Fuzzy Logic Based XQuery operations for Native XML Database Systems

[1]E.J.Thomson Fredrick and [2]Dr.G.Radhamani

[1]Senior Lecturer, Department of M.C.A
J.J College of Engineering & Technology
Trichy-9
[2]Director & Professor, Dr.G.R.D College of Science
Coimbatore
thomson500@yahoo.com, radhamanig@hotmail.com

Abstract

In recent years, many researchers focused on the research topic of XML database systems. The extraction of data from XML documents is an important issue for XML research and development. However, viewing XML documents determines the query methods. Firstly, this paper describes a natural way to extract data from XML documents using XQuery. This paper illustrates implementing Fuzzy Logic techniques to extract data from XML documents. This paper also describes a Fuzzy Logic based flexible XQuery language for XML. The proposed fuzzy XQuery processing techniques allow the users to use linguistic terms based user-defined functions in the XQueries. This paper demonstrates that how the flexible Fuzzy XQuery provides better output than normal XQuery language using eXist Native XML database. The proposed fuzzy query processing techniques can deal with the users' fuzzy XQueries in a more flexible and more intelligent manner.

Keywords: *XML, XQuery, Fuzzy XQuery,Native XML Database*

1. Introduction

XML is becoming a dominant standard for storing and exchanging information. With its increasing use in areas such as data warehousing and e-commerce, there is a rapidly growing need for rule-based technology to support reactive functionality on XML repositories. XML Database vendors rushed to enrich their products with more flexible and advanced features to make them satisfy the requirements of modern applications. X*ML* (Extensible Markup Language) has become a standard format to exchange information over the Internet, and the importance of database technologies that support storage, processing, and delivery of XML is still increasing.[6]

Most of the existing XML query languages are based on SQL. Unlike queries on traditional relational databases whose results are always flat relations, the results for XML queries are complex. Thus, XML queries have two components: querying part and result constructing part. The existing XML query languages have to intermix the querying part and result constructing in a nested way and thus make queries complicated, which is an inherent problem inherited from SQL.

This paper present new fuzzy logic based XQuery processing techniques for XML database systems, where the weights of attributes of the user's queries can be represented by fuzzy numbers. This paper compares normal XQuery processing techniques with Fuzzy logic based XQuery processing techniques. This paper also demonstrates how fuzzy XQuery processing techniques provide better output than normal XQuery operations. The proposed fuzzy query processing techniques can deal with the users' fuzzy queries in a more flexible and more intelligent manner.

The rest of this paper is organized as follows. In Section 2, this paper reviews the research works done on XQuery operations, Fuzzy SQL operations on Databases and Fuzzy Logic based operations on XML. In section 3, this paper demonstrates XQuery with user-defined functions. In section 4, this paper presents XQuery operations on Fuzzy Logic. The conclusion is discussed in section 5.

2. Literature Review

In [1], Bosc, P. Pivert,O proposed SQLf as a flexible querying language for relational databases conceived to be a complete extension of SQL with fuzzy logic. Fuzzy queries supported by SQLf involve fuzzy terms whose semantic depends of the user and the application domain. In [8], Shyi-Ming Chen and Yu-Chuan Chen presented new fuzzy query processing techniques for fuzzy database systems, where the weights of attributes of the user's queries can be represented by fuzzy numbers. The proposed fuzzy query processing techniques also allow the users to use linguistic terms in the queries represented by fuzzy sets. In [2], Buche *et al* proposed a fuzzy-based XML querying system that performs approximate comparisons between query and data trees. This technique supports imprecise data via possibility distributions. In [3], Calms *et al.* discussed some issues raised in fuzzy querying by handling semi-structured information.

In [7], Marlene Goncalves and Leonid Tineo were inspired by the research work of P. Bosc and O. Pivert [1] and proposed more flexibility to XQuery by means of fuzzy logic use. As a step for the XQuery extension, they have focused their attention to XPath expressions. They specified fuzzy terms through XML using the XML Schema language. But this paper proposes fuzzy query processing techniques allow the users to use linguistic terms based user-defined functions in the XQuery expressions represented by fuzzy sets. The linguistic terms based user-defined functions calculate fuzzy set values. Then the fuzzy set values will be defuzzified by using the centroid method.

3. XQuery with User-Defined Functions

According to [4], XQuery combines the features from several earlier XML query languages, in particular XPath . Through XPath, Document Fragments can be extracted from an XML document. Nested loops iterate over these fragments to further extract Document Fragments and construct sequences of output Document Fragments. Variable assignment supports complex computations based on content and structure of the input. This paper illustrates the XQuery operations with user-defined functions by means of a few examples. This paper extends the XQeries with Fuzzy Logic based User-defined functions in section 4.

Let us see how the problems occurring in XQuery Commands. Let us assume that a manager of an Electronics division wants to find the best performed salesman during the past 10 festival season days. He is fixing this criteria for getting bonus that salesman should do sales for Rs.4000 within 10 days. Then he can try the following XQuery command for matching the criteria in his mind.

```
<output>
{
  for $emp in doc("sale.xml")/salesman/emp
  let $eid := $emp/empid/text()
  let $en := $emp/ename/text()
  let $sa := $emp/sales/text()
  let $d:=$emp/days/text()

  return if  ($sa >=4000 and $d <=10) then
 <emp>
  <empid> {$eid} </empid>
  <ename> {$en} </ename>
  <sales>{$sa}</sales>
  <days>{$d}</days>
</emp>
else ()
}</output>
```

Then the output for the above command is :

```
<output>
<emp>
<empid>1003</empid>
<ename>Joshua</ename>
<sales>5000</sales>
<days>10</days>
</emp>
<emp>
<empid>1004</empid>
<ename>Kevin</ename>
<sales>4998</sales>
<days>3</days>
</emp>
<emp>
<empid>1005</empid>
<ename>Davidson</ename>
<sales>4100</sales>
<days>9</days>
</emp>
<emp>
<empid>1006</empid>
<ename>Martin</ename>
<sales>4095</sales>
<days>2</days>
</emp>
<emp>
<empid>1007</empid>
<ename>Jason</ename>
<sales>4000</sales>
```

<days>10</days>
</emp>
<empid>1009</empid>
<ename>George</ename>
<sales>4100</sales>
<days>11</days>
</emp>
<emp>
<empid>1008</empid>
<ename>Henry</ename>
<sales>3999</sales>
<days>3</days>
</emp>
</output>

If the above output is analyzed, the XQuery won't display the George and Henry salesman records***. (That's why George and Henry records have been striked out).*** Since the above XQuery operations implementing only Boolean conditions, it skips the George and Henry records. Although they are very close to the Manager's criteria, XQuery won't display their records. Moreover, if Joshua and Kevin records are compared, Kevin sales performance is better than Josha. But Kevin record order is behind Joshua. Similarly, if Davidson and Martin records are compared, Martin sales performance is better than Davidson. It is found from our analysis that the Manager won't get satisfaction from the XQuery output with his criteria in mind. According to the research, this paper found out the following limitations with XQuery operations.

- XQuery is forced to make arbitrary determinations about what it can do.
- XQuery does not fit the exact criteria people have in their minds.
- XQuery commands are executed on the basis of only crisp or classical logic.

In order to overcome the above limitations, this paper proposes XQuery operations based Fuzzy Logic in the next section.

4. XQuery Operations Based On Fuzzy Logic

Fuzzy XQueries are based on fuzzy set theory proposed by L. A. Zadeh,1965 [9], whose goals are to store imprecise data, to process user's imprecise queries, and to provide proper information to users to overcome the drawbacks of the normal XQuery Operations. Fuzzy XQueries provide a representation scheme for dealing with vague or uncertain concepts.
Unlike Boolean logic Fuzzy XQueries deal with data that is vague, ambiguous, incomplete and imprecise. Instead of applying crisp boundaries to delineate the search space, the space can be represented linguistically using the concept of fuzzy logic. [8]. Fuzzy logic provides a flexible and fluid method of defining semantic concepts within the Native XML database and provides the basis for a much richer and much more powerful method of looking through a XML database. In a fuzzy XQuery, the selected records are ranked according to their compatibility with the semantics – the intent - of the query. This provides a measure of how well a record fits in with the complete set of XML records retrieved. Fig.1 shows the working Fuzzy XQuery in Native XML Database system.

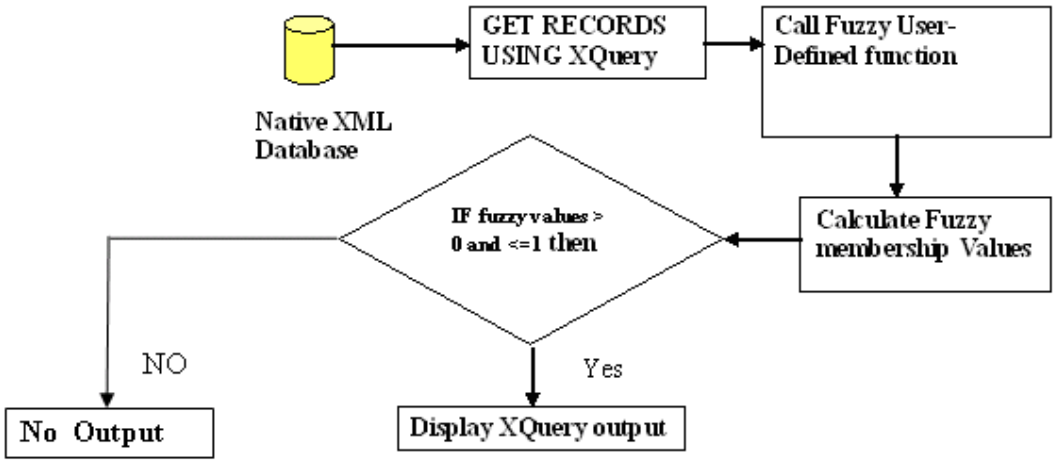

Figure 1. Working of Fuzzy XQuery

Fuzzy XQueries are working on the basis of Fuzzy sets. This paper consider three fuzzy sets that represent the concepts of good salesman, average salesman and bad salesman using the sales amount criteria. A reasonable expression of these concepts by triangular membership functions S_1, S_2, and S_3 is shown in Fig.2. These functions are defined on the interval [1000,5000] as follows :

$$S_1(x) = \begin{cases} 1 & \text{when } x<1000 \\ (3000-x)/2000 & \text{when } 1000 <= x <=3000 \\ 0 & \text{when } x>300 \end{cases}$$

$$S_2(x) = \begin{cases} 0 & \text{when } x<4000 \\ (x-4000)/1000 & \text{when } 4000 <= x <=5000 \\ 1 & \text{when } x>5000 \end{cases}$$

$$S_3(x) = \begin{cases} 0 & \text{when } x<2500 \\ (x-2500)/2000 & \text{when } 2500 <= x <=4500 \\ 1 & \text{when } x>4500 \end{cases}$$

Figure 2. Membership functions representing the concepts of a bad, average and good salesman.

In order to overcome the limitations of normal XQuery in section 3, this paper proposes Fuzzy XQuery operations on the Native XML Database system. Let us take the same problem described in the section 3. Now the manager of an Electronics division applies the Fuzzy XQuery like this to find out the best performed salesman.

```
<output>
{
  for $emp in doc("salesman.xml")/salesman/emp
  let $eid := $emp/empid/text()
  let $en := $emp/ename/text()
  let $sa := $emp/sales/text()
  let $d:=$emp/days/text()
  return if {salesman=good } then
<emp>
  <empid> {$eid} </empid>
  <ename> {$en} </ename>
  <sales>{$sa}</sales>
  <days>{$d}</days>
</emp>
else ()
}</output>
```

Output for the above Fuzzy XQuery is the following :

<output>

```
<emp>
<empid>1004</empid>
<ename>Kevin</ename>
<sales>4998</sales>
<days>3</days>
</emp>
<emp>
<empid>1003</empid>
<ename>Joshua</ename>
<sales>5000</sales>
<days>10</days>
</emp>
<emp>
<empid>1006</empid>
<ename>Martin</ename>
<sales>4095</sales>
<days>2</days>
</emp>
<emp>
<empid>1008</empid>
<ename>Henry</ename>
<sales>3999</sales>
<days>3</days>
</emp>
<emp>
<empid>1005</empid>
<ename>Davidson</ename>
<sales>4100</sales>
<days>9</days>
</emp>
<emp>
<empid>1009</empid>
<ename>George</ename>
<sales>4100</sales>
<days>11</days>
</emp>
<emp>
<empid>1007</empid>
<ename>Jason</ename>
<sales>4000</sales>
<days>10</days>
</emp>
</output>
```

If the above output is analyzed with section 3 Normal XQuery output, now the manager of the Electronics division will get satisfied output. In normal XQuery output, George and Henry records were skipped for getting bonus. Because they haven't satisfied the boolean criteria that sales amount should be above or equal to Rs.4000 and within 10 days. Since the Fuzzy XQuery follows human brain oriented approach, it considers the both the salesman

George and Henry for getting bonus. Moreover, the arranged order for the performance of the seven salesmen is confusing in Normal XQuery. But Fuzzy XQuery arranges all the salesmen according to the generated Fuzzy values. Normal XQuery gave only second position for Kevin. But Kevin is given first position in Fuzzy XQuery. Because Kevin collected Rs.4998 within 3 days, whereas Joshua collected only Rs.5000 in 10 days. Same like that Martin is given fourth position in Normal XQuery output, whereas Fuzzy XQuery gives third position. It is very clear that Fuzzy XQuery overcomes the limitations of Normal XQuery operations.

5. Conclusion And Future Work

Traditional querying languages suffer of rigidity, in the sense that fails in express user preferences and give discriminated answers. Despite expressive power XML querying languages, they also present this problem. In this paper we have deal with the problem of giving more flexibility to XQuery by means of fuzzy logic use. This paper proves from the research work that Fuzzy XQuery overcomes data retrieval problems in Normal XQuery operations. We can use linguistic terms based fuzzy user defined functions in Fuzzy XQuery operations. Fuzzy XQueries are useful for decision making in E-Commerce business transactions. Fuzzy XQuery mimics human decision making in retrieving records from Native XML database. In future, we plan to introduce Fuzzy triggers and Normal triggers in XQuery operations. We also plan to introduce a fuzzy logic based integrity constraints using XML Schema.

References

[1] Bosc, P. Pivert,O. (1995) 'SQLf: A Relational Database Language for Fuzzy Querying', IEEE Transactions on Fuzzy Systems,Vol 3, No. 1, February.

[2] Buche, P., Dibie-Barthèlemy, J., and Wattez, F. (2006). 'Approximate querying of XML fuzzy data'. In springer (Ed.), *Proceedings of the 7th international conference FQAS* , (Vol. 4027/2006). Milan,Italy.

[3] Calms, M. D., Prade, H., & Sdes, F. (2007). 'Flexible querying of semistructured data: A fuzzy-set based approach'. International Journal of Intelligent systems, Vol.22, pp. 723-737, July.

[4] Chamberlin,D. Robie,J. (2008) '*XQuery 1.1: An XML Query Language', W3C Working Draft*, W3C, December.

[5] Don Chamberlin, Michael Carey, Daniela Florescu, Donald Kossmann and Jonathan Robie (2006) '*XQueryP:Programming with XQuery',3rd ACM International Workshop on XQuery Implementation,Experience, and Perspectives*, June,Chicago, Illinois.

[6] Gang Gou, Chirkova,R.(2007) ' *Efficiently Querying Large XML Data Repositories: A Survey'*, IEEE Transactions on Knowledge and Data Engineering, October.

[7] Marlene Goncalves and Leonid Tineo (2007), 'A new step towards Flexible XQuery", Journal of Revista Avances en Sistemas e Informática',Vol.4 No.3, December.

[8] Shyi-Ming Chen and Yu-Chuan Chen (2003) '*New fuzzy query processing techniques for fuzzy database systems'*, International Journal of Fuzzy systems, Vol.5, pp. 161- 170.

[9] Zadeh,L.A.(1965) 'Fuzzy Sets', *Information and Control*, Vol. 8, pp. 338-353, .