FUZZY XML MODEL FOR REPRESENTING FUZZY RELATIONAL DATABASES IN FUZZY XML FORMAT

Alnaar Jiwani[‡], Yasin Alimohamed[‡], Krista Spence[‡], Tansel Özyer[‡], Reda Alhajj[‡,⊗]
[‡]*Department of Computer Science, University of Calgary, Calgary, Alberta, Canada*

[⊗]*Department of Computer Science, Global University, Beirut, Lebanon*

Keywords: XML schema, fuzzy Data, fuzzy relational database, database reengineering.

Abstract: This paper describes a fuzzy XML schema model for representing a fuzzy relational database in XML format. It outlines a simple translation algorithm to include fuzzy relations and similarity matrices with their associated conventional relation. We also describe an example implementation of a fuzzy relational database and the XML document resulting from the translation according to our schema.

1 INTRODUCTION

In the past two decades, there has been extensive research examining how imprecise and uncertain data can be represented in databases given that it is pervasive in most real-world applications. Examples of imprecise data include subjective opinions and judgments in areas such as personnel evaluation, policy preferences and economic forecasting. A particular vein of research that is immediately applicable to many applications is how the conventional relational model can be extended to incorporate this fuzzy data.

Another highly researched area focuses on how relational data can be represented in the Extensible Markup Language (XML). Lee *et al* (2002) and Turowski and Weng (2002) describe examples of XML representation for fuzzy data modeling. However, they do not describe how to incorporate fuzzy XML with data from conventional relations. The approach described by Lee *et al* applies to the object-oriented paradigm, and not simple relational data. Turowski's approach is more general; however, it does not utilize the currently accepted technique of XML schemas in defining the XML document class (it instead uses DTD format).

This paper presents a novel approach to incorporate fuzziness in the XML model and outlines one approach for transforming fuzzy relational data into fuzzy XML.

The rest of this paper is organized as follows. Section 2 reviews the existing literature in the area of fuzzy sets and fuzzy relational databases. Section 3 presents an overview of the representation of fuzzy data in XML. In section 4, we describe a basic algorithm for translation from fuzzy relational database to fuzzy XML. Section 5 is conclusions.

2 FUZZY RELATIONAL DATABASES

Zadeh (1965) formally introduced the fuzzy set concept to represent imprecise data. He identified that objects encountered in physical world do not have precisely defined criteria of membership; this imprecision is natural to human thinking. For example, a teacher may want to find a list of all 'good' students. The teacher may define the term 'good' based upon attendance, grades, behavior or many other criteria. The teacher may also want to include students who do not *exactly fit* any crisp definition of being 'good'. Conventional relational databases rely on crisp representations of data and do not have immediately obvious ways to incorporate such imprecision.

2.1 Overview of Fuzzy Set Theory

A classical set is a set with a crisp boundary, such that any object in the domain either belongs to the set, or does not belong to the set. A classical set C may be: $C = \{x \mid x \leq 25, \text{ for all real } x\}$. In contrast, a fuzzy set has a continuum of grades of membership. There is a gradual transition from "belonging to a

Jiwani A., Alimohamed Y., Spence K., Özyer T. and Alhajj R. (2006).
FUZZY XML MODEL FOR REPRESENTING FUZZY RELATIONAL DATABASES IN FUZZY XML FORMAT.
In *Proceedings of the Eighth International Conference on Enterprise Information Systems - DISI*, pages 163-168
DOI: 10.5220/0002465101630168

set" to "not belonging to a set". For example, a fuzzy set could be used to model linguistic expressions such as "the person is beautiful" or "the weather is bad". Fuzziness does not come from the randomness of members of the set, but from the uncertain and imprecise nature of abstract concepts.

The construction of a fuzzy set F relies first on identifying a domain of values that could belong to the set, generally referred to as the *Universe of Discourse* (Jang and Sun, 1995). The set is characterized by a membership function, denoted $\mu_F(x)$ that associates each value in the Universe of Discourse to a real number in interval [0, 1]. The specification of the membership function is completely subjective to the person who defines it.

A fuzzy set F with universe of discourse X is defined as ordered pairs: $F = \{(x, \mu_F(x)) \mid x \in X\}$

Fuzzy sets may be discrete or continuous. Let X be the set of possible grades a student may receive on a paper: $X = \{A, B, C, D, F\}$.

The discrete fuzzy set "High grades" (H) could be represented as: H = {A/1.0, B/0.7, C/ 0.2, D/ 0, F/ 0}

Table 1: An instance of a Student relation.

FName	LName	Avg_Marks	Attitude
Jeremy	Scott	A	Unhappy
Jenny	Wong	A	Negative
George	Yuzwak	C	Positive
Jose	Sanchez	B	Cheerful

Table 2: Similarity Relation for the Attitude attribute of the Student relation (Table 1).

	Unhappy	Negative	Positive	Cheerful
Unhappy	1	0.8	0.2	0
Negative	0.8	1	0	0
Positive	0.2	0	1	0.95
Cheerful	0	0	0.95	1

Alternately, let X be the set of possible ages for a human being. Then the continuous fuzzy set "about 50 years old" (G) could be represented as: $G = \{(x, \mu_G(x)) \mid x \in X\}$, where, $\mu_G(x) = 1/(1+((x-50)/5)^4)$

2.2 Fuzzy Data in Relational Databases

Existing literature discusses many different techniques for representing fuzziness within relational databases. In general, it seems that the following ideas are agreed upon: a fuzzy relational database (FRDB) either allows for queries that let preferences be expressed instead of exact Boolean conditions, or allows for the storage and querying of a new type of data that directly stores fuzzy sets. In other terms, a FRDB can accommodate two types of imprecision – impreciseness in the association among data values or impreciseness in the data values themselves (Medina et al, 1994). The two most common techniques used for handling imprecision are similarity relations and possibility distributions; or a combination of both.

2.2.1 Similarity-based Techniques

Buckles and Petry (1995) were the first to introduce the similarity-based relational model. The basis of this model is the replacement of equality with a similarity relation. A similarity relation s(x,y) is a mapping of every pair of elements within the domain of an attribute to the interval [0,1]. This is best visualized in the form of a matrix. An example of this, based on the Attitude attribute of the Student relation described in Table 1, is given in Table 2. The matrix illustrates that the similarity relation is reflexive and symmetric. In this model of FRDB, a similarity relation is defined over the elements in each attribute, in each relation. Where a crisp definition of equality is still desired, the matrix representation of the similarity relation is reduced to the identity matrix.

Another feature of the similarity-based FRDB, is that it allows for non–atomic domain values. In their model, Buckles and Petry (1995) define that any member of the power set of the domain may be a domain value except the null set. This feature allows uncertainty of data values to be expressed, but is not in first normal form and suffers the associated implementation problems. Similarity relations are best for finite and discrete domains of linguistic sets.

2.2.2 Possibility-based Techniques

Instead of understanding a membership function $\mu_F(x)$ as the grade of membership of x in F, possibility-based FRDBs interpret it as a measure of the possibility that a variable Y has a value x. Such fuzzy sets are referred to as possibility distributions and are represented by the symbol $\prod$. In a possibility-based FRDB, these possibility distributions can be used to indicate the possibility that a tuple has a particular value for an attribute. For example, if a tuple in a Person table has the value 'Young' for the attribute Age, a possibility distribution describes the likelihood that such person has a particular value for the age (Buckles and Petry, 1995): $\prod_{young} = \{1.0/22, 1.0/23, 0.8/24, 0.6/25, ...\}$

So the likelihood that the Young person is 24 years old is 0.8. This allows the linguistic identifier to be used as a value in the domain while the actual possibility distribution is given elsewhere in the database in the form of a relation having the name of the linguistic identifier.

Raju *et al* (1988), describe two different ways of implementing a possibility-based FRDB. Each represents a fuzzy relation r by a table with additional column for $\mu_r(t)$, showing the membership of tuple t in r. The first (Type-1) stipulates that the domain of each attribute is a fuzzy set (recall that a classical set is a special case of a fuzzy set). Given crisp values in a relation, there exist membership functions that map the values to linguistic terms with associated possibilities. The second implementation (Type-2) permits more uncertainty in the data values. It allows for ranges or possibility distributions to be the actual values of attributes. This cannot be implemented given current commercial frameworks for relational databases since it allows for different data types in the same column and/or multiple values.

2.2.3 Hybrid Techniques

Other techniques for representing fuzziness in relational databases have been proposed that include characteristics of both the similarity-based and possibility-based models. This allows them to work with more than one area of imprecision.

An example is GEFRED – a Generalized Model of Fuzzy Relational Databases. The model allows for linguistic terms in a column to be related via a 'proximity relation', which is identical to the similarity relation described by Buckles and Petry (1995). If no proximity relation exists for the attribute, it is assumed that the classical definition of equality applies for values in this domain.

3 XML SCHEMAS AND FUZZY DATA

XML is an excellent method of transmitting data between software applications. In order for an application to interpret the XML, certain constraints must be placed on an XML document. This can be accomplished by describing classes of XML documents through XML schemas or Document Type Declarations (DTDs). The application can then use the specified XML schema or DTD to parse the information contained in a specific XML document.

3.1 XML Schemas

An XML schema defines the structure of an XML document instance. Unlike DTDs, XML schemas allow for strong data typing, modularization, and reuse. The XML schema specification allows a developer to define new data types (using the <complexType> tag), and also use built-in data types provided by the specification. The developer can also define the structure of an XML document instance and constrain its contents. As well, the XML schema language supports inheritance, so that developers do not have to start from scratch when defining a new schema.

3.2 Fuzzy Relational Data in XML

There has already been some research completed on representing fuzzy data in XML.

The fuzzy object-oriented modeling technique (FOOM) schema proposed by Lee *et al* (2002) is one such approach. This method builds upon object-oriented modeling (OOM) to also capture requirements that are imprecise in nature and therefore 'fuzzy'. The FOOM schema defines a class of XML document that can describe fuzzy sets, fuzzy attributes, fuzzy rules, and fuzzy associations. This method would be useful in representing data contained in object-oriented databases. However, it is too specific in terms of its object-oriented nature to be applied directly to relational databases.

Another more general approach is proposed by Turowski and Weng (2002). The method described is aimed at creating a common interchange format for fuzzy information using XML to reduce integration problems with collaborating fuzzy applications. XML tags with a standardized meaning are used to encapsulate fuzzy information. A formal syntax for important fuzzy data types is also introduced.

This technique of using XML to represent fuzzy information is general enough to be built upon to apply to relational databases. However, it uses DTDs, rather than the currently accepted method of XML schemas to define and constrain the information held in an XML document. It would be beneficial to extend this approach to define the XML document class for holding data from fuzzy relational databases with an XML schema, rather than a DTD.

4 FROM FUZZY RDB TO FUZZY XML

In this section, we describe in detail our implementation of a fuzzy relational database, XML schema structure and the algorithm to convert database content to XML document conforming to the schema.

Table 3: Example 'Student' relation.

ID	FNAME	LNAME	ATTEND	AVG	ATTITUDE	ADVISOR
1	Jeremy	Scott	0.56	3.60	Unhappy	1
2	Jenny	Wong	0.98	3.87	Motivated	5
3	George	Yuzwak	0.80	2.74	Lazy	3
4	Jose	Sanchez	0.9	3.20	Cheerful	1
5	Eliza	Reichs	0.35	1.87	Lazy	1

4.1 Database Structure

We chose to create a hybrid-type fuzzy relational database that incorporates both similarity relations, to represent fuzzy equality, and possibility relations, which could be used to translate crisp data based on a number of linguistic terms or to represent a possibility distribution. Any attribute in a relation may have an associated fuzzy relation and/or an associated similarity relation. Each of these can be joined into a query to retrieve information based on imprecise conditions. The results of these queries themselves can then be considered a sort of fuzzy relation that has all the attributes requested by the query as well as an attribute that describes tuple's membership in the relation.

An example relation, 'Student', is illustrated in Table 3 (an extension of the relation in Table 1). We suppose that the information stored in ID, FNAME, LNAME and ADVISOR columns is crisp. The data in ATTEND and AVG is also crisp, but fuzzy relations based on linguistic terms are defined on each. The values within the domain of ATTITUDE have an associated similarity relation defined to provide fuzzy equivalence.

4.1.1 Similarity Relations

In our fuzzy relational database model, we allow any column to have an associated similarity relation, which assigns all elements in the domain a degree of similarity to all other elements in the domain. Normally, this is visually represented in a matrix, but to construct this matrix in a relation by naming attributes after each domain element is very inflexible and difficult to modify if one wanted to add another element to the domain. Instead, we flatten the matrix.

Every similarity matrix is named under the convention: 'SM_TABLENAME_COLNAME', where TABLENAME and COLNAME are the relation and attribute the similarity matrix applies to, respectively. Within the similarity relation there are three attributes: VALUE1, VALUE2, and MATCH. VALUE1 and VALUE2 hold the combination of domain values and will be assigned a type according to the type of the attribute being compared. MATCH contains the result of the similarity relation s(x,y) for the pair described in VALUE1 and VALUE2, and so will contain a value in the interval [0,1].

4.1.2 Fuzzy Relations

Our model also allows any attribute to have an associated fuzzy relation, which can contain the data for a number of fuzzy sets defined over the domain of the attribute. Each fuzzy set is identified by a linguistic term. If the set is discrete, the fuzzy relation will contain each ordered pair within the set. However, if the set is defined by a continuous function, the fuzzy relation will contain points along the graph of the function that can be interpolated to find the exact value of the membership function.

Fuzzy relations are named under the convention: 'FR_TABLENAME_COLNAME', where TABLENAME and COLNAME are the relation and attribute the fuzzy relation applies to, respectively. Within the fuzzy relation there are three attributes: LINGUISTIC_TERM, COLUMN_VALUE, and MEMBERSHIP. LINGUISTIC_TERM is the word that describes the meaning of the fuzzy set. COLUMN_VALUE and MEMBERSHIP can be interpreted as the (x,y) values of a point on the graph of the fuzzy set. COLUMN_VALUE is the same type as the attribute this relation applies to, and MEMBERSHIP is the result of the membership function that maps the COLUMN_VALUE to the unit interval.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  - <xs:element name="Database">
    - <xs:complexType>
      - <xs:sequence>
        - <xs:element name="Table" maxOccurs="unbounded">
          - <xs:complexType>
            + <xs:sequence>
              <xs:attribute name="name" type="xs:string" use="required" />
            </xs:complexType>
          </xs:element>
        </xs:sequence>
        <xs:attribute name="name" type="xs:string" use="required" />
      </xs:complexType>
    </xs:element>
  </xs:schema>
```

Figure 1: XML Schema – Top View.

4.2 XML Schema Structure

The schema structure we developed for representing our fuzzy relational database in XML provides a direct relationship between the database and the resulting XML document. This implementation allows for an XML representation of the data that is simple to interpret and query.

The schema defines the outermost element of the XML document as the Database element. A Database element contains a name attribute used to

indicate the name of the fuzzy relational database and a sequence of Table elements representing each relation in the database. A Table element also has a name attribute that will be set to the name of the table. This outer structure of the XML schema is represented in Figure 1.

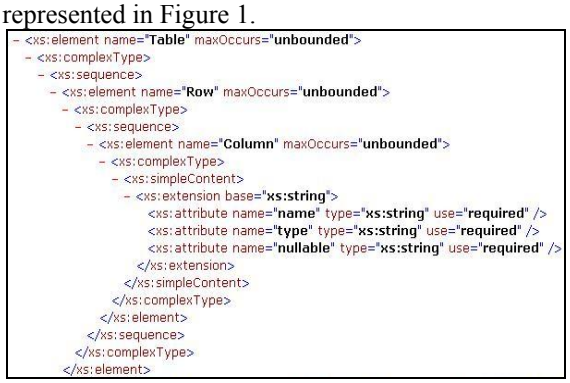

Figure 2: Definition of the Row Element for Storing Table Records.

Figure 3: Definition of the SimilarityMatrix Element.

The schema further defines a Table element from the database content related to each relation. This includes the relation's row and column data (database records) and any fuzzy relations or similarity relations associated with its attributes.

Figure 2 illustrates how record information is stored in the XML document. The schema defines a Table element as a complex type composed of Row elements. A Row element is composed of Column elements. Each Row element in the XML document holds a record, whose column values are stored as the value for each Column element. Column elements are also described by the name, type, and nullable attributes.

The database structure stores similarity and fuzzy relations as separate tables. To keep the XML document simple, our XML schema stores an attribute's fuzzy data along with the table that contains the attribute. The schema defines a Table element as a complex type with Row elements for each record (Figure 2), SimilarityMatrix elements for similarity relations (Figure 3), and FuzzyRelation elements for fuzzy relations (Figure 4).

Figure 4: Definition of the FuzzyRelation Element.

The flat conversion algorithm we chose to convert the fuzzy relational database to an XML document following our schema structure can be outlined as follows:

Add *Database* start tag to the XML document with the *name* attribute set to the name of the database.
Set the xsi:*schemaLocation* to point to the location of the XML Schema this document is to adhere to.
Retrieve all table names from the database
For each table:
Add *Table* start tag to the XML document with the *name* attribute set to the table name
Query the database to get the column data (name, type, nullable value) for the current table and then query for the row data using the column names
For each row:
Add *Row* start tag to the XML document
For each column:
Add *Column* start tag to the XML document and set the *name*, *type*, and *nullable* attributes to their corresponding values
Set the value of the *Column* element to the value retrieved for the current row/column
Add *Column* end tag to the XML document
Add *Row* end tag to the XML document
Query the database to get all similarity matrix data for the current table. A similarity matrix belonging

to a table is identified by appending 'SM' to the table name, followed by the name of the matrix.
For each similarity matrix:
Add *SimilarityMatrix* start tag to the XML document and set the *assocColumn* and *type* attributes to their corresponding values
For each cross reference:
Add *crossRef* start tag to the XML document
Add *Value1*, *Value2*, and *Match* elements and set their corresponding values
Add *crossRef* end tag to the XML document
Add *SimilarityMatrix* end tag to the XML document
Query the database to get all fuzzy relations for the current table. A fuzzy relation belonging to a table is identified by appending 'FR' to the name of the table, followed by the name of the relation
For each fuzzy relation:
Add *FuzzyRelation* start tag to the XML document
For each linguistic term:
Add *LinguisticTerm* start tag to the XML document and set the *term* attribute to its corresponding value
Add *FuzzySet* start tag to the XML document
For each point in the fuzzy set:
Add the *Point* start tag to the XML document
Add *x_value* and *membership* elements and set their corresponding values
Add *Point* end tag to the XML document
Add *FuzzySet* end tag to the XML document
Add *LinguisticTerm* end tag to the XML document
Add *FuzzyRelation* end tag to the XML document
Add the *Table* end tag to the XML document
Add the *Database* end tag to the XML document

5 CONCLUSIONS

We have described a fuzzy XML schema to represent an implementation of a fuzzy relational database that allows for similarity relations and fuzzy sets. We have also provided a flat translation algorithm to translate from the fuzzy database implementation to a fuzzy XML document that conforms to the suggested fuzzy XML schema.

REFERENCES

Anvari, M., Rose G. F., 1987. "Fuzzy Relational Databases," *Analysis of Fuzzy Information*, Bezdek ed., Vol II, CRC Press.

Bosc P., Galibourg M. and Hamon G., 1988. "Fuzzy querying with SQL: extensions and implementation aspects," *Fuzzy Sets and Systems*, Vol. 28, pp.333-349.

Buckles B. P. and Petry F. E., 1995. "Fuzzy Databases in the New Era," *Proc. of ACM SAC*, pp.497-502.

Dey D. and Sumit S., 1996. "A Probabilistic Relational Model and Algebra," *ACM TODS,* Vol.21, pp.339-369.

Duta A., Barker K. and Alhajj R., 2004. "Converting Relationships to XML Nested Structures," *Journal of Information and Organizational Sciences*, Vol.28, No.1-2.

Wang C., Lo A. Alhajj R. and Barker K., 2005. "Novel Approach for Reengineering Relational Databases into XML," *Proc. of XSDM*, (in conjunction with ICDE), Tokyo.

Fernandez M., Tan W.-C., and Suciu D., 2000. "SilkRoute: Trading between Relations and XML". *Proc. of WWW*, Amsterdam.

Fong J., Pang F. and Bloor C., 2001. "Converting Relational Database into XML Document," *Proc. of the International Workshop on Electronic Business Hubs*, pp.61-65.

Jang J. and Sun C., 1995. "Neuro-Fuzzy Modeling and Control," *Proc. of the IEEE*, 83, pp.378-406.

Yang K.Y., Lo A., Özyer T. and Alhajj R., 2005. "DWG2XML: Generating XML Nested Tree Structure from Directed Weighted Graph," *Proc. of ICEIS*.

Lee, D., et al, 2002. "NeT & CoT: translating relational schemas to XML schemas using semantic constraints," *Proc. of ACM CIKM*.

Lee, J., et al, 2002. "Modeling Imprecise Requirements with XML," *Fuzzy Systems*, 2, pp.861-866.

Medina J. M., Pons O. and Vila M. A., 1994. "GEFRED: A Generalized Model of Fuzzy Relational Databases Version 1.1," *Information Sciences*.

Raju. K. V. and Majumdar, A. K., 1988. "Fuzzy Functional Dependencies and Lossless Join Decomposition of Fuzzy Relational Database Systems," *ACM TODS*, Vol.13, pp.129-166.

Thompson H. S., et al, 2004. "XML Schema Part 1: Structures," W3C Recommendation.

Turowski K. and Weng U., 2002. "Representing and processing fuzzy information – an XML-based approach," *Knowledge-Based Systems*, Vol.15, pp.67-.

Wang S., et al, 2001. "Incremental Discovery of Functional Dependencies from Similarity-bases Fuzzy Relational Databases Using Partitions," *Proc. of the National Conference on Fuzzy Theory and Its Applications*, pp.629-636.

Zadeh L., 1965. "Fuzzy Sets," *Information and Control*, 8, pp.338-353.

Zvieli A. and Chen P. P., 1986"Entity-relationship modeling and fuzzy databases," *Proc. of IEEE ICDE*, Los Angeles, pp.320-327.

Zemankova M. and Kandel A.,1984. "Fuzzy Relational Data Bases – A key to Expert Systems," *Verlag TUV Rheinland*.