

Fuzzy Logic Programming for Implementing a Flexible XPath-based Query Language



Dep. of Languages and
Computation
University of Almería, Spain

Jesús M. Almendros-Jiménez
Luna Tedesqui

Ginés Moreno Valverde

RuleML11 - Barcelona, Spain
July 18, 2011



Dep. of Computing Systems,
University of Castilla-La
Mancha, Spain

Alejandro



Outline of the talk

- ✓ Introduction
- ✓ FuzzyXPath
- ✓ Examples
- ✓ Implementation
- ✓ Conclusions



Introduction and objectives

- We present an extension of the XPath query language that provide ranked answers.
- A Fuzzy variants of *and*, *or* and *avg* operators for Xpath conditions.
- Two structural constraints, called *down* and *deep*.
- Implemented by means of *Multi-Adjoint Logic Programming*(MALP) and the FL \diamond PER tool.



XML Path Language

- The *XPath* language has been proposed as a standard for XML querying.
- Based on the description of the path in the XML tree to be retrieved.
- XPath allows to specify the name of nodes (tags) and attributes together with boolean conditions.
- XPath querying mechanism is based on a boolean logic.

Examples:

- */bib/book/title*
- */bib/book/title[@text = "DonQuijote de laMancha"]*
Absolute Path
- */bib/book[@price < 30 and @year < 2006]*
- *//title*
Relative Path

- *//book[@price < 25]*



fuzzyXPath

FuzzyXPath is an extension of the XPath query language for the handling of flexible queries

- Two structural constraints called *deep* and *down* for which a certain degree of relevance can be associated.
- Fuzzy Operators *and*, *or* and *avg* for XPath conditions.

FuzzyXPath is defined by means of the following rules:

```

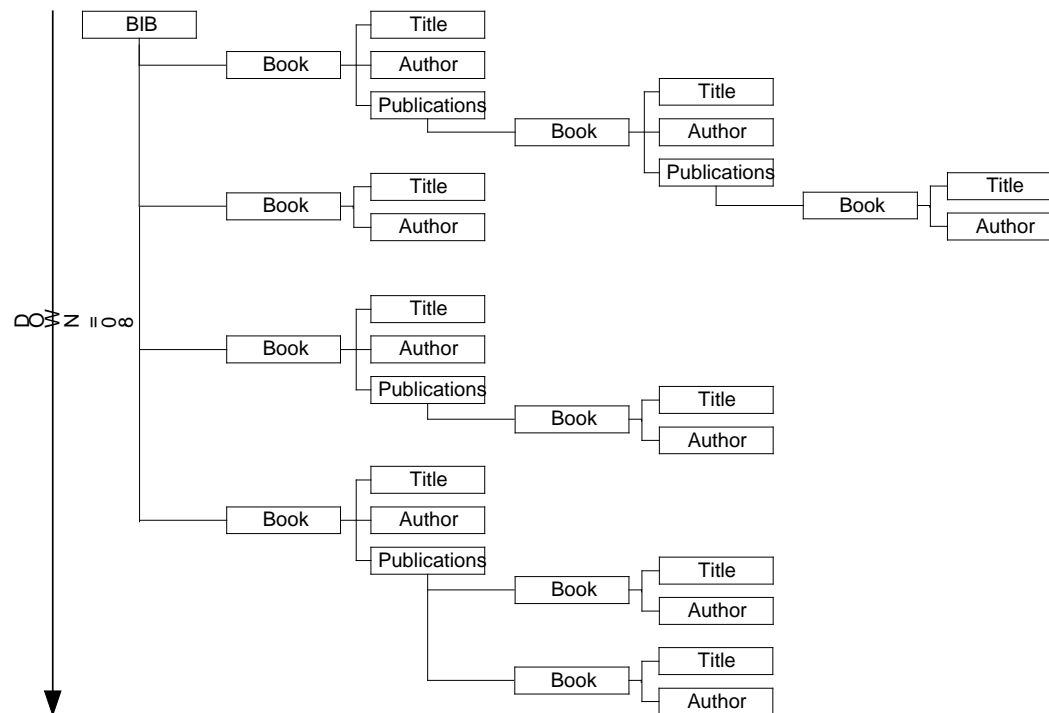
xpath := [[deepdown]]path
path := literal | text() | node
      | @att |
node := node/path | node//path
cond := QName | QName[[cond]]
deepdown := path op path
op := DEEP=degree; DOWN=degree
      > | = | < | and | or |
      avg
  
```

Structural constraints *DEEP/DOWN*

A Fuzzy XPath expression can be adorned with

$\ll [DEEP = r_1, DOWN = r_2] \gg$

which means that the *deepness* of elements is penalized by r_1 and that the *order* of elements is penalized by r_2 , and such penalization is proportional to the distance^{DEEP = 0.8}.





Fuzzy Logic Operators

The classical *and* and *or* connectives admit here a fuzzy behavior based on fuzzy logic, moreover, the *avg* operator.

- $\&_p(x; y) = x * y$
- $\vee_p(x; y) = x + y - x * y$
- $\@_{avg}(x; y) = (x + y) / 2$

p	q	p & q	p ∨ q	P @avg q
0	0	0	0	0
0	1	0	1	0.5
1	0	0	1	0.5
1	1	1	1	1
0.2	0.7	0.14	0.76	0.45
0.8	0.4	0.32	0.88	0.60
...

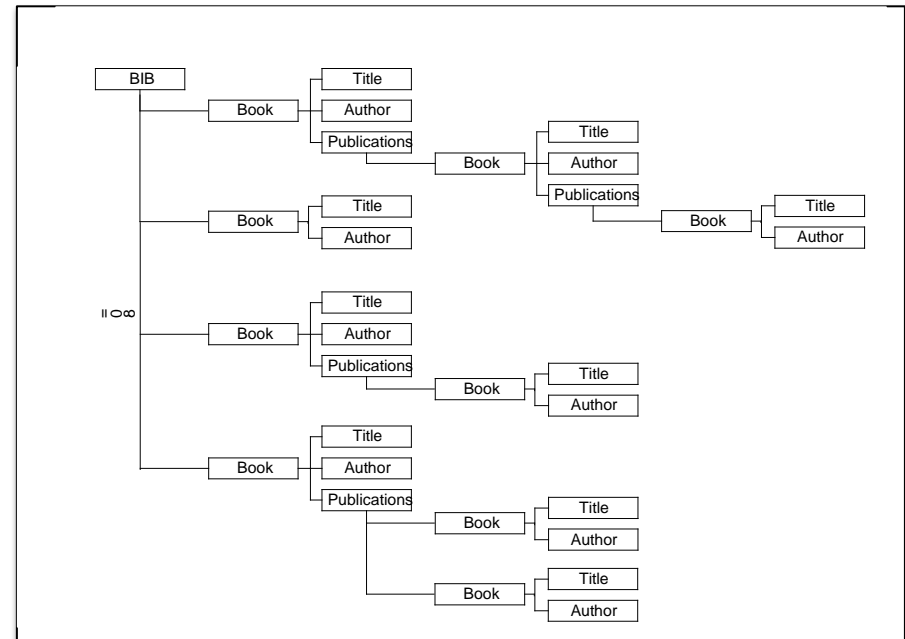


Input XML document in our examples

```

<bib>
  <book year="2001" price="45.95">
    <title>Don Quijote de la Mancha</title>
    <author>Miguel de Cervantes Saavedra</author>
    <publications>
      <book year="1997" price="35.99">
        <title>La Galatea</title>
        <author>Miguel de Cervantes Saavedra</author>
        <publications>
          <book year="1994" price="25.99">
            <title>Los trabajos de Persiles y Sigismunda</title>
            <author>Miguel de Cervantes Saavedra</author>
          </book>
        </publications>
      </book>
    </publications>
  </book>
  <book year="1999" price="25.65">
    <title>La Celestina</title>
    <author>Fernando de Rojas</author>
  </book>
  <book year="2005" price="29.95">
    <title>Hamlet</title>
    <author>William Shakespeare</author>
    <publications>
      <book year="2000" price="22.5">
        <title>Romeo y Julieta</title>
        <author>William Shakespeare</author>
      </book>
    </publications>
  </book>
  <book year="2007" price="22.95">
    <title>Las ferias de Madrid</title>
    <author>Felix Lope de Vega y Carpio</author>
    <publications>
      <book year="1996" price="27.5">
        <title>El remedio en la desdicha</title>
        <author>Felix Lope de Vega y Carpio</author>
      </book>
      <book year="1998" price="12.5">
        <title>La Dragontea</title>
        <author>Felix Lope de Vega y Carpio</author>
      </book>
    </publications>
  </book>
</bib>

```

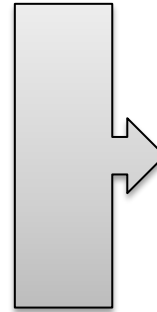


Example No 1

`[[DEEP=0.9;DOWN=0.8]]//title`

Resulting XML Document:

```
<result>
  <title rsv="0.81">Don Quijote de la Mancha</title>
  <title rsv="0.52488">La Galatea</title>
  <title rsv="0.340122">Los trabajos de Persiles ...</title>
  <title rsv="0.648">La Celestina</title>
  <title rsv="0.5184">Hamlet</title>
  <title rsv="0.335923">Romeo y Julieta</title>
  <title rsv="0.41472">Las ferias de Madrid</title>
  <title rsv="0.268739">El remedio en la desdicha</title>
  <title rsv="0.214991">La Dragontea</title>
</result>
```



```
0.81=0.92
0.52488=0.94*0.8
0.340122=0.96*0.82
0.648=0.92*0.8
0.5184=0.92*0.82
0.335923=0.94*0.83
0.41472=0.92*0.83
0.268739=0.94*0.84
0.214991=0.94*0.85
```

```
<bib>
  <book year="2001" price="45.95">
    <title>Don Quijote de la Mancha</title>
    <author>Miguel de Cervantes Saavedra</author>
    <publications>
      <book year="1997" price="35.99">
        <title>La Galatea</title>
        <author>Miguel de Cervantes Saavedra</author>
        <publications>
          <book year="1994" price="25.99">
            <title>Los trabajos de Persiles y Sigismunda</title>
            <author>Miguel de Cervantes Saavedra</author>
          </book>
        </publications>
      </book>
    </publications>
  </book>

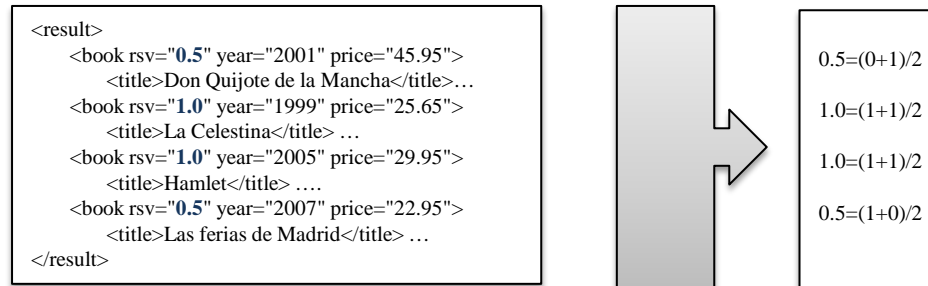
  <book year="1999" price="25.65">
    <title>La Celestina</title>
    <author>Fernando de Rojas</author>
  </book>

  <book year="2005" price="29.95">
    <title>Hamlet</title>
  </book>
</bib>
```

Example Nº 2

/bib/book[@price<30 avg @year<2006]

Resulting XML Document:



```
<bib>
  <book year="2001" price="45.95">
    <title>Don Quijote de la Mancha</title>
    <author>Miguel de Cervantes Saavedra</author>
    <publications>
      <book year="1997" price="35.99">
        <title>La Galatea</title>
        <author>Miguel de Cervantes Saavedra</author>
        <publications>
          <book year="1994" price="25.99">
            <title>Los trabajos de Persiles y Sigismunda</title>
            <author>Miguel de Cervantes Saavedra</author>
          </book>
        </publications>
      </book>
    </publications>
  </book>

  <book year="1999" price="25.65">
    <title>La Celestina</title>
    <author>Fernando de Rojas</author>
  </book>

  <book year="2005" price="29.95">
    <title>Hamlet</title>
```

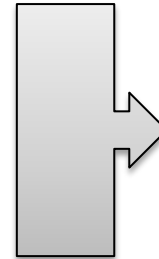


Example Nº 3

`[(DEEP=0.9;DOWN=0.8)]//book[(@price>25 and @price<30) avg (@year<2000 or @year>2006)]/title`

Resulting XML Document:

```
<result>
  <title rsv="0.3645">La Galatea</title>
  <title rsv="0.59049">Los trabajos de Persiles y Sigismunda</title>
  <title rsv="0.72">La Celestina</title>
  <title rsv="0.288">Hamlet</title>
  <title rsv="0.2304">Las ferias de Madrid</title>
  <title rsv="0.373248">El remedio en la desdicha</title>
  <title rsv="0.149299">La Dragonte</title>
</result>
```



```
0.3645=0.93*1/2
0.59049=0.95*1
0.72=0.9*0.8*1
0.288=0.9*0.82*1/2
0.2304=0.9*0.83*1/2
0.373248=0.93*0.83*1
0.149299=0.93*0.84*1/2
```

```
<bib>
  <book year="2001" price="45.95">
    <title>Don Quijote de la Mancha</title>
    <author>Miguel de Cervantes Saavedra</author>
    <publications>
      <book year="1997" price="35.99">
        <title>La Galatea</title>
        <author>Miguel de Cervantes Saavedra</author>
        <publications>
          <book year="1994" price="25.99">
            <title>Los trabajos de Persiles y Sigismunda</title>
            <author>Miguel de Cervantes Saavedra</author>
          </book>
        </publications>
      </book>
    </publications>
  </book>

  <book year="1999" price="25.65">
    <title>La Celestina</title>
    <author>Fernando de Rojas</author>
  </book>

  <book year="2005" price="29.95">
    <title>Hamlet</title>
  </book>
</bib>
```

Implementation is based on the following items:

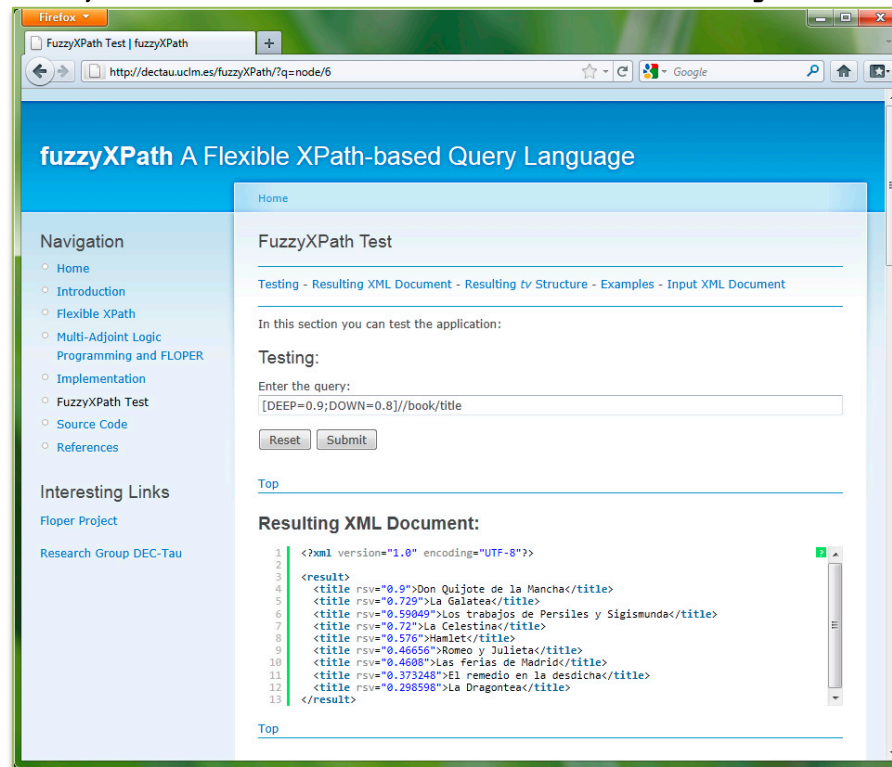
Our implementation is based on the following items:

1. We have reused/adapted several modules of our previous Prolog-based implementation of (crisp) XPath.
2. We have used the SWI-Prolog library for loading XML files, in order to represent a XML document by means of a Prolog term.
3. The parser of XPath has been extended to recognize the new keywords *deep*, *down*, *avg*, etc... with their proper arguments.
4. Each tag is represented as a data-term of the form: *element(Tag, Attributes, Subelements)*.
5. A predicate called fuzzyXPath where *fuzzyXPath(+ListXPath, +Tree, +Deep, +Down)*.
6. The evaluation of the query generates a *truth value* which has the form of a tree, called *tv tree*.
7. Finally, the *tv tree* is used for computing the output of the query, by multiplying the recorded values. A predicate called *tv_to_elem* has been

Web Version

More details about our implementation of the flexible version of XPath reported in this paper, are available on:

<http://dectau.uclm.es/fuzzyXPath/>






Conclusions

In this paper we have enriched XPath with new constructs (both Structural *deep/down*, and constraints *avg* together with fuzzy versions of classical *or/and* operators) in order to flexibly query XML documents.

This paper represents the first real-world application developed with the fuzzy logic language MALP, by showing its capabilities for easily modeling scenarios where concepts somehow based on fuzzy logic play a crucial role.

We think that this research line promises fruitful developments in the near future by reinforcing the power of fuzzy XPath commands, extensions to cope with XQuery and the semantic



Fuzzy Logic Programming for Implementing a Flexible XPath-based Query Language



Dep. of Languages and
Computation
University of Almería, Spain

Jesús M. Almendros-Jiménez
Luna Tedesqui

Ginés Moreno Valverde

RuleML11 - Barcelona, Spain
July 18, 2011



Dep. of Computing Systems,
University of Castilla-La
Mancha, Spain

Alejandro

fuzzyXPath Predicate

The predicate fuzzyXPath has the following form:

fuzzyXPath(Consult, Element, Deep, Down, Result)

where

Consult: Consultation has a list structure, which is touring until empty.

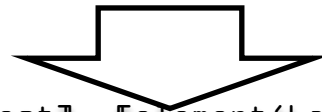
Element: Structure of SWI-Prolog used to handle XML files

Deep: Horizontal penalization value, when is deepened in the XML Tree

Down: Vertical penalization value

Result: Out the predicate, output with TV structure

```
fuzzyXPath([Label|LabelRest], Element(Label,
  ,Children)|Siblings],Deep, Down) <prod
@fuse(
  tv(1,[Element(Label, Attr, Children), [], []]),
  &prod(Deep, fuzzyXPath(LabelRest, Children, Deep, Down))
  &prod(Down, fuzzyXPath([Label|LabelRest], Siblings, Deep, Down))
) with tv(1,[])
```



```
fuzzyXPath([Label|LabelRest], Element(Label,
  ,Children)|Siblings],Deep, Down, TV Iam):-
  fuzzyXPath(LabelRest, Children, Deep, Down, TV Son),
  and prod(Deep, TV Son, TV Son0),
  fuzzyXPath([Label|LabelRest], Siblings, Deep, Down, TV Bro),
  and prod(Down, TV Bro, TV Bro0),
  agr fuse(tv(1,[Element(Label, Attr, Children), [], []]), TV Son0,
  TV Bro0, TV body),
  and prod(tv(1,[]), TV body, TV Iam).
```