Deriving an Emergent Relational Schema from RDF Data

Minh-Duc Pham  
duc@cw.nl

Linnea Passing  
linnea.passing@tum.de

Orri Erling  
oerling@openlinksw.com

Peter Boncz  
boncz@cw.nl
Main Problems in RDF Data Management [1]

- Bad query plans
- Low storage locality
- Lack of user schema insight

RDF de-emphasizes the need for a schema and the notion of structure in the data

Emergent schema = “rough” schema to which the majority of triples conforms
Recognize:
- **Classes** (Characteristic Sets - CS’s) – recognize “classes” of often co-occurring properties
- **Relationships** (CS) – recognize often-occurring references between such classes
  + give logical names to these

“Book”

“Author”

```xml
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://rdfs.org/sioc/ns#num_replies>
<http://purl.org/dc/terms/title>
<http://rdfs.org/sioc/ns#has_creator>
<http://purl.org/dc/terms/date>
<http://purl.org/dc/terms/created>
<http://purl.org/rss/1.0/modules/content/encoded>
```

```xml
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://xmlns.com/foaf/0.1/name>
<http://xmlns.com/foaf/0.1/page>
```
Foreign Key Relationship

<table>
<thead>
<tr>
<th>ID</th>
<th>type</th>
<th>creator</th>
<th>title</th>
<th>partOf</th>
</tr>
</thead>
<tbody>
<tr>
<td>inproc1</td>
<td>inproceeding</td>
<td>{author3, author4}</td>
<td>&quot;AAA&quot;</td>
<td>conf1</td>
</tr>
<tr>
<td>inproc2</td>
<td>inproceeding</td>
<td>author2</td>
<td>&quot;BBB&quot;</td>
<td>conf1</td>
</tr>
<tr>
<td>inproc3</td>
<td>inproceeding</td>
<td>author3</td>
<td>&quot;CCC&quot;</td>
<td>conf2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>type</th>
<th>title</th>
<th>issued</th>
</tr>
</thead>
<tbody>
<tr>
<td>conf1</td>
<td>Conference</td>
<td>&quot;conference1&quot;</td>
<td>2010</td>
</tr>
<tr>
<td>conf2</td>
<td>Proceedings</td>
<td>&quot;conference2&quot;</td>
<td>2011</td>
</tr>
</tbody>
</table>

Irregularity

Example of structure recognized from RDF graph
What does “schema” mean?

Relational Schema

- Describes the structure of the occurring data
- Concept mixing (for convenience)
- Designed for one database (=dataset)

Semantic Web Schema

- Purpose: knowledge representation
- Describing a concept universe (regardless data)
- Designed for interoperability in many contexts

Statement: it is useful to have both an (Emergent) Relational and Semantic Schema for RDF data
- useful for systems (higher efficiency)
- useful for humans (easier query formulation)
When is a **Emergent Schema** of RDF data useful?

- **Compact** Schema
  - as few tables as possible
  - homogeneous literal types (few NULLs in the tables)
- **Human-friendly** “Labels”
  - URIs + human-understandable table/column/relationship names
- **High “Coverage”**
  - the schema should match almost all triples in the dataset
- **Efficient** to compute
  - as fast as data import
Basic CS discovery

(\texttt{s1, offers, offer1})
(\texttt{s1, region, region1})
(\texttt{s2, offers, offer2})
(\texttt{s2, offers, offer3})
(\texttt{s2, region, region1})

... 
(\texttt{offer1, availableDeliveryMethods, DHL})
(\texttt{offer1, description, \textit{“Offer data”}})
(\texttt{offer1, hasBusinessFunction, \textit{“Sell”}})
(\texttt{offer1, hasEligibleQuantity, 1})
(\texttt{offer1, hasInventoryLevel, 1})
(\texttt{offer1, hasStockKeepingUnit, 112})
(\texttt{offer2, availableDeliveryMethods, DHL})
(\texttt{offer2, hasPriceSpec, price1})
(\texttt{offer2, hasStockKeepingUnit, 112})
(\texttt{offer2, type, Offering})

... 
(\texttt{price1, hasCurrency, \textit{“EUR”}})
(\texttt{price1, hasCurrencyValue, \textit{“35.99”}})
(\texttt{price1, hasUnitOfMeasurement, \textit{“€/kg”}})
(\texttt{price1, valueAddedTaxIncluded, \textit{“false”}})
(\texttt{price1, eligibleTransactionVolume, 0})
(\texttt{price1, ...}

... <Example RDF triples>
## Characteristic Sets in some well-known RDF datasets

<table>
<thead>
<tr>
<th>Datasets</th>
<th>#triples*</th>
<th>#CS’s</th>
<th>#CS’s to cover 90%</th>
<th>Avg. #prop.</th>
<th>#multi-type properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>LUBM</td>
<td>100M</td>
<td>17</td>
<td>7</td>
<td>5.71</td>
<td>0</td>
</tr>
<tr>
<td>BSBM</td>
<td>100M</td>
<td>49</td>
<td>14</td>
<td>12.61</td>
<td>0</td>
</tr>
<tr>
<td>SP2Bench</td>
<td>100M</td>
<td>554</td>
<td>7</td>
<td>9.8</td>
<td>0</td>
</tr>
<tr>
<td>synthetic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MusicBrainz</td>
<td>179M</td>
<td>27</td>
<td>10</td>
<td>4.7</td>
<td>0</td>
</tr>
<tr>
<td>EuroStat</td>
<td>70K</td>
<td>44</td>
<td>8</td>
<td>7.77</td>
<td>0</td>
</tr>
<tr>
<td>DBLP</td>
<td>56M</td>
<td>249</td>
<td>8</td>
<td>13.70</td>
<td>0</td>
</tr>
<tr>
<td>PubMed</td>
<td>1.82B</td>
<td>3340</td>
<td>35</td>
<td>19.27</td>
<td>0</td>
</tr>
<tr>
<td>relational</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WebData.</td>
<td>90M</td>
<td>13354</td>
<td>930</td>
<td>7.94</td>
<td>551</td>
</tr>
<tr>
<td>DBpedia</td>
<td>404M</td>
<td>439629</td>
<td>85922</td>
<td>24.36</td>
<td>1507</td>
</tr>
<tr>
<td>native</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Note: #triples* refers to the number of triples in the dataset.*

*Data created by benchmark data generator*

*RDF data from a relational database dump*

*real data originating as RDF*
Partial and Mixed Use of Ontologies

| Dataset       | mixed
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>LUBM</td>
<td>1.94</td>
</tr>
<tr>
<td>BSBM</td>
<td>3.96</td>
</tr>
<tr>
<td>SP2Bench</td>
<td>4.94</td>
</tr>
<tr>
<td>MusicBrainz</td>
<td>3.93</td>
</tr>
<tr>
<td>EuroStat</td>
<td>3.14</td>
</tr>
<tr>
<td>DBLP</td>
<td>6.58</td>
</tr>
<tr>
<td>PubMed</td>
<td>4.94</td>
</tr>
<tr>
<td>WebData.</td>
<td>2.27</td>
</tr>
<tr>
<td>DBpedia</td>
<td>8.35</td>
</tr>
</tbody>
</table>

\[
\text{cs}_4 \quad \begin{align*}
\text{dc:}&\text{description} \\
\text{gor:}&\text{validFrom} \\
\text{gor:}&\text{validThrough} \\
\text{gor:}&\text{hasCurrency} \\
\text{gor:}&\text{hasCurrencyValue} \\
\text{gor:}&\text{hasUnitOfMeasurement} \\
\text{gor:}&\text{valueAddedTaxIncluded} \\
\text{gor:}&\text{eligibleTransactionVolume}
\end{align*}
\]

\[
\text{PriceSpecification} \quad \begin{align*}
\text{gor:}&\text{description} \\
\text{gor:}&\text{name} \\
\text{gor:}&\text{eligibleTransactionVolume} \\
\text{gor:}&\text{validFrom} \\
\text{gor:}&\text{validThrough} \\
\text{gor:}&\text{hasCurrency} \\
\text{gor:}&\text{hasCurrencyValue} \\
\text{gor:}&\text{hasUnitOfMeasurement} \\
\text{gor:}&\text{valueAddedTaxIncluded} \\
\text{gor:}&\text{hasMaxCurrencyValue} \\
\text{gor:}&\text{hasMinCurrencyValue}
\end{align*}
\]

\[
\text{prefix gor:} \quad \text{http://purl.org/goodrelations/v1#} \\
\text{prefix dc:} \quad \text{http://purl.org/dc/elements/1.1/}
\]
Emerging a Relational Schema

1. Extract basic CS's

2. Labeling

3. Merge similar CS's

4. Schema Filtering

5. Instance Filtering

Physical relational schema

Triple table

Parameter Tuning

Basic CS's

Merged CS's
Results: compact schemas with high coverage

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Number of tables</th>
<th>Coverage – Metric $C$ (%)</th>
<th>final schema</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>before merging</td>
<td>after merging</td>
<td>remove small tables</td>
</tr>
<tr>
<td>LUBM</td>
<td>17</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td>BSBM</td>
<td>49</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>SP2B</td>
<td>554</td>
<td>13</td>
<td>10</td>
</tr>
<tr>
<td>MusicBrainz</td>
<td>27</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>EuroStat</td>
<td>44</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>DBLP</td>
<td>249</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>PubMed</td>
<td>3340</td>
<td>14</td>
<td>12</td>
</tr>
<tr>
<td>WebData.</td>
<td>13354</td>
<td>3000</td>
<td>253</td>
</tr>
<tr>
<td>DBpedia</td>
<td>439629</td>
<td>542</td>
<td>234</td>
</tr>
</tbody>
</table>
Results: understandable labels & performance

<table>
<thead>
<tr>
<th>RDF Store</th>
<th>Query 3</th>
<th></th>
<th></th>
<th>Query 5</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Cold</td>
<td>Hot</td>
<td>Opt. Time</td>
<td>Cold</td>
<td>Hot</td>
<td>Opt. Time</td>
</tr>
<tr>
<td>Virt-Quad</td>
<td>4210</td>
<td>53</td>
<td>40.2</td>
<td>3842</td>
<td>1350</td>
<td>18.6</td>
</tr>
<tr>
<td>Virt-CS</td>
<td>2965</td>
<td>9</td>
<td>5.4</td>
<td>2130</td>
<td>712</td>
<td>4.2</td>
</tr>
</tbody>
</table>

Table 3: Human survey results on Likert scale

Likert Score: 1=bad ..... 5=excellent

Table 5: Query time (msecs) w/wo the recognized schema
(Cold: First query runtime after re-starting the server
Hot : Run the query 3 times after and get the last runtime
Opt. Time: Query optimization time)
Thank You!