SCHEMA-BASED SEMANTIC MATCHING

Pavel Shvaiko

joint work on “semantic matching” with Fausto Giunchiglia and Mikalai Yatskevich
joint work on “ontology matching” with Jérôme Euzenat

1st European Semantic Technology Conference (ESTC’07), Semantic Web Technology Showcase
31 May 2007, Vienna, Austria

Outline

● Part I: The matching problem
● Part II: State of the art in ontology matching
● Part III: Schema-based semantic matching
● Part IV: Evaluation (technology showcase)
● Part V: Conclusions
Outline

- Part I: The matching problem
  - Problem statement
  - Applications
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
- Part IV: Evaluation (technology showcase)
- Part V: Conclusions

Matching operation

Matching operation takes as input ontologies, each consisting of a set of discrete entities (e.g., tables, XML elements, classes, properties) and determines as output the relationships (e.g., equivalence, subsumption) holding between these entities.
Example: two XML schemas

Example: two ontologies
Statement of the problem

Scope

- Reducing heterogeneity can be performed in two steps:
  - Match, thereby determine the alignment
  - Process the alignment (merge, transform, translate...)

Correspondence is a 5-tuple \(<id, e1, e2, R, n>\)

- \(id\) is a unique identifier of the given correspondence
- \(e1\) and \(e2\) are entities (XML elements, classes,...)
- \(R\) is a relation (equivalence, more general, disjointness,...)
- \(n\) is a confidence measure, typically in the [0,1] range

Alignment \((A)\) is a set of correspondences

- with some cardinality: 1-1, 1-n, ...
- some other properties (complete)
Statement of the problem

Matching process

\[ p \text{ (weights, ...)} \]

\[ r \text{ (WordNet, ...)} \]

Applications

Traditional
- Ontology evolution
- Schema integration
- Catalog integration
- Data integration

Emergent
- P2P information sharing
- Web service composition
- Agent communication
- Query answering on the web
Applications: Information integration

Q: find an article about Ontology Matching

Matcher

Matcher

Common Ontology

Alignment 1

Local Ontology 1

wrapper 1

Applications: summary

<table>
<thead>
<tr>
<th>Application</th>
<th>instances</th>
<th>run time</th>
<th>automatic</th>
<th>correct</th>
<th>complete</th>
<th>operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ontology evolution</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td>transformation</td>
</tr>
<tr>
<td>Schema integration</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td>merging</td>
</tr>
<tr>
<td>Catalog integration</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td></td>
<td></td>
<td>data translation</td>
</tr>
<tr>
<td>Data integration</td>
<td>√</td>
<td></td>
<td>√</td>
<td></td>
<td></td>
<td>query answering</td>
</tr>
<tr>
<td>P2P information sharing</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>query answering</td>
</tr>
<tr>
<td>Web service composition</td>
<td>√ √ √</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>data mediation</td>
</tr>
<tr>
<td>Multi-agent communication</td>
<td>√ √ √</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>data translation</td>
</tr>
<tr>
<td>Query answering</td>
<td>√ √ √</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>query reformulation</td>
</tr>
</tbody>
</table>
Outline

- Part I: The matching problem
- Part II: State of the art in ontology matching
  - Classification of matching techniques
  - Overview of matching systems
- Part III: Schema-based semantic matching
- Part IV: Evaluation (technology showcase)
- Part V: Conclusions

Classification of basic techniques

Three layers
- The upper layer
  - Granularity of match
  - Interpretation of the input information
- The middle layer represents classes of elementary (basic) matching techniques
- The lower layer is based on the kind of input which is used by elementary matching techniques
Basic techniques

String-based

- Edit distance

  - It takes as input two strings and calculates the number of insertions, deletions, and substitutions of characters required to transform one string into another, normalized by \( \text{max}(\text{length}(\text{string}1), \text{length}(\text{string}2)) \)

  - \( \text{EditDistance}(\text{NKN}, \text{Nikon}) = 0.4 \)
Basic techniques (cont’d)

Linguistic resources: WordNet

It computes relations between ontology entities by using (lexical) relationships of WordNet

- $A \subseteq B$ if $A$ is a hyponym or meronym of $B$
  - Brand $\subseteq$ Name
- $A \supseteq B$ if $A$ is a hypernym or holonym of $B$
  - Europe $\supseteq$ Greece
- $A = B$ if they are synonyms
  - Quantity = Amount
- $A \bot B$ if they are antonyms or siblings in part of hierarchy
  - Microprocessors $\bot$ PC Board

Systems: analytical comparison

~50 matching systems exist, …we consider some of them

<table>
<thead>
<tr>
<th></th>
<th>SF</th>
<th>Artemis</th>
<th>Cupid</th>
<th>COMA</th>
<th>Prompt</th>
<th>OLA</th>
<th>S-Match</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element-level</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Syntactic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>string-based, data types, key properties</td>
<td>domain compatibility, language-based</td>
<td>string-based, data types, key properties</td>
<td>string-based, language-based, data types</td>
<td>string-based, data types, ranges</td>
<td>string-based, data types, language-based</td>
<td>string-based, language-based</td>
</tr>
<tr>
<td>External</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Semantic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>common thesaurus (CT)</td>
<td>auxiliary dictionary</td>
<td>auxiliary dictionary</td>
<td>-</td>
<td>WordNet</td>
<td>WordNet</td>
<td></td>
</tr>
<tr>
<td>Structure-level</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Syntactic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>iterative fix-point computation</td>
<td>matching of neighbors via CT</td>
<td>tree matching weighted by leaves</td>
<td>DAG (tree) matching with a bias towards leaf or children structures</td>
<td>bounded path matching</td>
<td>iterative fix-point computation, matching of neighbors</td>
<td>-</td>
</tr>
<tr>
<td>Semantic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>SAT</td>
</tr>
</tbody>
</table>
Outline

- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
  - Semantic matching
  - Iterative semantic matching
- Part IV: Evaluation (technology showcase)
- Part V: Conclusions

Generic matching

Information sources (classifications, XML schemas, ...) can be viewed as graph-like structures containing terms and their inter-relationships.

Matching takes two graph-like structures and produces correspondences between the nodes of the graphs that are supposed to correspond to each other.
Semantic matching in a nutshell

**Semantic matching:** Given two graphs $G_1$ and $G_2$, for any node $n_{1i} \in G_1$, find the strongest semantic relation $R'$ holding with node $n_{2j} \in G_2$

**Computed $R$'s, listed in the decreasing binding strength order:**
- equivalence $\{ = \}$
- more general/specific $\{ \supseteq, \subseteq \}$
- disjointness $\{ \perp \}$
- I don’t know $\{ \text{idk} \}$

We compute semantic relations by analyzing the **meaning (concepts, not labels)** which is codified in the elements and the structures of ontologies.

Technically, labels at nodes written in natural language are translated into propositional logical formulas which explicitly codify the labels' intended meaning. This allows us to codify the matching problem into a propositional validity problem.

---

Concept of a label & concept at a node

**Concept of a label** is the propositional formula which stands for the set of documents that one would classify under a label it encodes.

**Concept at a node** is the propositional formula which represents the set of documents which one would classify under a node, given that it has a certain label and that it is in a certain position in a tree.
Four macro steps

Given two labeled trees T1 and T2, do:

1. For all labels in T1 and T2 compute concepts at labels
2. For all nodes in T1 and T2 compute concepts at nodes
3. For all pairs of labels in T1 and T2 compute relations between concepts at labels (background knowledge)
4. For all pairs of nodes in T1 and T2 compute relations between concepts at nodes

Steps 1 and 2 constitute the preprocessing phase, and are executed once and each time after the ontology is changed (OFF-LINE part)
Steps 3 and 4 constitute the matching phase, and are executed every time two ontologies are to be matched (ON-LINE part)

Step 1: compute concepts at labels

The idea
- Translate labels at nodes written in natural language into propositional logical formulas which explicitly codify the labels’ intended meaning

Preprocessing
- **Tokenization.** Labels (according to punctuation, spaces, etc.) are parsed into tokens. E.g., Photo and Cameras → <Photo, and, Cameras>
- **Lemmatization.** Tokens are morphologically analyzed in order to find all their possible basic forms. E.g., Cameras → Camera
- **Building atomic concepts.** An oracle (WordNet) is used to extract senses of lemmas. E.g., Camera has 2 senses
- **Building complex concepts.** Prepositions, conjunctions are translated into logical connectives and used to build complex concepts out of the atomic concepts
  - E.g., $C_{\text{Cameras and Photo}} = <\text{Cameras, }\{\text{WN}_{\text{Camera}}\} > \land <\text{Photo, }\{\text{WN}_{\text{Photo}}\}>$
Step 2: compute concepts at nodes

The idea

Extend concepts at labels by capturing the knowledge residing in a structure of a tree in order to define a context in which the given concept at a label occurs

Computation

Concept at a node for some node $n$ is computed as a conjunction of concepts at labels located above the given node, including the node itself

Two types of concepts of nodes

- Conjunctive:
  \[ C_2 = C_{\text{Electronics}} \land C_{\text{PC}} \]

- Disjunctive:
  \[ C_4 = C_{\text{Electronics}} \lor (C_{\text{Cameras}} \lor C_{\text{Photo}}) \lor C_{\text{Digital Cameras}} \]

Step 3: compute relations between (atomic) concepts at labels

The idea

- Exploit a priori knowledge, e.g., lexical, domain knowledge, with the help of element level semantic matchers

<table>
<thead>
<tr>
<th></th>
<th>Cameras$_2$</th>
<th>Photo$_2$</th>
<th>Digital Cameras$_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Photo$_1$</td>
<td>$idk$</td>
<td>$=$</td>
<td>$idk$</td>
</tr>
<tr>
<td>Cameras$_1$</td>
<td>$=$</td>
<td>$idk$</td>
<td>$?$</td>
</tr>
</tbody>
</table>

Semantic Web Technology Show Case at ESTC'07, Vienna, Austria
Step 3: 
Element level semantic matchers

Sense-based matchers have two WordNet senses in input and produce semantic relations exploiting (direct) lexical relations of WordNet.

String-based matchers have two labels in input and produce semantic relations exploiting string comparison techniques.

<table>
<thead>
<tr>
<th>Matcher name</th>
<th>Execution order</th>
<th>Approximation level</th>
<th>Matcher type</th>
<th>Schema info</th>
</tr>
</thead>
<tbody>
<tr>
<td>WordNet</td>
<td>1</td>
<td>1</td>
<td>Sense-based</td>
<td>WordNet senses</td>
</tr>
<tr>
<td>Prefix</td>
<td>2</td>
<td>2</td>
<td>String-based</td>
<td>Labels</td>
</tr>
<tr>
<td>Suffix</td>
<td>3</td>
<td>2</td>
<td>String-based</td>
<td>Labels</td>
</tr>
<tr>
<td>Edit distance</td>
<td>4</td>
<td>2</td>
<td>String-based</td>
<td>Labels</td>
</tr>
<tr>
<td>Ngram</td>
<td>5</td>
<td>2</td>
<td>String-based</td>
<td>Labels</td>
</tr>
</tbody>
</table>

Step 4: compute relations between concepts at nodes

The idea

- Decompose the tree matching problem into the set of node matching problems.
- Translate each node matching problem, namely pairs of nodes with possible relations between them, into a propositional formula.
- Check the propositional formula for validity.
Step 4: Efficient semantic matching

Conjunctive concepts at nodes
- Matching formula is Horn
  - Satisfiability can be determined in linear time
  - SAT solver requires quadratic time
- We developed ad hoc linear time reasoning procedure
  - Avoid conversion to propositional formula
  - Reason on the axioms matrix

Disjunctive concepts at nodes
- Matching formula is not in CNF by construction
  - Most SAT solvers require the input formula to be in CNF
  - Conversion to CNF may lead to exponential space explosion
- Exploit structure preserving transformation
  - Size of formula in CNF is linear with respect to original formula
Outline

- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
  - Semantic matching
  - Iterative semantic matching
- Part IV: Evaluation (technology showcase)
- Part V: Conclusions

Motivation:

Problem of low recall (incompleteness) - I

Facts

- Matching (usually) has two components: element level matching and structure level matching
- Contrarily to many other systems, the semantic matching structure level algorithm is correct and complete
- Still, the quality of results is not very good

Why? ... the problem of lack of knowledge
Motivation:

Problem of low recall (incompleteness) - II

Preliminary (analytical) evaluation

<table>
<thead>
<tr>
<th>Matching tasks</th>
<th>#nodes</th>
<th>max depth</th>
<th>#labels per tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google vs Looksmart</td>
<td>706/1081</td>
<td>11/16</td>
<td>1048/1715</td>
</tr>
<tr>
<td>Google vs Yahoo</td>
<td>561/665</td>
<td>11/11</td>
<td>722/945</td>
</tr>
<tr>
<td>Yahoo vs Looksmart</td>
<td>74/140</td>
<td>8/10</td>
<td>101/222</td>
</tr>
</tbody>
</table>

Dataset

[P. Avesani et al., ISWC'05]

On increasing the recall: an overview

Multiple strategies

- Strengthen element level matchers
- Reuse of previous match results from the same domain of interest
  - PO = Purchase Order
- Use general knowledge sources (unlikely to help)
  - WWW
- Use, if available (!), domain specific sources of knowledge
  - UMLS, FMA
Iterative semantic matching (ISM)

The idea
Repeat Step 3 and Step 4 of the matching algorithm for some critical (hard) matching tasks

ISM macro steps
- Discover critical points in the matching process
- Generate candidate missing axiom(s)
- Re-run SAT solver on a critical task taking into account the new axiom(s)
- If SAT returns false, save the newly discovered axiom(s) for future reuse

ISM: Discovering critical points - example

Google (T1) Looksmart (T2)
cLabsMatrix (result of Step 3) cNodesMatrix (result of Step 4)
ISM: Generating candidate axioms

- **Sense-based matchers** have two WordNet senses in input and produce semantic relations exploiting structural properties of WordNet hierarchies
  - Hierarchy Distance (HD)

- **Gloss-based matchers** have two WordNet senses as input and produce relations exploiting gloss comparison techniques
  - WordNet Gloss (WNG)
  - Extended WordNet Gloss (EWNG)
  - Gloss Comparison (GC)

ISM: generating candidate axioms

**Hierarchy Distance**

Hierarchy distance returns the equivalence relation if the distance between two input senses in WordNet hierarchy is less than a given threshold value (e.g., 3) and idk otherwise.

There is no direct relation between *games* and *entertainment* in WordNet.

Distance between these concepts is 2 (1 more general link and 1 less general). Thus, we can conclude that *games* and *entertainment* are close in their meaning and return the equivalence relation.

- diversion
- entertainment
- games
Outline

- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
- Part IV: Evaluation (technology showcase)
  - Evaluation setup
  - Evaluation results
- Part V: Conclusions

Evaluation (quality) measures

- Precision = \( \frac{|TP|}{|TP| + |FP|} \)
- Recall = \( \frac{|TP|}{|FN| + |TP|} \)

F-measure = \( 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \)

Overall = Recall \( \cdot \left( 2 - \frac{1}{\text{Precision}} \right) \)
Test cases

<table>
<thead>
<tr>
<th>#</th>
<th>Matching task</th>
<th>#nodes</th>
<th>max depth</th>
<th>#labels per tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Images vs Europe</td>
<td>4/5</td>
<td>2/2</td>
<td>6/5</td>
</tr>
<tr>
<td>2</td>
<td>Product schemas</td>
<td>13/14</td>
<td>4/4</td>
<td>14/15</td>
</tr>
<tr>
<td>3</td>
<td>Yahoo Finance vs Standard</td>
<td>10/16</td>
<td>2/2</td>
<td>22/45</td>
</tr>
<tr>
<td>4</td>
<td>Cornell vs Washington</td>
<td>34/39</td>
<td>3/3</td>
<td>62/64</td>
</tr>
<tr>
<td>5</td>
<td>CIDX vs Excel</td>
<td>34/39</td>
<td>3/3</td>
<td>56/58</td>
</tr>
<tr>
<td>6</td>
<td>Google vs Looksmart</td>
<td>706/1081</td>
<td>11/16</td>
<td>1048/1715</td>
</tr>
<tr>
<td>7</td>
<td>Google vs Yahoo</td>
<td>561/665</td>
<td>11/11</td>
<td>722/945</td>
</tr>
<tr>
<td>8</td>
<td>Yahoo vs Looksmart</td>
<td>74/140</td>
<td>8/10</td>
<td>101/222</td>
</tr>
<tr>
<td>9</td>
<td>Iconclass vs Aria</td>
<td>999/553</td>
<td>9/3</td>
<td>2688/835</td>
</tr>
</tbody>
</table>

Matching systems

Schema-based systems
- S-Match
- Cupid
- COMA
- Similarity Flooding as implemented in Rondo
- OAEI-2005 and OAEI-2006 participants

Systems were used in default configurations

PC: PIV 1.7Ghz; 512Mb. RAM; Win XP
Outline

- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
- Part IV: Evaluation (technology showcase)
  - Evaluation setup
  - Evaluation results
- Part V: Conclusions

Experimental results, test case #4

Cornell (mini) - Washington (mini)

Precision | Recall | Overall | F-measure | Time
---|---|---|---|---
Rondo | 0.8 | 0.9 | 0.85 | 1.0
Cupid | 0.7 | 0.8 | 0.75 | 0.8
COMA | 0.6 | 0.7 | 0.65 | 0.6
S-Match0 | 0.5 | 0.6 | 0.55 | 0.5
S-Match | 0.4 | 0.5 | 0.45 | 0.4

Experimental results, test case #5

BizTalk schemas: CIDX vs. Excel

Experimental results, #3,6,7,8: efficiency

Yahoo-Standard

Looksmart -Yahoo

Google-Yahoo

Google-Looksmart
Experimental results, #6,7,8: incompleteness

Recall, %

<table>
<thead>
<tr>
<th>Method</th>
<th>Recall (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>H-Match</td>
<td>13.38</td>
</tr>
<tr>
<td>Falcon</td>
<td>45.47</td>
</tr>
<tr>
<td>Autosms</td>
<td>15.72</td>
</tr>
<tr>
<td>RiMOM</td>
<td>26.84</td>
</tr>
<tr>
<td>OCIM</td>
<td>24.37</td>
</tr>
<tr>
<td>S-Match</td>
<td>46.1</td>
</tr>
</tbody>
</table>

OAEI-2006 contest results

Experimental results, #6,7,8: incompleteness (OAEI-2006 comparison)
Preliminary results, test case #9

<table>
<thead>
<tr>
<th></th>
<th>Precision, %</th>
<th>Recall, %</th>
<th>F-measure, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-Match</td>
<td>44.82</td>
<td>6.45</td>
<td>11.29</td>
</tr>
<tr>
<td>Iterative S-Match</td>
<td>47.69</td>
<td>6.6</td>
<td>11.59</td>
</tr>
</tbody>
</table>

Observations

- The dataset is hard and challenging
- Why do we have such a low recall?

- Gloss-like labels

  Aria: Top>Accessories>Jewelry
  Iconclass: Top>Nature>earth, world as celestial body>rock types; minerals and metals; soil types>rock types>precious and semiprecious stones>precious and semiprecious stones (with NAME)>precious and semiprecious stones: emerald

Outline

- Thesis contributions
- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
- Part IV: Evaluation (technology showcase)
- Part V: Conclusions
Summary

- Ontology matching applications and their requirements
- Overview of the state of the art, including classification of matching techniques and systems
- Semantic matching approach, including algorithms for basic, efficient and iterative semantic matching
- Evaluation of the approach on various data sets with encouraging results

Summary (cont’d)

- Automated reasoning techniques (e.g., SAT) provide good performance for industrial-strength matching tasks
- The issue is not efficiency but rather missing domain knowledge
  - This problem on the industrial size matching tasks is very hard
  - We have investigated it by examples of light weight ontologies, such as Google and Yahoo
  - Partial solution by applying semantic matching iteratively
Future challenges

- Missing background knowledge
- Interactive approaches
- Explanations of matching results
- Social and collaborative ontology matching
- Large-scale evaluation
- Infrastructures
- ...

Future challenges: scalability of visualization
References

- Project website - KNOWDIVE: http://www.dit.unitn.it/~knowdive/
Thank you for your attention and interest!