Towards the Fuzzification of XQuery Language

Pannipa Sae Ueng¹, Srdjan Skrbic², Wiphada Wettayaprasit¹
¹Prince of Songkla University, Faculty of Science, Thailand
²University of Novi Sad, Faculty of Science, Novi Sad, Serbia
email: pannipa.s@psu.ac.th, shkrba@uns.ac.rs, wwettayaprasit@yahoo.com

Abstract: In this paper, we propose a way to incorporate fuzzy information into XML documents and present an attempt to fuzzify XQuery language and implement an interpreter for such a derivate using XML native database. Fuzzy XQuery queries are translated to standard XQuery queries and executed against an existing XML native database. Obtained results are then processed using fuzzy logic mechanisms in order to obtain the final results.

Key Words: Fuzzy XQuery / XQuery / Native XML Database / XML

1. INTRODUCTION

XML (eXtensible Markup Language) is becoming a standard for data exchange, especially global exchange over the internet. On the other hand, each XML document is a text file. Then, it is obvious that XML can be stored by using a file system. However, access to querying, indexing, transactions and other functionalities that needed for efficient management of XML documents would be very limited. Moreover, the diverse use of XML leads to requirements such as frequent updating, analyzing, and selecting of required documents will be based on certain conditions. Thus, it is an imperative to use some kind of an XML database [1]. Most of the real world information are imprecise and incomplete values such as: “rich”, “slow”, and “tall”. The standard XML does not have facilities to support usage of this kind of information. For this reason, we attempt to incorporate fuzzy set into native XML database in an effort to manage vague data.

The organization of this paper is as follows: section two contains literature review. The third section, presents some basic knowledge on fuzzy set, XQuery, and XML database. The architecture of the future fuzzy XQuery system is presented in the fourth section. The last section is conclusion remarks.

2. RELATED WORK

XML database allows the use of uncertain and imprecise values in clearly defined database structure. In this type of papers, authors are trying to define fuzzy XML model that will rely on existing database architecture. Definition of uncertain value in database is achieved by reserving a part of database for fuzzy m

model that models fuzzy membership function. Using an unconstrained set of membership function would be difficult to implement. Then, authors define finite set of fuzzy membership function types that is enough for the most usage. For example, Rodrigues et al. [2] presented Apania - a new fuzzy database architecture.


Complexity of implementation of these extensions is problematic. Even if there is some kind of an implementation, it only contains some example functionalities.

3. BACKGROUND

3.1. Fuzzy sets

Fuzzy sets were introduced by Lotfi A. Zadeh [8]. A fuzzy set A of a universe X is determined by its membership function \( \mu_A : x \to [0,1] \), where for each \( x \in X \), \( \mu_A(x) \) presents membership degree of an element \( x \) to the fuzzy set \( A \). The fuzzy set formalism can be used in two different ways: (i) in the database, in order to represent imprecise data expressed in term of possibility distributions or (ii) in the queries, in order to represent fuzzy selection criteria which expresses the preferences of the end-user.

3.2. XQuery

XQuery is a standardized language that can be used to query XML documents such as SQL is used to query
relational database tables [9]. The main purpose of XQuery is to extract data from XML database. The XQuery is structured in FLWOR statement which is a form of a for loop. It is also similar to SQL select clause. FLWOR stands for “For, Let, Where, Order By, and Return”. For example, suppose we have documents in our container that contains a snippet that looks like the one given in Fig. 1.

```xml
@student>
  <name>John</name>
  <height>180</height>
</student>
```

Fig. 1. A snippet from XML file

In this case, queries against the container for these documents return the document in order by their document names. But suppose we want to select the students in our container that have height more than 160 cm, ordered by name. We can do this with a FLWOR expression as shows in Fig. 2.

```xml
for $i in collection("myContainer.dbxml")/student
  where $i/height > 160
  order by $i/name descending
return $i/name
```

Fig. 2. An example of an XQuery query

### 3.3. XML Database

There are two types of XML database: XML-enabled database and native XML database [10].

An XML-enabled database is a relational or object-relational database with added feature for supporting XML.

A native XML database is a database designed especially to store XML documents. It stores XML documents in their native form. Like other database, it supports features like transactions, multi-user access, security, query language, programmatic APIs, and so on.

### 4. SYSTEM ARCHITECTURE

The system architecture of our fuzzy XQuery interpreter consists of a FMKB (Fuzzy Meta-Knowledge Base), an XML native Database, and a client as shows in Fig. 3.

![Diagram](image)

Fig. 3. The architecture of our approach

When a client enters a fuzzy query and sends it to the server, the server that stores the linguistic terms and possibility distributions in an XML format will analyze the query and return the results to the client. We describe more details in continuation.

### 4.1. Fuzzy XQuery

The fuzzy XQuery is a name for various types of extension to the standard XQuery to allow flexible queries. User can specify linguistic terms defined inside the FMKB and threshold with each of the linguistic terms into the XQuery. We give an example of EBNF syntax for threshold definition [11] in Fig. 4.

FilterExpr ::= PrimaryExpr PredicateList
PredicateList ::= Predicate*
Predicate ::= 
  "[" (Expr | TholdExpr) "]"
TholdExpr ::= 
  "threshold" DecimalLiteral

Fig. 4. The EBNF syntax of the threshold definition

For example, Fig. 5 shows a query that retrieves students who are tall with a threshold of 0.6.

```xml
for $i in collection("myContainer.dbxml")/student
  where Sheight = tall [threshold 0.6]
  order by $i/name descending
return $i/name
```

Fig. 5. An example of the fuzzy XQuery

We plan to define our own variation of fuzzy extensions to XQuery. It will be assembled having in the previous work in this field and results related to usage of fuzzy logic in relational databases published in [12, 13, 14, 15].

### 4.2. Fuzzy Meta-Knowledge Base

The FMKB contains linguistic variables, linguistic terms of the linguistic variables and a possibility distribution as shows in Fig. 6. The linguistic variable "height" has a value of linguistic term “tall” and it is corresponding to the trapezoidal distribution with A, B, C, and D values of 160, 170, 190, and 200, respectively (see Fig. 7). It is worth of notion that linguistic variables are only names for certain membership functions defined inside the FMKB.

```xml
<?xml version="1.0"?>
<height>
  <labels>
    <tall A=160, B=170, C=190, D=200 />
  </labels>
</height>
```

Fig. 6. File height.xml

![Diagram](image)

Fig. 7. Trapezoidal distribution of the linguistic term “tall”
4.3. Native XML Database

The database is a container for the collection of XML documents and allows us to query and manipulate those documents as a set. Our main idea is to use an existing native XML database implementation – eXist which is an open source XML native database management system [16]. Fuzzy extensions will be built by having this open source software as foundation.

4.4. Query Processing

The query processing is intended to have 3 steps for execution. First, a query is checked against our fuzzy XQuery variation syntax. Then the fuzzy XQuery query will be transformed to classical XQuery query as shown in Fig. 8. The query processor consults the FMKB for the definition of the linguistic variable and calculates the range of values specified by this fuzzy expression and threshold. If a threshold is given, all the tuples in the result set that have a satisfaction degree below the threshold will be removed. Finally, server retrieves the records from database and shows the resulting data to client.

for $i$ in collection("myContainer.dbxml")/student
where $i/name$ > 166 and $i/name$ < 194
order by $i/name$ descending
return $i/name$

Fig. 8. The fuzzy XQuery after transformation

We use a grocery store application example to illustrate our approach. Detailed information about each product are stored in a XML native database collection with 6 tags: category, item, code, price, inventory, and vendor. Let us consider an example as shown in Fig. 9.

```xml
<?xml version="1.0" ?>
<product>
    <category>fruit</category>
    <item>Bananas</item>
    <code>Banafriuq6Kq</code>
    <price>0.55</price>
    <inventory>220</inventory>
    <vendor>Simply Fresh</vendor>
</product>
```

Fig. 9. The snippet of product.xml

Let us suppose that there are three linguistic variables defined for the amount of goods in inventory in the Fuzzy Meta-Knowledge Base. In practice, it means that they are defined inside an XML document residing inside an XML native database. The three linguistic variables are: low, medium, and high as shown in inventory.xml (Fig. 10). Figure 11 shows these membership functions graphically.

```xml
<?xml version="1.0" ?>
<inventory>
    <labels>
        <low type=fuzzyShoulder opt=desc A=100, B=150/>
        <medium type=trapezoidalFuzzyNumber A=100 B=150, C=250, D=300/>
        <high type=fuzzyShoulder opt=inc A=250, B=300/>
    </labels>
</inventory>
```

Fig. 10. The File inventory.xml

Let us consider an example

```xml
<inventory>
    <?xml version="1.0"?>
    <labels>
        <low type=fuzzyShoulder opt=desc A=100, B=150/>
        <medium type=trapezoidalFuzzyNumber A=100 B=150, C=250, D=300/>
        <high type=fuzzyShoulder opt=inc A=250, B=300/>
    </labels>
</inventory>
```

Fig. 10. The File inventory.xml

Fig. 11. The membership functions of the three linguistic terms related to inventory

If a manager wants to know the products that have a low amount with threshold = 0.5 then he could set the fuzzy XQuery query given in Fig. 12.

```xml
for $product$ in /grocery/product
where $product/inventory$ = low [threshold 0.5]
order by $product/inventory$
return <result>$product/item</result>
```

Fig. 12. The fuzzy XQuery

The server would calculate the result set and select the products that have the amount between 0 and 125 units.

5. CONCLUSION

We aim to extend classical XQuery implemented by a number of native XML databases with fuzzy logic. In this paper, we give a brief overview of attempts already made in this field. We present our approach and plan of what and how should be done in the future. The main idea is to translate a fuzzy XQuery query to the standard XQuery query and execute it against an existing database. After that, the results will be calculated by using the FMKB.

6. REFERENCES


