COMP4317: XML & Database
Tutorial 8: Streaming Xpath

Week 9

Thang Bui @ CSE.UNSW
Xpath Evaluation on Tree

- `void eval(exp, idx, node)`
  - Evaluate filters if any
  - If (`idx1 = slashslash(exp)`)
    - `eval(exp, idx1, node)`
  - If (`exp[idx] == node.tag`)
    - `isEnd(exp, idx)? => return node on result`
    - For any child `x` of `node`
      - `eval(exp, idx+1, x)"
Xpath Evaluation on Tree

- void eval(exp, idx, node)
  - Evaluate filters if any
  - If (idx1 = slashslash(exp))
    - eval(exp, idx1, node)
  - If (exp[idx] == node.tag)
    - isEnd(exp, idx)? => return node on result
    - For any child x of node
      - eval(exp, idx+1, x)
bool filter_eval(exp, idx, node)

- If exp[idx] == “following-sibling”
  - return filter_eval(exp, idx, following(node))

- If (exp[idx] == node.tag)
  - isEnd(exp, idx)? => return true
  - For any child x of node
    - filter_eval(exp, idx+1, x)

- If (idx1 = slashslash(exp))
  - filter_eval(exp, idx1, node)
Xpath Evaluation on Tree

- Travel the tree
  - eval(root)
- Print the result
  - printNodeList

- Disadvantages?
  - Travel a subtree many times
    - (filter, main expression, slashslash expressions)
Xpath Evaluation on Tree

- Multiple recursive function calling
- vs.
- Multiple comparison (on multiple rails)
Xpath Evaluation on Tree

- For each node
  - Evaluate filters if any
  - For each “active” rail r
    - If (idx1 = slashslash(r.exp))
      - Make a new rail r1.exp = r.exp, r1.idx = idx1
    - If (r.exp[r.idx] == node.tag)
      - isEnd(r.exp, r.idx)? => return node on result
      - r.idx++
Xpath Evaluation on Tree

- Active rail?
  - /a/b => level 0 & level 1 only
    - level > 1 => inactive
  - //a/b => level idx and idx+1 only
    - Rail 1: level 0 and 1
    - Rail 2: level 1 and 2 (dup.)
    - Rail 3: level 2 and 3 (dup. of rail 2)
  - e.t.c.
Xpath Evaluation on Tree

- Filter?
  - Filter rail
    - Stop at the first satisfied filter instance
      - Stop all other instance filter rail for the same subtree
    - Root node
  - End of a node?
    - All remained filter rails for that node are failed!
      - Remove a node from the candidate result list
Streaming Xpath Evaluation – Top-down evaluation

- **startElement**
  - For each active rail \( r \)
    - If (\( \text{idx1} = \text{slashslash}(\text{r.exp}) \))
      - Make a new rail \( \text{r1.exp} = \text{r.exp}, \text{r1.idx} = \text{idx1} \)
    - If (\( \text{r.exp}[\text{r.idx}] == \text{node.tag} \))
      - \( \text{isEnd}(\text{r.exp}, \text{r.idx})? \) => return node on result (building subtree from here!)
      - \( \text{r.idx}++ \)
Streaming XPath Evaluation – Bottom-up evaluation

- Reverse expression
- endElement
  - All nodes from the root to that node are on the stack (reverse order)
    - If exp is matched back to the root => result
      - Just like a string to other string!
Example

- **Expression:**
  - `/a//b/c`

- **Reverse:**
  - `c\b\\a\`
Bottom-up Evaluation

Ex:
- c\b\a\
Bottom-up Evaluation

Ex:
- $c\backslash b\backslash \backslash a\backslash$

Stack
Bottom-up Evaluation

Ex:
- `c\b\a`
Bottom-up Evaluation

Ex:
- c\b\a\
Bottom-up Evaluation

Ex:
- `c\b\a`
Bottom-up Evaluation

Ex:
\texttt{c\backslash b\backslash\backslash a\backslash}

Stack:

- c
- b
- c
- a
- m
- b
- a
- b
- a
- c
- b
- c
- d
- b
- c
- b
- c
- d