SMOQE: A System for Providing Secure Access to XML

Wenfei Fan, Floris Geerts, Xibei Jia, Anastasios Kementsietsidis

The University of Edinburgh

Abstract: XML views have been widely used to enforce access control, support data integration, and speed up query answering. In many applications, e.g., XML security enforcement, it is prohibitively expensive to materialize and maintain a large number of views. Therefore, views are necessarily virtual. This demo presents SMOQE, the first system to provide efficient support for answering queries over virtual and possibly recursively defined XML views. We demonstrate a set of novel techniques for the specification of views, the rewriting, evaluation and optimization of XML queries. Moreover, we provide insights into the internals of the engine by a set of visual tools.

Introduction

For all the reasons that views are essential to traditional databases, XML views are also important for XML data. In many applications, e.g., in XML security enforcement, views are necessarily *virtual*: it is prohibitively expensive to materialize and maintain a large number of views, one for each user group.

We have developed the Secure MOdular Query Engine (SMOQE) for facilitating the specification of XML views and answering of XML queries on virtual views. The main features of SMOQE are the following.

- SMOQE supports XML views defined by annotating an XML schema with Regular XPath [3] queries. It supports recursively defined schemas (and thus views).
- SMOQE is able to rewrite any Regular XPath query Q posed by users on a virtual view V to an equivalent Regular XPath query Q' on the underlying document T.
- SMOQE encompasses a query engine for Regular XPath queries, implementing an efficient evaluation algorithm and a novel indexing structure.

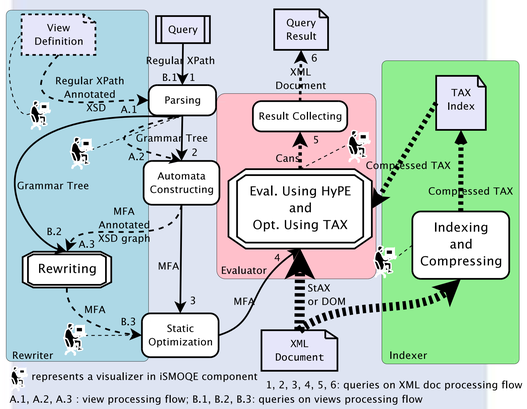


Figure 1. The SMOQE Architecture

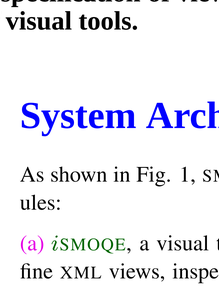
System Architecture

As shown in Fig. 1, SMOQE consists of four major modules:

(a) *i*SMOQE, a visual tool through which a user can define XML views, inspect the query rewriting and evaluation, and browse query results (a small user icon is used to indicate all the system components accessible through *i*SMOQE);
(b) a query rewriter (indicated by a blue box at the left of the figure) for translating user Regular XPath queries posed on XML views to equivalent Regular XPath queries on the underlying document;
(c) a query evaluator (indicated by a pink box in the middle of the figure) for processing Regular XPath queries; and
(d) an indexer (indicated by a green box at the right of the figure), which is used by the evaluator to build indexes and optimize queries.

Security Views

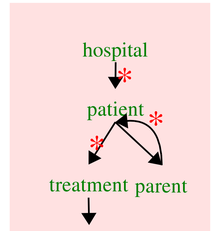
An *access specification* S is an extension of a document DTD D by associating security annotations with productions of D [1]. Here is an example:



hospital → patient*
σ0(hospital, patient) = patient[visit/treatment/medication = 'autism']
patient → treatment*, parent*
σ0(patient, treatment) = visit/treatment[medication]
σ0(patient, parent) = parent
parent → patient
σ0(parent, patient) = patient
treatment → medication
σ0(treatment, medication) = medication

document DTD D_0 | access specification S_0

A *security view specification* σ could be either manually defined by security administrators or automatically derived from access specifications. Moreover, a view DTD could be provided to help users posing queries by the view derivation algorithm in the latter case:



view specification σ_0 derived from S_0 | view DTD D_V

Regular XPath Evaluation

While it is always possible to rewrite a Regular XPath query Q on a view to an equivalent query Q' on the underlying document, the size of Q', if directly represented as Regular XPath expressions, may be exponential in the size of Q [2]. The SMOQE rewriter overcomes this challenge by employing an automaton characterization of Q', denoted by MFA (*mixed finite state automaton*) [2], which is *linear* in the size of Q.

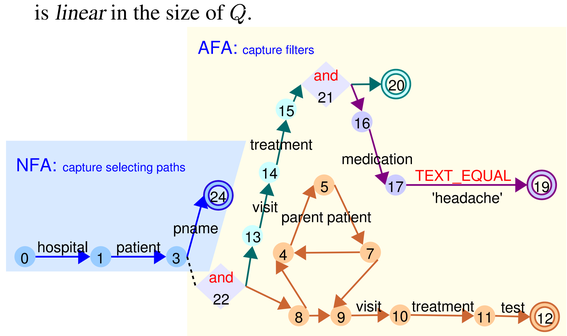


Figure 2: The MFA $\mathcal{M}_0$ for Q_0

For example, Fig. 2 depicts the MFA $\mathcal{M}_0$ characterizing the Regular XPath query:

Q_0 = hospital/patient[(parent/patient)*/visit/treatment/test/
and visit/treatment[medication/text()="headache"]]/pname

The SMOQE evaluator implements a novel algorithm, referred to as HyPE (Hybrid Pass Evaluation) [2], for processing Regular XPath queries represented by MFA's. Fig. 3 shows the evaluation of the MFA $\mathcal{M}_0$ given earlier on an XML document:

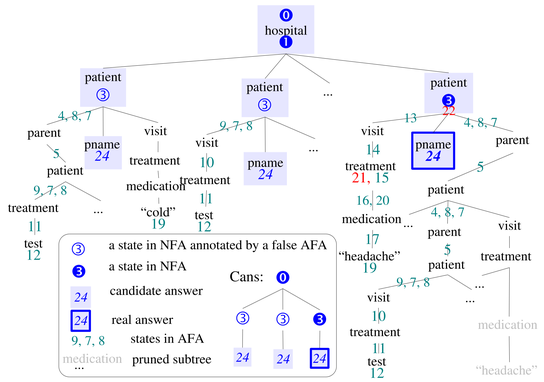


Figure 3: Evaluation of $\mathcal{M}_0$ using HyPE

A unique feature of HyPE is that it needs a single top-down depth-first traversal of the XML tree, during which HyPE both evaluates predicates of the input query (equivalently, AFA's of the MFA) and identifies potential answer nodes (by evaluating the NFA of the MFA). The potential answer nodes are collected and stored in an auxiliary structure, referred to as Cans (candidate answers), which is often much smaller than the XML document tree. A pass over Cans is needed to retrieve the real result nodes.

References

[1] W. Fan, C. Y. Chan, and M. Garofalakis. Secure XML querying with security views. In *SIGMOD*, 2004.
[2] W. Fan, F. Geerts, X. Jia, and A. Kementsietsidis. Rewriting regular XPath queries on XML views. *http://www.lfcs.inf.ed.ac.uk/research/database/rewriting.pdf.*
[3] M. Marx. XPath with conditional axis relations. In *EDBT*, 2004.

School of informatics