Chapter 7
Scalable XML Filtering for Content Subscriptions

Ryan Choi
KAIST, Korea

Raymond Wong
The University of New South Wales, Australia

ABSTRACT
Over the past few years, there have been an increasing number of Web applications that exchange various types of data on the Internet. In this article, we propose a technique for building efficient and scalable XML publish/subscribe applications. In particular, we look at the problem of processing streaming XML data efficiently against a large number of branch XPath queries. To improve the performance of XML data processing, the branch queries that have similar query characteristics are grouped, and common paths between the queries in the same group are identified. Then, these groups of queries are processed against an XML schema to validate query structures. After performing structural matching of queries, the queries are organized in a way that multiple queries can be evaluated simultaneously in the post-processing phase. In the post-processing phase, join operations are executed in a pipeline fashion, and intermediate join results are shared amongst the queries in the same group. The benefit of this approach is that, the total number of join operations performed in the post-processing phase is significantly reduced. In addition, we also present how to efficiently return all matching elements for each matching branch query. Experiments show that our proposal is efficient and scalable compared to previous works.

INTRODUCTION
With the development of the World Wide Web (WWW), a huge number of Web-based applications have been developed over the past few years. Web 2.0 (O’Reilly, 2005) goes one step further than traditional web applications in a way that, they provide personalized services, as well as letting users create, publish, and share contents amongst other users who share similar interests.
Recent works include online finance (Nah, Siau, & Tian, 2005), education (Siau, Sheng, & Nah, 2006), government (Siau & Long, 2006), healthcare (Siau & Shen, 2006), and firewall (Benedikt, Jeffrey, & Ley-Wild, 2008) applications. Another important difference between Web 2.0 and traditional web applications is that, users of Web 2.0 applications subscribe to the contents that they are interested in, and the web contents are delivered directly to users. Furthermore, users with similar interests can share their subscriptions to quickly discover other related contents. This is quite different from traditional web applications, where users obtain contents of interest by visiting web sites, following links, etc.

As a motivating example of a Web 2.0 application, let us consider an online-based news feed application that delivers latest news articles to users. A unique characteristic of this application is that, it receives various types of streaming data from multiple data publishers, selects data of interest, and forwards the selected data to various groups of users who are interested in receiving such data. One problem associated with this application is that, since each data publisher is designed and implemented differently, the data format from one data publisher is usually incompatible with the formats from other peer data publishers. Having a unique format for each data publisher causes problems when the data are collected and processed by a single application. XML (Bray, Paoli, Sperberg-McQueen, Maler, & Yergeau, 2008) solves this problem by providing a way to represent any data from different publishers in a universal format, such that the data can be collected and processed by a single application. Moreover, while data are converted to XML, irregular data, which may have been represented by multiple data and metadata tables in relational database systems, can be intuitively and logically represented. Let us now assume that all data publishers use XML to represent their data.

While information searching and retrieval are well studied areas in research communities, the problems in this context are different from the traditional search problems in many ways. In this context, new data continuously arrive to our news feed application, and the application must select or filter the right data according to user subscriptions. In our application, we represent user subscriptions in XPath (Clark & DeRose, 1999) queries. Then, the “query results” in this context are a set of matching XPath queries for each streaming XML document. The use of XML and XPath to implement a filtering mechanism has a number of advantages over similar but non XML-based approaches. First, more expressive user subscriptions can be supported. Unlike keyword-based subscriptions, which simply report matching sets of keywords for each document, users can also utilize structural information implicitly integrated to XML documents to precisely specify the exact content that they wish to receive. For example, while it is logical to express an XPath subscription to find news articles in a financial section that talk about the impact of US mortgage crisis, such subscription is not trivial in keyword-based subscriptions. Second, there are more opportunities to optimize a filtering processor. For example, since subscriptions are written in XPath queries, it is possible to group similar queries and process groups of queries simultaneously. Third, the use of XML is perfect to model the increasing amount of semi-structured data on the Web.

There have been a number of research works on how to filter a stream of XML data according to user subscriptions. The application that implements such functionality is commonly known as an XML publish/subscribe system (or XML pub/sub system for short) in the literature. The main focus of these works is how to evaluate a large number of XPath queries efficiently against streaming XML data. To do so, these works proposed a number of ways to build, combine, and share XPath queries. However, some works are not suitable for our need as: (1) XFilter (Altinel & Franklin, 2000), LazyDFA (Green, Gupta, Miklau, Onizuka, & Suciu, 2004), Bloom Filter
Related Content

3-D Virtual Worlds in Education: Applications, Benefits, Issues, and Opportunities
[www.igi-global.com/article/virtual-worlds-education/3397?camid=4v1a](www.igi-global.com/article/virtual-worlds-education/3397?camid=4v1a)

Identifying, Classifying, and Resolving Semantic Conflicts in Distributed Heterogeneous Databases: A Case Study
[www.igi-global.com/article/identifying-classifying-resolving-semantic-conflicts/51144?camid=4v1a](www.igi-global.com/article/identifying-classifying-resolving-semantic-conflicts/51144?camid=4v1a)

Database Support for Workflow Management Systems
[www.igi-global.com/chapter/database-support-workflow-management-systems/20705?camid=4v1a](www.igi-global.com/chapter/database-support-workflow-management-systems/20705?camid=4v1a)

Functional Dependencies For Value Based Identification In Object-Oriented Databases
[www.igi-global.com/chapter/functional-dependencies-value-based-identification/7884?camid=4v1a](www.igi-global.com/chapter/functional-dependencies-value-based-identification/7884?camid=4v1a)