Fuzzy Ordering Implementation Applied in Fuzzy XQuery

Supaporn Kansomkeat*, Sukgamon Sukpisit*, Apirada Thadadech*,
Pannipa Sae Ueng** and Srdjan Skrbic**
* Prince of Songkla University, Department of Computer Science, Songkhla, Thailand
** University of Novi Sad, Department of Mathematics and Informatics, Novi Sad, Serbia
supaporn.k@psu.ac.th, sukgamon.s@psu.ac.th, apirada.t@psu.ac.th,
pannipa@dmi.uns.ac.rs, srdjan.skrbic@dmi.uns.ac.rs

***Abstract* — Fuzzy XQuery is the extension of standard XQuery language that allows fuzzy values in the query condition statements. Relational operators are not only required and possible in crisp value cases but also for fuzzy values. When relational operators are included in the query, it is necessary to provide means for comparison between fuzzy sets. These fuzzy relational operators are typically used in two fuzzy sets comparison case, but can also be used with some aggregate functions like MIN, MAX, and SUM. The aim of this paper is to present the algorithms for the implementation of fuzzy relational operators. Our algorithms compare the horizontal positions of two fuzzy sets and calculate the ordering value based on partial fuzzy ordering proposed by Bodenhofer. Moreover, we developed a GUI application and evaluated our approach with 360 fuzzy ordering cases. The experimental results show that our algorithms are capable of calculating fuzzy ordering values with various types of fuzzy values correctly.**

I. INTRODUCTION

Recently, fuzzy extensions are proposed to handle vague, ambiguous, uncertain, imprecise or incomplete information. Campi et al. [1] introduced fuzzy extensions to XPath named FuzzyXPath that used to query XML data based on the fuzzy set theory. Fredrick and Radhamani [2] introduced fuzzy XQuery to retrieve data from native XML database. Skrbic et al. [3] introduced PFSQL (Prioritized Fuzzy Structured Query Language), which is an extension of SQL (Structured Query Language). PFSQL uses the prioritized fuzzy logic to retrieve data from a fuzzy relational database. In 2012, Ueng and Skrbic [4] proposed fuzzy extensions to standard XQuery. Their query system retrieves data from native XML database based on prioritized fuzzy logic. In 2014, they implemented an interpreter for fuzzy XQuery in their project called FXI (Fuzzy XQuery Interpreter). Users can query data with priority and threshold keywords in the condition statement and define fuzzy values used as search conditions in the query.

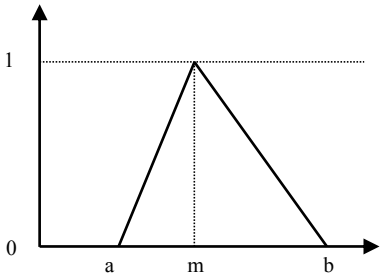
Including fuzzy relational operators in FXI is a very promising idea. In this way, fuzzy XQuery queries would be able to provide flexible comparisons between fuzzy sets that represent vague data. Relational operators on fuzzy sets are binary operators, which are able to compare two fuzzy sets: $<, \leq, \geq$ and $>$. Furthermore, fuzzy relational operators can be used with some aggregate functions like MIN, MAX, and SUM. In this paper, we propose a method to calculate fuzzy relational operations between two fuzzy sets and give its implementation. The proposed method is general and may be used with different types of problems. For example, it can be applied to fuzzy XQuery or PFSQL.

This paper is organized as follows. In the next section, we introduce algorithms for fuzzy relational operator calculations. Our implementation and testing results are presented in Sections 3 and 4, respectively. Section 5 is the conclusion.

II. FUZZY ORDERING CALCULATIONS

A. Membership functions

There are five different types of fuzzy membership functions used in [4]: triangle fuzzy number, trapezoidal fuzzy number, interval, fuzzy shoulder and crisp value. Figure 1 shows the shape of a fuzzy triangle membership function.



$$\mu_A(x) = \begin{cases} 0, & \forall x, x \leq a \wedge x \geq b \\ \dfrac{x-a}{m-a}, & \forall x \in (a, m] \\ \dfrac{b-x}{b-m}, & \forall x \in (m, b) \end{cases}$$

Figure 1 Fuzzy triangle number and its membership function

Definition 1 A fuzzy set A over universe X is determined by its characteristic (membership) function [5],

$$\mu_A: x \to [0,1],$$

where, for every $x \in X, \mu_A(x)$ is interpreted as membership degree of element x to fuzzy set A. Value $\mu_A(x) = 0$ denotes that element x does not belong to the set A, while $\mu_A(x) = 1$ denotes that element x belongs to the set A. Universe X is almost always the set of real numbers.

Definition 2 The set $x \in X \mid \mu_A > 0$ is called the support of A (*supp(A)*) and the set $\{x \in X \mid \mu_A = 1\}$ is called its kernel (*ker(A)*) [5]

B. Fuzzy ordering calculation

In 2008, fuzzy orderings were proposed by Bodenhofer in [6]. Here we recall some basic definitions used in our research; for a more extensive description see [6].

Definition 3 Consider a fuzzy equivalence relation, *T*-equivalence $E: X^2 \to [0,1]$ and a direct fuzzification, *T-E*-ordering $L: X^2 \to [0,1]$. Then, for given fuzzy set $A \in \mathcal{F}(X)$, where $\mathcal{F}(X)$ is a fuzzy superset of *X*. The fuzzy sets 'at least *A*' and 'at most *A*' (with respect to *L*), abbreviated ATL(*A*) and ATM(*A*), respectively, are defined as follow (for all $x \in X$):

$$ATL(A)(x) = \{T(A(y), L(y,x)) \mid y \in X\} \quad (1)$$

$$ATM(A)(x) = \{T(A(y), L(x,y)) \mid y \in X\} \quad (2)$$

ATL(*A*) is the smallest fuzzy superset of *A* that has a non-decreasing membership function with respect to *L*, while ATM(*A*) is the smallest fuzzy superset of *A* that has a non-increasing membership function with respect to *L*.

When *L* is a crisp ordering, the notations LTR(*A*) and RTL(*A*) are used instead of ATL(*A*) and ATM(*A*), respectively. LTR(*A*) stands for left-to-right closure and RTL(*A*) stands for right-to-left closure. The operator $\preccurlyeq$ is referred to crisp ordering.

$$LTR(A)(x) = \{A(y) \mid y \in X \wedge y \preccurlyeq x\} \quad (3)$$

$$RTL(A)(x) = \{A(y) \mid y \in X \wedge x \preccurlyeq y\} \quad (4)$$

First we describe a well-known ordering procedure for real intervals.

$$[a,b] \leq_I [c,d] \Leftrightarrow a \leq c \wedge b \leq \mathrm{d} \quad (5)$$

Equation (5) states that the only case that yields "true" or 1 value is $a \leq c$ and $b \leq d$. The inequality $a \leq c$ means that there are no elements of set [c, d] that are below the entire interval [a, b] and the inequality $b \leq d$ means that there are no elements of [a, b] that are completely above [c, d]. Equation (5) can be generalized to arbitrary crisp subsets of an ordered set $(x, \preccurlyeq)$ as follow:

$$M \preccurlyeq_I N \Leftrightarrow \big((\forall x \in N)(\exists y \in M) y \preccurlyeq x\big) \wedge \big((\forall x \in M)(\exists y \in N) x \preccurlyeq y\big) \quad (6)$$

By using the operators LTR and RTL, and considering a crisp ordering $\preccurlyeq$ on *X*, the following equivalences that hold for all $M, N \subseteq X$ are proved.

$$LTR(M) \supseteq LTR(N) \Leftrightarrow (\forall x \in N)(\exists y \in N)\, y \preccurlyeq x \quad (7)$$

$$RLT(M) \subseteq RTL(N) \Leftrightarrow (\forall x \in M)(\exists y \in N)\, x \preccurlyeq y \quad (8)$$

Since the operators LTR and RTL can be applied for fuzzy sets, an ordering of fuzzy sets $A, B \in \mathcal{F}(X)$ with respect to crisp ordering $\preccurlyeq$ is generalized as:

$$A \preccurlyeq_I B \Leftrightarrow (LTR(A) \supseteq LTR(B) \wedge RTL(A) \subseteq RTL(B)) \quad (9)$$

The inclusion $LTR(A) \supseteq LTR(B)$ means that the left flank of *A* is to the left of the left flank of *B* while $RTL(A) \subseteq RTL(B)$ means that the right flank of *A* is to the left of the right flank of *B*.

Considering fuzzy orderings above, the fuzzy ordering calculation can be determined by considering horizontal positions of comparing fuzzy sets. If the assertion (9) is fulfilled in both conditions, the fuzzy ordering value is *true* or 1. Otherwise, the operation returns *false* or 0. Figure 2 shows the comparison of fuzzy sets that yields value 1.

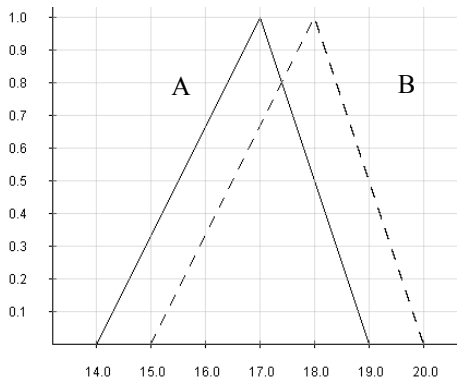


Figure 2. Comparison of fuzzy sets that satisfy (2)

From assertion (9) can be concluded that if only one condition is satisfied, it means that fuzzy sets cannot be compared - incomparable case. In this case, the fuzzy ordering operation will return *incomparable* or 0.5. Figure 3 shows the incomparable fuzzy sets.

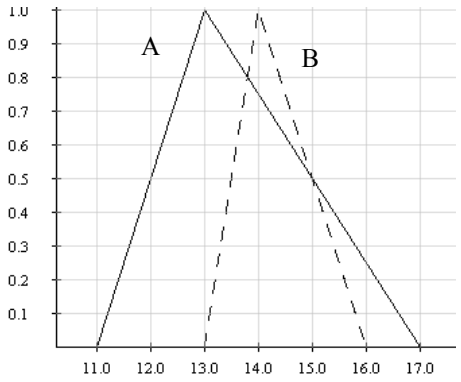


Figure 3. Incomparable fuzzy sets

Another incomparable case is the comparison of fuzzy sets having different heights. However, Skrbic and Rackovic proposed an idea to eliminate this problem in [5]. Fuzzy set A' is introduced as:

$$\mu_{A'} = \begin{cases} 1, \mu_A(x) = h(A) \\ \mu_A(x), otherwise \end{cases} \quad (10)$$

In this way, fuzzy relational operator $\leq_F$ is introduced by:

$$A \leq_F B \Leftrightarrow A' \leq'_F B' \quad (11)$$

Definition 4 Let A and B be two fuzzy sets over universe X. Order $\leq'_F$ over the set of all fuzzy sets over universe X, $\mathcal{F}(X)$ is defined by:

$$A \leq'_F B \Leftrightarrow (LTR(B) \subseteq LTR(A) \wedge RTL(A) \subseteq RTL(B)) \quad (12)$$

In the same way as with operators $<$ and $>$ on crisp domain, other relational operators, like $<_F$ and $>_F$ can be derived using the $\leq_F$ order.

C. Algorithm

As mentioned before, we consider five types of fuzzy set. Each type has different attributes that depict its properties. For example, a triangle fuzzy number contains three attributes (*LeftOffset*, *Maximum* and *RightOffset*). The *LeftOffset* refers to the beginning location of the support (*supp* in Definition 2) of fuzzy set (*LeftOffset*, 0). The *Maximum* refers to a location of its kernel (*Maximum*, 1) and the *RightOffset* refers to the end location of the support of fuzzy set (*RightOffset*, 0). Table I shows attributes for each type of characteristic function.

Attributes of fuzzy sets are used to calculate fuzzy relational operator values. Comparing two fuzzy sets, A and B, focuses on beginning, maximum and ending locations of A and B. For example, in Figure 1, two triangle fuzzy sets, A and B, are compared by operator $<$, the algorithm starts from comparing the *Maximum* attributes. If $Maximum_A$ is greater than $Maximum_B$, the result is 0 and the process ends. If not, the *LeftOffset* attributes will be compared. If $LeftOffset_A$ is not greater than $LeftOffset_B$, the process is still going onto compare *RightOffset*. If $RightOffset_A$ is greater than $RightOffset_B$, the result value is 0.5 (incomparable). If not, the result value is 1 (true). If $LeftOffset_A$ is greater than $LeftOffset_B$, $RightOffset_A$ and $RightOffset_B$ are compared. If $RightOffset_A$ is greater than $RightOffset_B$ then the result value is 0 (false), otherwise, the result value is 0.5 (incomparable). The algorithm for comparing two triangle fuzzy sets is shown in Listing 1.

TABLE I.
ATTRIBUTES OF EACH CHARACTERISTIC FUNCTION

Characteristic function	Attributes (A, μ_A)	Abbreviation
Triangle fuzzy number	LeftOffset $(A, 0)$	T-LO
	Maximum $(A, 1)$	T-MX
	RightOffset $(A, 0)$	T-RO
Trapezoidal fuzzy number	LeftOffset $(A, 0)$	TR-LO
	LeftMaximum $(A, 1)$	TR-LMX
	RightMaximum $(A, 1)$	TR-RMX
	RightOffset $(A, 0)$	TR-RO
Right shoulder	ZeroPoint $(A, 0)$	S-ZP
	Maximum $(\infty, 1)$	S-MX
Left shoulder	Maximum $(0, 1)$	S-MX
	ZeroPoint $(A, 0)$	S-ZP
Interval	LeftMaximum $(A, 1)$	I-LMX
	RightMaximum $(A, 1)$	I-RMX
Crip value	X (A)	C-X
	Y (μ_A)	C-Y

D. Crisp value

Unlike other fuzzy sets, the crisp value is a paired-value (A, μ_A). A comparison between crisp value and other fuzzy sets needs a special method.

For a relational operation between crisp value and another fuzzy set, we compare the value of attribute X of crisp value and boundary values of the compared fuzzy set. If a value X is less than the lower bound of the compared fuzzy set, the fuzzy ordering value is 1. If a value X is inside the boundary, the result value is 0.5. Otherwise, the result value is 0.

Comparing between crisp values is done in the same manner. For ordering between crisp values, A and B, following applies, if value X_A is not greater than value X_B, the result is 1. Otherwise the result is 0.

Listing 1. Algorithm for calculating fuzzy ordering between a triangle fuzzy number and another triangle fuzzy number.

```
Algorithm IsLessThan (FuzzyTriangle A, FuzzyTriangle B)
01. Compare Maximum_A and Maximum_B
02. If Maximum_A greater than Maximum_B
03.     Result is 0
04. Else
05.     Compare LeftOffset_A and LeftOffset_B
06.     If LeftOffset_A not greater than LeftOffset_B
07.         Compare RightOffset_A and RightOffset_B
08.         If RightOffset_A greater than RightOffset_B
09.             Result is 0.5
10.         Else
11.             Result is 1
12.         End if
13.     Else
18.         Compare RightOffset_A and RightOffset_B
19.         If RightOffset_A greater than RightOffset_B
20.             Result is 0
21.         Else
22.             Result is 0.5
23.         End if
23.     End if
24. End if
```

III. IMPLEMENTATION

To support our ideas, we developed the application that has two functions: manual fuzzy ordering testing and random fuzzy ordering testing. The manual testing function is used for a single test. In this case, the user can specify types of fuzzy sets and their attributes. When the process is done, the application shows an image of specified fuzzy sets and their fuzzy ordering value. Figure 4 illustrates the user interface for the manual testing function. The random testing function randomly generates comparison cases. In this function, the user can indicate types of fuzzy sets, number of generated cases, and boundary values. Figure 5 shows the user interface of the random testing function.

This application was developed on Java platform with the use of PostgreSQL to store fuzzy set attributes and cases of the random testing function.

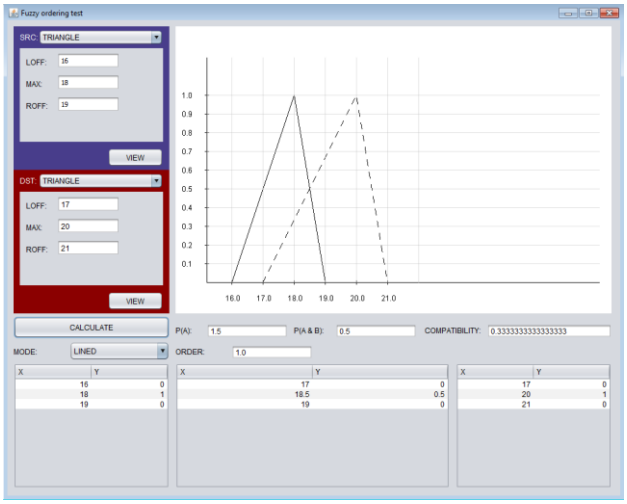


Figure 4. User interface of manual testing function

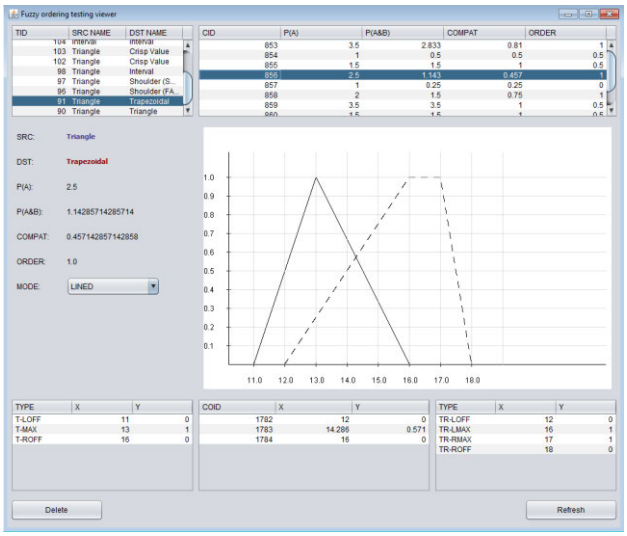
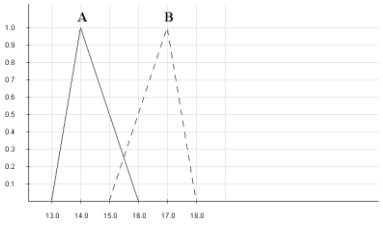
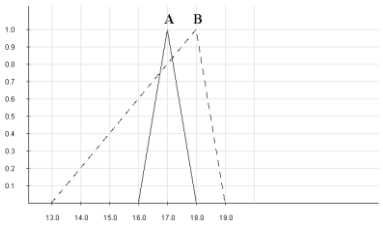
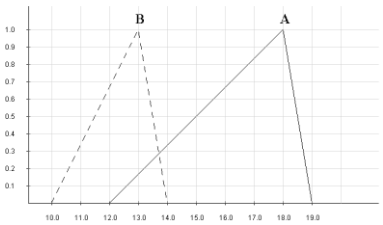
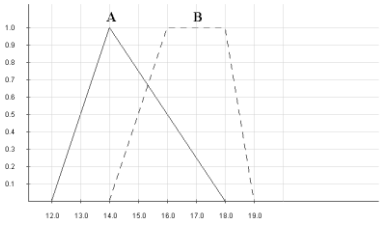


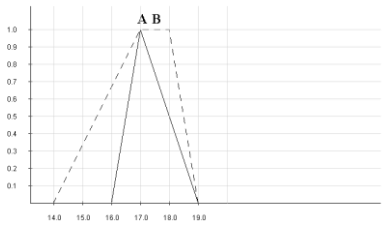
Figure 5. User interface of automated random testing function

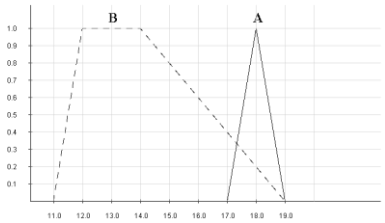
IV. TESTING RESULTS

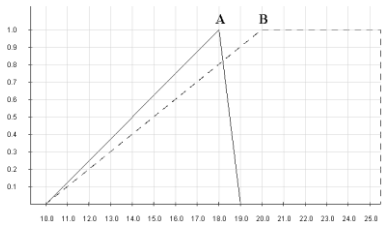
To prove the reliability of our proposed algorithms, some fuzzy ordering cases are generated randomly using random fuzzy ordering testing function of our application. As mentioned above, this paper considers five types of fuzzy sets. To cover all types of comparisons, each characteristic function is compared with the other four types including itself. Since there are two types of fuzzy shoulder, there are 36 comparison pairs. For a better variety in the comparison, the number of generated cases is set to be 10. Consequently, each pair has 10 cases of fuzzy ordering testing. Totally, there are 360 fuzzy ordering cases in our experimental results. The generated fuzzy sets are forced to position inside a boundary that is specified by the user. If the boundary is too wide, the fuzzy sets can be positioned too far from each other and have no incomparable cases. To avoid this problem, the lower bound and the upper bound are set to be 10 and 20, respectively. For the sake of brevity, some selected comparison cases between fuzzy triangle and other types are represented in Table II.

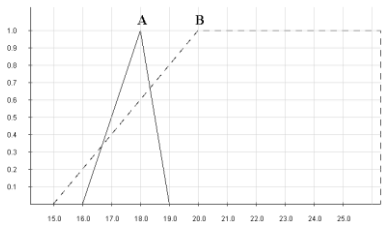
TABLE II.
A COMPARISONS BETWEEN FUZZY TRIANGLE AND OTHER TYPES

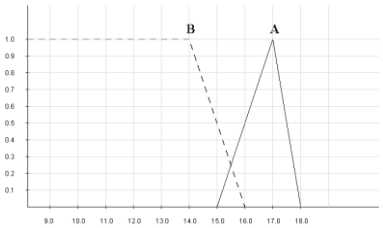
Case No.	Type of fuzzy set A		Type of fuzzy set B		Result
	Attributes	Value	Attributes	Value	
1	Fuzzy triangle		Fuzzy triangle		1
	$T\text{-}LO_A$	13	$T\text{-}RO_B$	15	
	$T\text{-}MX_A$	14	$T\text{-}MX_B$	17	
	$T\text{-}RO_A$	16	$T\text{-}RO_B$	18	
					
2	Fuzzy triangle		Fuzzy triangle		0.5
	$T\text{-}LO_A$	16	$T\text{-}RO_B$	13	
	$T\text{-}MX_A$	17	$T\text{-}MX_B$	18	
	$T\text{-}RO_A$	18	$T\text{-}RO_B$	19	
					
3	Fuzzy triangle		Fuzzy triangle		0
	$T\text{-}LO_A$	12	$T\text{-}RO_B$	10	
	$T\text{-}MX_A$	18	$T\text{-}MX_B$	13	
	$T\text{-}RO_A$	19	$T\text{-}RO_B$	14	
					
4	Fuzzy triangle		Fuzzy trapezoidal		1
	$T\text{-}LO_A$	12	$TR\text{-}LO_B$	14	
	$T\text{-}MX_A$	14	$TR\text{-}LMX_B$	16	
	$T\text{-}RO_A$	18	$TR\text{-}RMX_B$	18	
			$TR\text{-}RO_B$	19	
					

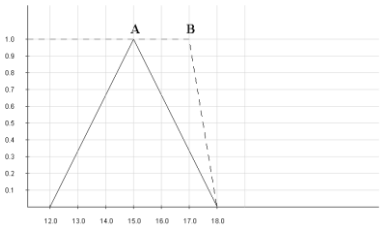
5	Fuzzy triangle		Fuzzy trapezoidal		0.5
	$T\text{-}LO_A$	16	$TR\text{-}LO_B$	14	
	$T\text{-}MX_A$	17	$TR\text{-}LMX_B$	17	
	$T\text{-}RO_A$	19	$TR\text{-}RMX_B$	18	
			$TR\text{-}RO_B$	19	
					

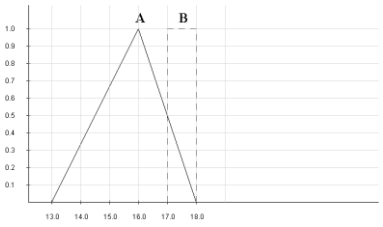
6	Fuzzy triangle		Fuzzy trapezoidal		0
	$T\text{-}LO_A$	17	$TR\text{-}LO_B$	11	
	$T\text{-}MX_A$	18	$TR\text{-}LMX_B$	12	
	$T\text{-}RO_A$	19	$TR\text{-}RMX_B$	14	
			$TR\text{-}RO_B$	19	
					

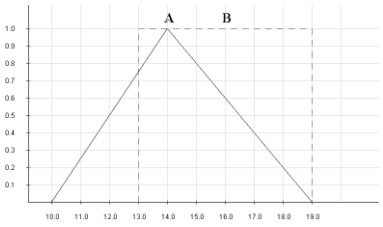
7	Fuzzy triangle		Right shoulder		1
	$T\text{-}LO_A$	10	$S\text{-}ZP_B$	10	
	$T\text{-}MX_A$	18	$S\text{-}MX_B$	20	
	$T\text{-}RO_A$	19			
					

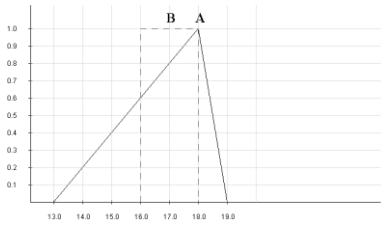
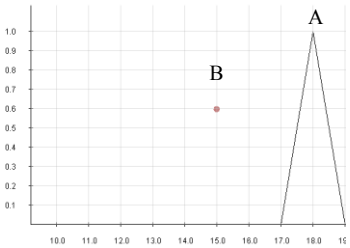
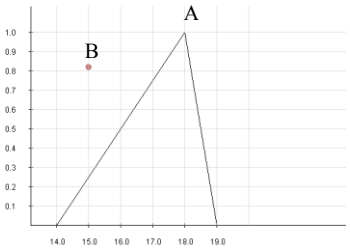
8	Fuzzy triangle		Right shoulder		0.5
	$T\text{-}LO_A$	16	$S\text{-}ZP_B$	15	
	$T\text{-}MX_A$	18	$S\text{-}MX_B$	20	
	$T\text{-}RO_A$	19			
					

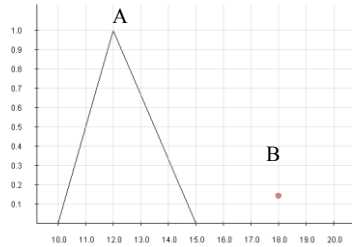
9	Fuzzy triangle		Left shoulder		0
	$T\text{-}LO_A$	15	$S\text{-}MX_B$	14	
	$T\text{-}MX_A$	17	$S\text{-}ZP_B$	16	
	$T\text{-}RO_A$	18			
					

10	Fuzzy triangle		Left shoulder		0.5
	$T\text{-}LO_A$	12	$S\text{-}MX_B$	17	
	$T\text{-}MX_A$	15	$S\text{-}ZP_B$	18	
	$T\text{-}RO_A$	18			
					

11	Fuzzy triangle		Interval		1
	$T\text{-}LO_A$	13	$I\text{-}LMX_B$	17	
	$T\text{-}MX_A$	16	$I\text{-}RMX_B$	18	
	$T\text{-}RO_A$	18			
					

12	Fuzzy triangle		Interval		0.5
	$T\text{-}LO_A$	10	$I\text{-}LMX_B$	13	
	$T\text{-}MX_A$	14	$I\text{-}RMX_B$	19	
	$T\text{-}RO_A$	19			
					

13	Fuzzy triangle		Interval		0
	$T\text{-}LO_A$	13	$I\text{-}LMX_B$	16	
	$T\text{-}MX_A$	18	$I\text{-}RMX_B$	18	
	$T\text{-}RO_A$	19			
					
14	Fuzzy triangle		Crisp value		0
	$T\text{-}LO_A$	17	C-X	15	
	$T\text{-}MX_A$	18	C-Y	0.595	
	$T\text{-}RO_A$	19			
					
15	Fuzzy triangle		Crisp value		0.5
	$T\text{-}LO_A$	14	C-X	15	
	$T\text{-}MX_A$	18	C-Y	0.819	
	$T\text{-}RO_A$	19			
					

16	Fuzzy triangle		Crisp value		1
	$T\text{-}LO_A$	10	C-X	18	
	$T\text{-}MX_A$	12	C-Y	0.143	
	$T\text{-}RO_A$	15			
					

V. CONCLUSION

This paper proposes the algorithm for binary fuzzy relational operators, which can be used to compare two fuzzy sets. Algorithms used to calculate fuzzy relational operator values are introduced. We developed an application that provides GUI and fuzzy relational operator calculations to prove the reliability of our algorithms. The testing results are generated randomly by this application. The results show that various comparisons are proved to be calculated correctly by our implementation. The proposed algorithms for fuzzy ordering will be used in FXI to enable comparison of two fuzzy sets.

Future research in this direction will tackle problems related to the implementation of aggregate functions, like MIN MAX, and SUM, using the proposed algorithms in FXI.

REFERENCES

[1] A. Champi, E. Damiani, S. Guinea, S. Marrara, G. Pasi, and P. Spoletini, "A Fuzzy Extension for the XPath Query Language" in *Flexible Query Answering Systems, Lecture Notes in Computer Science*, vol. 4027, 2006, pp. 210—221.

[2] E.J.T. Fredrick, and G. Radhamani, "Fuzzy Logic Based XQuery operations for Native XML Database Systems," in *International Journal of Database Theory and Application*, vol. 2, pp. 14—20.

[3] S. Skrbic, M. Rackovic, and M. Takaci, "Prioritized Fuzzy Logic Based Information Processing in Relational Databases," in *Knowledge-Based Systems*, vol. 38, 2013, pp. 62—73.

[4] P.S. Ueng, and S. Skrbic, "Implementing XQuery Fuzzy Extensions Using a Native XML Database," in *Proceeding of 13th IEEE International Symposium on Computational Intelligence and Informatics*, 2012, pp.305—309.

[5] S. Skrbic, and M. Rackovic, *Fuzzy databases*, Faculty of Sciences, University of Novi Sad, Novi Sad, 2013.

[6] U. Bodenhofer, "Orderings of Fuzzy Sets Based on Fuzzy Orderings Part I: The Basic Approach," in Mathware & Soft Computing, 2008, pp.201—218.