FuzzyXPath: using fuzzy logic an IR features to approximately query XML documents.

Ernesto Damiani[1], Stefania Marrara[1], and Gabriella Pasi[2]

(1)Università degli Studi di Milano
Dipartimento di Tecnologie dell'Informazione
via Bramante 65 26013 Crema (CR), Italy
{damiani, marrara}@dti.unimi.it

(2) Università degli Studi di Milano Bicocca
DISCO
Via Bicocca degli Arcimboldi, 8 20126 Milano (MI), Italy
pasi@disco.unimib.it

Abstract. XML has become a key technology for interoperability, providing a common data model to applications. However, diverse data modeling choices may lead to heterogeneous XML structure and content. In this paper, information retrieval and database-related techniques have been jointly applied to effectively tolerate XML data diversity in the evaluation of flexible queries. Approximate structure and content matching is supported via a straightforward extension to standard XPath syntax. Also, we outline a query execution technique representing a first step toward efficiently addressing structural pattern queries together with predicate support over XML elements content.

1 Introduction

Il the last few years, the problem of modeling and querying semi-structured information has been intensively studied by both database and information retrieval research communities, although with a slightly different focus. Database research mostly deals with data which do not conform to a strict database schema, i.e. whose structure is not regular [1]. Information retrieval focuses on documents sharing a basic logical structure, constituted by sub-components (or *sections*). Research on semi-structured information querying also comes in two different flavors: in the context of semi-structured databases, flexible query languages take into account the lack of a rigid schema of the database, thus allowing to enquiry both data and the type/schema [4,8], while in the context of IRSs, modeling flexibility means mainly to take into account the possibility to refer to a non-uniform structure of the documents when formulating queries [16]. Defacto standards for the definition of semi-structured documents such as HTML and XML further enriched this picture by adding some features of hierarchical data models. Today, the XML Infoset is widely employed as a basic data model for semi-structured information, and is now the basic standard for representing

semi-structured documents in Information Retrieval. Also, techniques for querying and updating XML data have been investigated and several standards like XQuery [26] and XUpdate [28] have been proposed. Looking at the amount and diversity of XML data items, it is possible to roughly divide them into two main classes:

- *document-centric* items where XML is used for logical markup of text,
- *data-centric* items where XML is used for exchanging structured data among applications.

For the sake of simplicity, in this work we concentrate on the selection aspect of queries. Namely, we extend the idea originally proposed by two of us in [6], and recently re-stated in a more formal garment in [9], in which fuzzy predicates are introduced into a XPath query to express flexible selection conditions and to perform fuzzy subtree matching. In addition to extending structural properties of flexible queries by means of fuzzy operators, we also outline some content-based properties adopting IR features similarly to the proposal of [16]. Our proposal includes modeling of vagueness and imprecision in XML retrieval by means of the following features:

- index term weighting to produce ranked query results,
- specificity-oriented search for retrieving only the most relevant parts of documents,
- use of vague predicates in query formulation to express flexible constraints on stored data,
- structural vagueness, in order to find close matches for structural query conditions.

The aim of the work is to set the foundations for a flexible XML selection language, *FuzzyXPath*, later to be extended to a fully fledged query language[1], which implements imprecision and vagueness for both structural and content-oriented query conditions. The paper is structured as follows: Section 2 motivates the ideas exposed in this work and proposes a running example that will be used in the rest of the paper, Section 3 gives details on query execution and show how it works by means of an example and, finally, Section 4 exposes our conclusions and outlines our future work.

1.1 Related work

Several approaches have been proposed to introduce flexibility in semi-structured information processing and, in particular, in XML querying. An early technique [14] was based on fuzzy encoding of XML data trees. That method did not attempt to define a query language; rather, it supported introducing new nodes in the XML tree structure in order to carry out fuzzy similarity comparison of

[1] Since the language name originally adopted in [6], *FXPath*, has recently been used with the meaning of *Functional XPath*, in this work we suggest the new name *FuzzyXPath* to overcome confusion.

XML data trees. This approach was later extended in [13] providing some flexibility in content comparison with the concept of XML data *smushing*. A later paper [2] proposed an approach based on XML query rewriting, supporting renaming and deletion of nodes in the query. Hybrid techniques [25] have also been proposed, where XML data are encoded and queries are rewritten. On the one hand, hybrid techniques can provide an accurate computation of the query cost; on the other hand, it is very difficult to implement them because they require ad hoc XML data indexing. A recent approach to this problem [20] proposes a dynamic summarization and indexing method, FLUX, based on Bloom filters and B^+-trees. Also, the work [12] presents an indexing method to execute approximate queries on XML documents taking into account approximation on both document structure and content. The proposed indexing aims to reduce the complexity of finding approximate query patterns, avoiding sequentially scanning all documents in the collection.

Another recent paper [7] proposes a fuzzy-based XML querying system that performs approximate comparisons between XML query and data trees. This technique supports imprecision on data via possibility distributions. However, while the authors claim that their querying system is fully compatible with XML querying standards since the final rewriting is performed in XQuery, their query rewriting is based on a *mediated architecture* called *MIEL++* that requires several rewriting steps and is unlikely to scale well. In [22] the authors propose an approach for approximate query answering in which, instead of working directly on the data, they interpret the structural component of the query by exploiting a reworking of the documents' schemas by means of a *schema matching* process.

The information retrieval angle has been explored in the seminal paper [16], which uses a probabilistic data modeling combined with some IR features in order to query document-centric XML. Of course, several works in the literature had already addressed the problem of defining IR models for structured documents [5,10,8,19,21,27]. They focused on two main aspects: how to index structured documents so as to usefully exploit their structure in their formal representation, and how to define query evaluation mechanisms that can retrieve also document subparts. Passage retrieval is mainly concerned with identify subparts of a text document as retrievable information units. Specifically, passage retrieval aims to identify short blocks of relevant information among irrelevant text [8,17]. In [21] a conceptual model for structured documents has been proposed which supports a query language enabling to retrieve passages based on their context as well as content. Aggregation-based approaches have also been explored for representation and retrieval of structured documents. These approaches estimate the relevance of document subparts based on the aggregation of estimated relevance of their content and of their structurally related parts [3,18,15,23,11]. In order to improve the effectiveness of IRSs some considerable efforts are being spent in trying to define new conceptual models aimed at indexing and querying structured documents [3,24,18,17,27,11]. In [11] the idea of producing a composite representation of structured documents is exploited, enabling focusing retrieval only on some document subparts. In [18] this ba-

sic scheme is enriched by modeling uncertainty in content representation. In [10,21,3] other approaches to representation and retrieval of structured documents are presented. Particularly, in [3] a fuzzy model for defining an indexing mechanism is described which exploits users' feedback to create personalized representations of the same structured document. This model exploits the vagueness in the indexing process and enables the system to learn users needs at the indexing level. This pioneering work has been further developed in [5], which is the starting step of this work.

2 Motivations and Running Example

Thanks to XML standards several application involving data interchange enjoy interoperability at the level of the data model. Nonetheless, they often have to face the lack of shared semantic models which causes different modeling choices. In turn, these different choices cause differences in XML schemata, leading to heterogeneous XML structure and content. Also, XML data stored in repositories constantly evolve over time. Such evolution is a challenge, as existing applications must continue to work with the evolved data. In our approach, flexible querying of XML data is aimed at tolerating schema-level heterogeneity due to either different modeling choices or to data evolution, allowing for posing the same query to multiple, possibly heterogeneous XML document collections loosely sharing the same semantics. For the sake of concreteness, in the remainder of the paper we shall discuss this notion based on a practical example involving the documents shown in Figure 1. Both documents contain information about people working at a University. In document (a) `people` is divided into several groups depending on the `department` they belong. Then each department is divided into `technicians`, `administrative staff` and `research staff`. Researchers include both professors and Ph.D. students. Inside each `professor` or `student` tag lie their respective `name`, `surname`, `cv` and `office`. The `cv` tag is usually a large text blob. In document (b) the `University` is directly divided into `departments`, each department containing two groups of people: `employees` and `students`. Employees include `administrative staff` and `professors`. Each professor's (or student's) personal details are enclosed in the tag `name` which includes their name, surname, and cv.

3 FuzzyXPath

We are now ready to illustrate our approach to fuzzy XML querying in some detail. According to the XML Infoset, XML documents can be represented as a tree of typed nodes. XPath uses a pattern expression to identify nodes in an XML document and retrieves portions of XML documents, namely the set of nodes matching the pattern. Due to space constraints, in this context we focus our attention on a subset of the XPath language informally defined as follows: `XPath*`:= $\varepsilon|l|*|p_1/p_2|\ //p_1|p[q]$ where p_1 and p_2 are `XPath*` expressions; ε, l, $*$ denote the empty path, a label and a wildcard, respectively; / and // stand for

Fig. 1. Two XML documents sharing the same content but with different structure

child-axis and descendant-or-self-axis; and finally, q is called a qualifier. Following [6] we extend the XPath syntax with three classes of characteristics:

- *Fuzzy Subtree Matching*: namely `NEAR`, `ABOUT`, `BESIDES` and `LIKE`, providing a ranked list of retrieved information items rather than the set oriented one typical of XPath.
- *Fuzzy Predicates*: specifying flexible selection conditions.
- *Fuzzy Quantification*: allowing the specification of linguistic quantifiers as aggregation operators.

XPath 1.0 presents two main features that are relevant to our approach:

- *Path-based selection*: the user formulates a *search path*, in the standard form of XPath expressions, that must be exactly matched against the structure of the target XML documents.
- *Set-oriented query result*: the selection mechanism retrieves the documents sets of nodes that exactly match the user-provided path.

XPath assumes that the user is fully aware of the target schema. This assumption is in itself debatable since most XML documents exist without schemas; even worse, it requires the user to write a different query for each variation of the target schema. In other words, XPath does not tolerate data structure or content

diversity. In order to tackle this problem, we extend XPath in order to perform *approximate queries* in which the search path only provides a loose example of the information the user is interested in.

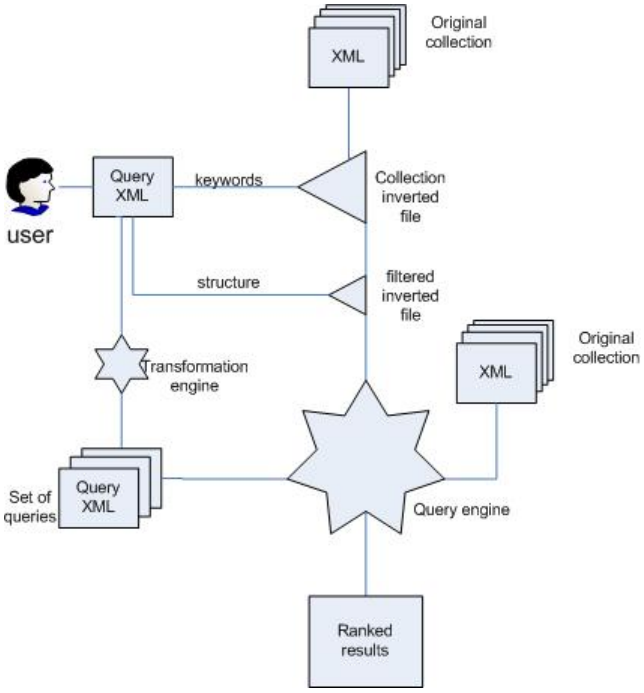

Fig. 2. Architecture of the approach

Our approach is based on three steps (Figure 2):

1. in the first step the query is analyzed to extract its keywords, and then these keywords are searched in an inverted file (detailed in Section 3.1). This operation reduces the cardinality of the document collection that will be target of the query execution and extracts a ranked list of candidates on the basis of the *index term weights* contained in the inverted file;
2. then, in order to "easily" process the structural selection conditions, the query is rewritten into a ranked set of queries that approximate the desired information. The ranking of the approximate query is computed on the basis of their distance from the user's query. Each approximate query is computed on the previously computed list of candidate documents by a common query engine and the retrieved results, if any, will be ranked according to the weight of the corresponding query's (*structure weight*).
3. finally retrieved results are shown in a bi-dimensional space in which one dimension is the content-based evaluation retrieval status value (*index term*

weight) and the other is the structure-based retrieval status value. The user will choose which dimension he/she prefers to rank the results in the list of retrieved items.

The proposed approach is based on exact tag matching; it would be interesting to address the problem of approximate or semantics-based tag matching. To face this issue, the use of a thesaurus/ontology is needed. At this stage of the work we do not address this problem.

3.1 Content based constraints

Most of the operating IRSs are based on indexing functions which take into account only marginally the structure of documents, being mainly based on their consideration as a monolithic object, thus providing their overall content synthesis. This allows for the retrieval of a document as a unit of information; in the case of structured documents this retrieval criterion can be in contrast with the real users' needs, as relevant information could be found in just some subparts (fragments) of a document. The work developed in [5] is the basis of the approach presented here. This model is based on the observation that the production of an information surrogate, i.e., a document synthesis used for representing the document content, is strongly subjective and depends on the personal view of the interpreter of this information. Despite of this, most of the existing indexing functions behave as a black box producing the same document representation to all users. To overcome this limitation the indexing model proposed in [3] is able to diversify the document representations, based on the users' view. The proposed indexing model is composed of two main components: a static component, aimed at the pre-calculation of terms occurrence information within the paragraphs (this information has to be stored in the inverted file), i.e., the building bricks of the documents, and an adaptive component that, based on the user's indications computes the index terms weights used by the matching mechanism. In [5] a flexible query language has been also proposed for expressing soft selection conditions on both the documents' structure and contents. The evaluation of these flexible queries makes it possible to select a fuzzy subset of semi-structured documents from a heterogeneous collection. For example, if one is interested in reading scientific papers he/she may filter documents having most of the following *sections: Title, abstract, authors, text references*. The linguistic quantifier most, that specifies the soft constraint on the document structure allows not to disregard potential interesting documents whose structure is not complete with respect to the specifications; for example documents lacking the *abstract* section. Further, users can indicate preferences on the desired sections. The two levels of soft conditions on the structure and on the content of documents are expressed in two distinct phases but are evaluated in a unique step so that the Retrieval Status Value reflects the satisfaction of the global query. In the extension of the XPath query language that we propose in this paper, in order to be able to evaluate content-based query constraints, an file inverted structure for organizing indexes has to be defined. Each term in this structure should point to blocks

of information, each block containing: `docId` (document identifier), `field-Id` (textual field corresponding to a leaf node), `path` to reach that field in docId, `n` number of occurrences of that term in that field, a normalization parameter `np` (or two normalization parameters: number of occurrences of the most occurring term in that field and number of total occurrences of words in that field). When a query which specifies both structural and content-based constraints is evaluated, we propose a first pre-processing based on the consideration of terms specified in the query for content based constraints. This pre-processing consists in selecting, based on the information contained in the inverted file, only paths related to the fragments pointed by a given query term (i.e. those which indexed by the term) thus pruning the paths which possibly satisfy the user query. Then on those selected paths the evaluation of structure-based constraints can be performed.

3.2 Enforcement of structure constraints

Once the document collection has been restricted to a finite set of candidate documents by the content based analysis, the query is analyzed on the basis of its structure. *FuzzyXPath* ([6]) constraints are used to indicate a degree of desired approximation in the query structure. In this step each FuzzyXPath constraint is used to create a set of queries that simulates the degree of approximation specified in the user query. Each of these queries is labeled by a *structure weight* that indicates the distance between the original query structure and the current one. Then, the resulting set of query structures is used to eliminate the documents that do not match at all (not even approximately) the user query from the set of candidate documents. Finally this set of queries is executed by a common query engine (this time the match is computed exactly) and each retrieved result is ranked on the basis of the structure weight of the query that produced it. For example, for the user query:

```
university//department{NEAR}/professor[CV cw "Information
Retrieval"]/name
```

the approach follows these steps:

- We separate the path that carries the desired information (called *info path*) from the path that carries the matching condition (called *match path*). In our example, we obtain the info path `university//department {NEAR}/professor/name` and the match path `university//department{NEAR}/professor[CV cw "Information Retrieval"]`.
- We identify the approximate clauses. In the example, the info path contains the clause `NEAR`, which means that the output node set will be ranked with respect to the number of steps from `department` to `professor`. The matching path contains the construct `cw` (*contains word/s*) that is resolved by means of standard retrieval procedures creating a set of candidate documents by filtering the collection's weighted inverted file. The result of this

step is a set of candidate documents represented by their inverted file tuples (`docId, field-Id, path, dw`[2]).
- The info path is then transformed in a set of weighted query paths. This translates our FuzzyXPath NEAR clause into standard XPath constructs. Again in our example we obtain the set of paths Q ={(`university//department/professor/name`,1), (`university//department/*/professor/name`, 0.5),(`university //department/*/*/professor/name`, 0.34), ...}. In our example we compute the structure weight as $w = 1/(1 + n)$ where n is the number of steps between `department` and `professor`, but different functions could be used on the basis of the desired approximation ranking. We can stop the generation of new paths in Q just by selecting a matching threshold mt and stopping when $w < mt$.
- Paths $\in Q$ are used to filter again the candidate documents inverted file eliminating those documents whose `path` does not match any path $\in Q$.
- The match path is used to construct a set of queries Q' where each query q' is composed by the match path and one info path $\in Q$.
- Finally, queries belonging to Q' are executed on the set of candidate documents identified by the filtered inverted file and results are ranked in a bidimensional space that considers both weights `w` and `dw`.

4 Conclusions

While fostering application interoperability by providing a common data model, XML Infoset poses unique challenges. Diverse modeling choices may cause differences in XML schemata, leading to heterogeneous XML structure and content. In this paper, information retrieval and database-related techniques have been jointly applied to effectively tolerate this diversity by supporting flexible queries. Fuzzy/approximate matching is supported via a straightforward extension to standard XPath syntax. Also, our indexing technique represents a first step toward efficiently addressing structural pattern queries jointly with predicate support over textual content of XML elements. We plan to address this issue in a future paper.

References

1. S. Abiteboul. Querying semi-structured data. *LECTURE NOTES IN COMPUTER SCIENCE*, 1186:1–18, 1997.
2. S. Amer-Yahia, S. Cho, and D. Srivastava. Tree pattern relaxation. In Springer, editor, *Proceedings of EDBT*, 2002.
3. G. Bordogna and G. Pasi. Controlling retrieval through a user adaptive representation of documents. *Int. J. of Apporximate Reasoning*, 12:317–339, 1995.
4. G. Bordogna and G. Pasi. Flexible representation and querying of heterogeneous structured documents. *Kibernetika*, 36(6):617–633, 2000.

[2] Note that `dw` $= \frac{n}{np}$

5. G. Bordogna and G. Pasi. Personalized indexing and retrieval of heterogeneous structured documents. *Information Retrieval*, 8(2):301–318, 2005.
6. D. Braga, A. Campi, E. Damiani, G. Pasi, and PL. Lanzi. FXPath: Flexible querying of xml documents. In *Proceedings of EuroFuse 2002, Varenna, Italy, Sep. 2002*.
7. Patrice Buche, Juliette Dibie-Barthèlemy, and Fanny Wattez. Approximate querying of XML fuzzy data. In Springer, editor, *Proceedings of the 7th International Conference FQAS 2006, Milan, Italy*, 2006.
8. J. Callan. Passage-level evidence in document retrieval. In ACM, editor, *Proceedings of SIGIR 94, Dublin, Ireland*, 1994.
9. A. Campi, S. Guinea, and P. Spoletini. A fuzzy extension for the XPath query language. In Springer, editor, *Proceedings of the 7th International Conference FQAS 2006, Milan, Italy*, 2006.
10. Y. Chiaramella. Information retrieval and structured documents. In F. Crestani M. Agosti and G. Pasi, editors, *Lectures on Information Retrieval*, Lecture Notes in Computer Science. Springer Verlag, 2000.
11. Y. Chiaramella, P. Mulhem, and F. Fourel. A model for multimedia information retrieval. Technical Report Fermi ESPRIT BRA 8134, University of Glasgow, 1996.
12. P. Ciaccia and W. Penzo. The *collection index* to support complex approximate queries. In Springer Verlag, editor, *Proceedings of XSym 2003*, volume 2824, pages 164–179.
13. Ernesto Damiani, Barbara Oliboni, and Letizia Tanca. Fuzzy techniques for XML data smushing. In *Proceedings of the International Conference, 7th Fuzzy Days on Computational Intelligence, Theory and Applications*, pages 637–652, London, UK, 2001. Springer-Verlag.
14. Ernesto Damiani and Letizia Tanca. Blind queries to XML data. In *Database and Expert Systems Applications*, pages 345–356, 2000.
15. M. Frisse. Searching for information in a hypertext medical handbook. *Communication of the ACM*, 31(7):880–886, 1988.
16. N. Fuhr and K. Grobjohann. XIRQL: A query language for information retrieval in xml documents. In ACM, editor, *Proceedings of SIGIR'01, New Orleans, Luisiana, USA*, 2001.
17. M. Kaszkiel and J. Zobel. Passage retrieval revisited. In N.J. Belkin, D. Narasimhalu, and P. Willett, editors, *Proceedings of the 20th SIGIR*, 1994.
18. M. Lalmas. Dempster-shafer's theory of evidence applied to structured documents: modelling uncertainty. In *Proceedings of ACM SIGIR, Philadelphia.*, 1997.
19. M. Lalmas and I. Ruthven. Representing and retrieving structured documents using the dempster-shafer theory of evidence: Modelling and evaluation. *Journal of Documentation*, 54(5):529–565, 1988.
20. Hua-Gang Li, S. Alireza Aghili, Divyakant Agrawal, and Amr El Abbadi. FLUX: Fuzzy content and structure matching of XML range queries. In *Proceedings of WWW 2006, May 23-26, 2006, Edinburgh, Scotland*, 2006.
21. I. Macleod. Storage and retrieval of structured documents. *Information Processing and Management*, 26(2):197–208, 1990.
22. F. Mandreoli, R. Martoglia, and P. Tiberio. Approximate query answering for a heterogeneous XML document base. In Springer, editor, *Proceedings of the 5th Int. Conf on Web Information Systems Engineering*, Brisbane, Australia, November 22-24, 2004.
23. S. Myaeng, D.H. Jang, M.S. Kim, and Z.C. Zhoo. A flexible model for retrieval of sgml documents. In *Proceedings of the 21st ACM SIGIR, Melbourne, Australia.*, pages 138–145, 1998.

24. G. Navarro and R Baeza-Yates. A language for queries on structure and content of textual databases. In *Proceedings of ACM SIGIR, Seattle.*, pages 93–101, 1995.
25. T. Schlieder. Schema-driven evaluation of approximate tree-pattern queries. In Springer, editor, *Proceedings of EDBT*, 2002.
26. W3C. Xquery 1.0: An xml query language, November 2006.
27. R. Wilkinson. Effective retrieval of structured documents. In *Proceedings of the 17th ACM-SIGIR, Dublin.*, pages 311–317, 1994.
28. XML:DB. Xupdate, November 2006.