Active XQuery

A. Bonifati, D. Braga, A. Campi, S. Ceri

Politecnico di Milano (Italy)

Outline

- Contributions
- Background
  - An update language for XQuery [TI*01]
- Syntax of Active XQuery
- Semantics of Active Xquery
  - Bulk Expansion Algorithm and Trigger Execution Model
- System Architecture
- Conclusions and Open issues

Main Contributions

- An active extension of the W3C query language
- Emulation of SQL3 trigger definition
- An algorithm for update expansion and a mechanism for interleaved execution of triggers and updates
- An architecture for the rapid prototyping of Active XQuery

Example Document

```
<!ELEMENT Lib (Shelf+, AuthorIndex)>
<!ELEMENT Shelf (Book*)>
<!ATTLIST Shelf  nr  ID  #REQUIRED>
<!ELEMENT Book (Author+, Title)>
<!ATTLIST Book  id  ID  #REQUIRED>
<!ELEMENT Author (#PCDATA)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT AuthorIndex (AuthorEntry*)>
<!ELEMENT AuthorEntry (Name, PubsCount)>
<!ATTLIST AuthorEntry  uni   CDATA   #IMPLIED
                       pubs  IDREFS  #IMPLIED>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT PubsCount (#PCDATA)>
```

Background

- An update language for Active Xquery presented by A. Y. Halevy et al. at Sigmod 2001 :

```
FOR $target IN document("Lib.xml")/Library,
    $frag   IN document("New.xml")/Shelves/Shelf
WHERE $frag/@nr="45"
UPDATE $target  {  INSERT $frag  }
```

Fragment to Insert

```
<Library>
 ...
 <Shelf nr="45">
  <Book id="AO97">
   <Author> J. Acute </Author>
   <Author> J. Obtuse </Author>
   <Title> Triangle Inequalities </Title>
  </Book>
  <Book id="So98">
   <Author> A. Sound </Author>
   <Title> Automated Reasoning </Title>
  </Book>
  ...
 </Shelf>
</Library>
```

The excerpt of Author Index

```
<AuthorIndex>
 ...
 <AuthorEntry uni="PoliMi" pubs=".. AO97 ..">
  <Name> J. Acute </Name>
  <PubsCount> ... </PubsCount>
 </AuthorEntry>
 ...
 <AuthorEntry uni="Princeton" pubs=".. So98 ..">
  <Name> A. Sound </Name>
  <PubsCount> ... </PubsCount>
 </AuthorEntry>
</AuthorIndex>
```

Referential Integrity

- NoDangling References:deletion of a `Book' element causes all its authors (listed in the index) to lose *dangling* references to that publication.

```
CREATE TRIGGER NoDangle
AFTER DELETE OF document("Lib.xml")//Book
FOR EACH NODE
DO ( FOR
  $AutIndex IN document("Lib.xml")//AuthorIndex,
  $MatchAut IN $AutIndex/AuthorEntry
                [Name = OLD_NODE/Author],
  $DangRef  IN $MatchAut/ref(pubs, OLD_NODE/@id)
UPDATE $AutIndex { DELETE $DangRef } )
```

Syntax of XQuery Triggers

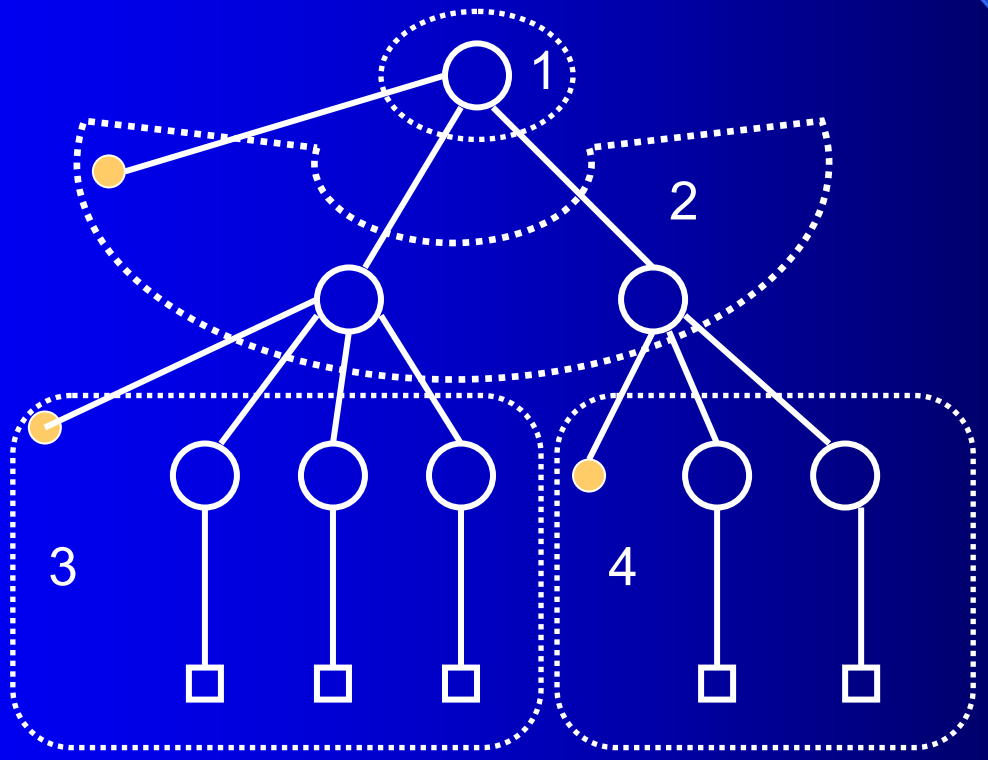
CREATE TRIGGER *Trigger-Name*
[WITH PRIORITY *Signed-Integer-Number*]
(BEFORE|AFTER)
(INSERT|DELETE|REPLACE|RENAME)+
OF *XPathExpression* (, *XPathExpression*)*
[FOR EACH (NODE|STATEMENT)]
[*XQuery-Let-Clause*]
[WHEN *XQuery-Where-Clause*]
DO (*XQuery-UpdateOp* | *ExternalOp*)

Semantics of Active XQuery

- Execution Model for Active XQuery
  - Semantics close to SQL3
    - (1) For each update, compute the set of affected nodes
    - (2) Each update is located exactly between BEFORE and AFTER triggers
    - (3) The procedure is recursively invoked if a trigger activates another trigger
  - SQL3 semantics does not work due to different granularity of updates
  - Need of an algorithmic expansion of *bulk* updates

Decomposition of updates in Active XQuery

- **Each "bulk" update is transposed into a sequence of smaller granularity updates on which rules are fired out**



An example of expansion(1)

```
EvalBefore(s1)
 Name:s1
 FOR $x IN document("Lib.xml")/Library,
  $frag IN document("New.xml")/Shelves/Shelf[@nr="45"]
 UPDATE $x
 {  INSERT <Shelf/>  }
 EvalBefore(s2)
 Name:s2
 FOR $x IN document("Lib.xml")/Library,
  $frag IN document("New.xml")/Shelves/Shelf[@nr="45"],
  $curfragment IN $x/*[empty($x/*[AFTER $curfragment])]
 UPDATE $curfragment
 {  INSERT new_attribute(nr, "45")
    INSERT <Book/>
    INSERT <Book/>  }
```

An example of expansion(2)

EvalBefore(s3)

```
Name:s3
FOR $x IN document("Lib.xml")/Library,
 $frag IN document("New.xml")/Shelves/Shelf[@nr="45"],
 $curfragment IN $x/*[empty($x/*[AFTER $curfragment])],
 $cur_node IN $curfragment/*[1]
UPDATE $cur_node
{  INSERT new_attribute(id, "AO97")
   INSERT <Author> J. Acute </Author>
   INSERT <Author> J. Obtuse </Author>
   INSERT <Title> Triangle Inequalities </Title>  }
```

EvalAfter(s3)

EvalBefore(s4)

An example of expansion(3)

Name:s4
FOR $x IN document("Lib.xml")/Library,
$frag IN document("New.xml")/Shelves/Shelf[@nr="45"],
$curfragment IN $x/*[empty($x/*[AFTER $curfragment])],
$cur_node IN $curfragment/*[2]
UPDATE $cur_node
{ INSERT new_attribute(id, "So98")
INSERT <Author> A. Sound </Author>
INSERT <Title> Automated Reasoning </Title> }

EvalAfter(s4)
EvalAfter(s2)
EvalAfter(s1)

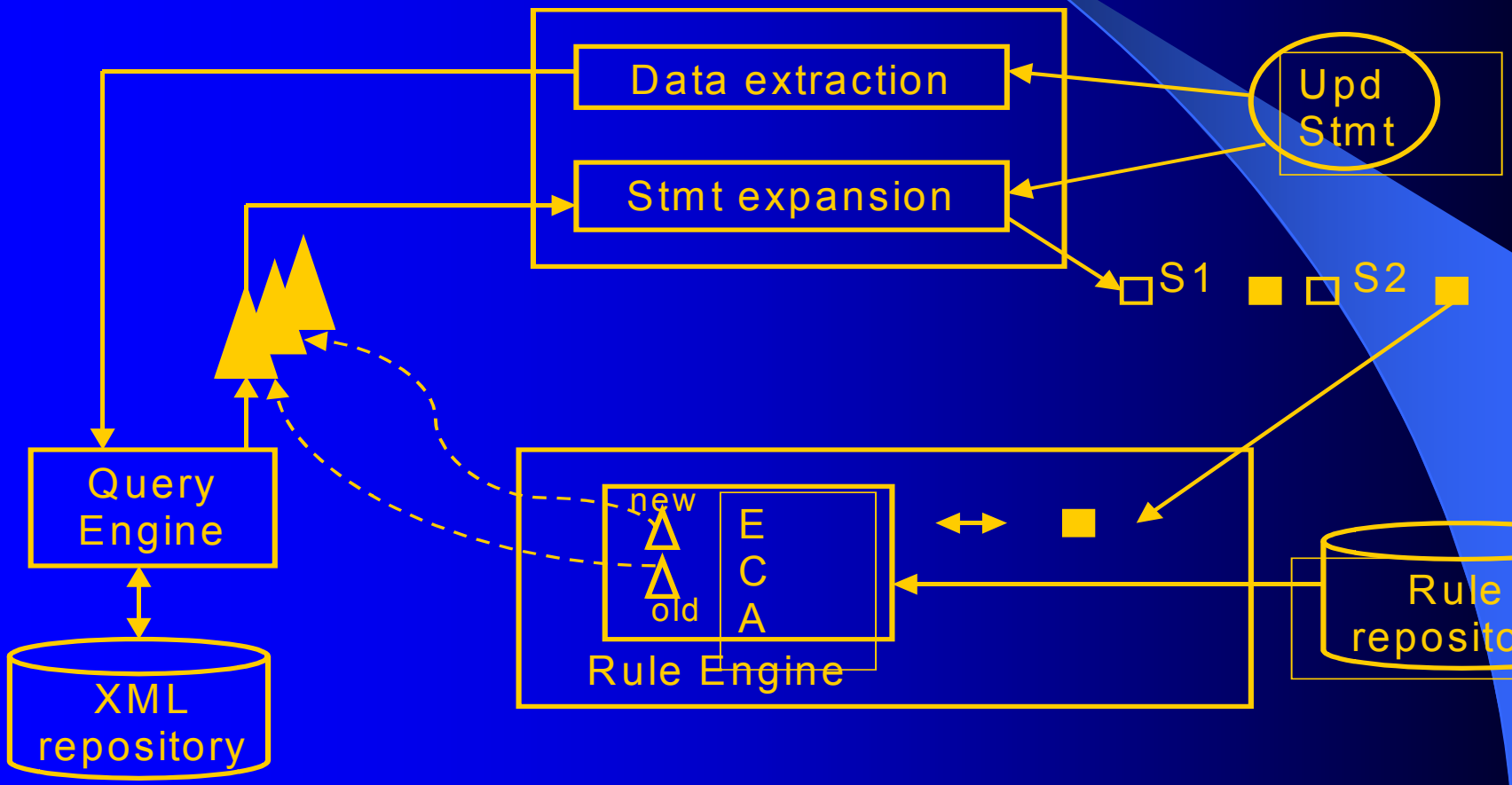
Advantages/Drawbacks

- Advantages:
  - Quick matching of triggers with updates
  - Simple composition of triggers with updates (possibly recursive)
  - Trigger engine and query engine physically detached
- Drawbacks:
  - Intermediate states and order dependence
  - Repeated executions of path traversals (a possible solution: caching mechanisms)

Detailed description of Active XQuery execution model

- Call procedure EXPAND_STATEMENT and store the retrieved fragments RF and the sequence of micro-updates SIL
- FOR EACH item in SIL:
  - Call COMPUTE_BEFORE_CS if it is an “EvalBefore” instruction
  - Call COMPUTE_AFTER_CS if it is an “EvalAfter” instruction
  - Execute it if it is an update statement

Active XQuery Architecture



Open Issues

- Schema-driven optimizations
- Indexing optimizations
- Management of BEFORE triggers
- Definition of illegal executions
- Compile-time trigger behavior analysis

Conclusions

- We have developed an extension of W3C Xquery for trigger definition and management.
- We propose an highly scalable modular architecture in which the Xquery Optimizer can be plugged in.
- We hope our paper will contribute to discussion upon XQuery desired features.

Automatic Indexing

- Insertion of a `Book' element causes a new reference to be inserted in all index entries that represent new book's authors.

```
CREATE TRIGGER AddNewReference
WITH PRIORITY -10
AFTER INSERT OF document("Lib.xml")//Book
FOR EACH NODE
DO ( FOR $ai IN document("Lib.xml")//AuthorIndex,
        $a  IN $ai/AuthorEntry[Name=$a]
     UPDATE $a
     { INSERT new_ref(pubs, NEW_NODE/@id)}
```

```
CREATE TRIGGER AddNewEntry
    AFTER INSERT OF document("Lib.xml")//Book
    FOR EACH NODE
    LET $AuthorsNotInList := (
      FOR $n IN NEW_NODE/Author
      WHERE empty(//AuthorIndex/AuthorEntry[Name=$n])
      RETURN $n )
    WHEN ( not( empty($AuthorsNotInList )) )
    DO ( FOR $ai IN document("Lib.xml")//AuthorIndex,
            $NewAuthor IN $AuthorsNotInList
         UPDATE $ai
         { INSERT <AuthorEntry>
                  <Name> {$NewAuthor/text()} </Name>
                   <PubsCount> 0 </PubsCount>
                  </AuthorEntry> } )
```

Counter of publications

```
CREATE TRIGGER IncrementCounter
 AFTER INSERT OF //new_ref(pubs)
 FOR EACH NODE
 LET $Counter := NEW_NODE/../PubsCount
 DO ( FOR $AuthorEntry IN NEW_NODE/..
      UPDATE $AuthorEntry
      { REPLACE $Counter WITH $Counter + 1 } )
```

More details in…

- My PhD Thesis (just defended):

  Angela Bonifati, "Reactive Services for XML Repositories", PhD Thesis, Politecnico di Milano, January 2002.

  (…soon available on my HP)