Synopsis Data Structures for XML Databases

Alfredo Cuzzocrea
ICAR-CNR and University of Calabria, Italy

INTRODUCTION

XML plays a leading role in the vest of "neutral" language/data-specification-format for next-generation *Intelligent Information Systems*, where the heterogeneity of data and processes pose novel and previously-unrecognized research challenges, beyond classical issues of conventional Database Systems. In fact, XML allows us to nicely overcome structural as well as semantic heterogeneities of distributed databases, such as those one can find in a *P2P network*. For these reasons, XML databases are becoming the most popular ones in distributed environments (e.g., Bonifati & Cuzzocrea, 2007).

Apart from data/schema integration, XML is also extremely useful in a wide spectrum of novel application scenarios, such as *schema mappings*, *data exchange*, and *metadata management*. On the other hand, XML processing is gaining momentum as it affects the reliability and performance of many computationally intensive applications, ranging from the recent Web-based and Grid-based infrastructures to the more traditional integrated and cooperative information systems. Looking at practical application scenarios, XML processing and management is relevant for a plethora of cases, ranging from *social networks* to *Geographical Information Systems* (GIS) and emerging *Cloud environments*.

In all such cases, efficiently querying native XML databases is a critical issue, due to the evidence that non-native databases may be too inefficient. This is due to several motivations, among which relevant ones are: (*i*) the XML data model is hierarchical in nature and not prone to be represented as a set of relations or objects; (*ii*) quite often, XML documents appear in a *schema-less* fashion (e.g., like those of corporate B2B and B2C e-commerce Web systems), thus making the relational translation more difficult; (*iii*) the inherent "richness" of the standard XML query language, which defines a comprehensive class of queries with possibly complex syntax and predicates (e.g., for clause of *XQuery* queries (Chamberlin et al., 2001), *twig* XML queries (Chamberlin et al., 2001), partial- and exact-match *XPath* queries (Clark & DeRose, 1999) etc.); (*iv*) the ambiguity of the XML semantics during query evaluation (Gyssens et al., 2006); (*v*) problematic update management issues posed by processing XML data (Benedikt et al., 2005; Boncz et al., 2006; Xu et al., 2005).

A possible solution to the above issues consists in computing and packaging *synopsis data structures* against the original XML documents. The aim of this approach is to obtain a small-size XML document $\tilde{X}$ (i.e., a proper synopsis data structure) starting from the input XML document X, in order to reduce the computational overhead due to processing data in X. The intuition behind such data structures is that the information content carried out by $\tilde{X}$ is "very similar" to the information content carried out by X. The measure of this similarity can be defined according to different alternatives depending on the target application. As an example, looking at the structure of documents (e.g., (Nierman & Jagadish, 2002)) is useful in the context of algorithms for clustering XML documents, and algorithms for detecting similarities among XML documents. In our approach, XML query processing aspects are taken into account, thus, given a query Q, the similarity measure is introduced in terms of the *approximation error* of Q, denoted by $E(Q)$. $E(Q)$ is quantified as the relative difference between Q *exact answer* (i.e., the answer to Q evaluated against X), denoted by $A(Q)$, and Q *approximate answer* (i.e., the answer to Q evaluated against $\tilde{X}$), denoted by $\tilde{A}(Q)$. $E(Q)$ is thus defined as follows:

$$E(Q) = \frac{|\, A(Q) - \tilde{A}(Q) \,|}{A(Q)}.$$

DOI: 10.4018/978-1-4666-5888-2.ch183

The goal of any query-centric technique for computing synopsis data structures from XML documents is that of making $E(Q)$ as minimal as possible. Lossless compression techniques ensures that, for any arbitrary query Q, $E(Q) = 0$, whereas lossy compression techniques do not guarantee this condition.

BACKGROUND

Synopsis data structures allow achieving two main goals. The first goal is to enable *selectivity estimation* for XML queries (e.g., (Aboulnaga et al., 2001)). In our context, given an XML query Q, the selectivity of Q, denoted by $\gamma(Q)$, is intended as the number of XML elements/nodes visited during the evaluation of Q. Similarly to conventional relational database systems (Kooi, 1980), the *Selectivity Estimator* is a fundamental component of the *Query Optimizer*, which, based on the selectivity of queries, determines the optimal query plan according to which the input query is evaluated. In turn, the latter plan represents the input of the *XML Query Engine*, which eventually provides the answer to the query. The second goal of synopsis data structures consists in efficiently supporting *approximate query answering* over XML documents (Polyzotis et al., 2004a), with is similar in spirit to what done in the context of databases and data cubes (Garofalakis & Gibbons, 2001; Cuzzocrea & Serafino, 2009). This means to evaluate input queries against the synopsis data structure $\tilde{X}$ rather than on the original XML document X, by allowing an approximation error that is meant to be negligible for the target application. Such an approximation, although generally applicable, makes more sense in particular application fields such as the integration of XML and OLAP (Jensen et al., 2001), due to the fact that imprecise query evaluations are perfectly tolerable in OLAP systems (Cuzzocrea, 2005a).

AN OVERVIEW OF SYNOPSIS DATA STRUCTURES FOR XML DATABASES

Data compression/reduction techniques for massive data sets have a long history. Among proposals appearing in literature, *sampling* (Gibbons & Matias, 1998), *histograms* (Ioannidis, 2003) and *wavelets* (Chakrabarti et al., 2000) are the widely-accepted methods for obtaining synopsis data structures from massive amount of data sets ranging from conventional relational databases to data warehouses and OLAP data cubes.

In the context of XML, there has been some preliminary work related to the problem of compressing large XML documents, and efficiently using such compressed representations to improve the performance of mining and query algorithms over XML data. Also, previous solutions for computing synopses from relational data sources are ineffective for XML data sets, as also recognized in (Polyzotis et al., 2004a). A first study on the problem of optimizing XML queries via computing so-called *data guides* having a statistical nature appears in (McHugh & Widom, 1999). Data guides can be viewed as a form of synopsis data structures. This solution has limited applicability, as it only supports approximate query answering for path queries of fixed length, which makes it not extensible in such a way to cover more complex kinds of XML queries.

Aboulnaga et al., (2001) investigates the problem of estimating the selectivity of path expression in an Internet-like applications scenario. The solution consists in two specialized data structures that, in total, form a synopsis data structure: (*i*) the *Summarized Path Tree* that, given the input XML document X, collapses its structure in a tree where low-frequency nodes are deleted or coalesced in summarizing nodes; (*ii*) the *Summarized Path Table* that stores the count of all frequent paths in X having length less than a parameter $k > 0$. Similarly to (McHugh & Widom, 1999), this proposal has limited applicability and, in addition to this, it does not consider at all the problem of maintaining compressed information related to the structure of the input XML document, which should not be instead ignored.

Polyzotis and Garofalakis (2002a, 2002b) move the problem from tree-structured XML databases to the more general case of handling *graph-structured XML databases* like those one can find in large-scale, complex Web systems, modeled by means of IDREF constructs. The solution in Polyzotis and Garofalakis (2002a, 2002b) consists in computing a statistical summary of the input graph-structured XML database X within a given space bound, thus devising a *graph-synopsis* called XSKETCH, which, starting from a *partitioned representation* of nodes in X, is obtained by means of successive refinements of the so-called *label-split*