On the Correctness of Query Results in XML P2P Databases

Carlo Sartiani
Dipartimento di Informatica - Università di Pisa
(This work has been funded by the GRID.IT Project)
for $p$ in input()//article
  $a$ in $b$/author,
  $t$ in $b$/title
return <author-title>
  {$a,$t}
</author-title>

{\{l_2,l_{15},l_{18},l_{33},l_{87},l_{114},l_{911}\}}: complete & correct result

{\{l_2,l_{15},l_{18},l_{87},l_{911}\}}: incomplete but correct result
for $a$ in input()//author
let $p_list := for $p$ in input()//article
where $p$/author = $a$ AND
$p$/year = 2003
return {$p}$
where count($p_list$) < 5
return <subIudice> {$a} </subIudice>

{12,15,18,33,87,114,911}: complete & correct result

{12,15,18,87,911}: incomplete & maybe incorrect result
Objectives

to study the problem of correctness of query results in XML P2P databases in the presence of incomplete query plans

to identify classes of statically correct queries

correctness enforced at compile-time
Outline

- background
- the nature of the problem
- the results
- future work
Background
Background

- XQuery FLWR queries w/o universal quantification
- P2P system hosting heterogeneous XML data [1]
- full autonomy of peers
Query Processing

- node \( l_1 \) submits the query \( Q \) to the overlay network
- the overlay network returns a query plan (algebraic expression + location assignment)
- node \( l_1 \) executes \( Q \)
Query Processing Example

for $p$ in input()//article
    $a$ in $b/author,$
    $t$ in $b/title$
return <author-title>/{$a,$t}</author-title>

return_{author-title}[{$a,$t}]
path(//,$p,in)article[(/,$a,in)author[∅],(/,$t,in)title[∅]](·))

+ $ullet i=1,11,13,17 loc_i$
The Nature of the Problem
Issues

- location assignments can be incomplete
- sudden changes in the topology
- data structures not up-to-date
- queries executed on a subset of relevant nodes
Consequences

- Dangerous for set manipulations
- Set predicates
- Aggregation functions
- `count($p_list) < 5`
- Errors may propagate from inner queries to outer queries
Errors

- errors are generated by non-monotone operators of XQuery
- `where` + set predicates
- `return` + aggregation functions
- errors may propagate from inner queries to outer queries
Incompleteness Sources

- path expressions spanning on multiple sites
- unguarded path expressions
- `input()//book`
- nested queries
Results
Approach

- studying the monotonicity properties of XQuery
- locating constructs originating incompleteness
- finding query classes for which correctness can be statically enforced
- warnings from the compiler
- using syntactical information only
The image contains a diagram illustrating the relationship between different classes of FLWR queries without universal quantification. The diagram shows the following relationships:

1. **FLWR queries without universal quantification**
2. **No aggs & No set pred to incomplete variables (5.18)**
3. **Corollary (5.22)**
4. **No aggs & No set predicates (5.20)**
5. **For each (5.21)**
6. **Complete sets (5.17)**

The diagram visually represents how these classes are nested within each other, indicating the hierarchy and exclusions based on the absence of aggregation functions and set predicates, and the presence of complete sets.
Class 5.18

- no nested queries returning incorrect results
- no set predicates or aggregation functions applied to incomplete sets
- semi-decidable
Class 5.22

- no nested queries returning incorrect results
- no set predicates or aggregation functions applied to the result of the evaluation of nested queries and unguarded path expressions
- statically enforced
Future Work

- using type information for refining the classification [2]
- extending XQuery typechecking to P2P environments
Bibliography
