similar across the three years, with the exception of the average interpolated precision, where we get important improvements for the years 2007 and 2008 but a small (not significant) worsening for 2006. This confirms that our method is very effective at lower recall levels but more unstable on the average.

In order to study whether our performance improvement is due to simple target restrictions or to the handling of the more complex structural requirements (because the base-CO system can return any element from the collection but our approach can only return elements having the same type as specified by the user), we are going to compare also with another additional baseline (which we will call base-CAS) which uses the same CO queries as the base-CO system, but additionally filters away all element that do not match the target element constraint. So, given a CAS query //A[B]//C[D], whereas the base-CO system transforms this query into //*[about(.,content words in B and D)], the base-CAS system transforms it as //A//C[about(.,content words in B and D)]. Table 3 and Fig. 6 display the results of the comparison between our augmented system and the base-CAS system being considered.

We can see that the augmented system still performs significantly better than the base-CAS system with respect to all the performance measures, although in this case the differences are lesser than in the previous case (except with the average interpolated precision). Therefore, we can obtain two conclusions: first, even a simple processing of the CAS queries can lead to improved performance with respect to using only CO queries; second, the proposed method improves further the results of this baseline treatment of CAS queries.

For the sake of completeness, detailed results for individual queries, for the iP[0.01] and AiP measures (which were the "official" measures at INEX, Kamps et al. (2008)) of the augmented CAS, the base-CO and the base-CAS, are displayed in Tables 11–13 in the Appendix. Table 14 shows the number of queries where our approach is either better, worse or equal than base-CO and base-CAS.

We want also to compare our proposal with another, state-of-the-art method for managing CAS queries. To this end, we have selected and implemented on top of the Garnata system the "full propagation" method proposed by Sigurbjörnsson et al.[12] previously mentioned (Sigurbjörnsson et al., 2004) (that we will call SKR), which was validated with the INEX IEEE Computer Society collection. The results of this comparison are displayed in Table 4 and Fig. 7.

[12] Among the different methods studied in their research, this one obtained the best overall results.

Table 3
Comparison between the augmented CAS and the base-CAS systems.

	Augmented CAS	Base-CAS	% Improvement	*p*-Value
iP[0.00]	0.670683	0.604653	10.92	1.15E−02*
iP[0.01]	0.609759	0.527010	15.70	1.01E−03**
iP[0.05]	0.475207	0.395017	20.30	4.51E−04**
iP[0.10]	0.377938	0.318322	18.73	5.59E−03**
AiP	0.143025	0.116574	22.69	8.99E−04**

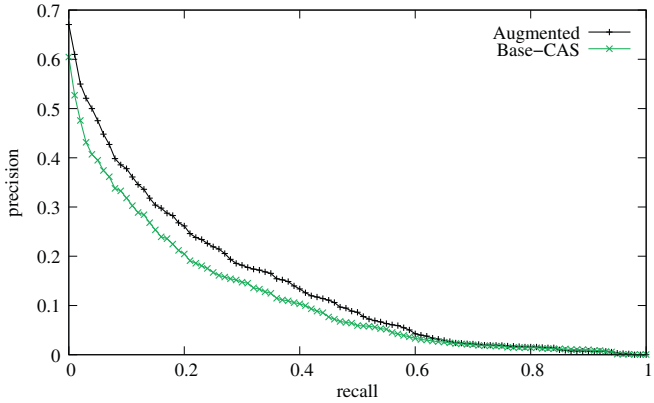

Fig. 6. Recall-precision curves for the augmented CAS and the base-CAS systems.

Table 4
Comparison between the augmented CAS and the SKR systems.

	Augmented CAS	SKR	% Improvement	*p*-Value
iP[0.00]	0.670683	0.512851	30.78	6.43E−07**
iP[0.01]	0.609759	0.469510	29.87	9.61E−07**
iP[0.05]	0.475207	0.345133	37.69	2.17E−06**
iP[0.10]	0.377938	0.294629	28.28	1.20E−04**
AiP	0.143025	0.120514	18.68	7.25E−03**

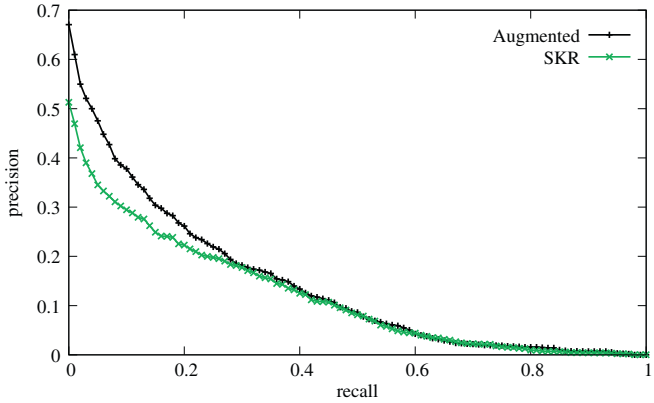

Fig. 7. Recall-precision curves for the augmented CAS and the SKR systems.

We can observe significant differences in performance between the two methods, favouring again our proposal. In fact the SKR method performs rather poorly in our experimental setting: it behaves worse than the base-CAS except in average interpolated precision, and scarcely outperforms the base-CO (again with the exception of the average interpolated precision).

Table 5
Results of the experiments with the Wikipedia collection using the PF/Tijah system.

	iP[0.00]	iP[0.01]	iP[0.05]	iP[0.10]	AiP
	Augmented CAS				
	0.568435	0.481378	0.373408	0.303367	0.115764
	Base-CO				
	0.485270	0.388033	0.298495	0.219913	0.067465
%	17.14	24.06	25.10	37.95	71.50
p-Value	2.09E−02*	4.22E−03**	1.57E−02*	4.06E−03**	3.46E−04**
	Base-CAS				
	0.530007	0.450899	0.332860	0.249575	0.098816
%	7.25	6.76	12.18	21.55	17.15
p-Value	1.64E−01	2.07E−01	1.37E−01	5.21E−02	9.89E−02
	SKR				
	0.415855	0.328691	0.274854	0.233593	0.097079
%	36.69	46.45	35.86	29.87	19.25
p-Value	1.56E−05**	4.58E−06**	7.98E−04**	7.30E−03**	4.29E−02*

Table 6
Results of the experiments with the IEEE Computer Society collection using Garnata.

	nxCG[10]	nxCG[25]	nxCG[50]	MAep
	Augmented CAS			
	0.1681	0.1546	0.1679	0.083731
	Base-CO			
	0.0971	0.1097	0.1220	0.053167
%	73.03	40.94	37.55	57.49
p-Value	2.59E−02*	8.91E−02	8.81E−02	1.36E−01
	Base-CAS			
	0.1541	0.1508	0.1676	0.075809
%	9.04	2.50	0.13	10.45
p-Value	2.36E−01	4.13E−01	4.94E−01	2.08E−01
	SKR			
	0.1214	0.1276	0.1365	0.060758
%	38.45	21.18	22.99	37.81
p-Value	2.41E−03**	2.35E−02*	2.13E−02*	9.37E−03**

In order to gain insight if there is a direct relationship between base and augmented system performance, we have also carried out experiments with another base system different from Garnata. We have selected the freely available PF/Tijah system[13] (Hiemstra, Rode, van Os, & Flokstra, 2006; Westerveld et al., 2007), part of the MonetDB/XQuery database system. Although PF/Tijah supports several retrieval models, we used the default language model. We have then repeated all the previous experiments but using PF/Tijah instead of Garnata. The results are summarized in Table 5. We can see that the augmented system always obtains better results than the other three systems, and these differences are still statistically significant for the base-CO and the SKR systems. Therefore, although the behaviour of the proposed method may be somewhat different depending on the base system being used, the trend is to improve the results of the base system.

Another interesting question to test is whether the proposed method for managing structured queries is also able to show consistent improvement using different document collections. To this end we have selected the IEEE Computer Society collection (Kazai, Lalmas, Fuhr, & Gövert, 2004) used in the first years of the INEX initiative. The original collection contained the full-texts, marked up in XML, of 12,107 articles of the IEEE Computer Society's publications from 12 magazines and 6 transactions, covering the period of 1995–2002, and totalling 494 megabytes in size. In 2005 the collection was extended with new articles from the period 2002–2004, giving a total of 16,819 articles and 764 megabytes in size. We have used a subset of 22 CAS queries from INEX'2003,[14] 19 from INEX'2004[15] and 14 from INEX'2005[16] (the list of queries is also displayed in the Appendix, in Tables 15–17).

In the first four editions of INEX (previous to 2006), the relevance judgments were done in a different way, using a two-dimensional scale taking into account exhaustivity and specificity, which were mapped to a single relevance scale by employing several quantization functions. We have used the so-called *strict quantization*, which focuses on components rated

[13] http://www.dbappl.cs.utwente.nl/pftijah/.
[14] http://www.inex.is.informatik.uni-duisburg.de:2003/.
[15] http://www.inex.is.informatik.uni-duisburg.de:2004/.
[16] http://www.inex.is.informatik.uni-duisburg.de/2005/.

Table 7
Results of the experiments with the Wikipedia collection using Garnata for different weights of the OR/AND gates.

w_o	w_a	iP[0.00]	iP[0.01]	iP[0.05]	iP[0.10]	AiP
1.0	*0.999*	*0.670683*	*0.609759*	*0.475207*	*0.377938*	*0.143025*
1.0	1.0	0.087623	0.082098	0.061717	0.051818	0.024452
1.0	0.99	0.640474	0.571272	0.436222	0.354053	0.131840
1.0	0.90	0.602101	0.538121	0.403327	0.329553	0.123633
1.0	0.75	0.557624	0.498115	0.380840	0.325173	0.119020
0.999	0.999	0.657806	0.590491	0.470164	0.381682	0.139561
0.99	0.999	0.670684	0.609759	0.475160	0.378033	0.143024
0.90	0.999	0.670315	0.607306	0.474859	0.377538	0.142859
0.75	0.999	0.653703	0.586629	0.465401	0.378000	0.137918

Table 8
List of the 34 INEX'2006 CAS queries in NEXI format used in the experiments with the Wikipedia collection.

291	//article//figure[about(.,olympian god goddess)]
292	//article//figure[about(.,renaissance painting italian flemish)]
293	//article[about(.,wifi)]//section[about(.,wifi security encryption)]
294	//article[about(.,user interface)]//section[about(.,design usability guidelines)]
295	//article[about(.,software)]//section[about(.,intellectual property) or about(.,patent license)]
304	//article[about(.,allergies)]//section[about(.,treatment)]
307	//article[about(.,islam islamic)]//section[about(.,qur'an) or about(.,prophet muhammad)]
308	//article[about(.,wedding)]//section[about(.,traditions customs)]
310	//article[about(.,novikov self-consistency principle) and about(.//section,time travel)]
313	//article[about(.,immanuel kant moral philosophy)]//section[about(.,categorical imperative)]
315	//article[about(.,spider)]//section[about(.,hunting) and about(.//p,insect)]
318	//article[about(.,atlantic ocean island)]//section[about(.,slave trade)]
327	//article[about(.,cloning animals)]//section[about(.,acceptance) and about(.,united states of america)]
331	//figure[about(.,tulips)]
334	//article[about(.,silk road)]//section[about(.,silk road china)]
337	//article[about(.,computer networks)]//section[about(.,security algorithms)]
347	//article[about(.,state machine)]//figure[about(.,mealy) or about(.,moore)]
355	//article[about(.,film)]//section[about(.,awards best actress academy award winner)]
357	//article[about(.,babylonia babylonian assyria assyriology)]//section[about(.,babylonia babylonian)]
360	//article[about(.,solar energy)]//section[about(.,domestic electricity heating)]
362	//article[about(.,nuclear power plant)]//section[about(.,effect accident)]
366	//article[about(.,fourier transform)]//p[about(.,applications)]
369	//article[about(.,pillars of hercules)]//section[about(.,mythology)]
373	//article[about(.,spy network)]//*[about(.,australia echelon)]
375	//article[about(.//p,states countries nuclear proliferation nonproliferation treaty npt) and about(.//p,weapons civilian)]
376	//article[about(.,diabetes mellitus)]//section[about(.,type 2 symptoms)]
382	//article[about(.,greek mythology) and about(.//caption,greek)]//*[about(.,aphrodite)]
386	//article[about(.,fencing)]//p[about(.,weapon)]
392	//article[about(.,australian aboriginals)]//section[about(.,stolen generation)]
399	//article[about(.,mobile phone country)]//section[about(.//table,umts) and about(.,umts)]
403	//article[about(.,analog color television)]//p[about(.,standard description)]
407	//article[about(.,football world cup)]//p[about(.,miracle of bern)]
409	//article[about(.,hybrid vehicle)]//p[about(.,fuel efficiency fuel sources model engine)]
413	//article[about(.,capital city europe)]//section[about(.,coordinates population)]

as highly exhaustive and highly specific (and whose exact definition varies slightly across years, Kazai & Lalmas (2006), Malik, Lalmas, & Fuhr (2005)), as done also in Kamps, Marx, de Rijke, and Sigurbjörnsson (2006), Sigurbjörnsson et al. (2004), and van Zwol (2006).

The measures of retrieval effectiveness are those used in INEX'2005, namely the (system-oriented) non-interpolated mean average effort-precision (MAep) and the (user-oriented) normalized extended cumulated gain (nxCG) at various fixed ranks (nxCG[10], nxCG[25] and nxCG[50]) (Kazai & Lalmas, 2006), averaged across the 55 queries. These measures have been computed using the EvalJ evaluation package, more specifically using the XCGEval package.[17] As in the previous experiments with Wikipedia, we still consider a focused task where overlap is not permitted (hence the raw results obtained by the IRS are filtered to remove overlapping components). Therefore, we always use the option `overlap=on` when evaluating with EvalJ. Table 6 displays the results of these experiments, where the base IR system used has been Garnata. The augmented system once again gets the best results, although the differences with the other systems are somewhat lesser than in the case of the

[17] http://www.sourceforge.net/projects/evalj.

Table 9
List of the 36 INEX'2007 CAS queries in NEXI format used in the experiments with the Wikipedia collection.

415	//article[about(.,space history)]//section[about(.,astronaut cosmonaut engineer)]
420	//article[about(.,shading models)]//section[about(.,phong shading)]
421	//article[about(.,neil gaiman novels)]//section[about(.,plot details)]
422	//article[about(.,bird) or about(.,passerine)]//p[about(.,song)]
428	//article[about(.,chinese traditional religion)]//section[about(.,taoism)]
429	//article[about(.,chinese dynasties)]
430	//article[about(.,steganography) or about(.,steganography techniques)]
431	//article[about(.,computer graphics)]//section[about(.,opengl)]
434	//article[about(.,nietzsche)]//section[about(.,book)]
444	//section[about(.//p,tcp ip)]//p[about(.,port forwarding)]
448	//article[about(.,french fifth republic)]//section[about(.,president)]
463	//article[about(.,healthy diet)]//section[about(.,diet features)]
464	//p[about(.,simpsons references)]
470	//section[about(.,operating system)]//p[about(.,page replacement policy)]
472	//article[about(.,prusik knot)]//figure[about(.,prusik)]
474	//section[about(.,engine diesel)]//p[about(.,fuel consumption)]
480	//article[about(.,computer) and about(.,monthly magazine)]
482	//article[about(.,characters of holy book ramayana) or about(.,ramayana)]
483	//article[about(.,theory) and about(.,origin universe)]
484	//article[about(.//category,artists)]//*[about(.,dutch artists paris)]
487	//article[about(.,brazil) and about(.//section,tourism)]
488	//article[about(.,football world cup fifa)]
489	//article[about(.,southern china)]//section[about(.,tourism)]
495	//article[about(.,machine learning algorithm) and about(.,machine learning theory)]
497	//article [about(.,first) and about(.,wikipedia)]
511	//article[about(.,george best)]//section[about(.,death) or about(.,died) or about(.,passed away)]
525	//*[about(.,potatoes) and about(.,paintings)]//figure[about(.,potatoes) and about(.,painting)]
526	//section[about(.,pyramids egypt) and about(.//(figure|image),pyramids)]
527	//article[about(.,walt disney land world) and about(.//(figure|image),disney land)]
530	//figure[about(.,hurricane)]
533	//(figure|image)[about(.,phone)]
537	//*[about(.,mont blanc) and about(.//(figure|image),mont blanc)]
538	//article[about(.,photographer) and about(.//(figure|image),photo)]
539	//figure[about(.//caption,self-portrait)]
542	//(figure|image)[about(.,tsunami)]
543	//(figure|image)[about(.,tux)]

Table 10
List of the 20 INEX'2008 CAS queries in NEXI format used in the experiments with the Wikipedia collection.

544	//article[about(.,philosophy)]//section[about(.,meaning of life)]
545	//article[about(.,dance)]//section[about(.,style)]
546	//article[about(.,history)]//section[about(.,19th century imperialism)]
550	//article[about(.,dna)]//section[about(.,test)]
555	//article[about(.//picture,amsterdam) or about(.//image,amsterdam)]
576	//article[about(.,aviation)]//p[about(.,aircraft formation)]
578	//section[about(.,childbirth tradition)]
581	//*[about(.,wine tasting) or about(.,wine)]
597	//article[about(.,database) or about(.,expert)]//section[about(.,database expert)]
598	//section[about(.,mahler symphony)]//*[about(.,song)]
600	//article[about(.,japanese culture)]//p[about(.,food)]
602	//article[about(.,dictionary)]//p[about(.,webster)]
603	//article[about(.,motors) or about(.,automobiles)]//p[about(.,tata)]
607	//article[about(.,nuclear)]//p[about(.,law) or about(.,legislation) or about(.,act)]
629	//article[about(.,film)]//section[about(.,science fiction)]
637	//article[about(.,java programming languages)]//section[about(.,applications of java)]
659	//section[about(.,technological singularity)]
668	//article[about(.,colossus code-breaking)]//section[about(.,bletchley park)]
669	//section[about(.,coin collecting)]
673	//section[about(.,intrusion detection)]

Wikipedia collection, specially with respect to the base-CAS system. These differences are not statistically significant in many cases (although in some cases this may be due in part to the less number of queries considered, 55 instead of 90).

Finally, in order to determine the sensitivity of the proposed model to the parameters considered for the AND and OR gates, w_a and w_o, we have carried out another series of experiments varying these parameters. We have used in this case the Wikipedia collection and the Garnata IR system. The results are displayed in Table 7.

Table 11
Detailed results per query for iP[0.01] and AiP using the 34 INEX'2006 CAS queries.

Query	iP[0.01]			AiP		
	Augm.-CAS	Base-CO	Base-CAS	Augm.-CAS	Base-CO	Base-CAS
291	0.04460	0.00871	0.04460	0.00833	0.00292	0.00833
292	0.66149	0.03105	0.66149	0.05995	0.00252	0.05995
293	1.00000	0.57261	1.00000	0.31303	0.43355	0.38455
294	1.00000	0.94037	1.00000	0.20720	0.30639	0.19421
295	0.84029	0.41597	0.74791	0.12757	0.12114	0.14363
304	0.39899	0.30393	1.00000	0.03704	0.02897	0.05400
307	0.21426	0.14873	0.13277	0.07113	0.07556	0.07129
308	0.75123	0.61193	0.64036	0.17494	0.08113	0.16735
310	1.00000	0.97749	1.00000	0.78168	0.23832	0.82698
313	0.97838	0.51579	1.00000	0.28906	0.23667	0.19598
315	0.76620	0.49489	0.71643	0.06069	0.05113	0.04409
318	0.08417	0.00336	0.03078	0.05109	0.00087	0.01841
327	1.00000	0.58446	0.31854	0.17109	0.08638	0.03195
331	0.68365	0.19140	0.68365	0.02344	0.09249	0.02344
334	0.83373	0.63629	0.99095	0.31647	0.21602	0.24332
337	0.07666	0.02895	0.11627	0.01758	0.00789	0.01943
347	0.84821	0.11736	0.68493	0.05879	0.04710	0.02962
355	0.03758	0.00762	0.03651	0.00531	0.00283	0.00555
357	0.94408	0.87974	0.97032	0.31471	0.44612	0.30204
360	0.84428	0.78447	1.00000	0.22690	0.54327	0.25231
362	0.26812	0.05146	0.06742	0.03389	0.03364	0.02744
366	0.39508	0.08766	0.61165	0.01511	0.03345	0.08299
369	0.65696	1.00000	0.65696	0.12635	0.03096	0.16079
373	0.20928	0.07189	0.07189	0.19685	0.03893	0.03893
375	0.08917	0.07989	0.09207	0.03358	0.05001	0.05317
376	0.61521	0.32973	0.61521	0.11630	0.13139	0.11324
382	0.93619	0.52754	0.52754	0.32437	0.31752	0.31752
386	0.74598	0.67702	0.85451	0.07032	0.12669	0.05494
392	0.51902	0.47099	0.60693	0.27754	0.27460	0.30169
399	1.00000	0.28935	0.76108	0.35452	0.22700	0.19310
403	0.41546	0.22920	0.32273	0.02023	0.12327	0.02144
407	1.00000	1.00000	1.00000	0.15604	0.62673	0.16864
409	1.00000	0.95235	0.63652	0.23839	0.50372	0.13596
413	0.01020	0.00425	0.00414	0.01190	0.00133	0.00030

The first fact that can be observed in Table 7 is that the values of the weights w_o and w_a matter, because they have influence on the results. Performance deteriorates systematically as the weight of the AND gate decreases, excluding the weight $w_a = 1.0$ (pure AND gate), which produces disastrous results. Therefore, a very high weight different from 1.0 for the AND gate, like 0.999 gives the best results. The value of the weight for the OR gate, w_o, has also influence on the results but at a lesser extent. The weight $w_o = 1.0$ (pure OR gate) gives almost the best results, together with $w_o = 0.99$. So, our initial selection of weights ($w_o = 1.0$ and $w_a = 0.999$) was, fortunately, a good choice.

In general, our results are considerably better than certain previous results reported in the literature by other researchers (Sigurbjörnsson et al., 2004; Trotman & Lalmas, 2006; Woodley, Geva, & Edwards, 2006), who did not find any significant improvement (or even no improvement at all) when managing CAS queries instead of CO queries. Our results are more in agreement with those in Kamps et al. (2006), who reported improvements in early precision, although we have also obtained good results in average measures. There is no simple explanation for this difference. The naive answer would be to say that some of these previous methods were not able to process structured queries effectively, but probably there are other reasons, such as the different document collections: our results are better for Wikipedia than for IEEE, which was the collection used to test these methods. This suggests that the effectiveness in managing CAS queries may depend on the type of collection, queries and performance measures being used. Another difference of our experimental setting with respect to previous studies is the type of task considered, focused in our case and thorough in the other cases. Even the base systems considered are also different. Therefore, although the proposed method has demonstrated its effectiveness, we believe that more research efforts should be invested in order to clarify when and how managing CAS queries is worthwhile.

6. Concluding remarks

In this paper, we have proposed a general methodology for managing structured queries within partially structured IRS able to only deal with content-only queries. Our method can be applied to any probabilistic IRS without the need to modify the system or interact with it in a complex way.

Table 12
Detailed results per query for iP[0.01] and AiP using the 36 INEX'2007 CAS queries.

	iP[0.01]			AiP		
Query	Augm.-CAS	Base-CO	Base-CAS	Augm.-CAS	Base-CO	Base-CAS
415	0.41188	0.10020	0.05003	0.03118	0.02157	0.00594
420	0.97421	0.46711	0.53599	0.27349	0.25737	0.21546
421	0.08287	0.07778	0.01844	0.00514	0.00570	0.00223
422	0.47726	0.33333	0.01886	0.15199	0.03845	0.00279
428	0.54670	0.19546	0.37765	0.19614	0.06826	0.09142
429	0.39120	0.39823	0.39120	0.17033	0.04076	0.17033
430	1.00000	0.95275	0.95271	0.63189	0.41167	0.50772
431	0.98273	0.38706	0.58617	0.34080	0.13278	0.20192
434	0.86993	0.61192	0.71744	0.06026	0.12089	0.31248
444	0.15366	0.04877	0.02276	0.01035	0.01428	0.00051
448	0.55402	0.57618	0.99219	0.06504	0.04714	0.11473
463	1.00000	0.91338	1.00000	0.56726	0.42323	0.40398
464	0.34535	0.17566	0.34535	0.02755	0.04294	0.02755
470	1.00000	0.94063	1.00000	0.22315	0.58218	0.12098
472	1.00000	0.67574	1.00000	0.01980	0.22313	0.01980
474	1.00000	0.98590	1.00000	0.16145	0.52697	0.11927
480	0.46862	0.42572	0.76274	0.10845	0.07676	0.10298
482	0.96576	0.99359	0.99359	0.37280	0.19099	0.20106
483	0.18047	0.09609	0.11841	0.11417	0.01299	0.07949
484	0.99471	0.99471	0.99471	0.15902	0.15902	0.15902
487	0.49939	0.10646	0.08869	0.04289	0.05133	0.05229
488	0.73738	0.52153	0.73738	0.16182	0.05725	0.16182
489	0.67209	0.12104	0.66923	0.10437	0.04810	0.10393
495	0.76830	0.78521	0.76830	0.13471	0.08503	0.10790
497	0.03354	0.00739	0.03354	0.02278	0.00070	0.02278
511	0.12060	0.00039	0.00055	0.04603	0.00034	0.00043
525	0.32174	0.01052	0.24511	0.03443	0.00115	0.03240
526	0.05524	0.01649	0.02773	0.00565	0.00620	0.00217
527	0.06865	0.02625	0.05442	0.03868	0.00506	0.03812
530	0.31607	0.01065	0.31607	0.01732	0.00084	0.01732
533	0.37813	0.06393	0.00000	0.00946	0.00208	0.00559
537	0.92133	0.19247	0.19247	0.38903	0.04696	0.04696
538	0.13871	0.84939	0.91789	0.01943	0.05668	0.07347
539	0.97346	0.51733	1.00000	0.28670	0.11084	0.29187
542	0.26275	0.00036	0.26275	0.05081	0.00002	0.05115
543	0.57692	0.00197	0.57692	0.20885	0.00018	0.20885

Table 13
Detailed results per query for iP[0.01] and AiP using the 20 INEX'2008 CAS queries.

	iP[0.01]			AiP		
Query	Augm.-CAS	Base-CO	Base-CAS	Augm.-CAS	Base-CO	Base-CAS
544	0.11635	0.26339	0.23657	0.03966	0.02368	0.05657
545	0.84679	0.79527	0.55989	0.09644	0.14366	0.11287
546	0.58191	0.40181	0.22365	0.05055	0.07570	0.03765
550	0.85852	0.29368	0.75747	0.13366	0.05163	0.12082
555	0.03218	0.00128	0.00487	0.00687	0.00004	0.00203
576	0.00692	0.00179	0.00929	0.00240	0.00119	0.00484
578	0.09163	0.10394	0.09163	0.02428	0.01047	0.02428
581	0.95469	0.42982	0.42982	0.02158	0.02137	0.02137
597	0.96844	0.26598	0.96844	0.05662	0.00533	0.02606
598	0.75385	0.14653	0.27434	0.04669	0.11108	0.03582
600	0.79630	0.44981	0.09364	0.02791	0.07512	0.00467
602	0.78538	0.44842	0.66351	0.08588	0.07277	0.08609
603	1.00000	0.99282	1.00000	0.31977	0.21643	0.06432
607	0.99940	0.05355	0.20583	0.32018	0.02284	0.03132
629	0.37466	0.17923	0.17953	0.02943	0.01165	0.02400
637	1.00000	0.55437	0.61479	0.19169	0.21442	0.10788
659	0.79146	0.89460	0.79146	0.15102	0.10743	0.15102
668	0.84929	0.99899	0.99439	0.21688	0.25280	0.26065
669	0.95996	0.75392	0.95996	0.13869	0.02914	0.13869
673	0.99838	0.97130	0.99838	0.35747	0.27335	0.35747

We have demonstrated the effectiveness of our approach by using two specific structured IRS and two benchmark XML document collections as the test bed. The experimental results confirm that a proper management of the structure present in CAS queries can significantly improve the retrieval capabilities of the system.

Table 14
Number of times that the Augmented CAS is better/worse/equal that Base-CO and Base-CAS methods for iP[0.01] and AiP, using the 90 INEX-Wikipedia CAS queries.

Times	iP[0.01]		AiP	
Augm.-CAS	Base-CO	Base-CAS	Base-CO	Base-CAS
Better	78	44	60	50
Worse	10	18	29	25
Equal	2	28	1	15

Table 15
List of the 22 INEX'2003 CAS queries in NEXI format used in the experiments with the IEEE Computer Society collection.

62	//article[about(.,security biometrics) and about(.//sec,facial recognition)]
63	//article[about(.,digital library) and about(.//p,authorization access control security)]
64	//article[about(.,hollerith)]//sec[about(.,DEHOMAG)]
67	//article//fm[about(.//(tig|abs),software architecture)]
68	//article[about(.,Smalltalk) or about(.,Lisp) or about(.,Erlang) or about(.,Java)]// bdy//sec[about(.,garbage collection algorithm)]
69	//article//bdy//sec[about(.//st,information retrieval)]
70	//article[about(.//fm//abs,information retrieval digital libraries)]
71	//article[about(.,formal methods verify correctness aviation systems)]// bdy//*[about(.,case study application model checking theorem proving)]
72	//article[about(.//fm//au//aff,United States of America)]//bdy//*[about(.,weather forecasting systems)]
74	//article[about(.,video streaming applications)]//sec[about(.,media stream synchronization) or about(.,stream delivery protocol)]
75	//article[about(.,Petri net) and about(.//sec,formal definition) and about(.//sec, algorithm efficiency computation approximation)]
77	//article[about(.//sec,reverse engineering)]//sec[about(.,legal) or about(.,legislation)]
78	//vt[about(.,Information Retrieval student)]
79	//article[about(.,XML) and about(.,database)]
80	//article//bdy//sec[about(.,clock synchronization distributed systems)]
81	//article[about(.//p,multi concurrency control) and about(.//p,algorithm) and about(.//fm//atl,databases)]
82	//article[about(.,handwriting recognition) and about(.//fm//au,kim)]
83	//article//fm//abs[about(.,data mining frequent itemset)]
84	//p[about(.,overview distributed query processing join)]
86	//sec[about(.,mobile electronic payment system)]
89	//article[about(.//bdy,clustering vector quantization fuzzy k-means c-means)]//bm//bb [about(.,vector quantization fuzzy clustering k-means c-means) and about(.//pdt,1999)]
90	//article[about(.//sec,trust authentication electronic commerce e-commerce e-business marketplace)]//abs[about(.,trust authentication)]

Table 16
List of the 19 INEX'2004 CAS queries in NEXI format used in the experiments with the IEEE Computer Society collection.

127	//sec//(p|fgc)[about(.,Godel Lukasiewicz other fuzzy implication definitions)]
128	//article[about(.,intelligent transport systems)]//sec[about(.,on-board route planning navigation system for automobiles)]
129	//article[about(.//atl,new book review bookshelf)]//sec[about(.,database data warehouse)]
130	//article[about(.//p,object database)]//p[about(.,version management)]
131	//article[about(.//au,Jiawei Han)]//abs[about(.,data mining)]
132	//article[about(.//abs,classification)]//sec[about(.,experiment compare)]
135	//article[about(.//atl,summaries)]//sec[about(.,Internet security) or about(.,network security)]
136	//bib[about(.,text categorisation) and about(.,Support Vector Machines SVM)]
137	//article[about(.//abs,digital library) or about(.//ip1,digital library)]
141	//article[about(.,java)]//sec[about(.,implementing threads)]
142	//abs[about(.,database access using perl)]
145	//article[about(.,information retrieval)]//p[about(.,relevance feedback)]
149	//article[about(.,animation)]//bdy//sec[about(.//st,inverse kinematics)]
150	//article[about(.//(abs|kwd),genetic algorithm)]//bdy//sec[about(.,simulated annealing)]
151	//article[about(.,web search engine)]//sec[about(.,vector space model)]
152	//article//p[about(.,linux word processor office programs)]
153	//article//bm//vt[about(.,phD student) OR about(.,phD final)]
155	//article[about(.//p,self organising feature map) and about(.//fm//yr,2000)]// fig[about(.//fgc,self organising map)]
156	//article[about(.//abs,spatial join)]//bdy//sec[about(.,performance evaluation)]

Our view of CAS queries in this work has been rather strict in the sense that the target-path constraints present in the query must be upheld for a result to be relevant. If a user asks (for example) for *section* units to be returned, these must be returned (although other aspects of the query are interpreted from the IR perspective, i.e. loosely). There is, however, another less strict, vague interpretation of a CAS query (Trotman & Sigurbjörnsson, 2005), where target-path requirements need not be fulfilled, and the path specifications should therefore be considered hints as to where to look. For future work, we plan to modify our method to deal with this vague interpretation of CAS queries. Another interesting future research

Table 17
List of the 14 INEX'2005 CAS queries in NEXI format used in the experiments with the IEEE Computer Society collection.

202	//article[about(.,ontologies)]//sec[about(.,ontologies case study)]
203	//sec[about(.,code signing verification)]
208	//article[about(.,Artificial Intelligence history)]
210	//article//(abs|sec)[about(.,multimedia document models content authoring)]
216	//sec[about(.,multimedia retrieval system architecture) or about(.//fig,multimedia retrieval architecture)]
219	//sec[about(.,learning object granularity)]
222	//article[about(.,bussiness strategies)]//sec[about(.,electronic commerce e-commerce)]
223	//article[about(.//sec,wireless ATM multimedia)]
229	//article[about(.//bdy,latent semantic analysis latent semantic indexing)]
230	//article//sec[about(.,brain research differential geometry)]
232	//article[about(.//abs,Dempster-Shafer theory)]//sec[about(.,Dempster Shafer database experiment)]
233	//article[about(.//bdy,synthesizers) and about(.//bdy, music)]
234	//article[about(.//atl,upcoming events) or about(.//atl,call for papers)]// sec[about(.,multimedia conference workshop)]
236	//article[about(.,machine translation approaches)]

could be to try to take advantage of the flexibility of the noisy-OR/AND used by our approach to improve further the results, by studying how to assign the weights $w(U_i)$ used in these noisy gates in a non-uniform way, depending for example on the type of query being processed or on the type of element being returned.

Acknowledgments

This work has been jointly supported by the Spanish 'Consejería de Innovación, Ciencia y Empresa de la Junta de Andalucía', 'Ministerio de Ciencia e Innovación' and the research programme Consolider Ingenio 2010, under Projects TIC-04526, TIN2008-06566-C04-01 and CSD2007-00018, respectively.

Appendix A.

See Tables 8–17.

References

Ali, M. S., Consens, M., Gu, X., Kanza, Y., Rizzolo, F., & Stasiu, R. (2007). Efficient, effective and flexible XML retrieval using summaries. *Lecture Notes in Computer Science, 4518*, 89–103.
Arvola, P., Kekäläinen, J., & Junkkari, M. (2006). Query evaluation with structural indices. *Lecture Notes in Computer Science, 3977*, 134–145.
Azevedo, M., Pantuza, L., & Ziviani, N. (2005). A universal model for XML information retrieval. *Lecture Notes in Computer Science, 3493*, 311–321.
Chiaramella, Y. (2001). Information retrieval and structured documents. *Lecture Notes in Computer Science, 1980*, 291–314.
Crouch, C. J., Mahajan, A., & Bellamkonda, A. (2005). Flexible retrieval based on the vector space model. *Lecture Notes in Computer Science, 3493*, 292–302.
de Campos, L. M., Fernández-Luna, J. M., & Huete, J. F. (2004). Using context information in structured document retrieval: An approach using influence diagrams. *Information Processing and Management, 40*(5), 829–847.
de Campos, L. M., Fernández-Luna, J. M., & Huete, J. F. (2005). Improving the context-based influence diagram for structured retrieval. *Lecture Notes in Computer Science, 3408*, 215–229.
de Campos, L. M., Fernández-Luna, J. M., Huete, J. F., Martín, C., & Romero, A. E. (2008). An information retrieval system for parliamentary documents. In P. Naim, O. Pourret, & B. Marcot (Eds.), *Bayesian Networks: A Practical Guide to Applications* (pp. 203–223). Wiley.
de Campos, L. M., Fernández-Luna, J. M., Huete, J. F., Martín-Dancausa, C., & Romero, A. E. (2008). The Garnata information retrieval system at INEX'07. *Lecture Notes in Computure in Science, 4862*, 57–69.
de Campos, L. M., Fernández-Luna, J. M., Huete, J. F., Martín-Dancausa, C., & Romero, A. E. (2009). New utility models for the Garnata information retrieval system at INEX'08. *Lecture Notes in Computer Science, 5631*, 39–45.
de Campos, L. M., Fernández-Luna, J. M., Huete, J. F., Martín-Dancausa, C., Tagua-Jiménez, A., & Tur-Vigil, C. (2009). An integrated system for managing the andalusian parliament's digital library. *Program – Electronic Library and Information Systems, 43*(2), 156–174.
de Campos, L. M., Fernández-Luna, J. M., Huete, J. F., & Romero, A. E. (2006). Garnata: An information retrieval system for structured documents based on probabilistic graphical models. In *Proceedings of the 11th international conference of information processing and management of uncertainty in knowledge-based systems (IPMU)* (pp. 1024–1031).
de Campos, L. M., Fernández-Luna, J. M., Huete, J. F., & Romero, A. E. (2007). Influence diagrams and structured retrieval: Garnata implementing the SID and CID models at INEX'06. *Lecture Notes in Computer Science, 4518*, 165–177.
Denoyer, L., & Gallinari, P. (2006). The Wikipedia XML corpus. *SIGIR Forum, 40*, 64–69.
Fuhr, N., Gövert, N., Kazai, G., & Lalmas, M. (Eds.). (2002). *Proceedings of the 1st international workshop of the initiative for the evaluation of XML retrieval, INEX 2002*.
Fuhr, N., Lalmas, M., Malik, S., & Kazai, G. (Eds.). (2006), Advances in XML information retrieval and evaluation, 4th international workshop of the initiative for the evaluation of XML retrieval, INEX 2005. *Lecture Notes in Computer Science, 3977*.
Fuhr, N., Lalmas, M., Malik, S., & Szlávik, Z. (Eds.). (2005). Advances in XML information retrieval, 3rd international workshop of the initiative for the evaluation of XML retrieval, INEX 2004. *Lecture Notes in Computer Science, 3493*.
Fuhr, N., Lalmas, M., & Trotman, A. (Eds.). (2007), Comparative evaluation of XML information retrieval systems, 5th international workshop of the initiative for the evaluation of XML retrieval, INEX 2006. *Lecture Notes in Computer Science, 4518*.
Fuhr, N., Lalmas, M., & Trotman, A. (Eds.). (2008), Focused access to XML documents, 6th international workshop of the initiative for the evaluation of XML retrieval, INEX 2007. *Lecture Notes in Computer Science, 4862*.
Fuhr, N., Malik, S., & Lalmas, M. (Eds.). (2003). *Proceedings of the 2nd international workshop of the initiative for the evaluation of XML retrieval, INEX 2003*.
Geva, S. (2005). GPX – Gardens Point XML information retrieval at INEX 2004. *Lecture Notes in Computer Science, 3493*, 211–223.

Hiemstra, D., Rode, H., van Os, R., & Flokstra, J. (2006). PF/Tijah: Text search in an XML database system. In *Proceedings of the 2nd international workshop on open source information retrieval* (pp. 12–17).
Kamps, J., Marx, M., de Rijke, M., & Sigurbjörnsson, B. (2006). Articulating information needs in XML query languages. *ACM Transactions on Information Systems, 24*(4), 407–436.
Kamps, J., Pehcevski, J., Kazai, G., Lalmas, M., & Robertson, S. (2008). INEX 2007 evaluation measures. *Lecture Notes in Computer Science, 4862*, 24–33.
Kazai, G., & Lalmas, M. (2006). INEX 2005 evaluation metrics. *Lecture Notes in Computer Science, 3977*, 16–29.
Kazai, G., Lalmas, M., Fuhr, N., & Gövert, N. (2004). A report on the first year of the INitiative for the evaluation of XML retrieval (INEX'02). *Journal of the American Society for Information Science and Technology, 55*(6), 551–556.
Lalmas, M. (2009). *XML Retrieval*. Morgan & Claypool Publishers.
Larson, R. (2005). Cheshire II at INEX'04: Fusion and feedback for the adhoc and heterogeneous tracks. *Lecture Notes in Computer Science, 3493*, 322–336.
Malik, S., Lalmas, M., & Fuhr, N. (2005). Overview of INEX 2004. *Lecture Notes in Computer Science, 3493*, 1–15.
Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge University Press.
Mass, Y., & Mandelbrod, M. (2005). Component ranking and automatic query refinement for XML retrieval. *Lecture Notes in Computer Science, 3493*, 73–84.
Mohan, S., & Sengupta, A. (2005). DocBase – The INEX evaluation experience. *Lecture Notes in Computer Science, 3493*, 261–275.
Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. San Mateo: Morgan and Kaufmann.
Pehcevski, J., Thom, J. A., Tahaghoghi, S. M. M., & Vercoustre, A. (2005). Hybrid XML retrieval revisited. *Lecture Notes in Computer Science, 3493*, 153–167.
Sauvagnat, K., & Boughanem, M. (2005). Using a relevance propagation method for adhoc and heterogeneous tracks at INEX 2004. *Lecture Notes in Computer Science, 3493*, 337–348.
Schweizer, B., & Sklar, A. (1983). *Probabilistic metric spaces*. Amsterdam: North-Holland.
Shimizu, T., Terada, N., & Yoshikawa, M. (2007). Development of an XML information retrieval system for queries on contents and structures. In *Proceedings of the second international conference on informatics research for development of knowledge society infrastructure* (pp. 161–168).
Sigurbjörnsson, B., Kamps, J., & de Rijke, M. (2004). Processing content-oriented XPath queries. In *Proceedings of the thirteenth ACM international conference on information and knowledge management* (pp. 371–380). ACM Press.
Sigurbjörnsson, B., Kamps, J., & de Rijke, M. (2005). Mixture models, overlap, and structural hints in XML element retrieval. *Lecture Notes in Computer Science, 3493*, 196–210.
Theobald, M., Broschart, A., Schenkel, R., Solomon, S., & Weikum, G. (2007). TopX – AdHoc track and feedback task. *Lecture Notes in Computer Science, 4518*, 233–242.
Trotman, A., & Lalmas, M. (2006). Why structural hints in queries do not help XML-retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 711–712). ACM Press.
Trotman, A., & Sigurbjörnsson, B. (2005). Narrowed Extended XPath I (NEXI). *Lecture Notes in Computer Science, 3493*, 16–40.
Vittaut, J., Piwowarski, B., & Gallinari, P. (2005). An algebra for structured queries in Bayesian networks. *Lecture Notes in Computer Science, 3493*, 100–112.
Weigel, F., Schulz, K., & Meuss, H. (2005). Ranked retrieval of structured documents with the S-term vector space model. *Lecture Notes in Computer Science, 3493*, 238–252.
Westerveld, T., Rode, H., van Os, R., Hiemstra, D., Ramirez, G., Mihajlovic, V., et al (2007). Evaluating structured information retrieval and multimedia retrieval using PF/Tijah. *Lecture Notes in Computer Science, 4518*, 104–114.
Woodley, A., Geva, S., & Edwards, S. L. (2006). Comparing XML-IR query formulation interfaces. *Australian Journal of Intelligent Information Processing Systems, 9*, 64–71.
W3C. Extensible Markup Language (XML) 1.0. (2006). W3C Recommendation (14th ed.). <http://www.w3.org/TR/REC-xml>.
W3C. XML Path Language (XPath) Version 2.0. (2007). W3C Recommendation. <http://www.w3.org/TR/xpath20>.
van Zwol, R. (2006). B^3-SDR and effective use of structural hints. *Lecture Notes in Computer Science, 3977*, 146–160.