Deep Set Operators for XQuery

Bo Luo, Dongwon Lee, Wang-Chien Lee, and Peng Liu
The Pennsylvania State University
University Park, PA, USA

Motivation: XML Access Control

- XML Access Control
- Rules and queries are described by XPath
- Positive rules: some nodes are allowed to access.
- Negative rules: access to some nodes is restricted.

Positive Rules

Query: //item

<item id="item0">
  <location>United States</location>
  <quantity>1</quantity>
  <name>duteous nine eighteen</name>
  <payment>Creditcard</payment>
  <description/>
  <shipping/>
  <incategory category="category0"/>
</item>

<item id="item1">
  <location>United States</location>
  <quantity>1</quantity>
  <name>great</name>
  <payment>Money order, Cash</payment>
  <shipping>Will ship internationally</shipping>
  <incategory category="category0"/>
</item>

Rule: (Role, //item/name, +, LC)

Positive Rules

- There is an "intersect" semantics between positive rules and query.
- Operands may not be at the same level.
  - E.g. //item intersect //item/name
- Existing intersect operator in XQuery could not handle this semantics.

Negative Rules

Negative rules:
  Query: /site/people
  Rule: (Role, /site/people/person/creditcard, -, LC)

<person id="person6">
  <name>Moheb Mersereau</name>
  <emailaddress>mailto:Mersereau@umass.edu</emailaddress>
  <creditcard>4462 9674 4373 8450</creditcard>
</person>

<person id="person7">
  <name>Yuanyuan Sydow</name>
  <emailaddress>mailto:Sydow@newpaltz.edu</emailaddress>
  <phone>+213 (600) 7118249</phone>
</person>

Negative Rules

- There is an "except" semantics between negative rules and query.
- They may not be at the same level.
  - E.g. //people except //people/person/creditcard
- Existing intersect operator in XQuery could not handle this semantics.
Deep set operators

• Existing set operators in XQuery
  – UNION, INTERSECT, EXCEPT
  – Operations are based on nodes (node_ids)
  – Take node sequences as operands
  – Compare, process and return nodes in these sequences

• The deep set operator
  – Deep_Union, Deep_Intersect, Deep_Except
  – Operations are based on subtrees (nodes and descendants)
  – Take node sequences as operands
  – Compare and process nodes and their descendants (deep)

Union vs. Deep-Union

Intersection vs. Deep-Intersection

Except vs. Deep-Except

Examples

• XML fragment:
  
  ```xml
  <a>
    <b> <c/> </b>
    <d/>
  </a>
  <a>
    <b> <c/> </b>
    <d/>
  </a>,
  <b> <c/> </b>
  <a>
    <b> <c/> </b>
    <d/>
  </a>
  <a>
    <b> <c/> </b>
    <d/>
  </a>
  <b> <c/> </b>
  <a>
    <d/>
  </a>
  ```

Definitions

• First, we denote node sequences as
  
  \[ P = \{ p_1, \ldots, p_n \} \]
  \[ Q = \{ q_1, \ldots, q_m \} \]

• And the enumeration of the nodes and all their descendant nodes as:
  
  \[ P_d = P\text{descendant} - \text{or - self :: *} \]
  \[ Q_d = Q\text{descendant} - \text{or - self :: *} \]
Definitions: DEEP-UNION

Deep-union operator takes two node sequences \( P \) and \( Q \) as operands, and returns a sequence of nodes:

- (1) who exist as a node or as a descendant of the nodes in "either" operand sequences
- (2) whose parent does not satisfy (1).

Mathematically:
\[
P \text{ Deep-Union } Q = \{ r | (r \in P\cap Q) \land (r::parent() \notin P \land r::parent() \notin Q) \}
\]

Definitions: DEEP-INTERSECT

Deep-intersect operator takes two node sequences \( P \) and \( Q \) as operands, and returns a sequence of nodes:

- (1) who exist as a node or as a descendant of the nodes in "both" operand sequences
- (2) whose parent does not satisfy (1).

Mathematically:
\[
P \text{ Deep-Intersect } Q = \{ r | (r \in P\cap Q) \land (r::parent() \notin P \lor r::parent() \notin Q) \}
\]

Definitions: DEEP-EXCEPT

- Deep-except (ad-hoc):
  \( P \text{ Deep-Except } Q \)
  Return the subtrees of \( P \), with subtrees rooting at \( Q \) been removed from the answer.
- Deep-except constructs new nodes.

Conclusions and Discussions

- Each XML node is "a set of set(s)"
- Regular set operators only rely on node-IDs to identify and compare nodes.
  - They ignore the comparability issue of operands
  - They neglect the tree-structured hierarchy of XML data.
- Deep set operators recognize the ancestor-descendant relationships in the tree-structured hierarchy of XML.
  - Nodes in input node sequences are not regarded as atomic entities
  - They are treated as hierarchically nested elements.
  - Traverse into descendants of nodes to conduct comparison and processing, in a "deep" manner.