Foundations of RDF Databases
(Overview)

Claudio Gutierrez

Department of Computer Science
Universidad de Chile

UPM - Madrid - Enero 2009
Joint Work With

- Renzo Angles
- Marcelo Arenas
- Carlos Hurtado
- Sergio Muñoz
- Jorge Pérez
To the memory of Alberto Mendelzon, database theoretician and Web enthusiast
1. RDF and Databases
2. RDF and Database models
3. RDF Query Language
   - Requirements and Domains
   - Manifold Views
4. SPARQL
   - Syntax and Semantics
   - Complexity
   - Expressive Power
1. RDF and Databases
2. RDF and Database models
3. RDF Query Language
   - Requirements and Domains
   - Manifold Views
4. SPARQL
Disclaimer

A particular view on the subject

Not a survey!
The base of the Semantic Web is RDF

“The **Semantic Web** is the representation of **data** on the World Wide Web. It is a collaborative effort led by W3C with participation from a large number of researchers and industrial partners. **It is based on the Resource Description Framework (RDF)**”

http://www.w3.org/2001/sw/
RDF Recommendation (1999)

Language for representing information about resources in the Web

Particularly metadata about Web resources

Automation of processing: "RDF is intended for situations in which this information needs to be processed by applications, rather than only displayed to people."

C. Gutierrez – Foundations of RDF Databases
Layers of the Semantic Web

- Self-desc. doc.
- Data
- Rules
- Logic
- Ontology vocabulary
- RDF + rdfschema
- XML + NS + xmscschema
- Unicode
- URI

C. Gutierrez – Foundations of RDF Databases
A Data Processing perspective

Logic + Ontology vocabulary
(Concepts + knowledge)
∀x ∃y (R(x, y) → ∃z Q(z))
A ∪ B, C → D ∩ E

RDF + rdfschema
(entities + relations)

XML + NS + xsmlschema
(Text + Links)

Digital Signature

Trust

Proof

KR + Logic

Databases

Information Retrieval

Unicode

C. Gutierrez – Foundations of RDF Databases
The Database Approach

- Manage huge volumes of data with logical precision
- Separate modeling from implementation levels
The Database Approach

- Manage huge volumes of data with logical precision
- Separate modeling from implementation levels

As opposed to AI: DB primary concern is scalability. Then expressive power
The Database Approach

- Manage huge volumes of data with logical precision
- Separate modeling from implementation levels

As opposed to AI: DB primary concern is scalability. Then expressive power.
As opposed to IR: DB primary concern is precision. Then scalability (recall).
This Course: Database Modeling Level

Data Structure: RDF Graphs

Query language

Native Data Store

Files

RDBMS

Applications

Services

APIs

C. Gutierrez – Foundations of RDF Databases
Hence leaving out:

- Visualization, APIs, Services, etc.
- Indexing, storing, transactions, etc.
Hence leaving out:

- Visualization, APIs, Services, etc.
- Indexing, storing, transactions, etc.

But also leaving out:

- Updating / Constraints / Temporality / Optimization / Aggregation / Flexibility / etc. / etc.
1. RDF and Databases
2. RDF and Database models
3. RDF Query Language
   - Requirements and Domains
   - Manifold Views
4. SPARQL
Database Models: Codd’s definition

- Query Language
- Integrity constraints
- Data structures
Database Models: Codd’s definition

Query Language

Data structures
Evolution of Database Models

C. Gutierrez – Foundations of RDF Databases
RDF Data Structure: three main blocks

- RDFS Vocabulary
  - class
  - property
  - subClass
  - type
  - range
  - domain

- Blank Nodes

- Graph (Triple) structure

C. Gutierrez – Foundations of RDF Databases
RDF Data Structure: the core
RDF Data Structure: the core

**Triple structure:** set of statements
RDF Data Structure: the core

**Graph structure:** linked network of statements.

**Triple structure:** set of statements

Graph (Triple) structure
RDF Data Structure: Relational Tables (Triple) view

- Triples as tuples
- Set of triples as Tables

<table>
<thead>
<tr>
<th>Subject</th>
<th>Predicate</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
RDF Data Structure: Relational Tables (Triple) view

- Triples as tuples
- Tables of triples

**Advantages:**

+ Well studied and well understood
+ Reuse relational technologies
RDF Data Structure: Relational Tables (Triple) view

- Triples as tuples
- Tables of triples

Advantages:
+ Well studied and well understood
+ Reuse relational technologies

Problems (Questions):
- Why yet another syntax for the relational model?
- Was this the intended objective of RDF?
- Expressive power limitations
Graph Database Models:

- Data and/or schema are represented by graphs
- Query language able to capture main graph operations and properties
- Studied by DB community, but still not well understood
Graph Database Models

- Data and/or schema are represented by graphs
- Query language able to capture main graph operations and properties
- Studied by DB community, but still not well understood
RDF Data Structure: Graph query languages

<table>
<thead>
<tr>
<th>PROPERTY</th>
<th>Neighbors</th>
<th>Adjacent Edges</th>
<th>Degree of a Node</th>
<th>Path</th>
<th>Fixed-length path</th>
<th>Distance</th>
<th>Diameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Query Language</td>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>G+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Graph Log</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gram</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Graph DB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lorel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>F-G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
RDF Data Structure: Graph query languages

<table>
<thead>
<tr>
<th>Graph Query Language</th>
<th>Neighbourhoods</th>
<th>Adjacent Edges</th>
<th>Degree of Node</th>
<th>Path</th>
<th>Fixed-length path</th>
<th>Distance</th>
<th>Diameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Graph Log</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gram</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Graph DB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lorel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F-G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Green light for graph features!
RDF Data Structure: Triple structure + Blank nodes

Blank Nodes

Graph (Triple) structure

Vocabulary

C. Gutierrez – Foundations of RDF Databases
RDF Data Structure: Triple structure + Blank nodes

Complexity / Semantics issues:
• Deciding entailment becomes NP-complete.
• Deciding core is DP-complete
• Semantics of querying not simple
RDF Data Structure: Ground fragment

Vocabulary

Graph (Triple) structure

Blank Nodes
RDF Data Structure: Ground fragment

Good News: Blank nodes can be treated orthogonally to ground fragment.
More good news:

- Vocabulary can be *reduced* to

\{ type, domain, range, subClassOf, subPropertyOf \}
More good news:

• Vocabulary can be reduced to
  \{ \text{type, domain, range, subClassOf, subPropertyOf} \}

• Complex semantic rules and axioms can be avoided
More good news:

- Vocabulary can be *reduced* to { type, domain, range, subClassOf, subPropertyOf }
- Complex semantic rules and axioms can be *avoided*
- Structural (internal) constraints of the language can be *separated* from user-features.
  e.g. (Class, type, Resource)
More good news:

- Vocabulary can be *reduced* to
  \{ type, domain, range, subClassOf, subPropertyOf \}
- Complex semantic rules and axioms can be *avoided*
- Structural (internal) constraints of the language can be *separated* from user-features.
  e.g. (Class, type, Resource)
- Features which do not add expressive power can be *avoided*, e.g. reflexivity of subClass and subProperty.
RDF Data Structure: A *minimal fragment*

\{subClass, subProperty, type, domain, range\}
Theorem: Simple proof system sound and complete for the semantics of RDF in this fragment. That is:

\[ G \models F \text{ under RDF semantics iff } G \models F \text{ under mRDF semantics} \]
RDF Data Structure: A minimal fragment

{subClass, subProperty, type, domain, range}

**Theorem:** Simple proof system sound and complete for the semantics of RDF in this fragment. That is:

\[ G \models F \text{ under RDF semantics iff } G \models F \text{ under mRDF semantics} \]

**Theorem:** Let \( G \) be a restricted graph in the fragment, and \( t \) a ground tuple. Deciding if \( G \models t \) can be done in time \( O(G \times \log(G)) \)
1. RDF and Databases
2. RDF and Database models
3. RDF Query Language
   – Requirements and Domains
   – Manifold Views
4. SPARQL
### RDF Query language: Social Networks domain

<table>
<thead>
<tr>
<th>Chapter title</th>
<th>Use Case (local)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Looking for Social Structure</td>
<td>+ Directed to undirected binary relations</td>
</tr>
<tr>
<td></td>
<td>+ Remove relations</td>
</tr>
<tr>
<td>Attributes and Relations</td>
<td>+ Extract a subnetwork based on attributes</td>
</tr>
<tr>
<td></td>
<td>+ Group actors based on attributes</td>
</tr>
<tr>
<td></td>
<td>+ Selective grouping of actors based on attributes</td>
</tr>
<tr>
<td>Cohesive Subgroups</td>
<td>+ Extract the subnetwork induced by cliques of size n</td>
</tr>
<tr>
<td></td>
<td>+ Build a hierarchy of cliques</td>
</tr>
<tr>
<td>Frienship</td>
<td>+ Extract subnetwork by time</td>
</tr>
<tr>
<td>Affiliations</td>
<td>+ Two-mode network to one-mode network</td>
</tr>
<tr>
<td>Center and Periphery</td>
<td>+ Group multiple binary relations</td>
</tr>
<tr>
<td>Brokers and Bridges</td>
<td>+ Extract egonetwork of an actor</td>
</tr>
<tr>
<td></td>
<td>+ Remove relations between groups</td>
</tr>
<tr>
<td>Diffusion</td>
<td>+ Selective counting of neighbors</td>
</tr>
<tr>
<td></td>
<td>+ Operations between attributes</td>
</tr>
<tr>
<td></td>
<td>+ Change relation direction based on attributes</td>
</tr>
<tr>
<td>Prestige</td>
<td>+ Discretize an attribute</td>
</tr>
<tr>
<td>Ranking</td>
<td>+ Find triads by type</td>
</tr>
<tr>
<td>Genealogies and Citations</td>
<td>+ Loop removal</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Subgraph family</th>
<th>Use Case (Global)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paths and Cycles</td>
<td>+ Geodesics</td>
</tr>
<tr>
<td>Groups</td>
<td>+ Detect cohesive subgroups</td>
</tr>
<tr>
<td></td>
<td>+ Egonetworks</td>
</tr>
<tr>
<td></td>
<td>+ Input Domain</td>
</tr>
<tr>
<td>Connected components</td>
<td>+ Connected components</td>
</tr>
<tr>
<td></td>
<td>+ Clustering</td>
</tr>
<tr>
<td></td>
<td>+ Bicomponents and brockers</td>
</tr>
</tbody>
</table>

C. Gutierrez – Foundations of RDF Databases
### RDF Query Language: Biology domain

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Graph Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chemical structure associated with a node</td>
<td>Node matching</td>
</tr>
<tr>
<td>Find the difference in metabolisms between two microbes</td>
<td>Graph intersection, union, difference</td>
</tr>
<tr>
<td>To combine multiple protein interaction graphs</td>
<td>Majority graph query</td>
</tr>
<tr>
<td>To construct pathways from individual reactions</td>
<td>Graph composition</td>
</tr>
<tr>
<td>To connect pathways, metabolism of co-existing organisms</td>
<td>Graph composition</td>
</tr>
<tr>
<td>Identify “important” paths from nutrients to chemical outputs</td>
<td>Shortest path queries</td>
</tr>
<tr>
<td>Find all products ultimately derived from a particular reaction</td>
<td>Transitive Closure</td>
</tr>
<tr>
<td>Observe multiple products are co-regulated</td>
<td>Least common ancestor</td>
</tr>
<tr>
<td>To find biopathways graph motifs</td>
<td>Frequent subgraph recognition</td>
</tr>
<tr>
<td>Chemical info retrieval</td>
<td>Subgraph isomorphism</td>
</tr>
<tr>
<td>Kinase enzyme</td>
<td>Subgraph homomorphism</td>
</tr>
<tr>
<td>Enzyme taxonomies</td>
<td>Subsumption testing</td>
</tr>
<tr>
<td>To find biopathways graph motifs</td>
<td>Frequent subgraph recognition</td>
</tr>
</tbody>
</table>
## RDF Query Language: Web domain

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Graph Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is/are the most cited paper/s?</td>
<td>Degree of a node</td>
</tr>
<tr>
<td>What is the influence of article D?</td>
<td>Paths</td>
</tr>
<tr>
<td>What is the Erdös distance between authors X and author Y?</td>
<td>Distance</td>
</tr>
<tr>
<td>Are suspects A and B related?</td>
<td>Paths</td>
</tr>
<tr>
<td>All relatives of degree one of Alice</td>
<td>Adjacency</td>
</tr>
</tbody>
</table>

### Friend Of A Friend (FOAF)

```
foaf:Person
foaf:name
foaf:mbox
foaf:Knows

foaf:Person
foaf:name
foaf:mbox

mailto:mm@example.com
mailto:dd@example.com
```

---

C. Gutierrez – Foundations of RDF Databases
Tags
A tag is simply a word you use to describe a bookmark. Unlike folders, you make up tags when you need them and you can use as many as you like.

Minimalist design:
- Tags + Bundles (classes)
- No inheritance, no intersection, etc.
- Renaming
• **SQL**: Great for finding data from *tabular representations*, can get complex when many tables are involved in a given query
• **SQL**: Great for finding data from *tabular representations*, can get complex when many tables are involved in a given query

• **XQuery**: Great for finding data in *tree representations*, can get complex when many relationships have to be traversed

---

**SQL, XQuery and SPARQL: What's Wrong with this Picture?**

Jim Melton (Oracle; XML Query Working Group, XML Coord. Group)

Sixth annual W3C Technical Plenary (March 2006)
• **SQL:** Great for finding data from tabular representations, can get complex when many tables are involved in a given query

• **XQuery:** Great for finding data in tree representations, can get complex when many relationships have to be traversed

• **SPARQL:** Good pattern matching paradigm, especially when relationships have to be used to answer a query
Standardization’s view (Jim Melton, Oracle, 2006)

- **SQL**: Great for finding data from tabular representations, can get complex when many tables are involved in a given query
- **XQuery**: Great for finding data in tree representations, can get complex when many relationships have to be traversed
- **SPARQL**: Good pattern matching paradigm, especially when relationships have to be used to answer a query

SQL, XQuery and SPARQL: What's Wrong with this Picture?
Jim Melton (Oracle; XML Query Working Group, XML Coord. Group)
Sixth annual W3C Technical Plenary (March 2006)
• RDF is the first level of a logical tower
• Emphasis in logic features of RDF model
• Keep an eye in extensions to more expressive logics
• Bad news: complexity issues
RDF Query Language: Developer’s view

• How do we answer the most common queries?
• How do we cope with APIs and store developments?
• Design usually influenced by current programming and system tools.
• Not always concerned with scalability and long term.
RDF Query Language: Database theoretician’s view

RDF as a graph data model?

RDF as a relational model?

C. Gutierrez – Foundations of RDF Databases
Theorem (Gaifman). A property of graphs is expressible by a closed first order formula iff it is equivalent to a combination of properties of the form
\[ \exists v_1, \ldots, v_s \left[ \bigwedge_{1 \leq i \leq s} P(N(v_i, r)) \land \bigwedge_{1 \leq i < j \leq s} d(v_i, v_j) > 2r \right] \]

where \( v_1, \ldots, v_s \) denote vertices and \( d(x,y) \) denotes distance
Theorem (Gaifman). A property of graphs is expressible by a closed first order formula iff it is equivalent to a combination of properties of the form

\[ \exists v_1, \ldots, v_s \left[ \bigwedge_{1 \leq i \leq s} P(N(v_i, r)) \land \bigwedge_{1 \leq i < j \leq s} d(v_i, v_j) > 2r \right] \]

where \( v_1, \ldots, v_s \) denote vertices and \( d(x, y) \) denotes distance.

Want Local (relational) or global (graph) queries?
SPARQL (W3C Recommendation, 2008)
- Relational view of querying
- RDF = triples + blanks
- Pattern matching
SPARQL (W3C Recommendation, 2008)

- Relational view of querying
- RDF = triples + blanks
- Pattern matching

C. Gutierrez – Foundations of RDF Databases - ESWC 2008
SPARQL Query (General Structure)

Query Form:
- CONSTRUCT
- DESCRIBE
- SELECT
- ASK

Dataset Clause:
- FROM
- FROM NAMED

Where Clause (Graph Pattern):
- FILTER
- OPTIONAL
- AND
- UNION

C. Gutierrez – Foundations of RDF Databases - ESWC 2008