AQAX: A System for Approximate XML Query Answers

Joshua Spiegel[†]
jspiegel@cs.ucsc.edu

Emmanuel Pontikakis[‡*]
manos@cs.stanford.edu

Suratna Budalakoti[†]
suratna@cs.ucsc.edu

Neoklis Polyzotis[†]
alkis@cs.ucsc.edu

[†]Department of Computer Science
University of California, Santa Cruz
Santa Cruz, CA 95064-1077

[‡]Department of Computer Science
Stanford University
Stanford, CA 94305-9025

ABSTRACT

On-line, interactive exploration of large databases becomes prohibitively expensive as the size of the database grows. Approximate query answering offers a cost-effective solution to this problem, by enabling the fast generation of approximate results based on concise data summaries. We apply this paradigm in the context of XML databases, where the increased complexity of data and queries amplifies the challenges behind interactive exploration. We have developed an on-line XML exploration system, termed AQAX that relies on accurate XML summaries in order to enable the rapid exploration of large data sets. To effectively support the exploration of semi-structured query answers, our system employs a tight coupling between the main query processor and the graphical clients that visualize the results. This demonstration will showcase the functionality of our system and the effectiveness of approximate query answering in the context of XML databases.

1. INTRODUCTION

The Extensible Mark-up Language [2] (XML) is rapidly gaining in popularity as a universal data model for data exchange and integration. This is evidenced by the increasing number of available XML data sets and the wide-spread support for XML data management from the major database system vendors. Given the growing popularity of XML, it is natural to expect the emergence of decision-support systems that will enable the on-line exploration of massive XML data stores. In this scenario, the user (typically, an analyst or a domain expert) analyzes the data in an interactive fashion, and it is thus crucial to maintain low response times in order to avoid disrupting the task of data exploration. This important goal, however, conflicts with the increased complexity of query evaluation over semi-structured data. Even though several research studies are actively exploring the important problem of efficient XML query evaluation, it is clear that interactive response times remain a very challenging issue.

A cost-effective solution for mitigating the increased cost of query evaluation is the use of *approximate query answers* [1, 3, 5]. In short, the system evaluates the query over a concise *synopsis* (or, summary) of the XML data, generating an answer that approximates well the actual result of the query. The key advantage of course is that the approximate answer is generated very efficiently, as it is based entirely on the highly compressed summary. The user can thus receive timely feedback on the results of the query, before or during query evaluation. In some cases, we expect the user to refine his/her query based on the feedback, while in others he/she may be satisfied with the approximate answer only. Overall, approximate query answering may reduce the number of costly queries that the system needs to evaluate, thus increasing the efficiency of interactive data exploration.

This demonstration presents AQAX (Approximate Query Answering for XML), a system that supports approximate query answers over large XML data sets. Our system is based on the recently proposed XCLUSTER [4] framework for the accurate summarization of large XML data sets. To support the generation and exploration of approximate tree-structured answers, AQAX employs a two-tier architecture. At the first tier, a query processor generates approximate answers to complex XML queries based on the XCLUSTER framework. At the second tier, a navigation client enables the effective exploration of the tree-structured answers through visual manipulation primitives that are applied on XCLUSTER synopses. Our demonstration presents the use of AQAX in a simulated data exploration scenario, showing the internal workings of the system at both tiers.

The following sections provide an overview of the XCLUSTER framework, and then present in more detail the architecture of the AQAX system. We conclude with a description of the demonstration.

[*] Work performed while author was at UC Santa Cruz

VLDB '06, September 12-15, 2006, Seoul, Korea.

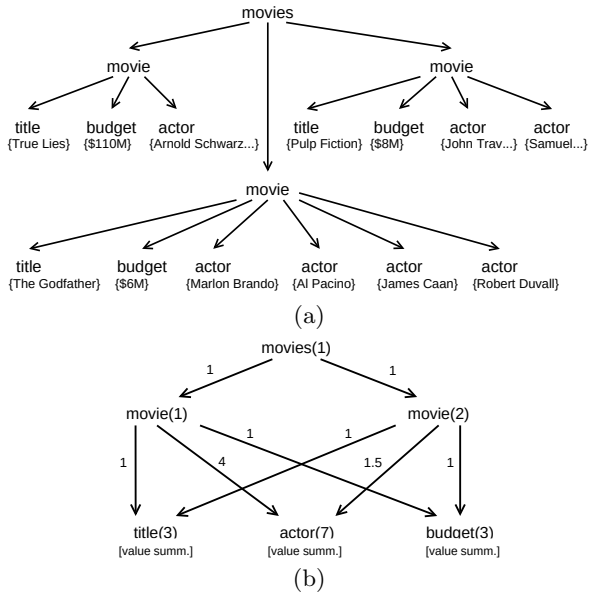


Figure 1: (a) an XML document (b) one possible XCluster **synopsis for the document**

2. OVERVIEW OF XCLUSTER

The recently introduced XCluster framework targets the difficult problem of summarizing the structure and value content of complex XML data. An XCluster synopsis is a node- and edge-labeled graph, where each node represents a sub-set of elements with the same tag, and an edge connects two nodes if an element of the source node is the parent of elements of the target node. To capture the structural and value-based properties of the underlying data, the synopsis records aggregate statistical information at nodes and edges. More precisely, each node maintains the count and tag of elements that it represents, and an optional *value summary* that captures the distribution of the corresponding element values. Each edge, on the other hand, maintains the average child count between source and target elements, thus capturing the connectivity between the corresponding element-sets.

An example of XCluster synopses is shown in Figure 1, that depicts a sample XML data set and one possible XCluster summary. In this particular case, the XCluster summary groups elements by their tag. Note that each node carries only the number of elements it represents and possibly a summary of their value distribution, while edges record the average number of children for the source elements.

XCluster captures the structural characteristics of the underlying data and supports the summarization of common types of values, namely, numeric, string, and textual values. Hence, a single XCluster summary can generate approximate answers for queries that reference the structure and the heterogeneous value content of the underlying XML data. We note that the original XCluster framework focuses primarily on the problem of selectivity estimation. Since our focus is on the generation of general approximate answers, we augment XCluster with query evaluation primitives that enable the computation of tree-structured results or aggregates (e.g., AVG, SUM) over the underlying XML data.

3. AQAX SYSTEM ARCHITECTURE

Figure 2 depicts the main components of the AQAX system architecture. The user interacts with the navigation client to create a visual query, which is subsequently translated to XQuery and submitted to the server. The server processes the query over the stored XCluster summaries and returns an approximate query answer, which is visualized and further manipulated at the graphical client. The following sections describe the client and server components in more detail.

3.1 The AQAX Server

The AQAX Server is responsible for evaluating queries over the stored summaries and generating approximate answers. The user has the option of selecting the synopsis for generating approximate answers, thus setting the desired accuracy of the generated approximate results. The user may also designate the actual data set as the "synopsis" of the data. In that case, AQAX forwards the query to an XML DBMS and returns the exact results.

The server employs an extension of the XCluster framework [4] in order to generate approximate answers for twig-queries with value predicates and optional aggregates. The generated results are returned to the Navigation Client in the form of an XCluster synopsis, henceforth referred to as the *result synopsis*.

3.2 The Navigation Client

The navigation client, shown in Figure 3(a), enables the user to formulate queries and explore their results. As shown in the screen-shot, the query formulation is performed graphically on the left side of the main window, by manipulating a tree-based representation of the twig query. At any point, the user may also view the XQuery equivalent of a graphically created query.

After submitting the query to the server, the client receives the result synopsis of the query. Since the result synopsis can be too complex to present on screen, the client visualizes a *working synopsis* that represents a coarser summarization of the query answers. Initially, the working synopsis is the coarsest (or, simplest) possible view of the result, grouping under a single node all the result elements with the same tag. The user can subsequently refine the working synopsis by applying a set of operations that essentially increase the level of detail of the visualization. (To the limit, the working synopsis becomes the same as the result synopsis.) These operations are processed entirely in the client and thus do not require any communication with the server. In what follows, we briefly describe the main operations that we have incorporated in our system.

Render: This operator produces statistics over an edge or a value summary in the working synopsis. The output of the operator is displayed to the user with an appropriate visualization that depends on the context. When applied to an edge, the operator generates statistics about the parent-child relationships between the corresponding element sets

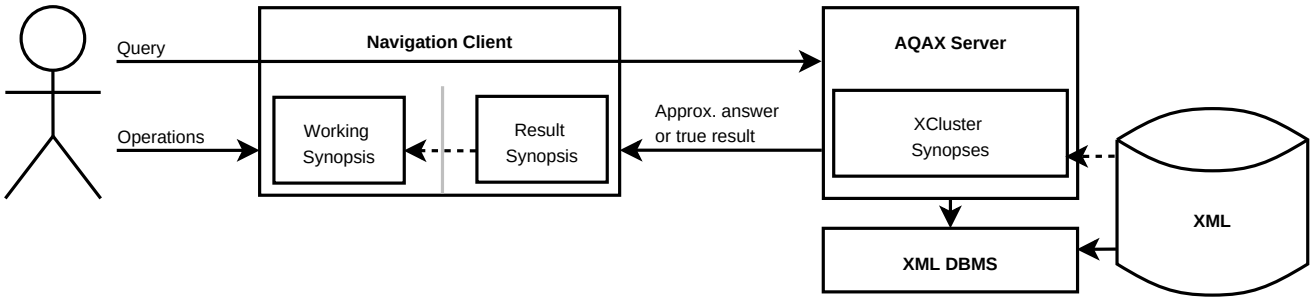


Figure 2: AQAX system architecture.

and visualizes them with a histogram. Consider, for instance, a movie database similar to the one shown in Figure 1 (a), where `movie` elements may be the parents of varying numbers of `actor` elements. Rendering the edge between a `movie` node and a `actor` node will present the user with a histogram that displays the distribution of actors per movie. Figure 3(b) depicts an example of this visualization, where the vertical axis indicates the number of parent elements and the horizontal axis indicates the number of children elements per parent.

Render can also be applied to a value summary of a given path in the working synopsis, in order to visualize the distribution of values under the given path. Since the XCLUSTER model provides support for numerical, textual, and string content, the summarization inherently depends on the type of values under the specified path. In the case of numerical values, the client displays an approximate value distribution in the form of a histogram, while for string or textual values, the user is presented with a list of the most frequent substrings and terms respectively.

Split Edge: The split operator allows the refinement of the working synopsis in areas where the user would like to see more information. More precisely, a split operation partitions the elements represented by the source node based on their child counts in the target node, and thus creates new source nodes and edges in the working synopsis. As an example, consider again the approximate result shown in Figure 3 and observe that `movie` elements may have a varying number of `actor` children elements. The working synopsis may represent the data in its coarsest form, with a single `movie` node, a single `actor` node, and an edge between them. (This would certainly be the case in the initial working synopsis). Assume that movies may have up to 100 actors each, but the user is only interested in the budget of movies with less than 10 actors. By applying the split operator to the edge with a *split point* of 10, two new edges and two new `movie` nodes will be created. The first `movie` node will represent `movie` elements with less than 10 `actor` children while the second will represent those elements with 10 or more. The user can then use the render operator to view the `budget` value distribution for movies with with less than 10 actors.

The set of split points may be entered manually, or by manipulating the histogram resulting from a render operation on the edge to be split. In the latter mode of interaction, the user can click on the bars of the histogram to designate the corresponding child counts as split points. Figure 3(b) shows an example of this manipulation, where the red lines next to histogram bars indicate the split points that the user has selected.

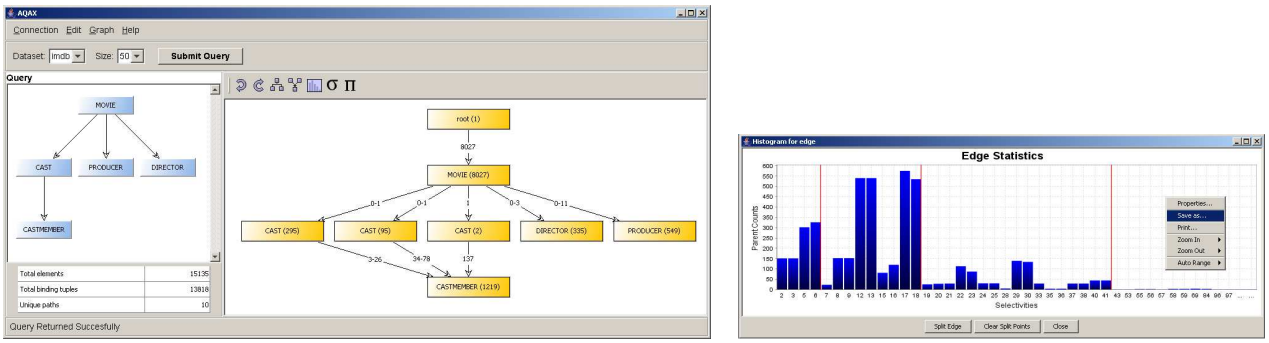
Merge Nodes: The merge operator is the inverse of the split operator. This operator is applied on a set of nodes in the working synopsis with the same tag name, and replaces them with a single node that acquires their aggregate characteristics. Overall, the combination of the merge and split operators allows the user to contract or expand different parts of the result synopsis depending on the goals of the exploration task.

Filter: The filter operator applies a value predicate to the working synopsis and essentially restricts the set of visualized results. This operation is thus similar to re-executing the query with the added predicate. We include it in this set of operations as the client already has the necessary information for applying the predicate. We also provide the option to integrate the filter in the current query, in case the user wishes to view results from a different data synopsis (e.g., of increased size).

4. DEMONSTRATION

Our demonstration aims to showcase the functionality of the AQAX system in exploring semi-structured query answers, and to demonstrate the effectiveness of approximate query answering in the exploration of large XML data sets.

The first part of our demonstration shows the effectiveness of our system in the interactive exploration of approximate query answers. More precisely, we simulate an interactive user session on a specific data set through a sequence of exploratory XML queries that are formulated on the visual client. Once a query is submitted to the server and the results returned, we show how the user can effectively explore the tree-structured answers by manipulating the working synopsis through the visual primitives that we introduced in Section 3.2. To illustrate the functionality of different operators, we introduce an auxiliary window that visualizes the result synopsis of the query. This window is a visual aid for our demonstration, showing how the updated working synopsis (after the application of an operation) relates to the result synopsis returned by the server.



(a) (b)

Figure 3: The Navigation Client: (a) the main window, showing a query and the working synopsis; (b) a histogram resulting from a render operation on an edge.

The next part of our demonstration illustrates the effectiveness of approximate query answers by showing that they are accurate and can be generated relatively fast. To aid in this part of the demonstration, we introduce a second window in the navigation client that displays a working synopsis of the true query result (henceforth referred to as the exact working synopsis). This auxiliary window is synchronized with the main window of the approximate working synopsis, in that any operations (splits, merges, etc.) that are applied to the approximate working synopsis in the first window are automatically applied to the exact working synopsis. The goal is to show the working synopsis that would result if the user chose to manipulate and explore the true results of the query, and hence to examine the effectiveness of approximate answers in capturing the main characteristics of the true query result. The demonstration simulates again a user session with several exploratory queries, comparing the execution time to obtain the initial approximate and exact working synopses, and evaluating the similarity between the two after the application of our visual operations.

5. REFERENCES

[1] S. Acharya, P. B. Gibbons, V. Poosala, and S. Ramaswamy. The Aqua Approximate Query Answering System. In *Proceedings of the 1999 ACM SIGMOD Intl. Conf. on Management of Data*, pages 574–576, 1999.

[2] T. Bray, J. Paoli, C. M. Sperberg-McQueen, and E. Maler. "Extensible Markup Language (XML) 1.0 (Second Edition)". W3C Recommendation (available from `http://www.w3.org/TR/REC-xml/`), October 2000.

[3] M. Garofalakis and P.B. Gibbons. Approximate Query Processing: Taming the TeraBytes. In *Proceedings of the 27th Intl. Conf. on Very Large Data Bases*, 2001.

[4] N. Polyzotis and M. Garofalakis. XCluster Synopses for Structured XML Content. In *Proceedings of the 22nd Intl. Conf. on Data Engineering*, pages to–appear, 2006.

[5] N. Polyzotis, M. Garofalakis, and Y. Ioannidis. Approximate XML Query Answers. In *Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data*, pages 263–274, 2004.