SEMANTIC CONFLICT DETECTION IN META-DATA
A RULE-BASED APPROACH

KARTHIKEYAN GIRILOGANATHAN

ADVISOR: Dr. I. BUDAK ARPINAR
COMMITTEE: Dr. AMIT P. SHETH
Dr. KHALED M. RASHEED
Introduction

- Massive amount of data is available on the Web
- Ability to annotate, extract, and query semantic meta-data has increased:
  - SWETO (Semantic Web Technology Evaluation Ontology):
    - populated with over 800,000 entities and 1.5 million explicit relationships between them in RDF or OWL
  - Freedom (Semagix):
    - uses SWETO and other domain ontologies to semantically annotate millions of documents or Web pages
  - Web Fountain (IBM):
    - annotated and disambiguated data from over a billion documents
Evolution of Meta-Data

Types of Metadata and Annotations

Ontology
(Example: Anatomy, Diagnostics, ...)

Semantic Metadata
(Example ontology-driven metadata:
Region: Upper Abdomen
Organ: Liver
Pathological Structure: Abscess, Abscess located in Liver)

Structural Metadata
(document structure: DTDs, XSL
clustering and similarity processing: concept extraction)

Syntactic Metadata
(language, format, document length, creation date, source,
audio bit rate, encryption, affiliation, date last reviewed, authorization, ...)

Data
(Structured, semi-structured and unstructured)

[Sheth 2003]
Meta-Data Concerns

- Next generation tools will focus on actionable information (with associated sources and supporting evidence) from existing (meta-)data
- Concerns about usage of meta-data
  - High quality (i.e., reliable, accurate, and trustworthy) semantic meta-data
    - Entity disambiguation
    - Inconsistency checking in OWL
    - Conflict detection
Motivating Factors

“Representing, identifying, discovering, validating, and exploiting complex relationships are important issues related to realizing the full power of the Semantic Web, and can help close the gap between highly separated information retrieval and decision-making steps” [Sheth, Arpinar & Kashyap 2003]

“The Web is decentralized, allowing anyone to say anything. As a result, different viewpoints may be contradictory, or even false information may be provided. In order to prevent agents from combining incompatible data or from taking consistent data and evolving it into an inconsistent state, it is important that inconsistencies can be detected automatically” [W3C 2004]

“... these problems manifest themselves in various ways, including poor recall of available resources and inconsistency of search results. They arise due to errors, omissions and ambiguities in the metadata...” [Currier & Barton 2003]
Semantic Conflict Identification

Introduction

Conflict Types

Simplification

Architecture

Results

Conclusion
Conflict illustration through Simplification

CONFLICT

John marriedTo Clauna

John fatherOf Mary

Clauna motherOf Bill

John fatherOf Mary

Bill marriedTo Mary

Introduction  Conflict Types  Simplification  Architecture  Results  Conclusion
Motivating Scenario

Various algorithms for semantic analytics

Query interface

EXTRACTORS (Freedom)

MetaData Store

SemDIS

SWETO

Introduction 
Conflict Types 
Simplification 
Architecture 
Results 
Conclusion
Outline

- Conflict types and definitions
- Simplification process
- System architecture
- Experimental results
- Conclusion and future work
Property Assertion Conflicts

- ‘daml:unambiguous’ or ‘owl:inverseFunctional-Property’ violation

- ‘daml:unique’ or ‘owl:Functional-Property’ violation
Property Assertion Conflicts

- ‘asymmetric’ property violation
- ‘disjoint’ property violation
Class Assertion Conflicts

- Classes c1 and c2 are ‘daml:disjoint’ or ‘owl:disjoint’

- Classes ‘Citizen’ and ‘Immigrant’ are ‘daml:disjoint’ or ‘owl:disjoint’
Class ‘Employee’ has a OWL or DAML restriction ‘maxCardinality’ of ‘1’ on a relation ‘hasDesignation’
We want to say that a person cannot be a superior and a friend to “John” at the same time.
Non-Assertional Conflicts

- Either the subject or the object alone is different between two RDF triples.
- Subjective Conflict
Conflict Definitions

- Two sets of triples T1 and T2 are said to be in conflict if their simplifications S(T1)→s1 and S(T2)→s2 are mutually non-agreeable.
- Two simplifications s1 and s2 are mutually non-agreeable if taken together they are in violation of U or E.

<table>
<thead>
<tr>
<th>T</th>
<th>A set of triples</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>A function denoting the process of simplification</td>
</tr>
<tr>
<td>s</td>
<td>The result of simplification (S(T)→s)</td>
</tr>
<tr>
<td>U</td>
<td>Constraints expressed in an ontology (e.g., the property ‘biologicalMother’ is unique)</td>
</tr>
<tr>
<td>E</td>
<td>Constraints supplied by an expert (e.g., person(x) can never do action(y))</td>
</tr>
</tbody>
</table>
Simplification Types

- An RDF triple is trivially a simplification because it is the most basic piece of knowledge.
- Composition of relations leads to simplification.
Composition of Relations

Consider a set of Triples T,
Let
\[ E = \{e_1, e_2, ..., e_n\} \]
\[ P = \{p_1, p_2, ..., p_m\} \]
Then,
Let
\[ C = \{(p_1, p_k), ..., (p_a, p_b, p_c, \ldots)\} \]
\[ R = \{r_1, r_2, ..., r_n\} \]
where \( r_1, r_2, ..., r_n \) are results of the composition.

The triple \((e_i, r_k, e_j)\) is a simplification if \( r_k \in R \) and \( e_i, e_j \in E \).
**Statement Simplification**

- There could be background knowledge based simplifications of the form:

\[ \text{statement}_1 \land \text{statement}_2 \land \ldots \land \text{statement}_n \rightarrow \text{statement}_t \]

- In this case \( \text{statement}_t \) is a simplification.
  - This type of simplification will depend on expert knowledge.
Statement Simplification

- Immigrant
  - associated
  - multipleDeposits
    - FinancialOrganization
  - suspected
    - MoneyLaundering
      - owner
        - BusinessOrganization
          - works
            - Person
              - underInvestigation
                - JudicialOrganization

Introduction | Conflict Types | **Simplification** | Architecture | Results | Conclusion
Defining Statement Simplification Rules

Introduction

Conflict Types

Simplification

Architecture

Results

Conclusion
RuleML (Overview)

- Explore rule systems suitable for the Web
- The syntax (in XML and RDF form)
- Semantics
- Tractability/efficiency
- Transformation
- Compilation
- Enable inferencing on Web data & interchange of rules between intelligent systems (ontology integration etc.)

We use RuleML

- Any inference engine that understands RuleML can evaluate our rules.
- We do not need to think about representation and translation.

http://www.ruleml.org/
Our Representation of an RDF triple

\[ \langle \text{subject} \rangle \langle \text{property} \rangle \langle \text{object} \rangle \]

- **Statement**\((x)\)
- **Subject**\((x, subject)\)
- **Property**\((x, property)\)
- **Object**\((x, object)\)
Conflict Rules

- Can be classified as Integrity Constraint Rules.

\[
\text{if statement}(x) \text{ and statement}(y) \text{ and subject}(x,a) \text{ and relation}(x,rel1) \text{ and object}(x,b) \text{ and subject}(y,a) \text{ and relation}(y,rel2) \text{ and object}(y,b) \text{ and disjoint}(rel1,rel2) \text{ then conflict}(x,y)
\]
Simplification Rules

- Can be classified as Production Rules

\[
\text{if } \text{statement}(x) \text{ and statement}(y) \text{ and } \\
\text{subject}(x,a) \text{ and relation}(x,rel1) \text{ and } \\
\text{object}(x,b) \text{ and subject}(y,a) \text{ and } \\
\text{relation}(y,rel2) \text{ and object}(y,b) \text{ then } \text{newStatement}(a, rel3, b) 
\]
Relational Ontology

- Relations are at the heart of semantic Web [Sheth, Arpinar & Kashyap 2003]
- Relations among relations need to be specified
  - Hierarchy of relations is similar to a taxonomy
- Just as we have moved from taxonomy to ontology the idea is to have an ontology for relations also.
Relationship Ontology

Introduction

Conflict Types

Simplification

Architecture

Results

Conclusion
Editing/Populating the Relationship Ontology
Template Based Rule Transformation

if statement(x) and statement(y) and subject(x,a) and relation(x,rel1) and object(x,b) and subject(y,a) and relation(y,rel2) and object(y,b) and disjoint(rel1,rel2) then conflict(x,y)


Transform

if statement(x) and statement(y) and subject(x,a) and relation(x, http://foo.com/test#likes) and object(x,b) and subject(y,a) and relation(y, http://foo.com/test#hates) and object(y,b) and disjoint(http://foo.com/test#likes, http://foo.com/test#hates) then conflict(x,y).

Introduction  Conflict Types  Simplification  Architecture  Results  Conclusion
System Architecture

- Conflict Engine
- MANDARAX API
- Rules (RuleML)
- Facts
- Relationship Ontology
- User Interface
- SIMPLIFICATION
- CONFIDER API
- SERIALIZER
- JENA API
- Semantic Metadata
- Ontology

Sections:
- Introduction
- Conflict Types
- Simplification
- Architecture
- Results
- Conclusion
Algorithm

Semantic Metadata → Confider

- Serialize the statements to triples and assign id to each triple
- Convert triples to FOL predicates

Simplification Rules?

- Find statements that correspond to constraints
- Convert those statements to conflict rules
- Convert the rest of the statements to facts

While new statements produced

- Apply simplification rules to get new stmts
- Assign internal ids to these stmts
- Attach the derivation to each stmt

CONFLICT ENGINE

- List of Statement pairs that are in conflict
- Derivation available for conflicting statements
- Derivation available if stmts generated internally

Simplification Rules?

- Find relations that correspond to constraints
- Generate Conflict rules
- Find relations that correspond to Composition
- Generate Simplification rules

Introduction Conflict Types Simplification Architecture Results Conclusion
Mandarax

- A open source java class library for deduction rules
- OO
  - Not a translation of a prolog interpreter from c to java
- Based on backward reasoning
- Easy integration of various databases
- Support for Web services, and EJB
- Rules specified as RuleML
- Jens Dietrich (Massey University, New Zealand)
  - A list of contributors available at http://mandarax.sourceforge.net/

www.mandarax.org
Conflict Identification Results

Introduction

Conflict Types

Simplification

Architecture

Results

Conclusion
Statement Provenance

Introduction
Conflict Types
Simplification
Architecture
Results
Conclusion
Performance Evaluation (1)

with increase in number of conflicts (500 triples)
Performance Evaluation (2)
with increase in number of triples (10 conflicts)
Conclusion

In this work we have:

- Defined conflicts in semantic meta-data and classified them.
- Discussed a rule-based approach to identify the conflicts.
- Shown the use of relations between relations to simplify the triples and identify conflicts.
- Demonstrated the applicability of the approach over a limited data set using a prototype.
Our future work directions include developing:

- Scalable conflict identification techniques for large amounts of semantic data and conflict rules
- Investigation of other rule evaluation methods to improve performance
- Experiment with ways of representing an RDF triple in predicate form to compare performance
- A mechanism for expressing, evaluating, and adjusting trust dynamically based on conflict detection
Questions?
Thank You