An Enterprise Inference Engine Inside Oracle Database 11g Release 2
Zhe Wu, Ph.D., Oracle
Vladimir Kolovski, Ph.D., Oracle
June 2010
Outline

• Overview of Oracle Database Semantic Technologies

• Design of Parallel and Incremental Inference Engine

• Support for User Defined Rules

• Best Practice for Performance

• Summary
THE FOLLOWING IS INTENDED TO OUTLINE OUR GENERAL PRODUCT DIRECTION. IT IS INTENDED FOR INFORMATION PURPOSES ONLY, AND MAY NOT BE INCORPORATED INTO ANY CONTRACT. IT IS NOT A COMMITMENT TO DELIVER ANY MATERIAL, CODE, OR FUNCTIONALITY, AND SHOULD NOT BE RELIED UPON IN MAKING PURCHASING DECISION. THE DEVELOPMENT, RELEASE, AND TIMING OF ANY FEATURES OR FUNCTIONALITY DESCRIBED FOR ORACLE’S PRODUCTS REMAINS AT THE SOLE DISCRETION OF ORACLE.
Semantic Application Workflow

Transform & Edit Tools
- Entity Extraction & Transform
  - OpenCalais
  - Linguamatics
  - GATE
  - D2RQ
- Ontology Eng.
  - TopQuadrant
  - Mondeca
  - Ontoprise
  - Protege
- Categorization
  - Cyc
- Custom Scripting

Load, Query & Inference
- RDF/OWL Data Management
- SQL & SPARQL
  - Sesame Adapter
  - Jena Adapter
- Native Inferencing
- Semantic Rules
- Scalability & Security
- Semantic Indexing

Applications & Analysis Tools
- BI, Analytics
  - Teranode
  - Metatomix
  - MedTrust
- Graph Visualization
  - Cytoscape
- Social Network Analysis
- Metadata Registry
- Faceted Search

Partner Tools
Oracle Database 11g Semantic Database

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

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
## Oracle’s Partners for Semantic Technologies

### Integrated Tools and Solution Providers:

<table>
<thead>
<tr>
<th>Ontology Engineering</th>
<th>Reasoners</th>
<th>Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>TopQuadrant</td>
<td>Clarkparsia, LLC</td>
<td>Teranode</td>
</tr>
<tr>
<td>protégé</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sesame</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Joseki</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cYcorp</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query Tool Interfaces</th>
<th>Standards</th>
<th>SI / Consulting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jena</td>
<td>W3C</td>
<td>Northrop Grumman</td>
</tr>
<tr>
<td>openRDF.org</td>
<td>World Wide Web</td>
<td>Raytheon</td>
</tr>
<tr>
<td>Sesame</td>
<td></td>
<td>Orbis Technologies, Inc.</td>
</tr>
<tr>
<td>Joseki</td>
<td></td>
<td>Boeing</td>
</tr>
<tr>
<td>cYcorp</td>
<td></td>
<td>McDonald Bradley</td>
</tr>
<tr>
<td>GATE</td>
<td></td>
<td>accenture</td>
</tr>
<tr>
<td>CALAIS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linguamatics</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NLP Entity Extractors</th>
<th>Standards</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALAIS</td>
<td></td>
</tr>
<tr>
<td>GATE</td>
<td></td>
</tr>
<tr>
<td>general architecture</td>
<td></td>
</tr>
</tbody>
</table>
Some Oracle Database Semantics Customers

Life Sciences
- Lilly
- Pfizer
- Swiss Institute of Bioinformatics

Defense/Intelligence
- National Geospatial-Intelligence Agency

Clinical Medicine & Research
- The University of Texas Health Science Center at Houston
- Cleveland Clinic

Education
- The University of Michigan

Telecomm & Networking
- Hutchinson 3G
- Austria

Publishing
- Cisco
- Westlaw
- Thomson Reuters
Release 11.1 Semantic Technologies

- W3C standards: RDF, RDFS, OWL, SPARQL
- SQL & DML access to RDF/OWL data and ontologies
  - Querying in SQL w/ SPARQL-like graph patterns
  - Querying w/ SPARQL in Java through Jena adaptor
  - Ontology-assisted querying of relational data
- Native Inferencing: RDFS/OWL & user-defined rules
  - Persistent, scalable, ahead of queries, pluggable reasoners
- Model level security
- Bulk and incremental loading of triples
  - Up to 8 exabytes, 1 million graphs w/ multi-graph querying
  - Integral Oracle Partitioning and also table compression
Release 11.2 Semantic Technologies

• Strong security for Semantic Technologies
  – Security policies and data classification for RDF data

• Semantic indexing of documents
  – Document indexing based on popular natural language tools

• Faster, more expressive reasoning
  – Parallel and incremental inference, owl:sameAs optimization
  – OWL: union, intersection, OWL 2 RL
  – W3C Simple Knowledge Organization System (SKOS) & SNOMED ontology
  – Pellet OWL DL reasoner Integration (PelletDb)

• Change management for collaboration

• Standards & open source support
  – SPARQL query support for Filter, Union in SEM_MATCH table function
  – Jena v2.6.2, ARQ 2.8.1, Sesame 2.3.1
  – Java SDK for SPARQL for 3rd party integration e.g., Sesame
  – Utility to swap, rename, and merge models, rename entailment, remove duplicates
Capabilities Overview of Release 11.2

NLP engines, Tools, Editors, Complete DL reasoners, …

SQL/PLSQL APIs & JAVA APIs (Jena, Sesame)

**INFER**
- RDF/S
- OWL/SKOS
- User defined rules

**QUERY**
- Query RDF/OWL data and ontologies
- Ontology-Assisted Query of Enterprise Data

**STORE**
- Incr. DML
- Batch-Load
- Bulk-Load

Built-in Security and Versioning for semantic data
- RDF/OWL data
- Ontologies & rule bases

Relational data

• RDF/OWL data
• Ontologies & rule bases
Applications of Rules-based Inference
Applications of Lightweight OWL

• “One very heavily used space is that where RDFS plus some minimal OWL is used to enhance data mapping or to develop simple schemas.”
  - James Hendler ¹

• Complexity distribution of existing ontologies (2006) ²
  - Out of 1,200+ real-world OWL ontologies:
    • 43.7% (or 556) ontologies are RDFS
    • 30.7% (or 391) ontologies are OWL Lite
    • 20.7% (or 264) ontologies are OWL DL.
    • Remaining OWL FULL

• OWL2 RL covers many of the real-world OWL ontologies

² A Survey of the web ontology landscape. ISWC 2006
OWL 2 RL

- Syntactic subset of OWL 2
  - W3C standard profile
  - Inspired by DLP, pD*
  - Has more than 70 entailment rules
  - Reasoning/conjunctive query answering is PTIME w.r.t data/taxonomy complexity
  - Defines a standard set of rules for implementation

- Oracle provides full support for OWL 2 RL/RDF ruleset

2 A Survey of the web ontology landscape. ISWC 2006
Example OWL2 RL Entailment Rules

• OWL2RL has 70+ entailment rules.

  – E.g. rule:
    
    \[
    T(?p, \text{owl:propertyChainAxiom}, ?x) \\
    \text{LIST}[?x, ?p_1, \ldots, ?p_n] \\
    T(?u_1, ?p_1, ?u_2) \\
    T(?u_2, ?p_2, ?u_3) \\
    \ldots \\
    T(?u_n, ?p_n, ?u_{n+1}) \implies T(?u_1, ?p, ?u_{n+1})
    \]

    \[
    T(?p, \text{rdf:type}, \text{owl:FunctionalProperty}) \\
    T(?x, ?p, ?y_1) \\
    T(?x, ?p, ?y_2) \implies T(?y_1, \text{owl:sameAs}, ?y_2)
    \]

• These rules have efficient implementations in RDBMS
Extending Rules to Cover SNOMED-CT Classification

• **SNOMED-CT is a major application of OWL2 EL**
  - Suitable for applications employing ontologies that define very large numbers of classes and/or properties
  - One of the largest commercial biomedical ontologies

• **Oracle supports the core (EL+) subset of OWL 2 EL**
  - Support SNOMED-CT and Gene Ontology inference
  - Implements more completion rules

• **Example rule for EL+ inference**
  
  ?A  rdfs:subClassOf  ?A1
  ...
  ?A  rdfs:subClassOf  ?An
  T(?C, owl:intersectionOf, ?x)
  LIST[?x, ?A_1, ..., ?A_n]

  ➔  ?A  rdfs:subClassOf  ?C
Design of Semantic Inference Engine in Oracle 11g Release 2
Core Inference Features in 11.2

• Oracle provides native inference in the database for
  • RDFS, RDFS++
  • OWLPRIME, OWL2RL, SKOS
  • User-defined rules
• Inference done using forward chaining
  • Triples inferred and stored ahead of query time
  • Removes on-the-fly reasoning and results in fast query times
• Proof generation
  • Shows one deduction path
Infer Semantic Data: APIs

**SEM_APISCREATE_ENTAILMENT**

- Index_name
- sem_models(\textquotesingle GraphTBox\textquotesingle, \textquotesingle GraphABox\textquotesingle, ...),
- sem_rulebases(\textquotesingle OWLPrime\textquotesingle),
- passes,
- inf_components,
- Options

Typical Usage:
- First load RDF/OWL data
- Call create_entailment to generate inferred graph
- Query both original graph and inferred data

Inferred graph contains only new triples! Saves time & resources

**SEM_APISVALIDATE_ENTAILMENT**

- sem_models(\textquotesingle GraphTBox\textquotesingle, \textquotesingle GraphABox\textquotesingle, ...),
- sem_rulebases(\textquotesingle OWLPrime\textquotesingle),
- Criteria,
- Max_conflicts,
- Options

Typical Usage:
- First load RDF/OWL data
- Call create_entailment to generate inferred graph
- Call validate_entailment to find inconsistencies

**Java API:** performInference method in

- OracleSailConnection (Sesame Adapter), GraphOracleSem (Jena Adapter)
OWL Subsets Supported in 11.2

- **OWL subsets for different applications**
  - RDFS++
    - RDFS plus owl:sameAs and owl:InverseFunctionalProperty
  - OWLSIF (OWL with IF semantics), OWLPrime
    - Based on Dr. Horst’s pD* vocabulary
  - OWLPrime
    - Includes RDFS++, OWLSIF with additional rules
    - Jointly determined with domain experts, customers and partners
  - OWL 2 RL
    - W3C Standard
    - Adds rules about keys, property chains, unions and intersections to OWLPrime
  - Inference components for SNOMED

---

1 Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary
How Inference Engine Works inside Oracle Database

Inference Start

1. Create
2. Un-indexed, Partitioned Temporary Table
   - SID
   - PID
   - OID
   
3. Insert
4. Copy
5. New triples?

Y
- New Partition for inferred graph

N
- Partition for a semantic model

Check/Fire Rule 1
Check/Fire Rule 2
... ...
Check/Fire Rule n

Un-indexed, Partitioned Temporary Table

Exchange Table

Exchange Partition

IdTriplesTable
Key Design Ideas: Parallel Inference

• Reduce data storage footprint
  – Combine all input models into a partitioned, single source table
  – Use native RAW8 columns and compression to reduce the size of source table

• Leverage native Oracle parallel execution
  – Use parallel query and parallel DML for some entailment rules

• Simplify Rules
  – Break up rules that contain 4,5 or more patterns in two separate rule evaluations
  – The simpler the rule, the more “parallelizable” it is
Key Design Ideas: Incremental Inference

- Use semi-naïve rule evaluation
  - Focus on delta: triples inserted since last inference
  - Break-up a rule into sub-rules
- Delay duplicate elimination
  - Perform only at end of inference round
- Apply semi-naïve evaluation in non-incremental case
  - Use inferences in round n as delta in round n+1
  - Apply selectively, only if delta is small
  - Works best when there are multiple rounds of inference (e.g., SNOMED)
Extending Semantics Supported by 11.2 OWL Inference

• **Option 1: add user-defined rules**
  • Both Oracle 10g Release 2 and Oracle 11g support user-defined rules in this form:

    | Antecedents | Consequents |
    |-------------|-------------|
    | ?z :parentOf ?x .
    | ?x owl:differentFrom ?y . | ➔ | ?x :siblingOf ?y |

• Filter expressions are allowed
  
  ?x :hasAge ?age.
Extending Semantics Supported by 11.2 OWL Inference

- Option 2: Separation in TBox and ABox reasoning through PelletDb (interfacing Oracle Jena Adapter)
  - TBox (schema related) tends to be small in size
    - Generate a complete class subsumption tree using complete DL reasoners (like Pellet, KAON2, Fact++, Racer, etc)
  - ABox (instance related) can be arbitrarily large
    - Use the native inference engine in Oracle to infer new knowledge based on the class subsumption tree from TBox
Inference Performance and Scalability Evaluation
Tuning Tips for Best Inference Performance

• Analyze models before running inference
  
  * execute immediate sem_apis.analyze_model(...);

• Need a **balanced** hardware setup to use parallel inference
  
  * E.g., a server with multi-core/multi-cpu processors and ample I/O throughput
  
  * Use Oracle Automatic Storage Management (ASM) to manage the disks

• Use RAW8=T option for compact data structures
  
  * Smaller data structures implies less I/O

• Additional optimizations
  
  * Dynamic incremental inference: selectively applies semi-naïve rule evaluation while generating the entailment
    
    * Off by default, could be turned on by DYN_INC_INF=T option
Performance on Desktop PC
• OWLPrime (11.1.0.7) inference performance scales really well with hardware. It is *not* a parallel inference engine though.
### 11.2 Inference Performance

| **Parallel Inference** | **Time to finish inference:** 12 hrs.  
(LUBM8000  
1.06 billion triples  
+ 860M inferred) | **3.3x faster compared to serial inference in release 11.1** |
|------------------------|-------------------------------------------------|---------------------------------------------------------------|
| **Parallel Inference** | **Time to finish inference:** 40 hrs.  
(LUBM25000  
3.3 billion triples  
+ 2.7 billion inferred) | **30% faster than nearest competitor**  
**1/5 cost of other hardware configurations** |
| **Incremental Inference** | **Time to update inference:** less than 30 seconds after adding 100 triples.  
(LUBM8000  
1.06 billion triples  
+ 860M inferred) | **At least 15x to 50x faster than a complete inference done with release 11.1** |
| **Large scale owl:sameAs Inference** | **60% less disk space required**  
(UniProt 1 Million sample) | **10x faster inference compared to release 11.1** |

- **Setup:** Intel Q6600 quad-core, 3 7200RPM SATA disks, 8GB DDR2 PC6400 RAM, No RAID.  
64-bit Linux 2.6.18. **Assembly cost: less than USD 1,000**
Performance on Server
Inference Performance on Server

- Inference performance for LUBM1000 (138M)
  - 24.6 minutes to infer 108M+ new triples (DOP=8)

- Inference performance for LUBM8000 (1B+)
  - 226 minutes to infer 860M+ new triples (DOP=8)

- Setup: Dual quad-core, Sun Storage F5100 Flash Array, 32 GB RAM
Performance on Exadata v2
Inference Performance on Exadata V2

- **LUBM 25K benchmark ontology**
  (3.3 Billion triples)
  - OWLPrime inference with new inference components took 247 minutes (4 hour 7 minutes)
  - More than 2.7 billion new triples inferred.
  - DOP = 32

- **Preliminary result on LUBM 100K benchmark ontology**
  (13 Billion+ triples)
  - One round of OWLPrime inference (limited to OWL Horst semantics) finish in 1.97 hour
  - 5 billion+ new triples inferred.
  - DOP = 32

- Setup: Full Rack Sun Oracle Data Machine and Exadata Storage Server (8 node cluster)
Summary

- **Oracle database 11g** is the leading commercial database with native RDF/OWL data management
  - Load (incremental, batch, bulk)
  - Query (SPARQL, SQL)
  - Inference
  - Security (graph level, fine grained access control)

- **Built-in enterprise class parallel & incremental inference engine**
  - Supports rich semantics including
    - OWL2 RL, OWLPrime
    - SNOMED
    - RDFS, SKOS, User-defined rules
  - Highly scalable
For More Information

search.oracle.com

Semantic Technologies

or

oracle.com

SOFTWARE. HARDWARE. COMPLETE.