Performance Evaluation on XML Schema Retrieval by Using XSPath

Trupti N. Mahale **Prof. Santosh Kumar**

Abstract — **The contents and structure of the XML document can be retrieved using schemas. As schemas are complex and large in size, hence it is difficult to handle them manually. The conversion of a single schema or bunch of schemas to query is used to get structural schema components. Hence, presenting a new query language, called as XSPath. This language works on graphical representation of XML schemas. Hence it provides navigation capacity by which it easily performs the selection of nodes. Also presenting XPath / XQuery based conversion which will help to evaluate XSPath queries. In this way the usability and efficiency of XSPath technique is been improved with the help of EX-up system.**

KeyWords — **EXup, Schema Querying, XML Schema, XPath, XQuery.**

I. INTRODUCTION

Different schema languages have been presented like DTD, XML Schema etc. After comparing their performance it is been said that W3C recommendation XML Schema is the most commonly used language which is based on XML representation[1]. XML is the easier way to represent loose structured data. Because of its flexibility, it became universal data representation format. As well it was used as medium for transferring data between different applications. However, XML Schemas are XML documents. Hence, the structure of a schema can be followed and its components (namely, element declarations, complex and simple type declarations, and complex types structures) can be optimized through a path language. Schemas can be small and regular but the data are big and complex in some of the large domains as in aviation or weather information. To overcome the problem of large database, logical representation of XML schemas came into picture. Some of the tools are also available to explore schemas into their graphical notation form. But it doesn't satisfy all needs of schema retrieval. Hence proposing a new query language called XSPath[1]. This language satisfies expressing retrieval needs on schemas without considering verbose XML schema syntax. As well it simplify retrieval tasks and offers flexibility to a query language. Schema contained in XML Schema is itself an XML document. To query different schemas, generally simple approaches could be used like XPath or XQuery. But these simple solutions does not reflect the complex expressions as per users intention. Suppose, we want to denote the global shiporder element in the schema as shown in following section:

A. Schema File:

Following is the schema file of shiporder having extension .xsd.

```
<?xml version="1.0"encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="shiporder">
<xs:complexType>
<xs:sequence>
<xs:element name="orderperson"type="xs:string"/>
<xs:element name="shipto">
<xs:complexType>
<xs:sequence>
<xs:element name="name" type="xs:string"/>
<xs:element name="address" type="xs:string"/>
<xs:element name="city" type="xs:string"/>
<xs:element name="country" type="xs:string"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="item" maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:element name="title" type="xs:string"/>
<xs:element name="note" type="xs:string" minOccurs="0"/>
<xs:element name="quantity" type="xs:positiveInteger"/>
<xs:element name="price" type="xs:decimal"/>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:attribute name="orderid" type="xs:string" use="required"/>
</xs:complexType>
</xs:element>
</xs:schema>
```

By considering above schema, the XPath expression /schema/ element [@name=shiporder] could be analyzed to retrieve the shiporder element. This expression is nothing but verbose and it generally prefers simpler expression. We can extend this query as, : find the city element declaration within shiporder which makes the XPath expression much more complicated but the extended form gives simple version of it. To specify the occurrence of referral element within global element requires expressions over internal links. Navigation of such links becomes difficult in XPath. The base of the proposed language is that the schema expressions can be specified and represented in graphical notation of tree structure. It makes navigation of all expressions easier. This proposed architecture transforms XSPath expression into XQuery expression. Using optimized EX-up technique, usability and efficiency of the given approach is been improved. This approach of EX-up system provides a navigational XML query language with efficiency and simpler approach. This language simplifies the retrieval task as well it is offering a power of query language over different tools. Main feature of

this language is two level graphical representation of the expression. This abstract representation makes the expression specification more easier and as well it solves the gap between graph based and textual representation of schemas. The language in this way navigates throughout the nesting structure of element declarations. Transformation of XSPath expressions into XPath expressions is also key feature of proposed language.

This work is explained in different sections. Most closely related work is surveyed in section 2. Section 3 defines overview of system model. Whereas Section 4 gives brief idea about Performance Analysis. Finally will Conclude the overall concept.

II. RELATED WORK

Lots of contributions focus on the satisfactory problem of XPath queries. So Distinguished schema mapping from the well known problem of the schema integration and discussed the similarities and differences between the two[2]. Xpath was modified with two fragments of XPath for which linear time query processing algorithm exists. Xpath can be processed more efficiently. Different algorithms to process query were presented i.e. 1)Axis evaluation, 2)Bottom up algorithm for XPath, 3)Mincontext, 4)Optmincontext [3]. Researchers then investigated important structural properties of a variety of XPath fragments, parameterized with modalities that include qualifiers, upward traversal and recursion[4]. To extend the previous work, they introduced an approach to optimize XPath query by minimizing its wildcard steps. Depending on the layer axis, wildcard steps were minimized by introducing a query rewriting algorithm[5]. Presented an algorithm for schema validation with and without modifications which avoids traversing subtrees of XML document. Algorithm described is in terms of abstraction of XML schemas which achieved 30-95 per performance improvement[6]. Some of researchers introduced a novel paradigm for specifying XML security constraints and investigated the enforcement of such constraints during XML query evaluation[7]. These algorithms were used to transform a query over a security view to an equivalent query over the original document[8]. Investigated the problem regarding XML schema evolution by proposing a set of rule of evolution primitives and studying the impact on XML documents[9]. The new algorithmic concept of iDTD was invented. Also discussed incremental computation, noise numerical predicates and generation of XML schemas[10]. Proposed a schema based approach for queries in XPath to detect semantic errors. Also to avoid unnecessary evaluation of not satisfiable queries[11]. They introduced a Spicy system, an approach which deals with the problem of automatically selecting the best mappings among the two data sources[12]. This spicy system binds together all schema matching as well as mapping generation tools which helps to automate the process with the help of all tools[13]. Overall comparison is done for schema evolution and versioning[14]. Hence developed an algorithm for inferring mapping rules from information schema correspondence and defined the set of rules of answering the query and developed query transformation algorithm[15]. After that it is showed that how stream Schema can be used to validate the consistency of streams. This can efficiently enhance programmability of stream processing systems[16]. Type based and projection based detection of corrupted XML schema mappings[17]. XML graph model was described comparatively more matured which was fully implemented, with all XML features and rules also allowing static validation against common schema mechanism[18]. To study types proposed an alternative type inference system which was more prompt and efficient without any performance deficiency. In this way, inference precision can be improved[19]. Developed a formalized a W3C semantics of property paths. The complexity of relative rules regarding query processing was significantly higher. Whereas previous semantics was in polynomial time complexity[20]. Developed an evaluation technique for two types of regular expressions. By using that technique, it was possible to represent them in simplified manner[21]. Data encoded in XML or other formats may use external schema definition files to specify the structure and content of dataset files[22]. To avoid performance degradation indexing technique is added to existing one. It increases efficiency of result[23].

III. OVERVIEW OF SYSTEM MODEL

Since a schema is an XML document, one could directly use the XPath language for the specification of navigational expressions on schemas. But the XPath doesn't support all features of XML retrieval. As well it takes much longer time to translate and evaluate the schema. Hence considering all previous specifications of Xpath, proposed a query language, named XSPath, specifically for XML schema that gives logical graph-based representations of schemas, which enables the traversal, and gives the way of node selection. Also proposed system specifies translation into tree structure for the retrieval of XSPath queries. Ex-up technique is used for translation and evolution of XSPath expression. The overall architecture is shown in the following Fig. 1. The prerequisites of the XSPath system are XML and its related schema file. User needs to browse both files and then apply XSPath query. At the time of execution, system will load XSPath function and necessary libraries. It will check the syntax of the query and will process it. The XSPath query gets converted into XPath query and gives output with/without graphical notation which depends on the type of query fired. In this way the XPath is been improved with expression length and evaluation time in the form of XSpath language.

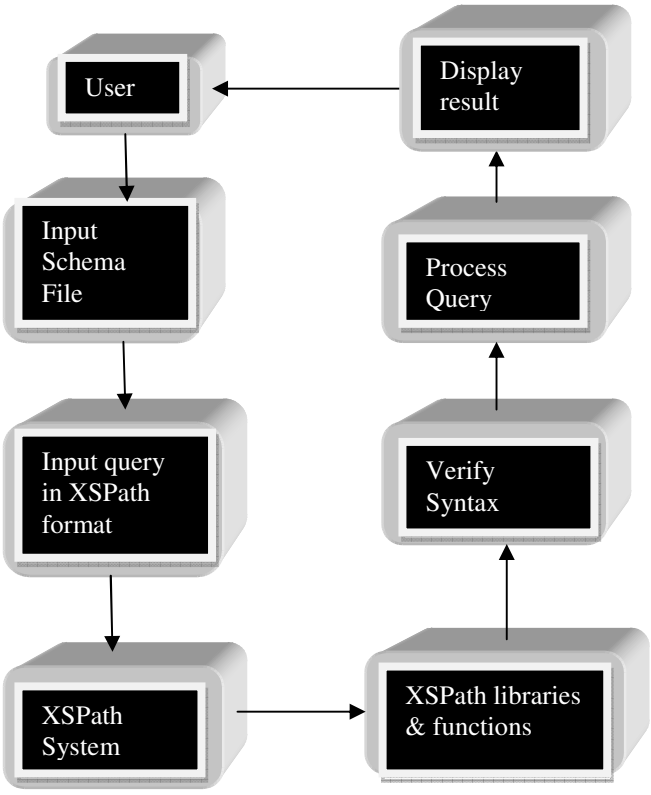

Fig. 1. Architectural view of XSPath system

Different modules of the proposed system are:

A. Loading Parser:

Here we are going to use dom/sax parser to handle(load/read/modify) xml files. The basic and important requirement of the system is parser.

B. Generating XSPath system:

Implementation of the proposed system query language, which is going to be used by user to work on xml and schema files will be achieved. It is equivalent to XPath and XQuery language which are existing systems. The Ex-UP technique will be implemented in this module.

C. Creating Compiler:

Will create a compiler to compile or verify the query(xspath) given by the user. It is used process on user request and to find the syntax error on user query.

D. Display Result:

The proposed system will be used to display the result into the graphical user interface(depends to query). In this section, the system will convert the query into XPath technique also. Translation and XPath evaluation algorithms need to be used for implementation of this module.

In this way the working of the whole system is been defined with the help of this architecture.

IV. PERFORMANCE ANALYSIS

The system is using windows based platform with Java swing as front end and Java XML Packages for logic development. Dom and Sax parsers will be used for processing query. To check the result we have used schema files of datasets from XML data Repository site as there are different datasets available for testing purpose[25]. The system is accepting a query in XSPath format and converting it into tree format for the understanding. The execution time will also show the efficiency of the system.

As per the analytical study of the existing language, XSPath and proposed language, it can be said that the presented system is more efficient, optimized as well as usable with respect to evaluation time. The time required for translation algorithm in ms is been reduced. The three different datasets are used to analyze the performance of the system. The following table gives idea about reduction in time for CD dataset which shows improved performance of proposed system in graph.

Table 1: Performance of CD

Sr. No.	Parameter	XSPath without translation	XSPath with translation	% Reduce time
1	element	22	6.4	70.9
2	Attribute	31.6	9.6	69.62
3	Child	18.2	9.4	48.3
4	type	28.4	12.4	56.33

As shown in the above table1, after applying translation algorithm, the time required to solve XSpath query is been reduced with respect to different attributes.The following graph is also showing the improved performance of the XSPath by using translation algorithm.

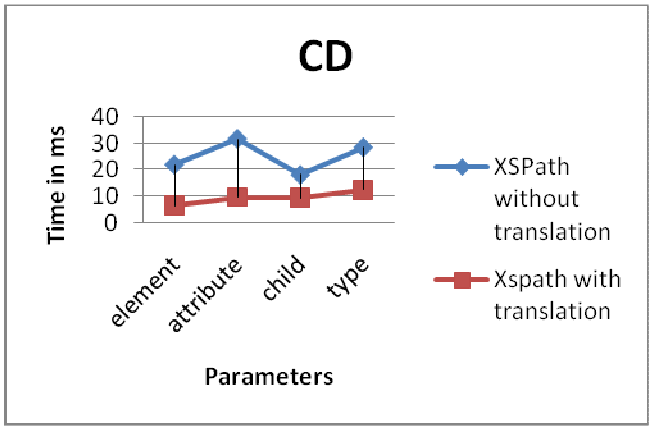

Fig.2. Performance of CD

The following table gives idea about reduction in time for National dataset which shows improved performance of proposed system in graph.

Table 2: Performance of National

Sr. No.	Parameter	XSPath without translation	XSPath with translation	% Reduce time
1	element	22.4	12.4	44.64
2	Attribute	26.6	6	77.44
3	Child	33.2	19.8	40.36
4	type	31.2	9.6	69.23

As shown in the above table2, after applying translation algorithm, the time required to solve XSpath query is been reduced with respect to different attributes.The following graph is also showing the improved performance of the XSPath by using translation algorithm.

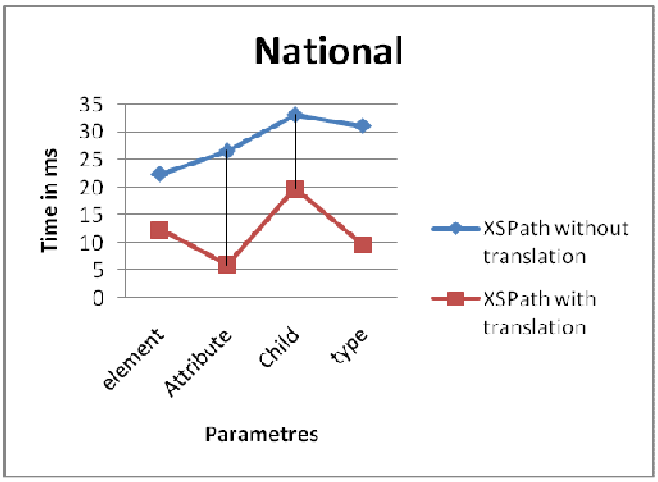

Fig.3 Performance of National

The following table gives idea about reduction in time for Supplier dataset which shows improved performance of proposed system in graph.

Table 3: Performance of Supplier

As shown in the above table3, after applying translation algorithm, the time required to solve XSpath query is been reduced with respect to different attributes.The following graph is also showing the improved performance of the XSPath by using translation algorithm.

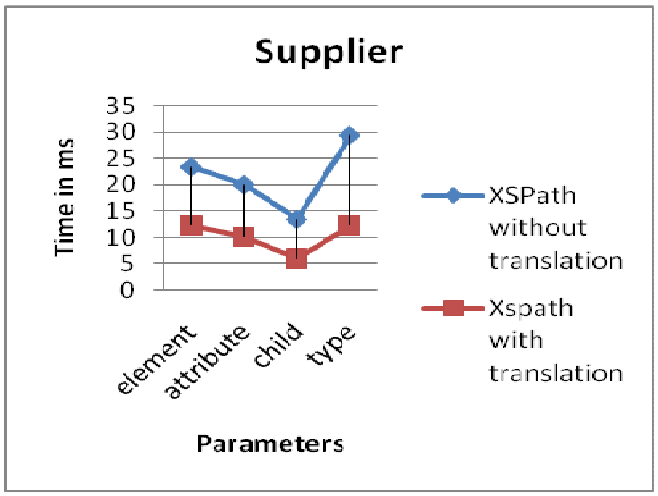

Fig.4 Performance of Supplier

In this way the working of proposed language is more efficient and improved as compared to the existing one. It can be summarized in following graph which is showing overall average reduction in time with respect to all three datasets.

Table 4. Average Time Reduction

Sr. No.	Dataset	Size	Avg. % reduce time
1	CD	2KB	61.28
2	National	1KB	57.91
3	Supplier	2KB	52.26

Sr. No.	Attribute	XSPath without translation	XSPath with translation	% Reduce time
1	element	23.4	12.4	47
2	Attribute	20	10.2	49
3	Child	13.4	6	55.22
4	type	29.4	12.4	57.82

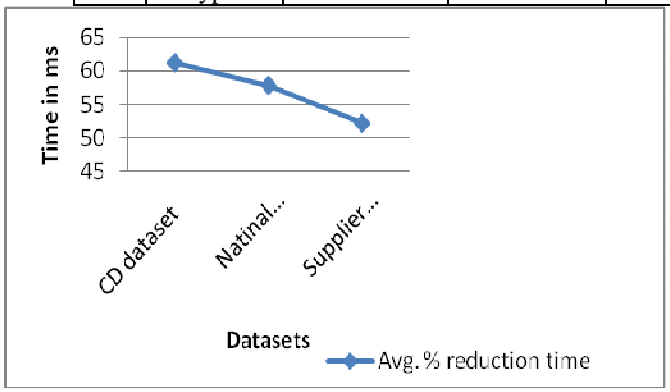

Fig.4 Average time reduction

Hence proposing an efficient navigation approach to querying on XML schemas using the Ex-up technique which improves the usability of XML schema using XSPath language that gives accuracy with the help of translation algorithm.

CONCLUSION

The XPath was designed to operate on schema file as schemas are the structural representation of XML file. But the performance of XPath was limited in some context of rules. Hence proposed a navigational XML schema query language which defines a type analysis for the given language as well it optimizes schema conversion with translation algorithm independently. Hence XSPath is been derived from XPath language. The proposed system gives the conversion of schema file to simplified tree format structure which will helps us to compare performance of both the languages. XPath translation algorithm is introduced to define correctness and complexity of result conversion. In this way Ex-up is an optimized technique for translation and evaluation of XSPath expression. The specified performance analysis using different datasets, shows the usability of the given system and its practical applicability.

ACKNOWLEDGEMENT

The authors would like to acknowledge Computer Engineering department, SITRC and all the people who provided with the facilities being required and conductive conditions for completion of the review paper.

REFERENCES

[1] F.Cavalieri, G. Guerrini, M. Mesiti, "XSPath: Navigation on XML Schemas Made Easy", IEEE Transactions on Knowledge and Data Engineering,vol.26, no. 2, 2014

[2]R. J. Miller, L. M. Haas, M. A. Hernandez, "Schema Mapping as Query Discovery", VLDB '00 Proceedings of the 26th International Conference on Very Large Data Bases ,pp 77-88 , 2000

[3]S. Yahia, S. Cho, Laks V. S. Lakshmanan, D. Srivastava, "Minimization of Tree Pattern Queries",SIGMOD '01 Proceedings of the 2001 ACM SIGMOD international conference on Management of data pp 497-508 , 2001

[4]G. Gottlob, C. Koch, and R. Pichler, "Efficient Algorithms for Processing XPath Queries", Proc. Int'l Conf. Very Large Data Bases, pp. 95-106, 2002.

[5]B. Lerner, "A Model for Compound Type Changes Encountered in Schema Evolution", Database Systems ,Volume 25 Issue 1, pp 83-127 , 2002

[6] M. Benedikt, W.i Fan, G. Kuper, "Structural Properties of XPath Fragments", ICDT '03 Proceedings of the 9th International Conference on Database Theory, pp. 79-95, 2003

[7]C.Y. Chan, W. Fan, and Y. Zeng, "Taming XPath Queries by Minimizing Wildcard Steps", Proc. 30th Int'l Conf. Very Large Data Bases, pp. 156-167, 2004.

[8]M. Raghavachari, O. Shmueli, "Efficient Schema-Based Revalidation of XML", IEEE Transactions on Knowledge and Data Engineering archive Volume 19 Issue 4, pp. 554-567, 2004

[9]W. Fan, C. Chan, "Secure XML Querying with Security Views", SIGMOD '04 Proceedings of the 2004 ACM SIGMOD international conference on Management of data pp. 587-598 , 2004

[10]G. Guerrini, M. Mesiti, and D. Rossi, "Impact of XML Schema Evolution on Valid Documents", Proc. Seventh Ann. ACM Int'l Workshop Web Information and Data Management, 2005.

[11]G. Bex, F. Neven, T. Schwentick, K. Tuyls, "Inference of Concise DTDs from XML Data", VLDB '06 Proceedings of the 32nd international conference on Very large data bases ,pp. 115-126, 2006

[12]J. Groppe and S. Groppe, "Filtering Unsatisfiable XPath Queries," J. Data Knowledge Eng., vol. 64, no. 1, pp. 134-169, 2008.

[13]Bonifati, G. Mecca, A. Pappalardo, S. Raunich, G. Summa, "The Spicy System: Towards a Notion of Mapping Quality ", SIGMOD '08 Proceedings of the 2008 international conference on Management of data pp. 1289-1294 , 2008

[14]G. Guerrini and M. Mesiti, "XML Schema Evolution and Versioning: Current Approaches and Future Trends", Open and Novel Issues in XML Database Applications, E. Pardede, eds., Idea Publishing, 2009.

[15]A. Bonifati, E. Chang, T. Ho, Laks V. S. Lakshmanan, R. Pottinger, Y. Chung ,"Schema mapping and query translation in heterogeneous P2P XML databases ", The VLDB Journal The International Journal on Very Large Data Bases archive Volume 19 Issue 2,Pages 231-256, April 2010

[16]P. Fischer, K. Esmaili, R. Miller, "Stream Schema: Providing and Exploiting Static Metadata for Data Stream Processing ", EDBT '10 Proceedings of the 13th International Conference on Extending Database Technology Pages 207-218 , 2010

[17]Dario Colazzo, "Schemas for safe and efficient XML processing ",IEEE 29th International Conference on Data Engineering (ICDE) (2011) pp: 1378-1379, 2011

[18]A. Mller and M.I. Schwartzbach, "XML Graphs in Program Analysis", Science Computer Programming, vol. 76, no. 6, pp. 492-515, 2011.

[19]D. Colazzo and C. Sartiani, "Precision and Complexity of XQuery Type Inference", Proc. 13th Int'l Symp. Principles and Practices Declarative Programming Languages, pp. 89-100, 2011.

[20]K. Losemann and W. Martens, "The Complexity of Evaluating Path Expressions in SPARQL", Proc. 31st Symp. Principles Database Systems, pp. 101-112, 2012.

[21]L. Libkin, D. Vrgoc, "Regular Path Queries on Graphs with Data", Proceedings of the 15th International Conference on Database Theory pp. 7485, 2012

[22]Jeppesen, "Schema Definition and Validation Files", Computer Programming, vol. 76, 2012

[23]D.Karthiga, S.Gunasekaran, " Optimization of Query Processing in XML Document Using Association and Path Based Indexing", International Journal of Innovative Research in Computer and Communication Engineering Vol. 1, Issue 2, April 2013

[24]T. Mahale, S. Kumar, "Review Paper on Querying XML Schema with EX-up Technique for Accuracy", IJEEM,Vol. 3, Issue 1,pp.5-9, 2014

[25]XML data Repository , online at http://www.cs.washington.edu /research/projects/xmltk/xmldata

Trupti N. Mahale
B.E. Computer (Savitribai Phule Pune University)
PG Student,
Savitribai Phule Pune University,
SITRC College,
Nashik-422213
Trupti.mahale31@gmail.com

Prof. Santosh Kumar
Assistant Professor,
Savitribai Phule Pune University,
ME Coordinator, Computer Department, SITRC, Nashik.
Santosh.kumar@sitrc.org
ISTE Member