RDF-Constraints and SPARQL

Thomas Hornung and Michael Meier, 15.04.2008

joint work with Georg Lausen, Norbert Küchlin, and Michael Schmidt
Table of Contents

• Part One:  Constraints
  – Encoding Constraints in RDF
  – Checking Constraints

• Part Two:  SPARQL Performance Benchmark
  – RDF Data Generator & 16 Queries
  – Evaluation Results
Motivation

• Export relational databases to RDF
  – Data consistency
  – Updates
An example
An example

No representation of key and foreign key constraints.
An example

Encoding in the same graph as the data via a new namespace rdfc.
An example

FK: inProject(Pid) → Project(Pid)
Checking a foreign key

ASK {
  ?x rdf:type inProject; Pid ?Pid.
  OPTIONAL {
    ?y rdf:type Project; Pid ?Pid.
  } FILTER (!bound(?y))
}
Part Two: SPARQL Performance Benchmark
Motivation

• Up-to-date no benchmark for SPARQL has been proposed
• SP²B fills this gap
  – Settled in the DBLP scenario
  – Data generator for creating arbitrarily large datasets + 16 benchmark queries
The SP²Bench Data Generator

• Creates bibliography documents similar to DBLP in a deterministic and incremental way
• Mirrors vital key characteristics found in original DBLP data
  – Structure of entities (Articles, Journals, Books, ...)
  – Relations between authors
  – Quantity of entities (development over time)
  – Citation system
The SP²Bench Queries

• 14 SELECT and 2 ASK Queries
• Different SPARQL solution modifiers
• Closed World Negation
• (Comparably) long predicate chains and bushy patterns
• Amenable to a variety of SPARQL optimizations
• Varying selectivity, output size, etc.
Benchmark Results

- Considered SPARQL Engines:
  - ARQ (memory/SDB)
  - Sesame (memory/native)
  - Virtuoso
  - Redland

- Findings:
  - Significant differences between engines
  - Room for improvement in current implementation
  - Poor support for several SPARQL specifics
Current Focus

• Benchmarking of RDF stores on top of relational DBMS:
  – Oracle
  – MySQL
  – MonetDB

• Different storage schemes:
  – Triple tables
  – Property tables
  – Vertical partitioning  Abadi et. al. (VLDB 2007)
Initial Results

• Column-oriented DBMS are clearly preferable to row-store DBMS
• Vertical partitioning does not solve general problems
• General drawbacks of relational approach:
  – Selectivity estimation is often suboptimal
  – Translating more complex patterns, such as CWN, results in queries with left joins and filters
Part Three:
Additional Resources
The SP$^2$Bench Data Generator

Distribution of Citations

Distribution of Publications
The DBLP RDF Schema
Benchmark Queries

SELECT ?yr
WHERE {
}

• Simple
• Constant result size (exactly 1 result)
• Might be answered very fast with index
Benchmark Queries

Q5a

SELECT DISTINCT ?person ?name Q5
WHERE {
  ?article rdf:type bench:Article.
  ?article dc:creator ?person.
  ?person foaf:name ?name.
  FILTER(?name=?name2).
}

Q5b

SELECT DISTINCT ?person ?name
WHERE {
  ?article rdf:type bench:Article.
  ?article dc:creator ?person.
  ?person foaf:name ?name.
}

- Equivalent in our scenario
- Tests implicit vs. explicit joins
- We found that Q5a is much more challenging for current engines
Benchmark Queries

SELECT ?yr ?name ?doc
WHERE {
  ?class rdfs:subClassOf foaf:Document.
  ?author foaf:name ?name.
  OPTIONAL {
    ?class2 rdfs:subClassOf foaf:Document.
  FILTER(!bound(?author2)). }

• Closed-World-Negation

• Returns, for each year, the set of all publications authored by persons that have not published in years before
Benchmark Query (Triples)

(a) SELECT DISTINCT T2.obj AS person,
    T3.obj AS name
FROM Triples T1
    JOIN Triples T2 ON T1.subj=T2.subj
    JOIN Triples T3 ON T2.obj=T3.subj,
    Triples T4
    JOIN Triples T5 ON T4.subj=T5.subj
    JOIN Triples T6 ON T5.obj=T6.subj
WHERE T1.pred='rdf:type'
    AND T2.pred='dc:creator' AND T3.pred='foaf:name'
    AND T4.pred='rdf:type'    AND T5.pred='dc:creator'
    AND T6.pred='foaf:name'   AND T1.obj='bench:Article'
    AND T4.obj='bench:Inproceedings' AND T3.obj=T6.obj

(b) SELECT DISTINCT T2.obj AS person,
    T5.obj AS name
FROM Triples T1
    JOIN Triples T2 ON T1.subj=T2.subj
    JOIN Triples T3 ON T2.obj=T3.obj
    JOIN Triples T4 ON T3.subj=T4.subj
    JOIN Triples T5 ON T3.obj=T5.subj
WHERE T1.pred='rdf:type'    AND T2.pred='dc:creator'
    AND T3.pred='dc:creator' AND T4.pred='rdf:type'
    AND T5.pred='foaf:name'   AND T1.obj='bench:Article'
    AND T4.obj='bench:Inproceedings'
<table>
<thead>
<tr>
<th>Subject</th>
<th>Predicate</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>foaf:name</td>
<td>Alice</td>
</tr>
<tr>
<td>s1</td>
<td>foaf:mbox</td>
<td><a href="mailto:alice@gmx.net">alice@gmx.net</a></td>
</tr>
<tr>
<td>s2</td>
<td>foaf:name</td>
<td>Bob</td>
</tr>
<tr>
<td>s2</td>
<td>foaf:mbox</td>
<td><a href="mailto:bob@web.de">bob@web.de</a></td>
</tr>
</tbody>
</table>
## Partition Table

<table>
<thead>
<tr>
<th>Subject</th>
<th>foaf_name</th>
<th>foaf_mbox</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>Alice</td>
<td><a href="mailto:alice@gmx.net">alice@gmx.net</a></td>
</tr>
<tr>
<td>s2</td>
<td>Bob</td>
<td><a href="mailto:bob@web.de">bob@web.de</a></td>
</tr>
</tbody>
</table>
Vertical Partitioning

<table>
<thead>
<tr>
<th>Subject</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>Alice</td>
</tr>
<tr>
<td>s2</td>
<td>Bob</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Subject</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td><a href="mailto:alice@gmx.net">alice@gmx.net</a></td>
</tr>
<tr>
<td>s2</td>
<td><a href="mailto:bob@web.de">bob@web.de</a></td>
</tr>
</tbody>
</table>
More Constraints

• Let $C, C_1, C_2$ be classes and $Q_i, R_i$ properties

  – **Primary Keys:** $\text{Key}(C,[Q_1,...Q_n])$
  – **Foreign Keys:** $\text{FKey}(C_1,[Q_1,...Q_n],C_2,[R_1,...R_n])$
  – **Cardinality Constraints:** $\text{Min}(C,n,R), \text{Max}(C,n,R)$ for $n \in \mathbb{N}$
  – **Singleton Constraints:** $\text{Single}(C)$
  – **Subclass Constraint:** $\text{SubC}(C_1,C_2)$
  – **Subproperty Constraint:** $\text{SubP}(Q_1,Q_2)$
  – **Property Domain/Range:** $\text{PropD}(Q,C), \text{PropR}(Q,C)$
Satisfiability

Given a set of constraints is there a non-empty RDF graph that satisfies the constraints?
Satisfiability

- Primary keys + Foreign Keys
- Singleton
- Max-Cardinality
- Subclass + Subproperty
- Property Domain + Property Range
- Min-Cardinality

undecidable
Satisfiability

- Unary primary keys
- Unary foreign keys
- Min-Cardinality + Max-Cardinality
- Subclass + Subproperty
- Property Domain + Property Range

EXPTIME