Information Retrieval manuscript No.
(will be inserted by the editor)

Specificity Aboutness in XML Retrieval

Tobias Blanke · **Mounia Lalmas**

Received: date / Accepted: date

Abstract This paper presents a theoretical methodology to evaluate XML retrieval systems and their filters. Theoretical evaluation is concerned with the formal investigation of qualitative properties of retrieval models. XML retrieval deals with retrieving those document components that specifically answer a query, and filters are a method of delivering the most focussed answers. Our theoretical evaluation critically analyzes how filters achieve this.

Keywords Information Retrieval Theory, Theoretical Evaluation, XML Retrieval

1 Introduction

According to INEX, the evaluation initiative for XML retrieval [11], the aim of XML retrieval is to retrieve not only relevant document components, but those at the right level of granularity, i.e. those that specifically answer a query. XML, contrary to HTML, separates the logical structure of documents from the layout. The logical structure of an XML document forms a tree of elements, which starts with a root element and has edges between elements. XML retrieval uses the logical structure of elements and edges between them to return more precise results to user information needs. To evaluate how effectively XML retrieval approaches deliver precise results, it is necessary to consider whether the 'right' level of the structure is correctly identified. For this purpose, INEX has developed a new relevance criterion next to general relevance, which measures how focussed an XML element is with respect to an information need. The general relevance of an element is captured in the INEX exhaustivity dimension[1] while the specificity dimension indicates the focus.

In INEX, the retrieval task that aims at finding the most specific answers has been referred to as the focussed task. This is to be compared to the thorough task, that aims at estimating the relevance of document components to a query. In this latter task, all relevant document components are to be identified, and then ranked according to their degree

Department of Computing Science
University of Glasgow , G12 8QQ, Glasgow, UK
Tel.: +44 141 330 1647
Fax: +44 141 330 1651
E-mail: tobias.blanke@dcs.gla.ac.uk
E-mail: mounia@acm.org

[1] Since 2006, INEX does not refer to exhaustivity anymore, just relevance and specificity.

of relevance. In the focussed task, the result set should consist of non-overlapping document components, ranked according to how specific they are to the query. Overlap occurs when a document component (e.g. a section) and one of its descendents (e.g. a paragraph in the section) or ascendents (e.g. the chapter containing that section) are both returned as answers. The aim of the focussed task is therefore to identify the component that is the most specific to the query among overlapping document components, and to return it as what is referred to as a *focussed* answer. In this paper, we concentrate on retrieval models developed for the focussed task at INEX 2005, as the fundamentals of these models have not changed much since. In addition, we restrict ourselves to models aiming at delivering these focussed answers for content-only queries, i.e. without structural constraints.

Models developed at INEX to implement the focussed retrieval task can be viewed as filters. Indeed, these models mostly consist of the post-processing of an answer set produced by models aiming at implementing the thorough retrieval task. The post-processing phase consists of eliminating all but the most focussed document components from the answer set. Several types of filters have been developed in INEX. The most popular filter is the so-called brute-force one, which eliminates all but the most relevant elements[2] on a particular XML path. However, in the experimental evaluation, it performs less well than others that look at the structural relationships between elements [17]. It is this kind of behaviour that we investigate in this paper, by looking at an alternative for evaluating retrieval methods. We suggest a theoretical evaluation to analyse retrieval models and develop explanations for their experimental behaviour.

In this paper, we analyze retrieval models developed at INEX that aimed at delivering results that specifically answer a query. Delivering these so-called *most specific* answers has proven to be a complex retrieval task. This paper proposes an alternative theoretical evaluation that complements an experimental evaluation, especially when dealing with complex retrieval tasks such as those developed for XML retrieval.

This paper is organised as follows. In Sections 2 and 3, we introduce the background of our theoretical evaluation methodology. In Section 4, we draw on earlier results to demonstrate parts of the theoretical evaluation of two XML retrieval approaches implementing the thorough task. We then introduce in Section 5 our theoretical methodology to analyse filters as aboutness decisions, before applying it to the brute-force and re-ranking filtering models in Section 6. Finally, we relate our findings to those of the experimental evaluation performed at INEX in Section 7.

2 Theoretical Evaluation

Experimental evaluation in INEX and TREC is used to study the behaviour of IR systems [13]. A typical task in an experimental evaluation is to compare IR models with each other, where system A is tested against system B using a collection C. This test is repeated by manipulating various parameters in systems A and B or by changing the collection C by, for instance, adding new documents to the collection. After the experiments have been carried out and results evaluated, a hypothesis is formulated that explains the experimental results. To support the hypothesis, standardized statistical evaluation values such as recall and precision are employed. Often this hypothesis concludes with a specification of why and when system A performs better than system B, or vice versa, using precision/recall graphs and other means, such as mean average precision [24].

[2] In this paper, elements and document components are used interchangeably.

Standardized test collections and experimental evaluation, however, do not always deliver sufficient information about a system's behaviour. There are clear disadvantages to such an evaluation approach, if test collections are incomplete or if precision and recall depend on knowledge about the set of all relevant documents [13]. In this paper, we suggest another approach to analyze how XML retrieval systems deliver specific answers, which complements an experimental evaluation such as the one typically performed at INEX. We present an evaluation based on describing the activities of a retrieval model as a set of logical reasoning rules using a theoretical framework that we have developed to specifically analyse XML retrieval systems' behaviour. We deliver a formal means of comparing system behaviour, with which we are able to go deeper into the details of how particular systems within INEX achieve specific answers.

Research on theoretical evaluation in IR is not new. Van Rijsbergen suggested earlier [21] that an experimental approach to IR should be complemented with a theoretical evaluation to match the increasing complexity of the retrieval task in new areas of IR like multimedia and XML retrieval. More recently the Renaissance project[3] uses new formal models based on quantum theory to explain the complex behaviour of multimedia and XML retrieval systems [18]. Such approaches expect that a theoretical evaluation can offer new insights into the reasoning behaviour of complex retrieval tasks. We, and others, see in theoretical evaluation a complementary means to an experimental one if it helps to clarify the assumptions of retrieval models and if it can identify the characteristics leading to a particular experimental behaviour.

There are many approaches for a theoretical evaluation in IR. We focus here on logic-based ones, as we aim to prove claims we make about XML retrieval systems. These logic-based evaluation approaches to IR are generally based on Cooper's definition of 'logical relevance' for an objective view on relevance [9, p. 24]:

> "A stored sentence is logically relevant to a representation of an information need if and only if it is a member of some minimal premise set of stored sentences for some component statement of that need."

In this definition, logical relevance is characterized by the topical bearing of a document on an information need. Based on Cooper's definition, van Rijsbergen and others have expressed logical relevance as the conditional $d \rightarrow q$ [21]. Since then, much research has gone into logical models of IR based on the work of Cooper and van Rijsbergen [19].

Huibers [13] has offered a different logic-based theoretical approach, with the aim to evaluate retrieval models. Following Huibers' formalism and approach, we call topical implications between query and document 'aboutness', where aboutness is described by formally deriving the reasoning process involved in IR models. Huibers has developed a general IR framework based on aboutness that he uses to evaluate existing retrieval models rather than to develop new ones. Such a framework needs to have the formal means to model the underlying concepts and behaviour of any particular information retrieval model. It should therefore abstract from specific constructs and implementation details.

'Aboutness' has been frequently discussed in IR literature, most notably in the work of Lalmas and van Rijsbergen [22], Hutchins [14], Huibers and Bruza [5], and recently Bruza, Song and Wong [25] and [6]. Huibers demonstrates the power of an aboutness based framework for the theoretical evaluation of IR. He successfully derived *aboutness proof systems* to capture several aspects of the reasoning process involved in commonly used 'flat'

[3] http://renaissance.dcs.gla.ac.uk/

document retrieval models [13]. Wong et al. [25] present theoretical evaluations of flat document document retrieval systems grounded in a well-defined set of steps as a complement to traditional experimental evaluation of IR systems. They found explanations for model behaviour that escaped more traditional evaluation methods, e.g. the effect monotonicity has on aboutness, which is discussed in this paper in more detail. Furthermore, they could derive the conditions and thresholds many flat document IR models use to adjust their reasoning behaviour to particular retrieval tasks. We show in Section 4 how methods of controlling monotonic behaviour are also widely used in XML retrieval to deliver more precise answers to an information need. Wong et al. also claimed that logic-based evaluation is more open to debate, as sometimes underlying assumptions of IR model performance can be hidden by tuning a priori assigned parameters in such a way that they fit best the evaluation task. They discussed this for performance differences of the general vector space model compared to the one that uses thresholds. In this paper, we identify similar strategies to deliver effective XML retrieval models.

These existing results of an aboutness based theoretical evaluation in standard document IR indicate that it can be a powerful methodology to analyse the more complex tasks in XML retrieval. Our thesis is that particularly in the domain of XML retrieval, aboutness based theoretical evaluation presents a powerful methodology to analyse the complex interaction of structure and content in XML retrieval, and to understand which interaction leads to the best performances in the experimental evaluation. In this paper, we use the concept of aboutness to evaluate XML retrieval models that aimed at identifying the most specific document components for a query. XML retrieval systems realise this specificity aboutness by narrowing the answer set of XML retrieval models.

3 Methodology background

In this section, we introduce the steps of our theoretical evaluation methodology. A theoretical evaluation methodology needs a formalism powerful enough to characterize the fundamental properties of retrieval models. Following Huibers, we use Situation Theory (ST), developed by Barwise and Perry [1], for this purpose. ST is a mathematical theory of meaning and information with *situations* as primitives [13]. Situations are partial descriptions of the world and are composed of *infons* [10]. Infons are 'items of information' that specify a certain part of the world or a 'situation'. For IR modelling, queries and documents are modelled as situations, while infons represent a model's information items like keywords or phrases.

Using ST, we model documents and queries as situations [5]. Let document D and query Q be situations, then $D \,\Box\!\!\rightsquigarrow\, Q$ means that the information in D is about the information need expressed in Q. For instance, in standard IR, a document containing 'garden' and 'house' would be about a query asking for 'garden'. Then, $D \,\Box\!\!\not\rightsquigarrow Q$ symbolises that D is not about Q. With $\otimes$, we formalise the composition of situations, e.g. 'house' and 'garden' can be combined to 'garden house'. $\equiv$ states that two situations are equivalent, i.e. they contain the same information.

Chiaramella [7] demonstrated the theoretical underpinnings for structured document retrieval in general and XML retrieval in particular. He uses the conditional $d \rightarrow q$ by van Rijsbergen to model the general relevance (exhaustivity) of an element to an information need, but adds to this a second one $q \rightarrow d$ to model the focus of an element (specificity). Translating this into our more abstract aboutness framework, we use Chiaramella's distinction and say that $D \,\Box\!\!\rightsquigarrow\, Q$ symbolizes exhaustivity and $Q \,\Box\!\!\rightsquigarrow\, D$ specificity.

We call *translation* the symbolic representation of an IR model's handling of information using a formal language. It is formally represented by a function *map* that 'maps' situations to their formal representation. In IR, mapping a document (or a document component) to its formal representation corresponds to the indexing process.

Infons as a result of the indexing are represented by $\langle\langle k \rangle\rangle$, where k stands for any indexed term. They stand for the 'facts' an IR system recognizes as information items. A set of information items is a situation: $\{\langle\langle k_1 \rangle\rangle, ..., \langle\langle k_n \rangle\rangle\}$. N-ary relationships R between information items i_j are themselves information items and are modelled by $\langle\langle R, i_1, ..., i_n \rangle\rangle$[4].

For representing information in XML retrieval, we use N-ary relationships R between infons i_j, to model structural relationships. For instance, a section with two paragraphs will be symbolized by: $\{\langle\langle ElementType, Sec, s \rangle\rangle, \langle\langle ElementType, Para, p_1 \rangle\rangle,$ $\langle\langle Value, garden, p_1 \rangle\rangle$, $\langle\langle ElementType, Para, p_2 \rangle\rangle, \langle\langle Value, house, p_2 \rangle\rangle, \langle\langle Parent, s, p_1 \rangle\rangle,$ $\langle\langle Parent, s, p_2 \rangle\rangle\}$. This reflects the fact that each XML element has an element type infon, expressed with the relation $ElementType$. Content infons (i.e. the actual text in the element) are modeled as *Values*. The relation *Parent* expresses that the two paragraphs (p_1 and p_2) are the children of the section (s). Translation is therefore based on building a document representation through indexing, which according to van Rijsbergen [23] leads to the view that index terms represent properties of documents (or document components), which may then be studied[5].

Next to the symbolic characterization of an IR model, we need means to describe the functional behaviour of the model, i.e. what makes a document (or document component in XML retrieval) about a query. This is done through so-called *reasoning rules*. Indeed, an IR model's aboutness decision is specified by the reasoning rules it incorporates. These can be either fully, partially, or not at all supported. Together with the symbolic representation of documents, queries and information for an IR model, they make up the *aboutness decision system* that characterizes how the model decides that a document (or document component) is relevant to query.

An example of a rule is Left Monotonic Union (LMU), which plays an important role in our theoretical study of filters. LMU states that if a document D is about a query Q, then so is the composition of D and D':

- LMU: If $D \sqsubset\!\!\sim Q$, then also $D \otimes D' \sqsubset\!\!\sim Q$.

By comparing the reasoning rules each decision system incorporates and the way it does so, we can give an overall comparison of the behaviour of the retrieval model characterized by the aboutness decision system.

Huibers [13] and Wong et al. [25] both see the careful consideration of monotonicity as an important feature of IR. There are also right variants of LMU, called Right Monotonic Union (RMU), where the information is added on the right side of the aboutness relation. In an aboutness model unconditionally supporting LMU, a query containing 'house' is not only about documents with 'house', but equally valid answers are components with 'house' and 'garden'. This means, aboutness decisions supporting LMU are not affected by document length. For systems supporting RMU, query expansion does not change the aboutness

[4] Note that according to Devlin's notation [10] the information item $\langle\langle garden \rangle\rangle$ would be written $\langle\langle R, garden; 1 \rangle\rangle$, where R is an unary relation and the polarity 1 indicates that the information garden holds for an XML element situation. As the investigated XML retrieval systems do not use polarities in their indexing, we simply use $\langle\langle garden \rangle\rangle$.

[5] In the work of Barwise and Perry [1] and Devlin [10], infons are theoretical constructs used to investigate information flow. In our case, we use them to model how information contained in XML elements can be about an information need, which could be viewed as a special case of information flow.

decision. This means that systems with RMU can expand the original query and gain a higher recall base while at the same time not loosing what the original query was about. Both document component and query length are decisive aspects of aboutness decisions, which underlines the importance of monotonicity [25].

There are over 20 reasoning rules to be considered in the theoretical analysis of retrieval models (see [13] and [25]). In this paper, we restrict ourselves to the following rules (including LMU), as they are sufficient for our investigation:

- Reflexivity: $S \; \Box\!\!\rightsquigarrow \; S$.
- Transitivity: If $S \; \Box\!\!\rightsquigarrow \; T$ and $T \; \Box\!\!\rightsquigarrow \; U$, then also $S \; \Box\!\!\rightsquigarrow \; U$.
- Symmetry: If $S \; \Box\!\!\rightsquigarrow \; T$, then also $T \; \Box\!\!\rightsquigarrow \; S$.
- Euclid: If $S \; \Box\!\!\rightsquigarrow \; T$ and $S \; \Box\!\!\rightsquigarrow \; U$, then also $T \; \Box\!\!\rightsquigarrow \; U$.
- Mix: If $S \; \Box\!\!\rightsquigarrow \; T$ and $U \; \Box\!\!\rightsquigarrow \; T$, then also $S \otimes U \; \Box\!\!\rightsquigarrow \; T$.

These rules have proven to be important for determining specificity in XML retrieval [3] and for understanding the impact of filters on underlying aboutness systems, as we describe in Section 6.

Throughout the paper, we use upper case letters from the middle of the alphabet such as S, T, for situations if we are not talking about queries and document components. In that case we use Q and D. Anything that situations are made of, e.g. keywords but also structural relationships, is symbolized with letters from the beginning of the alphabet like A or B.

In the next section, we illustrate the aboutness relation in XML retrieval.

4 Aboutness in XML retrieval

To carry out our theoretical evaluation of filters (the focussed retrieval task at INEX), it is necessary to present the theoretical evaluation of models developed for the thorough task. This is because of the relationships between the tasks, one being a post-processing of the other. This also provides an illustration of our theoretical methodology.

In this section, we look at XML retrieval models evaluated at INEX and their logical properties. We do not fully specify all the properties of these models; instead, we focus on demonstrating some aspects of our theoretical evaluation approach. We concentrate on those characteristics that our investigations have shown to be most important when looking at differences in XML retrieval models. We investigate two models that build upon existing flat document retrieval strategies, namely a vector space and a language models. As a reference implementation we use the retrieval runs generated by these two models and submitted to INEX 2005. Since then, the fundamentals of the models have not changed much. In Section 7, we use the same reference implementation and its experimental evaluation results to see what a theoretical evaluation adds to the analysis of XML retrieval.

For each model, we look at the corresponding translation into formal situations and the reasoning rules the model incorporates. In a full theoretical evaluation, we would cover over 20 rules but for the purpose of this paper, we focus on the Transitivity, Reflexivity, Symmetry and Monotonicity. These often form the basis of a theoretical evaluation.

4.1 XML Vector Space Model

A vector space model for XML retrieval is presented in [17]. Here, XML documents are split into several disjoint indexes of components, which have in INEX proven to be most

'informative', as the authors call it. 'Informative' elements are simply those most often chosen by human judges in their assessments of relevant document components. In the model, structure is therefore addressed by using different indexes for 'informative' elements. Thus, we reuse the standard vector space model but need to change the indexing unit from a complete document into separate document components. The query (Q) vector is compared to a document component (D) vector instead of a complete document vector:

$$rsv(D,Q) = \frac{\sum_{t_i \in \{Q \cap D\}} w_Q(t_i) * w_D(t_i) * idf(t_i)}{||Q|| * ||D||} \quad (1)$$

$$idf(t) = log(\frac{|D|}{|D(t)|})$$

where $w_Q(t) = \frac{log(TF_Q(t))}{log(AvgTF_Q)}$ and $w_D(t) = \frac{log(TF_D(t))}{log(AvgTF_D)}$. $|D|$ stands for the number of components in the collection and $|D(t)|$ for those containing t. $||Q||$ and $||D||$ are the numbers of unique terms in Q and D, respectively. Both are scaled by the average document length in the collection [17]. The obtained scores are modified using an Automatic Query Refinement (AQR) approach based on Lexical Affinity (LA). LA describes pairs of terms where exactly one of the pair is part of the query and both appear close in relevant documents. They are chosen according to the degree they separate relevant from non-relevant document components. LA's with the highest information gain are found, added to the query Q, and the final retrieval status values *rsv* are recalculated. We do not go into the details of the calculations of the information gain, but rather focus on how it impacts on the aboutness decision. After calculating the ranking based on the vector space model, applying AQR and further normalisation steps, the result sets are merged into a single result set of all element types.

Structure is used in the model mainly to allocate XML elements across different indexes. Each of the elements is inserted into the index separately as a paragraph, an article, a section, etc. The translation is limited to those infons of document components that have shown to be most 'informative' in previous INEX years such as 'section (ss)' or 'paragraph (p)'. Therefore, the translation is defined as follows. Let D be a document component, e be an element type, i be an identifier, and t be a value of an element:

$$map(D) = \{\langle\langle ElementType, e, i\rangle\rangle, \langle\langle Value, t, i\rangle\rangle | e \in \{article, abs, sec, ss1, ss2, p, p1\}\}$$

For the aboutness decision, let Q be a query and D be a document component. As each XML element is inserted separately in the index, the main difference to flat vector space retrieval is that we consider elements instead of full documents. The *XML retrieval vector space aboutness decision* is then defined by:

$$D \ \square\!\!\rightarrow Q \text{ if and only if } rsv(D,Q) \geq n$$

For the AQR only the top N documents are considered. We call the value that has to be reached in order to be part of the top N documents n. Thus, the model implements thresholded vector space retrieval [25]. The final retrieval status value $rsv(D,Q)$ includes all the steps described above and some further refinements of the component score, which are currently of no interest here.

Regarding the reasoning rules, we concentrate on LMU and briefly cover Reflexivity, Transitivity and Symmetry for the purpose of demonstrating some distinctive features of the model. Reflexivity is clearly given, as we always have $rsv(D,D) \geq n$. The model is also symmetric. Indeed, $rsv(D,Q) > n$ is equivalent to $rsv(Q,D) > n$ as in Equation 1 the numerator of the retrieval status value fraction is the sum of products depending on Q and D

and the denominator is a purely symmetrical product. Vector space XML retrieval, however, does not support Transitivity. We can easily construct an example so that $rsv(D,Q) \geq n$ and $rsv(D',Q) \geq n$ but not $rsv(D,D') \geq n$. The terms t_i in Equation 1 responsible for $rsv(D,Q) \geq n$ and the terms t_j responsible for $rsv(D',Q) \geq n$ do not have to be overlapping so that $rsv(D,D') \geq n$. As the aboutness decision is based on overlap of terms t_i and t_j, Euclid is not given either.

Next, we derive the reasoning behaviour for LMU. Does for any situations S and T, $S \boxdot\!\!\rightsquigarrow T$ imply that $S \otimes U \boxdot\!\!\rightsquigarrow T$? LMU is conditionally given. We split the discussion of the involved calculations into two steps, the first includes the vector space based relevance calculation rsv and the second the AQR step. For the first step, let us assume that $S \equiv map(A)$, $T \equiv map(B)$ and $U \equiv map(C)$. Then, $S \boxdot\!\!\rightsquigarrow T$ means that $rsv(A,B) \geq n$. We are not interested in the details of the top part of the rsv fraction and rewrite $rsv(A,B) = \frac{f(AB)}{||A||*||B||} \cdot$ $||.||$ stands for the number of unique infons, whereas $f(AB)$ describes a function dependent on the informational overlap of A and B: $A \cap B \not\equiv \emptyset$. This is the case, as the product $w_Q(t_i) * w_D(t_i) * idf(t_i)$ is only > 0, if none of its expressions is 0, because there is an overlap in information between D and Q.

It is typical for a theoretical evaluation to relate the behaviour of the elements of the rsv function to each other. This way we can find out when and how the threshold is missed and therefore aboutness is given or not given. The value $f(AB)$ increases if either the information overlap in A and B is larger or if the frequencies for A and B are much higher than the averages, or if the elements in the overlap of A and B are not spread out across too many documents. Then, the overall behaviour of $rsv(A,B)$ is determined by the size of $f(AB)$ in relation to the number of unique terms in A and B. It is clear that the more unique terms a document component has and the less it has a significant overlap with the query, the more difficult it will be to pass the threshold n. This is appropriate for structured document retrieval, as document components focussed on specific topics will have less unique terms. The newly added information on the left side of the aboutness relation can, however, have many unique terms. Then, extending $f(AB)$ to $f(ABC)$ could have a negative effect and the threshold would be missed.

For our discussion of the XML retrieval specific effects of the reasoning, we reuse Chiaramella's distinction of exhaustivity and specificity. We look at the impact on $D \boxdot\!\!\rightsquigarrow Q$ and $Q \boxdot\!\!\rightsquigarrow D$, respectively. For exhaustivity ($D \boxdot\!\!\rightsquigarrow Q$), the need to pass the threshold to maintain aboutness with LMU, implies that new document information is added and this document information is not too general, as otherwise $||D||$ would increase. Regarding specificity ($Q \boxdot\!\!\rightsquigarrow D$), if we add new query information, this new information has to specify the existing one in order to maintain $Q \otimes Q' \boxdot\!\!\rightsquigarrow D$ for LMU. An increasing number of unique query terms would be a sign of a loss of focus. The rsv function therefore uses a threshold mainly to emphasize focussed answers.

On top of the rsv function, AQR is applied. We consider AQR to be a second reasoning step. Say, Q_{AQR} is the added query part. The refined scores deliver a larger overlap for those document components that were relevant in the first step. This emphasis on relevant documents has, however, only a positive effect as long as the newly added information does not contribute to a significant increase of $||Q||$, which can easily happen, as per assumption only information not present in the original query is to be added. So, $||Q||$ has to increase. This increase can, however, be outweighed to the desired effect if $f(ABC)$ is large enough. Being dependent on the size of the information overlap in D and Q ($|D \cap Q|$) and their number of unique terms ($||D||$ and $||Q||$), LMU is only conditionally supported.

LMU is only conditionally supported. This control of monotonic reasoning ensures a more conservative approach to monotonicity, which means that aboutness is only preserved under certain conditions [25]. This allows for better control of this important quality and adds to the model's convincing performance in the INEX campaigns [12]. However, the condition is chosen a priori by setting N, and is external to the aboutness decision. Also, it does not relate to the quality of the information overlap but to the number of unique terms in relevant XML elements.

Another monotonic reasoning property is Mix as a specific variant of LMU. It makes sure that the added information is about the target situation. Let us assume that $S \equiv map(A)$, $T \equiv map(B)$ and $U \equiv map(C)$. Mix implies that $rsv(A,C) \geq n$ as well as $rsv(B,C) \geq n$. The latter condition means that in the fraction in Equation 1 only new terms are added that are part of $Q \cap D$. Under no circumstances can therefore the sum in the denominator decrease. But still, this does not change that the constants $||Q||$ and $||D||$ might increase so that $rsv(A,B,C) < n$. This demonstrates that Mix is only conditionally supported as well as that the control of the monotonic behaviour for the XML vector space retrieval model is not dependent on the actual information overlap.

4.2 Language Models

Another approach to XML retrieval that performed well at INEX is that of [20]. With language modelling, another successful approach to flat document retrieval is used. A language model for each document component is calculated by interpolating element ($P_{mle}(t_i|e)$), document ($P_{mle}(t_i|d)$) and collection ($P_{mle}(t_i)$) language models:

$$P(t_i|e) = \lambda_e * P_{mle}(t_i|e) + \lambda_d * P_{mle}(t_i|d) + (1 - \lambda_e - \lambda_d) * P_{mle}(t_i)$$

This model is built on the decision that an element D is about a query Q if and if only the information in Q can be found in the element (its representation). We have different aboutness decisions depending on which elements are actually indexed, e.g. all the elements, those above a given size, those that correspond to types frequently assessed as relevant, or a specific type of elements only (e.g. section). We therefore must provide a *map* function to translate infons for each chosen approach. We only demonstrate 2 of them, as the others are built very similarly. Let D be a document component with term t and an element type e:

- The length based index is limited to elements with a size larger than κ: $map_{length}(D) \equiv \{\langle\langle ElementType, e, i\rangle\rangle, \langle\langle Value, t, i\rangle\rangle | |D| > \kappa\}$ where κ is a threshold to eliminate too small elements.
- The section index only stores sections: $map_{sec}(D) \equiv \{\langle\langle ElementType, e, i\rangle\rangle, \langle\langle Value, t, i\rangle\rangle | e \in \{sec\}\}$.

Apart from the article index, the main difference to a flat document language model is the division into document components instead of documents. This was the same for the vector space model and is a popular method throughout INEX to adjust flat document retrieval models to the requirements of XML retrieval.

The *XML language model retrieval aboutness decision* is the same as for the flat document language model: $D \boxdot\!\!\rightsquigarrow Q$ if and if only $P(t_i|e) > \theta$. The threshold θ is the smoothing value, which is the collection language model $(1 - \lambda_e - \lambda_d) * P_{mle}(t_i)$. Contrary to the vector space model threshold, it is internal to the aboutness decision, as it is dependent on the overall distribution of terms in the collection.

It can be shown that the proposition $P(t_i|e) > \theta \Leftrightarrow A \cap B \not\equiv \emptyset$ holds, which means that the model is essentially built on information overlap. We omit the proof here. Huibers [13] has shown that this means that the model supports Symmetry and Reflexivity. Because it is based on overlap, the model also does not support Transitivity, as the following counter-example using the section index demonstrates. Let us assume that $S \equiv \{\langle\langle ElementType, Section, i_1\rangle\rangle, \langle\langle Value, House, i_1\rangle\rangle, \langle\langle Value, Garden, i_1\rangle\rangle\}$, $T \equiv \{\langle\langle ElementType, Section, i_2\rangle\rangle, \langle\langle Value, House, i_2\rangle\rangle, \langle\langle Value, Car, i_2\rangle\rangle\}$ as well as $U \equiv \{\langle\langle ElementType, Section, i_3\rangle\rangle, \langle\langle Value, Garage, i_3\rangle\rangle, \langle\langle Value, car, i_3\rangle\rangle\}$. Then, $S \; \square\!\!\rightsquigarrow \; T$ and $T \; \square\!\!\rightsquigarrow \; U$, but not $S \; \square\!\!\rightsquigarrow \; U$. Transitivity is not given. For similar reasons Euclid is not given either.

Regarding LMU, say that $S \equiv map(A)$, $T \equiv map(B)$ and $S \otimes U \equiv map(C)$. Hence, we have $C \cap B \not\equiv \emptyset$ and $C \supseteq A$. Thus, $A \cap B \not\equiv \emptyset$, and LMU is unconditionally supported. For exhaustivity, this implies that new document component information can be added, and for specificity, new query information, without restrictions. LMU is fully given, if the element remains unchanged but the collection gets extended, as the aboutness decision is based on an interpolation of element, document and collection language models. However, the monotonic extension only takes place on the content level. We cannot add a structurally different element, as this would involve a completely changed aboutness decision, which is not represented in $P(t_i|e)$. Structurally different elements might be part of another index. If, e.g., a paragraph element is added, the section translation could not be used for the aboutness decision anymore. Therefore, LMU is unconditionally given as long as we add information only on the content level. As Mix is a special case of LMU with an additional condition, it is fully supported, too.

This XML language model demonstrates the power of relatively simple adjustments to flat document retrieval models. However, it lacks a means to express structure. In particular, we do not know which XML elements are related to each other. For the model, generally the same rules hold as for flat document language model retrieval systems. A generalization of the model to other XML structures than the one predominant in INEX is not easily possible, as the indexes are based on element types specific to the collection(s) used at INEX. XML language modelling uses, however, an internal threshold to control its aboutness behaviour, which allows it to be adjusted well in specific collections. The XML language model therefore performed relatively well at INEX, as we discuss in Section 7.

Table 1 summarizes the results of our theoretical evaluation for both models. We see that XML language modelling and XML vector space retrieval only differ in the fact that XML vector space retrieval controls the monotonic behaviour of LMU and Mix by adding conditions. Otherwise, the models have similar reasoning characteristics, as both are essentially based on information overlap between D and Q.

The two models described in this section performed better for the thorough retrieval task than for the task aiming at returning the most focussed elements, i.e. the focussed retrieval task. In the rest of the paper, we provide a theoretical explanation for this behaviour by investigating specificity aboutness realized through XML retrieval filters in more detail. For this purpose, we present an addition to our theoretical methodology in Section 5, which allows us to describe the reasoning of two INEX filters in Section 6. In Section 7, we use our theoretical analysis of filters to explain some experimental results obtained at INEX 2005.

5 Defining specificity aboutness

In this section, we describe our theoretical methodology to evaluate filters, which we then use to understand how models achieve specificity aboutness on top of their underlying gen-

XML Retrieval Model	*Supported*	*Not Supported*	*Conditionally Supported*
XML Vector Space Retrieval	Reflexivity		
	Symmetry		
		Transitivity	
		Euclid	
			LMU, dependent on $\|D \cap Q\|$ as well as $\|\|D\|\|$ and $\|\|Q\|\|$
			Mix, dependent on $\|\|D\|\|$ and $\|\|Q\|\|$
XML Language Modelling	Reflexivity		
	Symmetry		
		Transitivity	
		Euclid	
	LMU		
	Mix		

Table 1 Summary of theoretical evaluation results

eral reasoning behaviour, as analysed in the previous section. We rely on some initial work by Huibers on the relationship between the filter aboutness system (characterizing the focussed task) and the corresponding underlying aboutness system (characterizing the thorough task) [13], which we adapt to the requirements of XML retrieval. We go beyond his work by applying his theoretical work to analyze two filters developed at INEX in Section 6.

As already explained, the task of finding the most focussed elements consists of filtering the ranked result list produced by an XML retrieval model like the ones described in Section 4. Generating this ranked result list is itself based on an aboutness decision system, which characterizes the model used to deliver that list. Thus, with filtering, a further aboutness decision is applied, one which removes overlapping elements from the result list.[6]

Huibers [13] describes that one aboutness decision system is a filter to another aboutness decision system if the two corresponding aboutness systems are embedded — meaning their reasoning behaviour is related by supporting the same or sufficiently similar properties. In the context of XML retrieval, this translates to having to relate the aboutness decision system associated with the model for the focussed task to that of the underlying aboutness system associated with the model used to generate the ranked list (the thorough task) to then be filtered.

The theoretical analysis of filter is done in three steps. We first formalize the translation process, as we did in Section 4. Secondly, we identify the reasoning rules associated with the filter. Finally, we analyse the relationship between the filter and the underlying aboutness systems. For the later, we make use of the filtering function f-*answer* defined in [13], which we adapt to XML retrieval:

Definition 1 *Let A_p, B_p be aboutness systems and $\mathscr{D}$ be a set of documents and Q be a query. The filtering function f-answer of A_p with respect to B_p is defined by: f-answer$(A_p;B_p;Q;\mathscr{D})$ = answer$(A_p;Q;$answer$(B_p;Q;\mathscr{D}))$, where answer describes a function that delivers an answer set from the set $\mathscr{D}$ based on query Q.*

[6] It should be pointed out that the use of filters is not exclusive to XML retrieval. Filters are used in IR to improve performance [13], if, for instance, at first a fast but less accurate approach is used to identify relevant documents from a very large set documents, and then a second retrieval system is used to search the initial result set more accurately. Pseudo-relevance feedback and passage retrieval are examples of such a process.

Using this definition, we can investigate the filtering process by looking at the relationship between f-*answer* and *answer*. Without going into detail, Huibers identified three important distinctions between f-*answer* and *answer* [13]:

- A filtering function f-$answer(A_p;B_p;Q;\mathscr{D})$ is called *useless* if for all sets of documents $\mathscr{D}$ and queries Q f-$answer(A_p;B_p;Q;\mathscr{D}) = answer(B_p;Q;\mathscr{D})$. An example of a useless filter is the application of the coordinate retrieval model as a filter to an answer set generated by simple vector space retrieval, as both are based on the same aboutness decisions, according to which a document D is about a query Q if both share information.
- The aboutness systems A_p and B_p are said to be *f-equivalent* if and only if f-$answer(A_p;B_p;Q;\mathscr{D}) = answer(A_p;Q;\mathscr{D})$. An example of an *f-equivalent* filter is to use strict coordinate retrieval to filter a result set generated by vector space retrieval. Strict coordinate retrieval defines that a document D is about a query Q if and only if the information items of Q are a subset of the information items in D. This delivers a subset of the answer set from simple vector space retrieval, for which D is about Q if they share information. Strict coordinate retrieval therefore fully determines the final answer set.
- A_p and B_p are said to *intersect* if and only if the filter is neither useless nor f-*equivalent*.

In our analysis of the relationship between f-*answer* and *answer*, we first determine whether a filter is 'useless', i.e. the filtering function does not change the original answer set. If this is not the case, next we investigate whether the filter f-*answer* uses f-*equivalent* aboutness systems. We call a filter aboutness system to be f-*equivalent*, if its A_p alone will determine the final result set. If the filter is not useless and not f-*equivalent* with regard to the underlying aboutness system, we then define how the filter and underlying aboutness system 'intersect' by comparing their aboutness properties.

The following section demonstrate the presented methodology for the analysis of two filers at INEX.

6 Applying specificity aboutness at INEX

Two main types of approaches have been proposed for the focussed task at INEX: a simple one that keeps the highest ranked element of each XML path, and a more complex one that takes into account the relations in the tree hierarchy between retrieved elements. For each of the filters, we go through the three steps described in Section 5.

6.1 Brute-force filter

One method of removing overlap in the result set of an XML retrieval model has been referred to as 'brute-force filter', because only the highest ranked element from each of the paths is selected. The advantage of this filter is that it is relatively easy to implement and that it can be used on top of any kind of underlying aboutness system.

6.1.1 Aboutness decision

The aboutness decision of brute-force filtering can be defined by:

$$D \text{ about } Q \text{ if and only if } rsv(D,Q) = max(rsv_u(D,Q))$$

Here, $max(rsv_u(Q,D))$ is delivering the XML element with the maximum retrieval status value for the underlying aboutness system. For the translation, let D be a document component, e_i an element type, k_i a value in an element, and i an identifier to enumerate all $\{1,...,n\}$ elements in an XML tree in a depth-first traversal manner:
$map(D) = \{\langle\langle ElementType, e_1, i_1\rangle\rangle, \langle\langle ElementType, e_2, i_2\rangle\rangle, \langle\langle Parent, i_1, i_2\rangle\rangle, ..., \langle\langle Element Type, e_n, i_n\rangle\rangle, \langle\langle Parent, i_{n-1}, i_n\rangle\rangle, \langle\langle Value, e_n, k_1\rangle\rangle, ..., \langle\langle Value, e_n, k_n\rangle\rangle\} | \forall i_i \in \{\langle\langle Parent, i_i, i_k \rangle\rangle\}, count(i_i) = 1\}$.
The translation expresses that we only consider elements on the same XPath, meaning each element is the parent and the child of exactly one other element, unless it is the root or leaf element.

6.1.2 Reasoning behaviour

We now analyse the functional behaviour of brute-force filtering using the reasoning rules from Section 3. Reflexivity holds for brute-force filtering. A maximum element will be about itself. For Symmetry, there is no contradiction in the statement that if $S \boxdot\!\!\rightsquigarrow T$ or S is the highest ranked answer to T then also $T \boxdot\!\!\rightsquigarrow S$ or T will be the highest ranked answer to S. It is possible that the same aboutness system allows this. Brute-force filtering is symmetric.

Let us use the example of Symmetry to look at how the second aboutness system of filters influences underlying aboutness systems. As Symmetry is fully supported, it does not change the underlying aboutness behaviour. All systems, for instance, that are based on overlap aboutness, are symmetric. Examples included the XML vector space retrieval model and language modelling presented in Section 4. All these systems remain symmetric if brute-force filtering is applied on top of them. The underlying asymmetric aboutness systems remain asymmetric. This means that those reasoning properties that are fully supported by a filter do not change the underlying aboutness behaviour of the system that produced the original ranking. If, however, a reasoning property such as Symmetry is not fully supported by a filter or, for instance, a threshold is applied, then this changes the underlying aboutness behaviour, as we see next when we discuss Transitivity.

More interesting are those reasoning rules that are not supported. The Transitivity rule, for instance, is not supported, as two situations cannot be the maximum scoring answers towards the same query. If T is the maximum scoring answer to U, S cannot be the maximum scoring answer to U, too. This means whatever the status of Transitivity in an aboutness system, if we apply brute-force filtering on top of it, it will not be supported. The same applies for Euclid from Section 3. If S is the maximum scoring answer to U, how could T be the maximum scoring answer to the same U, too? This means Euclid is never supported.

Mix is another rule that cannot be supported. It states that with the assumptions $S \boxdot\!\!\rightsquigarrow U$ and $T \boxdot\!\!\rightsquigarrow U$, we can also say that $S \otimes T \boxdot\!\!\rightsquigarrow U$. S and T, however, cannot be at the same time the maximum answer to U. The assumptions contradict each other. LMU would imply in the context of brute-force filtering that if one extends S to $S \otimes U$ and aboutness would be preserved for both, both S and $S \otimes U$ would be maximum scoring answers, which is a contradiction. This means LMU is not supported either.

All the rules analyzed in this section are important in the analysis of XML retrieval models' behaviour [3]. When we analyze the experimental results related to brute-force filtering in Section 7, we see the impact of excluding the rules' reasoning behaviour.

6.1.3 F-answer

In this section, we look at the relation between f-*answer* and *answer*. First, we need to show that the brute-force filter is not *useless*. Formally, this can be proven by demonstrating that the aboutness systems of filter and underlying system differ in at least one aboutness characteristics. This can be either a rule or a condition. We have just seen that brute-force filtering disallows LMU, Transitivity, etc., which means it is not useless as a filter for both XML retrieval models presented in Section 4. These models at least conditionally support LMU. As $max(rsv_u(D,Q))$ is dependent on the underlying retrieval status value rsv_u, brute-force filtering is also not f-*equivalent*.

As the filter is neither useless nor f-*equivalent*, neither brute-force filtering nor the underlying aboutness systems from Section 4 fully determine the outcome of combining both. They 'intersect'. If we therefore look at the differences in reasoning behaviour, the filter creates, it is key that LMU reasoning is excluded. This will change any aboutness system that follows the strict structural constraints of XML documents. If an element is a child, it will share information with its parent. This means for language modelling as presented in Section 4.2, for instance, that both are about the same queries. However, such aboutness due to overlap in (redundant) information is what is supposed to be excluded by brute-force filtering.

We analyze the impact of brute-force filtering on the experimental results in INEX 2005 in Section 7, but let us first look at an alternative approach to dealing with overlapping elements.

6.2 Controlling the overlap via re-ranking

The next approach [8] we present re-ranks elements with a new context-dependent retrieval status value, but not entirely eliminate overlapping elements. The approach is based on iteratively reducing the score of those elements that contain highly relevant elements (which would have been retrieved at higher ranks). The input into the re-ranking method is a list of XML elements x. These are each associated with $x.\overrightarrow{f}$ as the term frequency vector per query term and with $x.\overrightarrow{g}$ as the adjustment vector, and other information required to process the algorithm such as the set of children per element. The adjustment of each term x_t is based on $x_t = f_t - \alpha * g_t$, where α is an adjustment weight. For parents y containing a highly ranked child x their adjustment score $y.\overrightarrow{g}$ is increased. As the information of children of highly ranked parents has already been considered in the reported parent element, its $x.\overrightarrow{g}$ becomes $y.\overrightarrow{f}$. The tree is traversed until all elements are covered and re-ranked.

6.2.1 Aboutness decision

Regarding the translation, we need to take into account that the complete XML tree structure is preserved in the index [8]. To represent the complete XML tree structure in Situation Theory, we propose to translate XML elements together with their values into infons. If these elements are connected, they form a situation. To represent these connected situations, we use the work by Brown et al. [4], who showed that the denotations of XML schema types can be organized into a lattice structure of types. Then:

- For each XML element p with a type U, *map* is defined as $\{\langle\langle ElementType, U, p\rangle\rangle\}$.

- For each XML element p with a type U containing term k, map is $\{\langle\langle ElementType, U, p\rangle\rangle, \langle\langle Value, k, p\rangle\rangle\}$.
- Say R is a relation between two XML trees A and B. Let E_1 and E_2 be element types and $\{\langle\langle ElementType, E_1, p\rangle\rangle\} \in map(A)$ and $\{\langle\langle ElementType, E_2, q\rangle\rangle\} \in map(B)$. We can then say that $map(R(AB)) = \{\langle\langle R, p, q\rangle\rangle\} \cup map(A) \cup map(B)$.

This definition of *map* reflects the fact that each XML element has an element type infon, which is expressed with the relation $ElementType$ of two arguments, the name of the element type and an identifier p. Content infons can only be found as $Values$ of element types. Situations can be combined with $\cup$ to form larger situations. Using this translation, the aboutness decision of this re-ranking approach is described by:

$$D \text{ about } Q \text{ if and only if } rsv_{adjusted}(D,Q) > 0$$

The re-ranking does not use a threshold, because no element is filtered out unless the adjusted score becomes 0.

6.2.2 Reasoning behaviour

The first reasoning property we look at is Reflexivity, which as seen in Section 3 states that $S \sqsubset\!\!\sim S$. Reflexivity is not given. With $S \sqsubset\!\!\sim S$, then $f_t = g_t$. If in $x_t = f_t - \alpha * g_t$, $\alpha = 1$ [8], then $x_t = f_t - 1 * g_t$, which means $rsv_{adjusted} = 0$, with $f_t = g_t$. Thus, Reflexivity is not supported. However, Reflexivity is a special case and the exception. Generally, re-ranking does not fundamentally change the aboutness decision of the XML retrieval models but adds emphasis to the ranking of elements. For our analysis of the impact of filters we therefore need to relate it to the models we have presented in Section 4 directly. For these models, Symmetry, Transitivity and Euclid behaviour will not be changed. All of these are either fully or not at all supported.

We concentrate on LMU, as it is a reasoning property controlled by thresholds (see Section 4). LMU would be given if $S \otimes U \sqsubset\!\!\sim T$ and $S \sqsubset\!\!\sim T$ are given. Regarding the XML vector space model, re-ranking with $x_t = f_t - \alpha * g_t$ can reduce the extension to fall below the threshold n. This mainly affects the children of the highly ranked parents. LMU is only conditionally supported if re-ranking does not lower the retrieval result to fall below n. The language modelling approach in Section 4.2 forms an interesting case. Its internal threshold based on the smoothing value might be missed if the added information leads to a re-ranking below the smoothing value. Therefore, applying re-ranking on top of language modelling means that LMU is now conditionally supported, while language modelling alone fully supported LMU. Similar arguments apply to Mix for XML vector space retrieval. Its condition is also influenced by the re-ranking. Mix reasoning behaviour, however, does not change for language modelling, as here it is fully supported.

6.2.3 F-answer

Re-ranking is certainly not *useless*, because the LMU thresholds for vector space retrieval and language modelling have been changed. It is not f-*equivalent* either, as it is dependent on the underlying aboutness decision, because re-ranking is a function of the original retrieval status value. Thus, re-ranking is also 'intersecting'. Reflexivity is changed through the impact of α. That Transitivity and Mix behaviour is preserved is a clear advantage towards the brute-force filtering approach, as both are important properties of XML retrieval

behaviour [2]. In particular, the support for Mix adds to the better performance of the model in the experimental results.

Table 2 summarizes the results of our theoretical evaluation for both filters. As discussed, for filters we have to discriminate three cases: (1) If they support a reasoning property, they will not change the underlying reasoning behaviour. (2) If they conditionally support a reasoning property, they will add a condition to it or change the existing one. (3) If they do not support a reasoning property, they will eliminate it from the overall aboutness behaviour. Looking then at a comparison of Table 1 and Table 2, we can identify the strong impact of brute-force filtering on the two XML retrieval models presented in Section 4. In the next section, we look at how this impact is materialised in the experimental behaviour observed at INEX 2005.

XML Retrieval Filter	*Supported*	*Not Supported*	*Conditionally Supported*
Brute-force	Reflexivity		
	Symmetry		
		Transitivity	
		Euclid	
		LMU	
		Mix	
Re-ranking		Reflexivity	
	Symmetry		
	Transitivity		
	Euclid		
			LMU
	Mix		

Table 2 Summary of filter evaluation results

7 Impact of filters on experimental behaviour at INEX 2005

To analyse the impact of filters on experimental behaviour, we proceed as follows. First, in Section 7.1, we introduce the background, in particular the evaluation metrics used at INEX 2005 and the reasoning behaviour associated with these metrics. In Section 7.2, we discuss how the changes in reasoning behaviour impact on the experimental outcomes.

We concentrate on the impact of brute-force filtering on the experimental behaviour in INEX 2005. Looking at the second filter, re-ranking, it is difficult to make general statements regarding its impact on XML retrieval, as it has been developed for a particular model [8]. The authors, however, report limitations of their algorithm according to their experimental evaluation [8]. From a theoretical evaluation point of view, an immediate recommendation on how to potentially improve the approach would be to introduce a threshold to control the monotonic behaviour of the re-ranking aboutness decision. Indeed, only if $rsv_{adjusted}(D,Q) > \theta$, the element would be reported. We have seen in Section 4.1 how thresholds effectively constrain monotonic behaviour and improve performance.

The impact of brute-force filtering has been noted in various retrieval runs submitted to INEX. For instance, the XML vector space retrieval model has been overall very successful in the experimental evaluation in INEX 2005 [17], but its performance decreases if only non-overlapping document components were supposed to be returned, ranked according to how

specific they are to the query. XML vector space retrieval implemented these tasks by using various filters. Particularly, in the run that used simple brute-force filtering, the performance was much worse. The situation is similar for the XML retrieval model based on language modelling [20]. Its performance decreases, too, when brute-force filtering is used to filter the original language modelling retrieval results. Our theoretical evaluation can provide an explanation for this decrease in performance.

7.1 INEX 2005 metrics and associated reasoning behaviour

To understand why the reasoning implied by brute-force filtering leads to a performance decrease, it is important to understand first more about the overall aims and metrics of the evaluation strategy adopted at INEX 2005. According to the INEX focussed evaluation task, retrieval systems were rewarded most for the return of focussed XML elements, i.e. those at the right level of granularity. Overlapping elements should not be returned. For the focussed task, an official evaluation metric was the eXtended Cumulated Gain (XCG) metrics [16], which are based on the cumulated gain (CG) metrics proposed in [15]. XCG consider the dependency of XML elements (e.g. overlap and near-misses) within the evaluation. One of these metrics is the normalized extended cumulated gain (*nxCG*). Given a rank i, the value of $nxCG(i)$ reflects the relative gain the user accumulated up to that rank, compared to the gain he or she could have attained if the system would have produced the optimum ranking. Better performance means therefore that systems are able to return only the most relevant elements early.

Looking at this evaluation strategy, two changes of reasoning properties are highly conclusive, if we try to understand why brute-force filtering decreases performance for the focussed task. Both LMU and Mix reasoning are not supported by brute-force filtering. Both are interesting, particularly from the perspective of good results under *nxCG*, as both are reasoning rules that preserve aboutness if information is added. Mix demands that the added information is also about the query, whereas LMU does not constrain what the added information might be. A full support for LMU is therefore counterproductive if the aim is to deliver only the most relevant elements early.

As an example, we look next at how the elimination of monotonic reasoning, in particular LMU and MIX reasoning, helps explain experimental results under *nxCG*. First, we compare the performance of two retrieval systems based on support for LMU and changes to it through filtering. Second, we investigate the impact of LMU changes on one particular model. Finally, we look at the general consequences of eliminating Mix in the reasoning.

7.2 Changes in reasoning behaviour

LMU support can first explain differences in the results for XML language modelling and XML vector space modelling. Here, it comes to no surprise that XML language modelling performs overall worse than XML vector space retrieval under *nxCG*, as the latter successfully used conditions on LMU reasoning to adjust the behaviour of flat document vector space retrieval to the requirements of XML retrieval. Especially, the left monotonic behaviour is controlled by disallowing the addition of information, as seen in Section 4. However, this control over left monotonic reasoning is lost for XML vector space modelling, once the brute-force filter is applied, as seen in Section 6.1.2. LMU reasoning is completely eliminated. The XML vector space retrieval model performance therefore decreases, and it

loses its advantage towards XML language modelling, which also does not support LMU anymore after applying the brute-force filter. Thus, differences in support for LMU reasoning help explain different experimental results for XML vector space and XML language modelling.

Looking at the impact of brute-force filtering on one individual model, we now investigate in more detail the impact of LMU reasoning changes to XML language modelling. In particular, we want to answer the question why the performance of XML language modelling is worst for higher ranks and relate this to the way structure is considered in the aboutness decision. We have identified two advantages of the language modelling approach compared to the vector space model in Section 4.2. First, some structural context of an XML element is taken into account, because the language model is interpolated with collection and document model. Second, the threshold in the aboutness decision is internal. However, as this threshold is only related to the overall collection language model, the aboutness decision is still derived from the overlap of information in document components and query, and not really considering structure. We concluded in Section 4.2 that structure is only indirectly considered in XML language modelling.

Three types of retrieval runs based on XML language modelling were submitted to the INEX focussed task in 2005, one for all elements, one for just article elements and one for section elements. Good performance were obtained for the all-element run at lower ranks in *nxCG*, but not for the section and the article runs. In higher ranks even the performance for the all-element run fell behind the performance of other models [16]. For the lower ranks of the focussed task, it is not surprising that the best performing run for the language modelling approach is the all-element one. As LMU is fully supported, it will return all the most relevant elements first — independent of whether they are articles or sections. If now brute-force filtering is applied, this relatively positive impact of LMU is eliminated for the element run and its performance will decrease for higher ranks, where we would find most of overlapping but still relevant elements.

Using our insight from Section 4.2 that structure is only indirectly considered in XML language modelling, we can finally answer the question why the performance of XML language modelling is worst for higher ranks in the elements' index. One reason is that the model is not able to deliver in the lower ranks those elements that have similar content to highly ranked XML elements but might be on a different XML path. As seen in Section 4.2, XML language modelling uses structure only to allocate elements into several different indexes and not in the actual aboutness decision. The brute-force filter, which is applied on top of the language modelling aboutness model, will be based solely on the overlap in information and not the structural relatedness of two elements. The XML language modelling aboutness decision only relates XML elements according to the information overlap, and brute-force filtering is dependent on this underlying aboutness decision, as they are intersecting according to Section 6.1.3.

To elucidate the negative impact of a brute-force filtering based only on information overlap, let us assume that the all-element index is used, and that we have two document components $D1$ and $D2$ on two unrelated XML paths, and $D1$ has achieved a higher *rsv* for language modelling than $D2$. Let us further assume that $D1 \equiv D2 \otimes D2'$. Then, both $D1$ and $D2$ will be about the same query Q. That $D1$ and $D2$ are structurally different does not play any role. Using brute-force filtering, once $D1$ is traversed $D2$ is removed from the (focussed) result list. This means that it is not delivered as an alternative answer and *nxCG* performance decreases for higher ranks. This effect is particularly noticeable in higher ranks, as the relevant XML elements, which have been eliminated through the brute-force filter, would have been found here.

Finally, let us examine the impact of the elimination of the Mix reasoning rule from the aboutness reasoning through brute-force filtering. We can generally say that its elimination through brute-force filtering leads to a performance decrease for the focussed task. The unwanted side effects of LMU, that also irrelevant information might be added, do not apply here. To show the desired reasoning behaviour that is eliminated with Mix, let us consider the following example. Among other things, Mix describes that, if two children $D1$ and $D2$ are about a query, then their parent $D1 \otimes D2$ are also about the same query. This behaviour is typical to XML based reasoning. If it is not supported, problems might arise, such as the elimination of potentially highly relevant children. Say, we have one relevant child and a more relevant parent, then the child is eliminated from the result set after applying brute-force filtering. Another child of the same parent that is about the same query, is also eliminated, as the parent is already chosen. However, this child might be highly relevant, too.

8 Conclusion and Future Work

In this paper, we have shown how a theoretical evaluation based on Situation Theory can aid the analysis of filters in XML retrieval. To this end, we first presented the potential of a theoretical aboutness approach for a theoretical evaluation of XML retrieval models. We introduced an aboutness system to demonstrate the reasoning properties and under which conditions these properties are supported. We used this aboutness system to compare the behaviour of two XML retrieval models, both evaluated within INEX. We were able to show commonalities as well as differences between those models. The main commonality was that none of the models presented radically breaks with methods applied in flat document retrieval. The structure is never directly included in the aboutness decision. The main difference in the models then was how they attempt to adjust a flat document retrieval model to the specific requirement of XML retrieval to deliver focussed answers. Here, the control of monotonic reasoning behaviours and other standard reasoning rules like Symmetry and Transitivity, were found to be particularly important.

The analysis of the XML retrieval systems was the basis for our discussion of specificity aboutness and filters in the second part of the paper. We have considered filters as a second layer aboutness decision, which changes the reasoning of the underlying aboutness system, and asked how they influence the underlying aboutness system. We could do so, as we regarded them as aboutness systems. This has led to conclusions about why and how they change the performance of their underlying systems in the experimental evaluation in INEX. Our primary interest has been whether the filters are suitable extensions of the underlying aboutness decision. We showed how particularly brute-force filters significantly change the underlying aboutness behaviour of the retrieval models.

In the future, we will first investigate further how to theoretically describe specificity aboutness and develop additional reasoning behaviour and new rules that define it. We will also continue our work by elaborating our theoretical evaluation and by going into more detail regarding different INEX evaluation criteria. A theoretical evaluation could help understand better the implications of the two measures of exhaustivity and specificity. In INEX, up to 2005, exhaustivity and specificity are drawn together in so-called quantisation functions [16], each corresponding to a particular user behaviour. In earlier work [3], we represented user reasoning in the same formal logical framework as system reasoning. In the future, we intend to bring together user and system reasoning to see how systems behave with respect to particular user expectations.

Acknowledgements

Mounia Lalmas is currently funded by Microsoft Research/Royal Academy of Engineering.

References

1. J. Barwise and J. Perry. *Situations and Attitudes*. Cambridge, MA: MIT Press, 1983.
2. T. Blanke and M. Lalmas. Theoretical benchmarks of XML retrieval. In *ACM SIGIR '06*, pages 613–614, New York, NY, USA, 2006. ACM Press.
3. T. Blanke and M. Lalmas. Theoretical evaluation of XML retrieval evaluation. In *DIR '07*, 2007.
4. A. L. Brown and M. Fuchs. Lattices and documents: A schema language independent model of types for XML and information interchange. In *XML 2002 Conference Proceedings*, Baltimore, 2002.
5. P. D. Bruza and T. W. C. Huibers. Investigating aboutness axioms using information fields. In *ACM SIGIR '94*, pages 112–121, New York, NY, USA, 1994. Springer-Verlag New York, Inc.
6. P. D. Bruza, D. W. Song, and K. F. Wong. Aboutness from a commonsense perspective. *J. Am. Soc. Inf. Sci.*, 51(12):1090–1105, 2000.
7. Y. Chiaramella. Information retrieval and structured documents. In *Lectures on information retrieval*, pages 286–309. Springer-Verlag, New York, 2001.
8. C. L. A. Clarke. Controlling overlap in content-oriented XML retrieval. In *ACM SIGIR '05*, pages 314–321, New York, NY, USA, 2005. ACM.
9. W. Cooper. A definition of relevance for information retrieval. *Information Storage and Retrieval*, 7:19–37, 1971.
10. K. Devlin. *Logic and information*. Cambridge University Press, New York, NY, USA, 1991.
11. N. Fuhr, M. Lalmas, S. Malik, and G. Kazai, editors. *Advances in XML Information Retrieval and Evaluation, INEX 2005, Dagstuhl*, volume 3977 of *Lecture Notes in Computer Science*. Springer, 2006.
12. N. Gövert, G. Kazai, N. Fuhr, and M. Lalmas. Evaluating the effectiveness of content-oriented XML retrieval. *Journal of Information Retrieval*, 9(6):699–722, 2006.
13. T. W. Huibers. *An Axiomatic Theory for Information Retrieval*. Universiteit Utrecht, Utrecht, 1996.
14. W. J. Hutchins. The concept of "aboutness" in subject indexing. pages 93–97, 1997.
15. K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.
16. G. Kazai and M. Lalmas. *INEX 2005 Evaluation Measures*. Springer Berlin/Heidelberg, 2006.
17. Y. Mass and M. Mandelbrod. Using the INEX environment as a test bed for various user models for XML retrieval. In Fuhr et al. [11], pages 187–195.
18. B. Piwowarski and M. Lalmas. Structured information retrieval and quantum theory. In *3rd Quantum Interaction Symposium, DFKI, Saarbruecken*, 2009.
19. F. Sebastiani. On the role of logic in information retrieval. In *Information Processing and Management*, pages 1–18, 1998.
20. B. Sigurbjörnsson and J. Kamps. The effect of structured queries and selective indexing on XML retrieval. In Fuhr et al. [11], pages 104–118.
21. C. J. van Rijsbergen. Towards an information logic. In *ACM SIGIR '89*, pages 77–86, New York, NY, USA, 1989. ACM.
22. C. J. van Rijsbergen and M. Lalmas. Information calculus for information retrieval. *J. Am. Soc. Inf. Sci.*, 47(5):385–398, 1996.
23. C. J. v. van Rijsbergen. *The Geometry of Information Retrieval*. Cambridge University Press, 2004.
24. E. M. Voorhees and D. K. Harman. *TREC: Experiment and Evaluation in Information Retrieval*. Digital Libraries and Electronic Publishing. MIT Press, September 2005.
25. K.-F. Wong, D. Song, P. Bruza, and C.-H. Cheng. Application of aboutness to functional benchmarking in information retrieval. *ACM Trans. Inf. Syst.*, 19(4):337–370, 2001.