Anup Athavale
May 5th, 2008
CSE 718 – Advanced Topics in Database Systems

Lazy Query Evaluation for Active XML

Overview:

This paper studies an approach to efficiently query AXML documents. AXML documents are XML documents whose content is given partly by data elements and partly embedded calls to web services which can be invoked to generate data. Since these calls to the web services can be present anywhere in the document, a major challenge is to make only those calls which get relevant data and ignore the other calls which are irrelevant with context to the query at hand. The authors have formalized the problem and provided algorithms which can be used together for faster and efficient query evaluation for active XML.

The web service calls in the AXML documents can appear anywhere within the scope of the document and also in the result of previously invoked calls. Moreover, the relevance of one call may depend upon the result of another and return types should also be taken into consideration to avoid irrelevant calls. The main contribution of this paper is to gain a better understanding of the possible relationships among the service calls embedded within the document and their influence on the relevance of calls to queries. Thus the approach given in this paper is multi-step and involves the following key techniques:

1. **Computing the set of relevant service calls:** The algorithm generates a set of queries that retrieve all service calls relevant because of there position.
2. **Service calls sequencing**: Relationships among the calls are analyzed, to derive a sequence of call invocations appropriate to answer a query.
3. **Pruning via typing:** Return types of services are used to rule out more service calls.
4. **Service calls guide:** A specialized access structure is used to speed up the detection of relevant calls.
5. **Pushing queries:** Precise knowledge of the interaction between the query and each service call enables pushing queries to capable Web services, like mediators do with data sources.

The paper first describes the methods used for finding relevant calls. The authors have listed two means of achieving this, namely, Linear Path Queries (LPQ's) and Node Focused Queries (NFQ's). The NFQ's take into account the conditions within the query for matching and hence can be used to find a more compact subset of possible calls.

After finding possible candidates for service calls, their sequencing is taken into account. The NFQ algorithm is repeatedly used to find the sequence of these calls as the document grows. The NFQA algorithm computes a (possibly infinite) relevant rewriting. If it terminates, the obtained document is complete for the query q. The relationship between these calls is taken into consideration to avoid making useless rewriting. The NFQ layers are determined and their influence on each other is studied. If two NFQ's lie on the same layer and do not influence each other, the can be processed in parallel.

After sequencing the calls, type checking is done to avoid making calls which return data of a different type. Also optimizations methods like F-guides are used to increase the performance. Wherever possible, the queries are pushed to get only relevant information from the service, which saves time taken during transmission of data as well as its processing.

Detailed comments:

1. **Main strength**: The main strength of this paper lies in the approach wherein mappings between data sources are captured by service calls embedded in the data, with new relationships discovered at run-time, in the answers of service calls. This approach is based on ad-hoc dynamic discovery of mappings. Another key aspect is the use of Node focused queries for finding relevant calls and sequencing along with the use of F-guides to achieve better and faster results.

   Weakness: The weakness of this paper is that it never compares the experimental results with a hybrid approach described at the start. Although it saves the time taken for getting data from services, the NFQ filtering is a time consuming process in itself when applied repeatedly on the re-written document. There is a possibility of developing a light-weight parser extension which uses on the fly evaluation techniques and multithreading to achieve query processing and making service calls simultaneously. The paper rules out the possibility of such an approach and states that it is not possible to club the logic of finding relevant calls within the query processor.

2. **Is the paper technically sound?**
   The paper is well balanced in terms of technical depth. The details of the NFQ algorithm and sequencing relevant calls are described in technical depth. Except for one instance (Testing satisfiability) the authors have sufficiently covered the technical intricacies in depth.

3. **Is the paper technically sound?**
   The paper is technically sound. The authors have thoroughly described all the algorithms used for achieving lazy query evaluation. However, some definitions are superfluous and there is a considerable redundancy. For example, the definitions 2, 3 and 4 as stated in the paper, could have been clubbed into a single comprehensive definition for relevant rewriting and completeness. Overall, the

authors have done justice to the complex nature of the evaluation techniques by putting them in a lucid yet technical language.

4. **How does the paper compare with the related work?**
   This paper mainly deals with an approach to efficiently evaluate queries on a given AXML document. The closest in relevance to this paper is an unpublished work which also deals with evaluation of queries over AXML documents with lazy service calls. The difference is that the focus of that paper is on minimizing data materialization at a global level (using a generalization of Query-Sub-query technique discussed in one of the referred papers) for systems with many interrelated documents in a peer-to-peer setting. Also set in the context of AXML, another paper considers the problem of distributing and replicating AXML data and services in a peer-to-peer setting. In contrast to both, this paper addresses the specific problem of optimizing the evaluation of queries over one local AXML document. Thus, this work is clearly complementary to both, and indispensable for them to obtain good performances.