A Method of XML Twig Query Processing based on XML Document Schema

Yi Yu

School of National Education, Nanchang Institute of science & Technology, Nanchang, 330108, China

lunwenzju@163.com

Keywords: XML Schema, Query Processing, Twig Query, XML Document

Abstract. The uncertainty caused by the Ancestor-Descendant relations and wildcard could affect query efficiency in the query of Twig query. For this problem this paper proposes a query method of the Twig query processing based on document schema. Firstly, it matches the Twig query to the schema tree, so as to achieve the specific query type, and remove some indefinite factors of query. Secondly, it computes the Twig query matching result by using the general whole matching algorithm TwigStack. Finally, this paper implements some related experiments and the operation efficiency of the STwigStack algorithm and the TwigStack algorithm, so as to verify the effectiveness of the algorithm.

1. Introduction

With the arrival of the big data era, all kinds of the Internet of Things and Internet have produced massive amounts of data. Most of data is unstructured and semi-structured data, and the semi-structured data is represented mainly as XML form. XML is a markup language which is proposed by the W3C. The XML language has some advantages, such as the good extension, platform-independent and structure, which has become standard of data presentation and transmission. This paper mainly discusses the problems of Twig query processing of semi-structure XML data.

The general query language includes Xpath and XQuery in XML data. The core part of these query languages is generally represented by Twig query schema. How to improve the efficiency of Twig query processing is an important topic in XML data analysis. Some experts and scholars had proposed some methods in accordance with Twig query processing, such as the dual structure match algorithm and whole match algorithm, etc. In these algorithms, the whole match algorithm is mainstream strategy [1], and the Twig-stack [2] is the foundation of many algorithms in whole match algorithm. But if there are uncertainties (Parent-Child and Ancestor-Descendant relations) in data processing, the whole match algorithm has some useless operations. For this problem, this paper proposes an algorithm of XML Twig query processing based on XML document schema. And each node is marked as tuple <start, end, level> in the range encoding.

2. Background Knowledge and the Relevant Definitions

2.1 Data Model and Range Encoding.

XML data is usually represented as a DOM tree. A XML document tree is shown in Fig.1. In the case of a tree, the XML elements represents node, the edges represents structure relationships of element-child element, element-attribute and attribute-value. In query processing, it mainly judges relations between nodes and nodes through location information of node encoding, such as Parent-Child and Ancestor-Descendant relationship. In the XML encoding, the most common encoding is the rage encoding and prefix encoding, and this paper takes the range encoding as node encoding. And each node is marked as tuple <*start, end, level*> in the range encoding [3]. In the preorder traversal of the DOM document, the root node is firstly visited, and the *start* need be assigned value. In the next operation, the node will be taken as the root node. When all nodes of sub-tree are visited, the *end* need be assigned value. The *level* represents the level information of node, and the value of the root node is 1.

In Figure 1, the node subject (2, 187, 2) and the node books (6, 23, 4) meet the ancestor-descendant relations, which also meets the requirement of property 2<6 and 23<187. The node books (5, 186, 3) and the node books (6, 23, 4) meet the Parent-Child relations, and also meets the requirement of property 5<6 and 23<186.

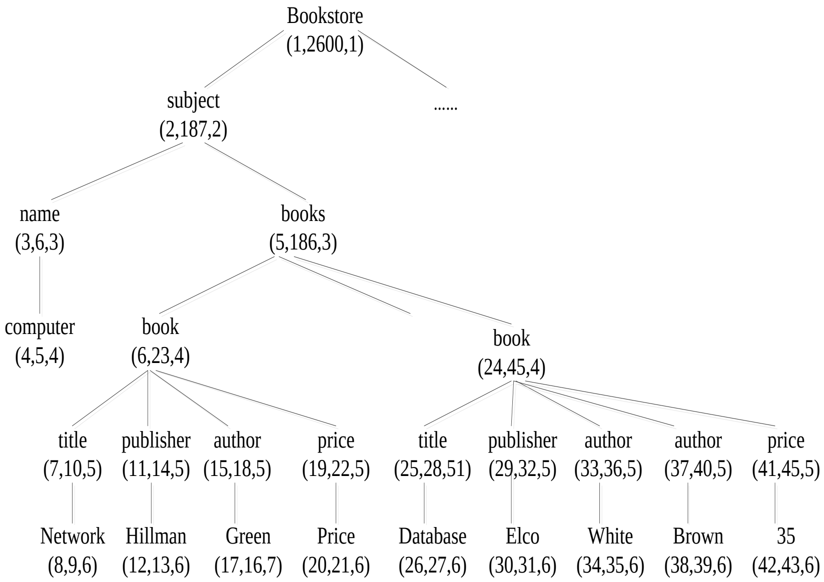


Fig.1 XML DOM model

2.2 Twig Query.

In the XML data query language, the core parts of Xpath and XQuery is represented as Twig query schema. The Twig query is also generally represents as tree structure. For example, the query *Q1://book [/title="Database"]//price* is represented as the tree (Figure 2), the "//" indicates that nodes must meet the requirement of Ancestor-Descendant relations, the "/" indicates that nodes must meet the requirement of Parent-Child relations. The match of The Twig query and XML data are defined as follows:

Definition 1. The match definition of the Twig query *Q* is defined as mapping *e,* which should meet the following conditions:

(1) For any node *n* of query *Q*, the corresponding node e (n) of XML document should meet predicate constraints of node *n*;

(2) For any node *n* and node *m* of query *Q*, the node *e(n)* and node *e(m)* of XML document should meet structure relations of node *n* and node *m*;

2.3 XML Schema Definition.

In order to guarantee the standardization and effectiveness of XML document, W3C defines the DTD (Document Type Definition) and XML Schema to be XML constraints. The DTD and XML Schema are abstracted as schema tree form in this paper, so as to extract query type. In the schema tree definition, the "?" means zero or one occurrence of the element, the "?" means zero or multiple times occurrence of the element, the "+" means one or multiple times occurrence of the element, the "1" means a choice among elements. In addition, the "#text" represents text or numeric data one or multiple times occurrence of the element.

According to XML document of Figure.1, the corresponding XML schema can be defined. The XML schema definition is shown in Figure 3.

In order to implement matching of nodes type, the schema definition will be converted to schema tree format. Thus, type definition computer will be defined as matching of Twig query and schema tree, so as to use the existing Twig query processing algorithm.

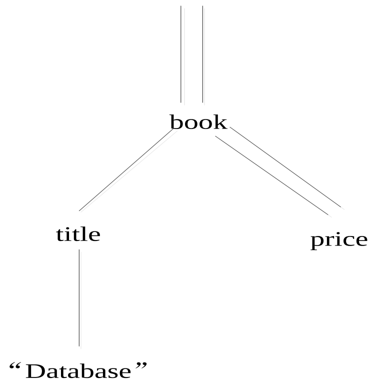
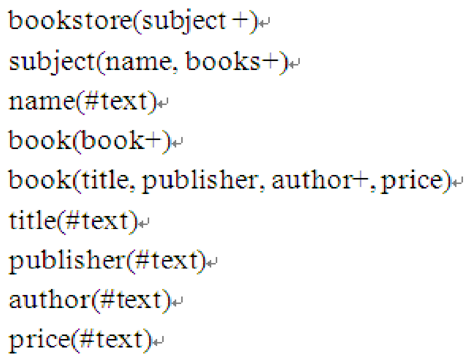


Fig.2 Twig query schema tree



```
bookstore(subject +)↵
subject(name, books+)↵
name(#text)↵
book(book+)↵
book(title, publisher, author+, price)↵
title(#text)↵
publisher(#text)↵
author(#text)↵
price(#text)↵
```

Fig.3 XML schema definition

3. The Design and Implementation of Algorithm

This section will give the whole thinking and the specific implementations of Twig query based on schema processing.

The whole schema of the XML Twig based on the schema processing will be discussed, and the concrete instances will be given to explain its implement processing. Firstly, the query request is converted to the Twig expressive format, and the schema definition of source data is converted to the schema tree. Secondly, it matches the Twig query to schema tree, so as to remove some indefinite factors of query and achieve the Twig query with type. Finally, it computes the Twig query matching result by using the general whole matching algorithm.

In the whole matching method based on the Twig query processing, most of them are improved in accordance with the TwigStack, so the TwigStack method proposed in this paper can fit all the improved TwigStack.

4. The Experiment Results and Analyses

In order to verify the effectiveness and Operating Efficiency of the algorithm, this paper implements comparisons of the STwigStack algorithm and TwigStack algorithm with Java language. In experiments, it takes the general XMark [4] and DBLP [5] as XML testing dataset. The specific testing environment is shown as follows: Window 7 operation system, Intel(R) Core(TM) 2 Quad CPU, 2.66GHZ, 3.49G.

Some related information of the two datasets and query expressions are shown in table 1. In the query expressions, the M1 and N1 do not contain “//” and “*”. There are “//” and “*” in the other query expressions.

Table 1. Testing dataset and query expressions

Dataset	Query	Path expressions
Xmark 180M	M1	/site/open_auctions/open_auction
	M2	/site/people//*/interest/
	M3	/site//close_auction[price>125]/type
	M4	//regions/africa/item[//mailbox//mail/from=Libero Rive]//keyword
DBLP 300M	N1	/dplp/proceedings[year>1979]/sibn
	N2	//article[author=‘Jim Gray‘]/title
	N3	//book[year>2000][publisher=‘Springer‘]/booktitle
	N4	/dblp/*/booktitle

The experiment implements some comparisons of the STwigStack algorithm and TwigStack algorithm. Some different datasets is used in experiment test. They are XMark and DBLP datasets respectively.

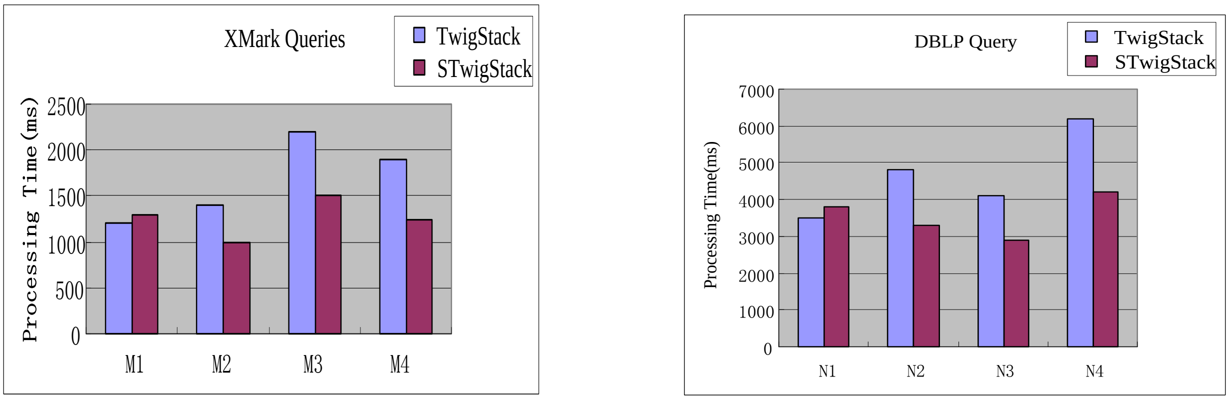


Fig.4 Comparison of running time

The comparison of running time of the TwigStack algorithm and the STwigStack algorithm is shown in Figure 4. From running time analysis of the Figure 4, the STwigStack algorithm couldn't its advantage if there is not judgment of Ancestor-Descendant relations and wildcard in query expressions, and the running time of the STwigStack algorithm is longer than the TwigStack algorithm. While for the other query, the STwigStack algorithm has high query efficiency because the STwigStack algorithm implements certainty processing. Although these degrees of query improvement are different, these degrees of query improvement are up over 30 percent.

5. Summary

The A-D axis and wildcard can cause some uncertain problems in Twig query processing, which will affect query efficiency. For this problem, this paper proposes an algorithm of XML Twig query processing based on XML document schema. This method is based on the TwigStack algorithm. It firstly matches the query to schema, and implements certainty processing for the uncertain elements. Then it computes query result by using the general Twig query processing algorithm. The experiment shows that the STwigStack algorithm can effectively implement query processing and improve the query efficiency. The next step in this work is to consider more query processing and further optimize the algorithm.

Acknowledgments

This research was financially supported by the financial aid of the Science & Technology of Nanchang Institute of science & Technology, under Grant no. GJKJ-15-47.

References

[1]. Bi X, Wang, G.R, and Zhao X.G. Review on Twig Query Processing and Optimization Techniques of XML Data. Journal of Frontiers of Computer Science & Technology. Vol. 7(2013) No. 9, p.769-782.

[2]. Brouno N, Koudas N, and Srivastava D. Holistic Twig joins: Optimal XML pattern Matching, In Proceedings of SIGMOD, Tokyo, 2002, p.310-321.

[3]. Lu J, Chen T, Ling T. W. Efficient Processing of XML Twig Pattern With Parent Child Edges: a Look-ahead Approach. In Proceeding of CIKM, 2010, p. 533-542.

[4]. Wan L. Y, A Fast Approach for SLCA in Keyword Query over XML Document, 2013 the 4th International Conference on Frontiers of Manufacturing and Design Science (ICFMD 2013), Hong Kong, 2013, p.133-136.

[5]. Schmidt A, Waas F, Kersten M. XMark: A Benchmark for XML Data Management. In Proceeding of VLDB Endowment, 2002, p.974-985.