Cytoscape Plugin for Oracle Database Semantic Technologies

Seema Sundara, Consulting Member Technical Staff
Jags Srinivasan, Consulting Member Technical Staff
Cytoscape Plugin

- Overview
- Basic Elements
- Basic Operations
- A Quick Tour
- Browsing UniProt Data
- Browsing Entailed RDF Data (TSNET & LUBM)
- Browsing Wikipedia Data
- Setup
- Interactive Visualization
- Components
Overview
Semantic Technologies in a Nut Shell

- Model complex real-world relationships beyond Boolean in the data as a graph
  
  Allow schemas to continuously and dynamically evolve

- Inference among relationships with rules, standard concepts and terms to discover new information
  
  Enable machine-driven creation of new data relationships without restructuring the data model

- Query using graph patterns to provide more semantically complete information for decision-making
  
  Support discovery workflows
Why Organizations use Oracle RDF Store

• Standards Compliance
  – RDF, RDFS, OWL; ontologies – SNOMED, SKOS, etc.

• Open Source Technology Support
  – Jena, Joseki, ARQ, TDB, SDB, Sesame, Pellet, D2RQ, Jetty, Cytoscape, GATE, Protégé…

• Scalability & Performance
  – storage, native inference engine, querying
  – Customers tell us other triple-stores routinely fail with large datasets
  – Leading Lehigh University Benchmark (LUBM) results

• Query RDF data using SPARQL or SQL
  – SEM_MATCH, SEM_CONTAINS, Ontology-assisted SQL queries

• Security
  – Virtual Private Database, Oracle Label Security to control access

• Semantic Indexing for Documents stored in Oracle Database
Oracle Database 11g Semantic Technologies

- Only leading commercial database with native semantic data management
- Scalable & secure platform scales to repositories w/ billions of triples
- RAC & page-level compression support
- Choice of SQL or SPARQL query
- Native inferencing and 3rd party reasoner support e.g., PelletDB
- Industry leading 3rd party & open source tools, services, apps support
- W3C standards-based technologies

Key Capabilities:

**Load / Storage**
- Native RDF graph data store
- Manages billions of triples
- Fast batch, bulk and incremental load

**Query**
- SPARQL-Jena/Joseki, Sesame
- SQL: SEM_Match
- Ontology assisted query of relational data

**Reasoning**
- RDFS, OWL 2 RL support
- User-defined SWRL-like rules
- Plug-in architecture
Key Features of the Cytoscape Plugin

- Plugin from Oracle that connects the open source Cytoscape RDF data visualizer to Oracle Database
- Visualizes RDF and OWL datasets in Oracle Database
- Visualizes original and entailed (inferred) graphs
- Enables Fish Eye views by building hybrid (summary-detail) graphs
- Performant with large RDF data sets
- Available for download from the OTN Semantic Technologies page

Fish eye views show the point of interest in detail and the overview of the graph in the same window. The picture is scaled non uniformly. Objects far away from the focus point are shrunk while objects near the focus point are magnified. See Furnas, G.W.: Generalized Fisheye Views, Proc. ACM SIGCHI'86, 1986
**Key Features**

- Visualization rendered using the state-of-art Open Source CytoScape* by implementing a Oracle RDF Plugin Class.

(*Uses CytoScape 2.6 :for more details see [http://cytoscape.org/](http://cytoscape.org/) *)
Key Features

- Powered by Oracle’s PL/SQL SEM_ANALYSIS package providing fast and optimized access to Oracle Semantic Store.
Large-scale RDF Data Visualization

A 3-S approach

**Subset Browsing**
- Static subsets such as root classes
- Dynamic subsets specified via SPARQL graph pattern

**Summary Graphs**
- Reduces clutter by representing portions of graph by its summary (user can create their own *Fish Eye* views)
- Improves response time

**Sampling**
- For very large datasets, allows browsing a sample of the dataset
Basic Elements
Basic Elements

• Detail Graph
  – Whole graph or a subgraph

• Summary Graph
  – Static Summaries
    • Representative Instance
    • Particular Instance
  – Dynamic Summaries
    • Summary for SPARQL-pattern based dynamic subgraph

• (Summary-Detail) Hybrid Graph
Example 1: Detail Graph
Example 2: Detail SubGraphs
Example 3: Representative & Particular Instance Summary Graphs

Note: The right graph was manually re-arranged to simplify comparison.
Example 4: Dynamic Subset-based Summary
Example 5: Particular Instance also an example of hybrid (detail-summar y) graph
Basic Operations
### Basic Operations on a Model

<table>
<thead>
<tr>
<th>Operation</th>
<th>Type</th>
<th>Max. Result Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default (Get Root Classes)</td>
<td>Static Subset</td>
<td>Num_Nodes</td>
</tr>
<tr>
<td>Get Demographics</td>
<td>Static Summary</td>
<td>1</td>
</tr>
<tr>
<td>Get Class Demographics</td>
<td>Static Summary</td>
<td>1</td>
</tr>
<tr>
<td>SEM_MATCH (with focal point) based Summary</td>
<td>Dynamic Subset</td>
<td>2*(Num_Predicates)</td>
</tr>
<tr>
<td>SEM_MATCH based subset</td>
<td>Dynamic Subset</td>
<td>Num_Triples</td>
</tr>
</tbody>
</table>
### Basic Operations on a Class

<table>
<thead>
<tr>
<th>Operation</th>
<th>Type</th>
<th>Max. Result Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get (Class) Representative Summary</td>
<td>Static Subset Summary</td>
<td>2*(Num_Predicates)</td>
</tr>
<tr>
<td>Get Particular Instance Summary</td>
<td>Static Subset Summary</td>
<td>2*(Num_Predicates)</td>
</tr>
<tr>
<td>Expand Class</td>
<td>Dynamic Subset</td>
<td>Num_Triples</td>
</tr>
</tbody>
</table>
## Basic Operations on a Hybrid Graph

<table>
<thead>
<tr>
<th>Operation</th>
<th>Type</th>
<th>Max. Result Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expand Class</td>
<td>Dynamic Subset</td>
<td>Num_Triples</td>
</tr>
<tr>
<td>Expand Node</td>
<td>Static Subset Summary</td>
<td>2*(Num_Predicates)</td>
</tr>
<tr>
<td>Expand Property</td>
<td>Dynamic Subset</td>
<td>Num_Triples</td>
</tr>
</tbody>
</table>
Basic Operation Overview

- SPARQL-pattern based Summary
- Root Classes
- Class Representative Summary
- Particular Instance Summary
- SPARQL-pattern based subset

Expand Node
Expand Class
Expand Property

Hybrid (Summary-Detail) Graph
Handling Operation returning Large Results

- Namely, Get Root Classes, SEM_MATCH based subset, Expand Class, Expand Property.
- Two threshold are used for the result set
  - **Database Result Set Limit**: Identifies the maximum number of triples returned by database for an operation (currently, set to 2,000). Thus, the above operations would return at most FIRST 2,000 rows to the rendering engine.
  - **Display Set Limit**: Identifies the maximum number of triples displayed in a single view (currently, set to 250). The rendering engine currently uses a *shuffle* operation to randomly pick 250 triples after combining the existing set of triples with the newly obtained triples.
Handling Operation returning Large Results: Selective Expand Property

- Support `selectiveExpandProperty()`, which allows user to restrict expansion based on look-ahead filter specified by a SPARQL pattern.
- [Benefits: Could be useful when property being expanded has a very high count]
- **Example:** the type edge has 76529 edge count, which can be restricted using `sem_match` pattern to publications containing ‘Publication1’
A Quick Tour of the Cytoscape Plugin
Steps to Browse RDF Data

• Select Model, Rulebase, Overlay option
  – Overlay option decides whether you want to see object/property/class expansions in the same view or a new separate view

• You get all root classes

• Expand Class to get all its subclasses or get a Representative Instance of a class

• Get a Particular Instance of a class
  – Either from root class view
  – Or from representative instance view
  – Properties having >1 count of particular instance are shown as an aggregate edge.

• Perform Expand Property on an aggregate edge or perform expand object on any named node in the given view.

• In general:
  – Expand Property is allowed on any aggregate property edge having either source or destination as a named node
  – Expand Object and Expand Class are allowed on any named node.
Browsing RDF Data

Get Root Class

#Person

Expand Person Class

Get Particular Instance: Jack

Get Representative for ITProfessional
Browsing RDF Data: Expand Node(John)
After Multiple Property & Node Expansions
Browsing UniProt Data
Examining protein subsets w.r.t `encodedIn` property
Examining protein subsets w.r.t. sequence property
Dynamic Subset of Proteins

Summary
Dynamic subset of Gene Instances
Dynamic subset of Gene Instances
Browsing Entailed RDF Data (TSNET & LUBM)
Browsing Wikipedia Data
Setup
Database Setup

• After creating semantic network
  – use Flexible Indexing API to create two additional indexes
    EXEC SEM_APIS.ADD_SEM_INDEX('CPS');
    EXEC SEM_APIS.ADD_SEM_INDEX('SP');
    [Index created to allow efficient querying for incoming edges]
• Load models as before
• Perform entailment as before (if needed).
  [For visualization shown, we used minimal ‘RDFS’, as shown below]
  EXEC SEM_APIS.CREATE_ENTAILMENT('sem_tsnet',
    SDO_RDF_MODELS('tsnet'),
    SDO_RDF_RULEBASES('RDFS'),
    SEM_APIS.REACH_CLOSURE,
    'RDFS4A-, RDFS4B-, RDFS6-, RDFS8-, RDFS10-, RDFS13-',
    'ENTAIL_ANYWAY=T, DISTANCE=T, USER_RULES=F');
Database Setup

- Generate summary for model data and entailed model

```sql
DECLARE
    model VARCHAR2(32):='tsnet';
    rulebase VARCHAR2(32) := 'rdfs';
    sample_size number := 100;
    cnt1 NUMBER;
    cnt2 NUMBER;
BEGIN
    SEM_ANALYSIS.GEN_SUMMARY(
        model, rulebase, sample_size, cnt1, cnt2);
    DBMS_OUTPUT.PUT_LINE('Model '||model);
    DBMS_OUTPUT.PUT_LINE('genclassinscnt:'||cnt1||' genclasspropcnt:'||cnt2||' ----');
END;
```
CytoScape Setup

- Need Cytoscape 2.6; Install Oracle RDF Visualizer Plugin jar in plugin directory
- Launch Cytoscape
  - From Plugin Menu select Oracle RDF Visualizer
  - Provide database connect parameters and connect. It will bring the following form
Interactive Visualization
Interactive Visualization of a representative instance

- Queries performed on a model with and without a rulebase
- Queries performed with and without a summary table
- Summary tables facilitate the response time to be within several seconds.
- Improved performance with summary tables
## Summary tables

**Summary information for the dataset**

<table>
<thead>
<tr>
<th></th>
<th>LUBM 1 million</th>
<th>UNIPROT 5 million</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary creation time</td>
<td>1 min, 24 sec</td>
<td>4 min, 41 sec</td>
</tr>
<tr>
<td>Class inst. count rows</td>
<td>20</td>
<td>89</td>
</tr>
<tr>
<td>(without rulebase)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Property Count rows</td>
<td>150</td>
<td>467</td>
</tr>
<tr>
<td>(without rulebase)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class inst. count rows</td>
<td>36</td>
<td>94</td>
</tr>
<tr>
<td>(with rulebase)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Property Count rows</td>
<td>390</td>
<td>574</td>
</tr>
<tr>
<td>(with rulebase)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Interactive visualization for LUBM 1 million data set

• Query to get a representative instance of the “department” class

```sql
select * from table(sem_analysis.get_class_representative('LUBM1M_MODEL','http://www.lehigh.edu/~zhp2/2004/0401/univbench.owl#Department', 'RDFS'));
```

<table>
<thead>
<tr>
<th></th>
<th>Without Rulebase</th>
<th>With Rulebase</th>
</tr>
</thead>
<tbody>
<tr>
<td>With Summary</td>
<td>0.01 sec</td>
<td>0.01 sec</td>
</tr>
<tr>
<td>Without Summary</td>
<td>0.5 sec</td>
<td>1.62 sec</td>
</tr>
</tbody>
</table>
• Query to get a representative instance of the “protein” class

```sql
select *
from table(sem_analysis.get_class_representative('UNIPROT5M_MODEL', 'urn:lsid:uniprot.org:ontology:Protein', 'RDFS'));
```

<table>
<thead>
<tr>
<th></th>
<th>Without Rulebase</th>
<th>With Rulebase</th>
</tr>
</thead>
<tbody>
<tr>
<td>With Summary</td>
<td>0.01 sec</td>
<td>0.01 sec</td>
</tr>
<tr>
<td>Without Summary</td>
<td>2.8 sec</td>
<td>14.4 sec</td>
</tr>
</tbody>
</table>
Oracle RDF Visualizer Component Overview
Oracle RDF Visualizer Component Overview

*The Cytoscape Core Architecture shown above (in the left) is from [1]*
Cytoscape Plugin
Java Class
Cytoscape Plugin Java Class

- Extend `CytoscapePlugin` abstract class exposed by Cytoscape
- Add items to `JMenu` which are action listeners to appear under particular plugin
- Add an action listener for the item added
- Implement a class for the action listener which implements `actionPerformed` method
- Make a `jar` file of plugin and include in Cytoscape `plugins` directory
  - It gets picked up automatically when Cytoscape is started
SEM_ANALYSIS
PL/SQL Package
SEM_ANALYSIS PL/SQL Package

- Utility package that can be used standalone or with any application (such as Cytoscape)
- Contains set of functions/procedures to get both static/dynamic RDF graph information
- Contains set of functions to dynamically generate summaries for model or model with inferred triples
SEM_ANALYSIS PL/SQL package

• Summary Tables
  – RDF_CLASS_INSTANCE_SUMMARY$
    – Instance count summary per distinct class
      
      | Model Name | Rulebase Name | Class Name       | Instance Count |
      |------------|---------------|------------------|----------------|
      | TSNET      | NULL          | sc:ITProfessional| 2              |
      | TSNET      | NULL          | sc:Receptionist  | 1              |

  – RDF_CLASS_PROPERTY_SUMMARY$
    – Summary of distinct property count
      
      | Model Name | Rulebase Name | Class Name       | Property   | Property count |
      |------------|---------------|------------------|------------|----------------|
      | TSNET      | NULL          | sc:ITProfessional| sc:Age     | 1              |
      | TSNET      | NULL          | sc:ITProfessional| sc:residesIn | 1              |
SEM_ANALYSIS PL/SQL package

• Demographics gathering functions
  – get_demographics - returns
    • Total number of triples in the graph
    • Total number of DISTINCT subject/predicate/object/literals in the graph
  – get_class_demographics - returns
    • Size of the graph (number of unique nodes, excluding root nodes)
    • Maximum Fanout for class hierarchy
    • Class hierarchy (DAG or a Tree)
SEM_ANALYSIS PL/SQL Package

- Functions to get graph information (modeled as Table functions)
  - get_class_representative
    - gets the representative (one-hop) connectivity graph for a given class
  - get_matching_triples
    - gets the sub graph satisfying the specified triple pattern
  - get_expanded_prop
    - get all property value triples for the given class instance via the specified property

Example get_expanded_prop(:Jack, sc:friendOf)
returns
(:Jack sc:friendOf :John)(:Jack sc:friendOf :Jill)
SEM_ANALYSIS PL/SQL Package

- Functions to get graph information (modeled as Table functions)
  - `get_propcount_gtone`
    - gets the properties of a given instance that have an occurrence count of more than one
  - `get_propcount_one`
    - gets the properties of a given instance that have an occurrence count of one

The above two functions, along with `get_class_representative`, determine the summary graph for an instance
SEM_ANALYSIS PL/SQL Package

• Functions to get schema information
  – get_root_classes
    - gets the root classes in the given model
  – get_sub_classes
    - gets all the subclasses of the given class
Summary – Cytoscape Plugin for Oracle Database

- Visualize RDF and OWL datasets in Oracle Database
- Visualize original and entailed (inferenced) graphs
- Create Fish Eye views w/ summary-detail graphs
- Performant for large RDF data sets
For More Information and Software Download

http://search.oracle.com

Semantic Technologies

Google “Oracle RDF”

Or go to

oracle.com
SOFTWARE. HARDWARE. COMPLETE.