Containment of Partially Specified Tree-Pattern Queries

Dimitri Theodoratos (NJIT, USA)
Theodore Dalamagas (NTUA, GREECE)
Pawel Placek (NJIT, USA)
Stefanos Souldatos (NTUA, GREECE)
Timos Sellis (NTUA, GREECE)
Motivating example

- Consider a database including shops that sell spare parts for motorbikes.
- The authors of the paper need spare parts for their motorbikes.
- BUT…

```
root
  ├── GREECE
  │    ├── ATHENS
  │    │    ├── HONDA
  │    │    │    ├── TRAVEL
  │    │    │    │    └── VARADERO
  │    │    │    │        └── 125cc
  │    │    │    └── YAMAHA
  │    │    │        └── ON-OFF
  │    │    └── BMW
  │    │        └── TRAVEL
  │    └── 200cc
  └── USA
      ├── YAMAHA
      │    └── TRAVEL
      │        └── F650GS
      │            └── 650cc
      └── BMW
          └── ON-OFF
              └── TRAVEL
                  └── 200cc
```

Stefanos Souldatos - SSDBM 2006
Dimitri Theodoratos lives in NY. He owns a Yamaha Serrow motorbike in Greece. So, he searches for spare parts in Greece or USA.

A **structural difference** emerges…
Theodore Dalamagas owns a BMW motorbike and looks for spare parts worldwide. A structural inconsistency emerges…

650cc/F650GS vs F650GS/650cc
3: Unknown Structure

- Stefanos Souldatos owns a Honda Varadero.
- But, he is not fully aware of the hierarchy...
Pawel Placek wants to buy a motorbike that he can easily find spare parts for.

He searches in many hierarchies with different structure…
Motivation

- Querying tree-structured data requires exact knowledge of the structure.
- BUT, structure is not always strictly defined!

MOREOVER, the user needs to set queries without always dealing with structure:

- Find shops for Honda spare parts in Athens, Greece
- …where Athens is a descendant of Greece
- …but I don’t know if Athens is an ancestor or a descendant of Honda.
Previous Approaches

- **Keyword-based search approach**
  - Total absence of structure.

- **Approximation techniques**
  - Relax the initial query and retrieve more answers.

- **Naive approach**
  - All possible query patterns are generated and evaluated against the tree structure.

- **Traditional integration approach**
  - Global structure and mapping rules.
Our Approach

- We have designed a **Query Language** that allows for partial specification of the structure (**Partially Specified Tree-Pattern Queries**).
- Query formulation and evaluation is based on **Dimension Graphs** that summarize the structure of the trees.
- Dimension graphs group sets of semantically related nodes called **Dimensions**.

- We study the problem of **Query Containment** for Partially Specified Tree-Pattern Queries.
Data Model
Dimension graphs summarize the structure of the tree.

**DIMENSIONS**
- R (root)
- C (country) = \{GREECE, USA\}
- L (location)
- B (brand)
- T (type)
- M (model)
- E (engine)
Dimension Graphs...

- **Offer** a summary of the structure of the tree.
- **Provide** the necessary semantics for query formulation.
- **Set** the framework for querying sources with structural differences and inconsistencies.
- **Support** query evaluation and optimization.

**DIMENSIONS**

- R (oot)
- C (ountry)
- L (ocation)
- B (rand)
- T (ype)
- M (odel)
- E (ngine)
**Query**: Find shops with spare parts for all models and engines of **BMW** motorbikes in **Greece**.
**Partially Specified Tree-pattern Queries**

**Query:** Find shops with spare parts for all models and engines of BMW motorbikes in Greece.

### DIMENSIONS

- R (oot)
- C (ountry)
- L (ocation)
- B (rand)
- T (ype)
- M (odel)
- E (ngine)

**C = {Greece}**

**B = {BMW}**

**E = ?**

**M = ?**

**PSP p1**

**PSP *p2**

**Partially specified paths (PSP)**
Partially-Specified Tree-pattern Queries

**Query:** Find shops with spare parts for all models and engines of BMW motorbikes in Greece.

![Diagram showing partially specified paths (PSP) and output path (*) for finding shops with spare parts for BMW motorbikes in Greece.](image)
**Query**: Find shops with spare parts for all models and engines of BMW motorbikes in Greece.
**Query:** Find shops with spare parts for all models and engines of BMW motorbikes in Greece.
Dimension Trees and Query Evaluation
Dimension Trees

1) We apply Inference Rules to the query to produce its Full Form.

2) We search for paths in the graph that match the paths of the query.

3) Paths in the graph correspond to Dimension Trees.

4) Dimension trees are translated into XPath queries.
1) We apply Inference Rules to the query to produce its **Full Form**.

2) We search for **paths in the graph** that match the paths of the query.

3) Paths in the graph correspond to **Dimension Trees**.

4) Dimension trees are translated into **XPath queries**.
1) We apply Inference Rules to the query to produce its **Full Form**.

2) We search for paths in the graph that match the paths of the query.

Paths in the graph correspond to **Dimension Trees**.

3) Dimension trees are translated into **XPath queries**.
Dimension Trees

1) We apply Inference Rules to the query to produce its Full Form.
2) We search for paths in the graph that match the paths of the query.
3) Paths in the graph correspond to Dimension Trees.
4) Dimension trees are translated into XPath queries.
Dimension Trees

So, Dimension Trees are equivalent to the Query in a specific Dimension Graph:

“Dimension Trees = Query + Graph”
Query Containment
Query Containment

- **Absolute Containment**
  - Query Q1 is absolutely contained in query Q2 if the answers of Q1 in every dimension graph are also answers of Q2.

- **Relative Containment (with respect to graph)**
  - Query Q1 is contained in query Q2 with respect to graph G if the answers of Q1 in G are also answers of Q2 in G.
Query Q1 is absolutely contained in query Q2 if there is a mapping from Q2 to Q1.
Query Q1 is contained in query Q2 with respect to graph G if every dimension tree of Q1 is contained in a dimension tree of Q2.

A dimension tree of Q1 \(\subseteq\) A dimension tree of Q2
Absolute & Relative Containment

- Checking Absolute Containment:
  - 1msec

- Checking Relative Containment:
  - 100msec – 1sec

- We suggest a technique that improves the performance of Relative Containment check. (Relative Containment 2 – RC2)
- It is a sound but not complete technique
- However, it maintains high accuracy for containment check.
Relative Containment 2 (RC2)

For each possible parent-child or ancestor-descendant relation in the query, we extract relations that hold in all graph paths that include it:

- \( R \Rightarrow C \) : \( R \rightarrow C \)
- \( L \Rightarrow B \) : \( R \rightarrow C, \ C \rightarrow L \)

We add these relations in \( Q_1 \) and check absolutely whether \( Q_1' \subseteq Q_2 \).
Example

- Q1 is NOT absolutely contained in Q2.
- But, Q1 is relatively contained in Q2.
Example

- Q1 is NOT absolutely contained to Q2.
- But, Q1 is relatively contained to Q2.

Relative Containment 2

\[ \text{L} \rightarrow \text{B} : \text{R} \rightarrow \text{C}, \text{C} \rightarrow \text{L} \]

\[ \text{Q1} \subseteq \text{G} \]

\[ \text{Q2} \]

\[ \text{PSP} \ast p1 \]

\[ \text{PSP} \ast p2 \]
Experiments
Experiments

- We measured…
  - execution time for checking Absolute Containment, Relative Containment and Relative Containment 2 (RC2)
  - the accuracy of RC2

- …varying
  - the density of the dimension graphs (dimensions, root-to-leaf paths)
  - the size of the queries (PSPs, nodes per PSP)
Varying Graphs

30 dimensions

RC2 accuracy
> 80%

RC2 accuracy
> 50%

Relative Containment

Relative Containment 2

Absolute Containment

graph paths

time (sec)

> 80%

> 50%
Varying Queries

2 PSPs

RC2 accuracy > 93%

Relative Containment

Relative Containment 2

Absolute Containment

time (sec)

dimensions per PSP
Conclusion

- We addressed the problem of **Query Containment** for Partially Specified Tree-Pattern Queries (PSTPQs).

- We suggested a **sound technique** for checking Relative Query Containment.

- **Experiments** showed that the technique improves checking of relative containment
  - one order of magnitude
  - with accuracy over 80% for graphs of common sizes (based on XML benchmarks e.g. XMark, XMach etc).
Future Work

- Suggest and experiment with a set of heuristics for checking Relative Query Containment.
- Discuss the trade-off between time and accuracy for each of those heuristics.
Questions?
Links

- Introduction (2-10)
- Data Model (11-18)
- Query Evaluation (19-24)
- Query Containment (25-32)
- Experiments (33-36)
- Conclusion – Future Work (37-39)
Appendix
Dimension Trees

\[ R \rightarrow C \rightarrow L \rightarrow E \rightarrow M \]

\[ R \rightarrow C = \{\text{Greece}\} \rightarrow B = \{\text{BMW}\} \rightarrow T \rightarrow E \rightarrow M \]

\[ r / \text{Greece} / \text{BMW} / \]

\[ r / \text{Greece} / \text{BMW} / *_{T}[*_{E}]/*_{M} \]

\[ r / \text{Greece} / \text{BMW} / *_{T}[*_{M}/*_{E}]/*_{E} *_{M} \]

\[ r / \text{Greece} / \text{BMW} / *_{T}/*_{M}[*_{E}] \]

\[ r / \text{Greece} / \text{BMW} / *_{T}/*_{E}/*_{M} \]

\[ C = \{\text{Greece}\} \rightarrow B = \{\text{BMW}\} \rightarrow E = ? \rightarrow M = ? \]

\[ \text{PSP } p_1 \]

\[ \text{PSP } *p_2 \]