A XPath Debugger based on Fuzzy Chance Degrees

Jesús M. Almendros-Jiménez[1], Alejandro Luna[2], and Ginés Moreno[2]

[1] Dept. of Languages and Computation, University of Almería, Spain
Email: `jalmen@ual.es`
[2] Dept. of Computing Systems, University of Castilla-La Mancha, Spain
Emails: `Alejandro.Luna@alu.uclm.es`, `Gines.Moreno@uclm.es`

Abstract. We describe our recently designed/implemented method for debugging XPath queries which produces a set of alternative XPath expressions with higher chances for retrieving answers from XML files.

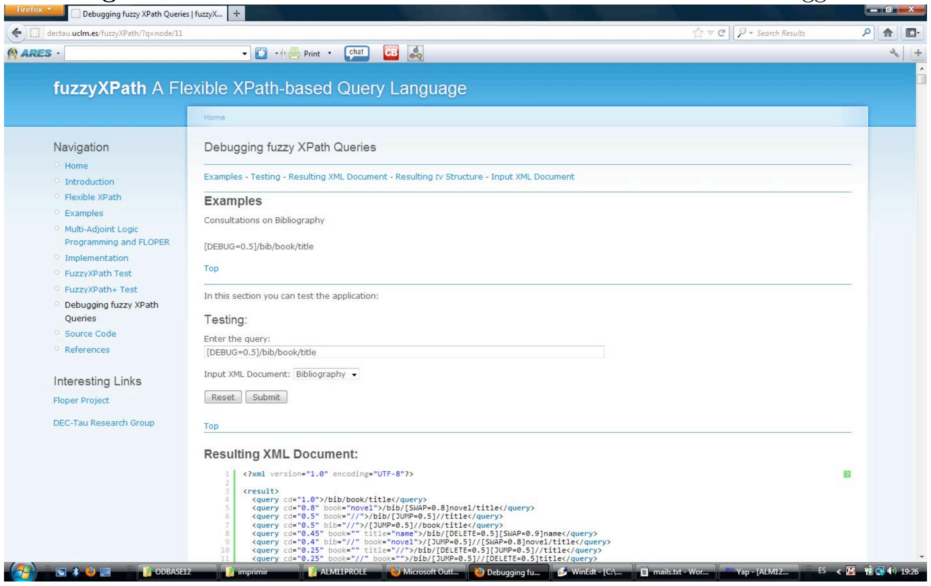
The eXtensible Markup Language (XML) provides a very simple language to represent the structure of data, using tags to label pieces of textual content, and a tree structure to describe the hierarchical content. The XPath language [2] was designed as a query language for XML in which the path of the tree is used to describe the query. This work is motivated by the evidence that users frequently omit some tags on paths, add more than necessary, or employ wrong tags[3], in order to help them when formulating XPath queries.

Our debugging method acts on a given initial XPath query Q preceded by the [`DEBUG` $= r$] command, where r is a real number in the unit interval used at debugging time for labeling *deletion* and *jumping* actions. So, assume that we plan to process "[`DEBUG`=0.5]/bib/book/title" w.r.t. the following XML document:

```
<bib>
    <name>Classic Literature</name>
    <book year="2001" price="45.95">
        <title>Don Quijote de la Mancha</title>
        <author>Miguel de Cervantes Saavedra</author>
        <references>
            <novel year="1997" price="35.99">
                <name>La Galatea</name>
                <author>Miguel de Cervantes Saavedra</author>
                <references>
                    <book year="1994" price="25.99">
                        <title>Los trabajos de Persiles y Sigismunda</title>
                        <author>Miguel de Cervantes Saavedra</author>
                    </book>
                </references>
            </novel>
        </references>
    </book>
    <novel year="1999" price="25.65">
        <title>La Celestina</title>
        <author>Fernando de Rojas</author>
    </novel>
</bib>
```

[3] We are nowadays equipping our tool with techniques for automatically extracting *similarity degrees* between tags from standard internet resources such as `WordNet`.

Fig. 1. Screen-shot of an on-line work session with the XPath debugger.



fuzzyXPath A Flexible XPath-based Query Language

Home

Navigation
- Home
- Introduction
- Flexible XPath
- Examples
- Multi-Adjoint Logic Programming and FLOPER
- Implementation
- FuzzyXPath Test
- FuzzyXPath+ Test
- Debugging fuzzy XPath Queries
- Source Code
- References

Interesting Links

Floper Project

DEC-Tau Research Group

Debugging fuzzy XPath Queries

Examples - Testing - Resulting XML Document - Resulting *tv* Structure - Input XML Document

Examples

Consultations on Bibliography

[DEBUG=0.5]/bib/book/title

Top

In this section you can test the application:

Testing:

Enter the query: [DEBUG=0.5]/bib/book/title

Input XML Document: Bibliography

Reset Submit

Top

Resulting XML Document:

```
<?xml version="1.0" encoding="UTF-8"?>

<result>
  <query cd="1.0">/bib/book/title</query>
  <query cd="0.8" book="novel">/bib/[SWAP=0.8]novel/title</query>
  <query cd="0.5" book="//">/bib/[JUMP=0.5]//title</query>
  <query cd="0.5" bib="//">/[JUMP=0.5]//book/title</query>
  <query cd="0.45" book="" title="name">/bib/[DELETE=0.5][SWAP=0.9]name</query>
  <query cd="0.4" bib="//" book="novel">/[JUMP=0.5]//[SWAP=0.8]novel/title</query>
  <query cd="0.25" book="" title="//">/bib/[DELETE=0.5][JUMP=0.5]//title</query>
  <query cd="0.25" book="//" book="">/bib/[JUMP=0.5]//[DELETE=0.5]title</query>
```

Our technique produces a set of alternative queries $Q_1, ..., Q_n$ (each one adorned with attributes and special keywords informing about all changes that deviates Q_i from Q) packed into an output XML document like the following one, where the set of proposals is sorted w.r.t. a CD key meaning that, as much changes are performed on Q_i and as more *traumatic* they are w.r.t to Q, then its "*Chance Degree*" becomes lower according a policy based on the *product fuzzy logic*:

<result>
  <query **cd**="1.0">/bib/book/title</query>
  <query **cd**="0.8" **book**="novel">/bib/[SWAP=0.8]novel/title</query>
  <query **cd**="0.5" **bib**="//">/[JUMP=0.5]//book/title</query>
  <query **cd**="0.45" **book**="" **title**="name">/bib/[DELETE=0.5][SWAP=0.9]name</query>
  <query **cd**="0.225" **bib**="" **book**="//" **title**="name">/[DELETE=0.5][JUMP=0.5]//[SWAP =0.9]name</query> ...
  ...
</result>

So, after executing (please, see Figure 1, consult [1] and visit `http://dectau.uclm.es/fuzzyXPath/` where it is possible too to perform an on-line debugging session) the first proposed alternative -which coincides with the original query-, we can retrieve "Don Quijote de La Mancha", the second query returns "La Celestina", the third one adds "Los trabajos de Persiles y Sigismunda" to "Don Quijote", whereas the fully-annotated case with commands `DELETE`, `JUMP` and `SWAP` (which justifies its lower CD value 0.225) is even able to produce "Classic Literature", as shown in Figure 2, where the **rsv** key -*retrieved status value*- means the satisfaction degree of each solution w.r.t. the considered query.

Fig. 2. Execution of query "/[DELETE=0.5][JUMP=0.5]//[SWAP=0.9]name"

```
<result>
  <name rsv="0.225">Classic Literature</name>
  <name rsv="0.028125">La Galatea</name>
</result>
```

In order to explain how our technique works, let us consider a path P of the form "$/tag_1/.../tag_i/tag_{i+1}/...$", where $P[i]$ references tag_i in P: this notation is used in the following sketched algorithm where symbols `Q` and `B` refers to the original Xpath query and any *branch* of the input XML document, respectively.

```
For each branch B in the input document
  For each tag Q[i] in the input query
    If Q[i] and B[i] are ‘‘syntactically’’ different tags then
      Add a new query with SWAP if Q[i] and B[i] are ‘‘similar’’ tags
      Add a new query with JUMP if Q[i] coincides with B[j], being j>i
      Add a new query with DELETE if Q[i] coincides with B[i+1]
```

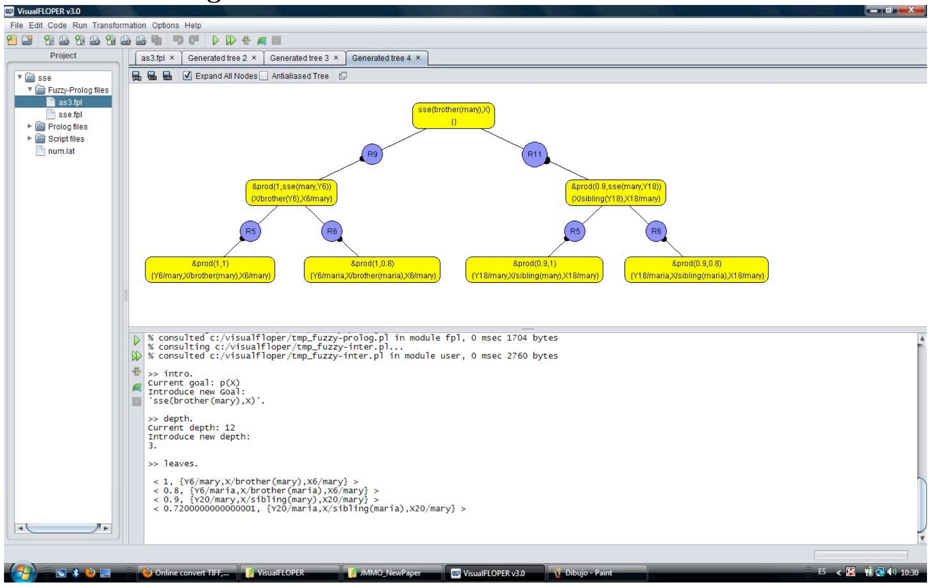
We would like to remark that even when we have worked with a very simple query with three tags in our examples, our technique works with more complex queries with larger paths and connectives in conditions. For instance, in Figure 3 we debug a query which needs to SWAP in its condition the wrong occurrence of "cost" by the similar word "price": note that the first alternative deletes tag "classic", but our debugger produces too more chances based on JUMP and SWAP commands, whose further execution are intended to produce new interesting results.

Fig. 3. Debugging effects on XPath conditions associated to the complex query: [DEBUG=0.6]/bib/classic/[DEEP=0.8]//book[@year < 2000 avg{3,1} @cost < 50]/title

```
<result>
    <query cd="0.54" classic="" cost="price">/bib/[DELETE=0.6][DEEP=0.8]//book
                [(@year < 2000) avg{3,1} ([SWAP=0.9]@price < 50)]/title</query>
    <query cd="0.54" classic="//" cost="price">/bib/[JUMP=0.6]//[DEEP=0.8]//book
                [(@year < 2000) avg{3,1} ([SWAP=0.9]@price <50)]/title</query>
    <query cd="0.432" classic="" book="novel" cost="price">
                /bib/[DELETE=0.6][DEEP=0.8]//[SWAP=0.8] novel
                [(@year < 2000) avg{3,1} ([SWAP=0.9]@price < 50)]/title</query>
   .........
   ......
   ..
</result>
```

To finish, we wish to mention that both the interpreter and the debugger of the fuzzy-XPath dialect considered in this paper, have been implemented with a fuzzy logic language by exploiting the clear synergies between the source code and the target tools (see Figure 4 and visit `http://dectau.uclm.es/floper/`).

Fig. 4. Screen-shot of a work session with FLOPER.



```
% consulted c:/visualfloper/tmp_fuzzy-prolog.pl in module fpl, 0 msec 1704 bytes
% consulting c:/visualfloper/tmp_fuzzy-inter.pl...
% consulted c:/visualfloper/tmp_fuzzy-inter.pl in module user, 0 msec 2760 bytes

>> intro.
Current goal: p(X)
Introduce new Goal:
'sse(brother(mary),X)'.

>> depth.
Current depth: 12
Introduce new depth:
3.

>> leaves.

 < 1, {Y6/mary,X/brother(mary),X6/mary} >
 < 0.8, {Y6/maria,X/brother(maria),X6/mary} >
 < 0.9, {Y20/mary,X/sibling(mary),X20/mary} >
 < 0.7200000000000001, {Y20/maria,X/sibling(maria),X20/mary} >
```

Acknowledgements: Work partially supported by the EU, under FEDER, and the Spanish Science and Innovation Ministry (MICINN) under grant TIN2008-06622-C03-03, as well as by Ingenieros Alborada IDI under grant TRA2009-0309, and the JUNTA ANDALUCIA administration under grant TIC-6114.

References

1. J. Almendros, A. Luna, G. Moreno. A Flexible XPath-based Query Language Implemented with Fuzzy Logic Programming. Springer LNCS 6826, pp. 186-193. 2011.
2. A. Berglund, S. Boag, D. Chamberlin, M.F. Fernandez, M. Kay, J. Robie, and J. Siméon. XML path language (XPath) 2.0. *W3C*, 2007.