

XQGen - An Algebra-based XPath Query Generator for Micro-Benchmarking

Yuqing Wu, Namrata Lele, Rashmi Aroskar, Sharanya Chinnusamy, Sofia Brenes
Indiana University, Bloomington, USA
{yuqwu, nlele, rároskar, sharchin, sbrenesb}@cs.indiana.edu

ABSTRACT

We propose XQGen, a stand-alone, algebra-based XPath generator to aid engineers in testing and improving the design of XML query engines. XQGen takes an XML schema sketch and user configurations, such as number of queries, query types, duplication factors, and branching factors as input, and generates a set of queries that conform to the schema and configurations. In addition, given a set of label-paths as workload input, XQGen is capable of generating query sets that honor the workload.

Categories and Subject Descriptors

H.2.m [Information Systems]: Database Management—*Miscellaneous*

General Terms

Performance

Keywords

Micro benchmark, query generator, XPath algebra, workload

1. INTRODUCTION

Since XML has become a popular data format for representing semi-structured data, a significant amount of effort has been made by both academia and industry in the research of efficient query processing techniques and the implementation of XML query engines. Naturally, to assess the performance of these techniques and systems, XML benchmarks were proposed. Most of them were macro-benchmarks that tried to stress a system in handling large volumes of data and complicated queries at the application level [1, 5]. Micro-benchmarks [4] focused on capturing the rich variety of data structures and distributions possible in XML, and on isolating their effects on an XML query engine. All these benchmarks come with an XML schema, a data generator that is capable of generating data sets of various scales, and a set of *pre-defined* queries against the XML data sets.

However, it is important to vary the queries and capture the query distribution of certain key XML query sub-languages in order to isolate their effects on an XML query engine. We present XQGen, an algebra-based XPath query

generator that can generate query sets for a user specified XML document on demand, with the capability to satisfy the requirements on the characteristics of the query set specified by the users, such as the number of queries, the sub-languages the queries belong to, and the percentage of queries that yield empty results, or are duplicate, or have predicates. Furthermore, XQGen is capable of taking query workload into consideration and generating query sets that satisfy a given workload. This assists users in assessing the performance of their systems where special technologies have been designed and implemented to take advantage of query workload in pursuit of better query throughput.

2. SYSTEM DESCRIPTION

XQGen is a query generator for micro-benchmarking; our design goal is to generate query sets that feature certain characteristics to help researchers and developers to fine-tune and improve the performance of their systems,

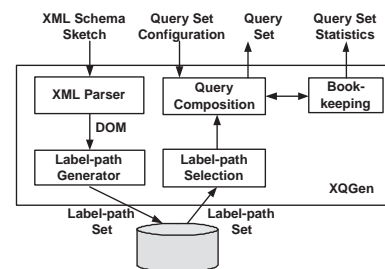


Figure 1: XQGen Architecture

rather than to generate complicated queries that cover the syntax of a query language and to test the completeness of an XML data engine. For this purpose, XQGen features an algebra-based query generation algorithm and sophisticated query set configurations. The XQGen system architecture is shown in Figure 1. It takes the sketch of an XML schema in the form of an XML document as input and generates a set of XPath queries that are meaningful against the data set that conforms to the given schema.

Algebra-based Query Generation. The methodology for identifying sub-languages of a query language and reasoning about whether a given database technique is suitable for answering such queries has been established in the context of relational database systems. For example, a hash index is good for point queries but not range queries, while a B+ tree index is good for both. This methodology was recently introduced in the context of XML to reason about the suitability of structural indices in answering sub-classes of XPath queries [3]. In XQGen, we adopt the two most frequently used sub-algebras $\mathcal{D}[k]$ and $\mathcal{D}^{\downarrow}[k]$, which represent linear downward queries of length up to k and down-

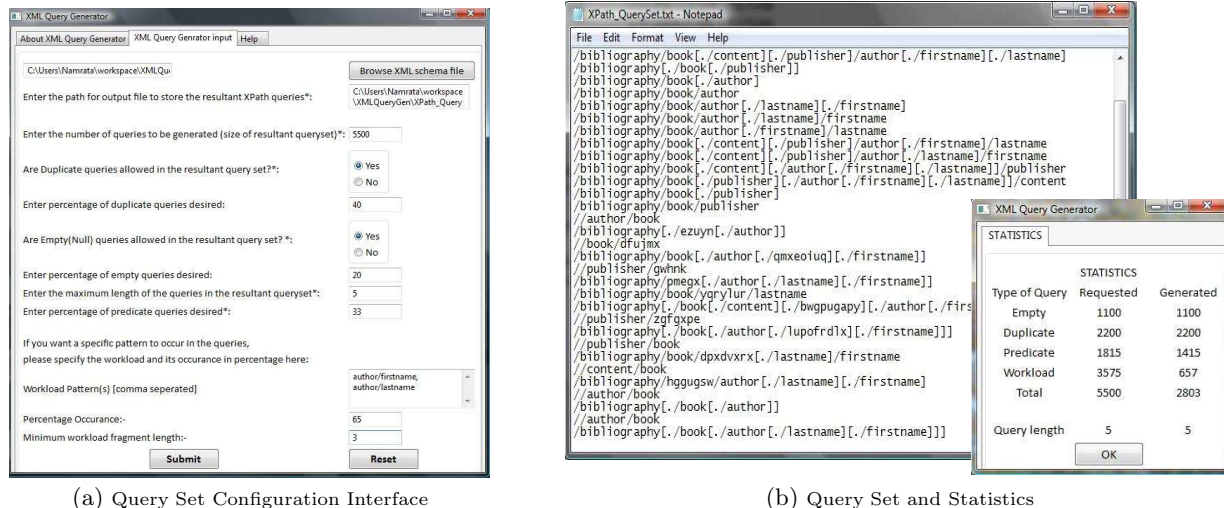


Figure 2: XQGen User Interface

ward predicated queries of length up to k , respectively. The label-paths of an XML document, e.g. $\mathcal{D}[\infty]$ queries, are the building blocks of any queries XQGen may generate. Users are given the choice to configure the system to generate a set of queries that belong to a certain sub-language, such as $\mathcal{D}[3]$, or to generate a set of queries where a portion of the set belongs to a certain sub-language, such as $\mathcal{D}[3]$ with 30% $\mathcal{D}^{\square}[3]$.

Query Set Configuration. In the study of XML query processing, indexing, and optimization techniques (including our own work [2, 3, 6]), it is claimed that various properties of XPath queries have significant impact on query performance, such as the size of the query, including the length of the path and the number of nodes in the query pattern, and the location of the return nodes. An XML engine may use view or caching techniques to maximize the overall throughput of a set of queries. To assess these capabilities, one may want to generate query sets with duplicate queries, queries that yield empty results, or even query sets that feature a certain query workload. To address these issues, XQGen is designed with a sophisticated set of configuration options through which users can specify the characteristics of the query set they expect to generate. A few major options are:

- the total number of queries in the query set;
- the class of queries, e.g. $\mathcal{D}[3]$;
- the percentage of queries that yield empty results;
- the percentage of duplicate queries;
- the percentage of queries with predicates; and
- whether query workload is to be considered. If so, the users are asked to provide a set of label-paths that represent the workload and the frequency threshold for sub-queries in the query set generated to be qualified as workload.

3. DEMONSTRATION PROPOSAL

We will demonstrate XQGen [7], focusing on its strong configuration support and its capability in generating query sets under sophisticated configurations.

Query Set Configuration. XQGen features a graphical interface, as shown in Figure 2(a), from which users can upload the schema sketch of their XML dataset and configure the characteristics of the query set they wish to generate. Users are required to provide the total number of queries in the query set to be generated. Other parameters are op-

tional, such as the duplication ratio, branching ratio, ratio of queries with empty results, workload, and workload ratio. We also welcome users to select from the sample XML schema sketches we provide, or modify our samples to create their own schema sketch for query generation.

Query Set Generation. We will demonstrate the query generation process of XQGen by showing the query sets generated as well as statistical information about the query sets, as shown in Figure 2(b). Please note that not all user configurations can be fulfilled due to schema restrictions, for example, XQGen may identify that an XML document has only 20 distinct labels, but the user requires for a query set of 100 distinct linear queries of length 1. The statistical information summarizes the statistical characteristics of the query set to assist the users in obtaining a better understanding of the query set the user requested and the set that XQGen could generate. It also helps users to conduct more sophisticated analysis when using the query set in their experimental evaluations.

4. REFERENCES

- [1] T. Böhme, *et al.* XMach-1: A Benchmark for XML Data Management. In *BTW*, 2001.
- [2] S. Brenes, *et al.* Trie Indexes for Efficient XML Query Evaluation. In *WebDB*, 2008.
- [3] G. H. L. Fletcher, *et al.* A Methodology for Coupling Fragments of XPath with Structural Indexes for XML Documents. *Information Systems*, 2009.
- [4] K. Runapongsa, *et al.* The Michigan Benchmark: Towards XML Query Performance Diagnostics. *Inf. Syst.*, 31(2):73–97, 2006.
- [5] A. Schmidt, *et al.* XMark: A Benchmark for XML Data Management. In *VLDB*, 2002.
- [6] Y. Wu, *et al.* Workload-aware Trie Indices for XML. In *CIKM*, 2009.
- [7] XQGen. Available at <http://sites.google.com/site/xmlqgen>.