BiFluX: A Bidirectional Functional Update Language for XML

Hugo Pacheco
Joint work with Tao Zan and Zhenjiang Hu

National Institute of Informatics, Tokyo, Japan

BIRS workshop
BX – Theory and Applications Across Disciplines

Banff, December 5th, 2013
“A mechanism for maintaining the consistency of two (or more) related sources of information.”

[Czarnecki et al., ICMT 2009]
• **relational**: derive both transformations from a relation between the two schemas

• **bidirectionalization**: write one transformation, derive the other

• **combinatorial**: write a single program that denotes both transformations
• due to the latent ambiguity of BXs
• existing approaches focus mainly on enforcing consistency
• from the programmer’s perspective, they suffer either from:
  • supporting only “trivial” BXs
  • providing arbitrary bidirectional behavior
  • giving little control of what the BX does
  • being impractical to specify complex BXs
“Intuitively, a BX translates updates on a source model into updates on a target model, and vice-versa, so that the updated models are consistent.”
• XML transformation languages (XQuery, XSLT, XDuce) are bad for specifying small updates
• a few dedicated languages for in-place XML updates:
  • XQuery Update Facility [W3C]:
    • imperative language
    • ill-understood semantics (aliasing, side-effects, depends on traversal order)
  • Flux (Functional Lightweight Updates for XML) [Cheney, ICFP 2008]:
    • functional language
    • clear semantics
    • straightforward type-checking
• XUpdate, XQuery!, etc…
Our proposal: BiFluX

- we propose BiFluX, a bidirectional variant of Flux
- particular class of BXs: lenses, view updating
- modest syntactic extension
  - notion of view (feat. pattern matching, non-in-place updates)
  - static restrictions to ensure well-behavedness

- Flux: fixed input schema & new output schema
- unidirectional in-place semantics

- BiFluX: fixed source and view schemas
- bidirectional semantics as lenses
Is this a bidirectional *update*?

UPDATE $source/books/book BY
    INSERT BEFORE title
    VALUE <author>$view</author>
WHERE title = "Through the Looking-Glass"

\[ S = \text{books} [\text{book} [\text{author} [\text{String}] +, \text{title} [\text{String}]]*] \]
\[ V = \text{String} \]
Is this a bidirectional *update*?

```
UPDATE $source/books/book BY
  INSERT BEFORE title
  VALUE <author>$view</author>
WHERE title = "Through the Looking-Glass"
```

\[
S = books [book [author [String]+, title [String]]*]
\]

\[
V = String
\]

- adds the view as the last author to the source authors
- violates the GETPUT law of lenses!
Is this a bidirectional *update*?

UPDATE $source/books/book BY
    REPLACE author[last()]
    WITH <author>$view</author>
WHERE title = "Through the Looking-Glass"

\[
V = String
\]
Is this a bidirectional *update*?

```
UPDATE $source/books/book BY
    REPLACE author[last()]
    WITH <author>$view</author>
WHERE title = "Through the Looking-Glass"
```

\[ S = books [book [author [String]+, title [String]]*] \]
\[ V = String \]

- replaces the last author in the source with the view author
- well-behaved *update*!
BiFluX Core Language

- BiFluX → core language → lenses
- we consider two different semantics
  - default **bidirectional** semantics as lenses
    - Hugo Pacheco and Zhenjiang Hu and Sebastian Fischer
      Monadic Combinators for “Putback” Style Bidirectional Programming
      *PEPM 2014.*
  - Flux “standard” **in-place** semantics (insert, delete, ...)
    - James Cheney
      FLUX: Functional Updates for XML
      *ICFP 2008.*

- core BiFluX language:

  \[ e ::= \text{“core XQuery expressions”} \]

  \[ p ::= \text{“simple XPath expressions”} \]

  \[ pat ::= \text{“linear, sequence-based XDuce patterns”} \]

  \[ u ::= \text{“Flux in-place updates”} \]

  \[ s ::= \text{“BiFluX bidirectional updates”} \]
• BiFluX high-level language (changes to Flux in red):

\[
Stmt ::= \text{Upd} [\text{WHERE } Expr] \mid \text{IF } Expr \text{ THEN } Stmt \text{ ELSE } Stmt \\
\mid \text{Stmt ; Stmt} \mid \{ \text{Stmt} \} \mid \text{LET } Pat = Expr \text{ IN } Stmt \\
\mid \text{CASE } Expr \text{ OF } \{ \text{Cases} \}
\]

\[
Upd ::= \text{INSERT (BEFORE | AFTER) PatPath VALUE Expr} \\
\mid \text{INSERT AS (FIRST | LAST) INTO PatPath VALUE Expr} \\
\mid \text{DELETE [FROM] PatPath | REPLACE [IN] PatPath WITH Expr} \\
\mid \text{UPDATE PatPath BY Stmt} \\
\mid \text{UPDATE PatPath BY VStmt FOR VIEW PatPath [Match]} \\
\mid \text{KEEP Path AS (FIRST | LAST) | CREATE VALUE Expr}
\]

\[
Cases ::= Pat \rightarrow Stmt \mid \text{Cases }\mid\mid \text{Cases} \\
VStmt ::= VUpd \mid\mid VUpd \mid VUpd \\
VUpd ::= \text{MATCH } \rightarrow Stmt \\
\mid \text{UNMATCHS } \rightarrow Stmt \\
\mid \text{UNMATCHV } \rightarrow Stmt \\
Match ::= \text{MATCHING BY Path} \\
\mid \text{MATCHING SOURCE BY Path VIEW BY Path} \\
PatPath ::= [Pat \text{ IN}] \text{ Path}
\]
UPDATE $book IN $source/book BY
{
    MATCH -> REPLACE price WITH $price
| UNMATCHV -> CREATE VALUE <book category='undefined'>
            <title/>
            <author>??</author>
            <year>??</year>
            <price/>
    </book>
}
MATCHING SOURCE BY $book/title VIEW BY $title
A bookstore BiFluX Example: Forward

- Source:

```xml
<bookstore>
  <book>
    <title>Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category='Programming'>
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

- View:

```xml
<books>
  <book>
    <title>Harry Potter</title>
    <price>29.99</price>
  </book>
  <book>
    <title>Learning XML</title>
    <price>39.95</price>
  </book>
</books>
```
A bookstore BiFluX Example: Update

• Source:

```xml
<bookstore>
  <book>
    <title>Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category='Programming'>
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

• Updated View:

```xml
<books>
  <book>
    <title>XPath for Dummies</title>
    <price>19.99</price>
  </book>
  <book>
    <title>Harry Potter</title>
    <price>19.99</price>
  </book>
  <book>
    <title>Learning XML</title>
    <price>19.99</price>
  </book>
</books>
```
A bookstore BiFluX Example: Backward

- Updated Source:

```xml
<bookstore>
  <book category='undefined'>
    <title>XPath for Dummies</title>
    <author>??</author> <year>??</year>
    <price>19.99</price>
  </book>
  <book>
    <title>Harry Potter</title>
    <author>J K. Rowling</author> <year>2005</year>
    <price>19.99</price>
  </book>
  <book category='Programming'>
    <title>Learning XML</title>
    <author>Erik T. Ray</author> <year>2003</year>
    <price>19.99</price>
  </book>
</bookstore>
```
• proposed a novel *programming by update* bidirectional paradigm
• presented BiFluX, a bidirectional XML update language
• BiFluX is work in progress (much more under the hood)
• for demos and more info, see...

http://www.prg.nii.ac.jp/projects/BiFluX