## Goals of the tutorial

- Illustrate the role of ontology matching
- Provide an overview of basic matching techniques
- Demonstrate the use of basic matching techniques in state of the art systems
- Motivate future research
<table>
<thead>
<tr>
<th>Problem</th>
<th>Classification</th>
<th>Basic techniques</th>
<th>Process</th>
<th>Systems</th>
<th>Other</th>
<th>Conclusions</th>
</tr>
</thead>
</table>

## Outline

1. The ontology matching problem
2. Classification
3. Basic techniques
4. Matching process
5. Systems
6. Other topics
7. Conclusions
Heterogeneity problem

Resources being expressed in different ways must be reconciled before being used.

Mismatch between formalized knowledge can occur when:

▶ different languages are used;
▶ different terminologies are used;
▶ different modelling is used.

Reconciliation can be achieved online or offline with different constraints.

Scope

▶ Reducing heterogeneity can be performed in 2 steps
  ▶ Match, thereby determine the alignment
  ▶ Process the alignment (merging, transforming, etc.)

▶ When do we match?
  ▶ Design time
  ▶ Run time
The Matching operation

- takes as input ontologies, 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.
- possibly exploiting techniques developed in a variety of fields, including linguistics, automated reasoning, statistics and data analysis, machine learning, etc.
Motivation: two ontologies

Transformation and mediation

```
SELECT x.doi
WHERE x : Book
AND x.author = "Bertrand Russell"
AND x.topic = "Bertrand Russell"

SELECT x.isbn
WHERE x : Autobiography
AND x.author = "Bertrand Russell"
```

mediator

x.doi=http://dx.doi.org/10.1080/041522862X
x.isbn=041522862X
Correspondence

Definition (Correspondence)
Given two ontologies $o$ and $o'$, a correspondence between $o$ and $o'$ is a 5-uple: $\langle id, e, e', r, n \rangle$ such that:

- $id$ is a unique identifier of the correspondence
- $e$ and $e'$ are entities of $o$ and $o'$ (e.g., XML elements, classes)
- $r$ is a relation (e.g., equivalence ($=$), more general ($\sqsubseteq$), disjointness ($\perp$))
- $n$ is a confidence measure in some mathematical structure (typically in the $[0,1]$ range)

Alignment

Definition (Alignment)
Given two ontologies $o$ and $o'$, an alignment ($A$) between $o$ and $o'$:

- is a set of correspondences on $o$ and $o'$
- with some additional metadata (multiplicity: 1-1, 1-*, method, date, properties, etc.)
Matching process

Application domains

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

- **Emergent**
  - P2P information sharing
  - Agent communication
  - Web service composition
  - Query answering on the web
Application: ontology evolution

Application: Catalog integration
Applications: P2P information sharing

Applications: Peer-to-peer and emergent semantics
Applications: Web service composition

Applications: Agent communication
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>

1. The ontology matching problem
2. Classification
3. Basic techniques
4. Matching process
5. Systems
6. Other topics
7. Conclusions
Matching dimensions

▶ Input dimensions
  ▶ Underlying models (e.g., XML, OWL)
  ▶ Schema-level vs. Instance-level

▶ Process dimensions
  ▶ Approximate vs. Exact
  ▶ Interpretation of the input

▶ Output dimensions
  ▶ Cardinality (e.g., 1-1, 1-*)
  ▶ Equivalence vs. Diverse relations (e.g., subsumption)
  ▶ Graded vs. Absolute confidence

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
Classification of schema-based techniques

Outline

1. The ontology matching problem
2. Classification
3. Basic techniques
4. Matching process
5. Systems
6. Other topics
7. Conclusions
Basic techniques: classification

Techniques are presented according to our classification:

- **Element-level techniques**
  - Terminological
    - String-based
    - Language-based
  - Constraint-based
  - Based on external resources
    - linguistic resources
    - ontologies

- **Global techniques**
  - Taxonomy-based
  - Graph-based

- **Extensional techniques**

- **Semantic techniques**

Distance, similarity, dissimilarity

Many of the techniques used are based on computing a distance or a similarity between ontology elements. These distances are for the sake of comparability normalized over the unit interval. They can turned into a boolean value by applying thresholds (e.g., S-match).
Element-level techniques: String-based

▶ **Prefix**
  ▶ takes as input two strings and checks whether the first string starts with the second one
  ▶ `net = network`; but also `hot = hotel`

▶ **Suffix**
  ▶ takes as input two strings and checks whether the first string ends with the second one
  ▶ `ID = PID`; but also `word = sword`

(e.g., COMA, SF, S-Match, OLA)

▶ **Edit distance**
  ▶ takes as input two strings and calculates the number of edition operations, (e.g., insertions, deletions, substitutions) of characters required to transform one string into another,
  ▶ normalized by length of the maximum string
  ▶ `EditDistance(NKN,Nikon) = 0.4`

(e.g., S-Match, OLA, Anchor-Prompt)
Element-level techniques: String-based

► N-gram
  ▶ takes as input two strings and calculates the number of common n-grams (i.e., sequences of $n$ characters) between them, normalized by $\max(\text{length}(\text{string1}), \text{length}(\text{string2}))$
  ▶ trigram(3) for the string nikon are nik, iko, kon

(e.g., COMA, S-Match)

Element-level techniques: Language-based

► Tokenization
  ▶ parses names into tokens by recognizing punctuation, cases
  ▶ Hands-Free_Kits $\rightarrow$ \texttt{\{hands, free, kits\}}

► Lemmatization
  ▶ analyses morphologically tokens in order to find all their possible basic forms
  ▶ Kits $\rightarrow$ Kit

(e.g., COMA, Cupid, S-Match, OLA)
Element-level techniques: Language-based

- **Elimination**
  - discards “empty” tokens that are articles, prepositions, conjunctions, etc.
  - a, the, by, type of, their, from

  (e.g., Cupid, S-Match)

Element-level techniques: Constraint-based

- **Datatype comparison**
  - integer < real
  - \{a, c, g, t\}[1 - 10] < \{a, c, g, u, t\} +

- **Multiplicity comparison**
  - [1 1] < [0 10]

Can be turned into a distance by estimating the ratio of domain coverage of each datatype.

(e.g., OLA, COMA)
Element-level techniques: Linguistic resources

- **Sense-based: WordNet**
  - A ⊆ B if A is a hyponym or meronym of B
  - Brand ⊆ Name
  - A ⊑ B if A is a hyponym or holonym of B
  - Europe ⊑ Greece
  - A = B if they are synonyms
  - Quantity = Amount
  - A ⊥ B if they are antonyms or the siblings in the part of hierarchy
    - Microprocessors ⊥ PC Board

(e.g., Artemis, CtxMatch, S-Match)

---

Element-level techniques: Linguistic resources

- **Sense-based: WordNet hierarchy distance**

Some other measures (e.g., Resnik measure) depends on the frequency of the terms in the corpus made of all the labels of the ontologies. (e.g., S-Match)
Element-level techniques: Linguistic resources

▶ Gloss-based: WordNet gloss comparison
  ▶ The number of the same words occurring in both input glosses increases the similarity value. The equivalence relation is returned if the resulting similarity value exceeds a given threshold
  ▶ Maltese dog is a breed of toy dogs having a long straight silky white coat
    Afghan hound is a tall graceful breed of hound with a long silky coat

(e.g., S-Match)

▶ Specific thesauri
  ▶ These usually store specific domain knowledge
  ▶ PO = Purchase Order
    uom = UnitOfMeasure
    line = item

(e.g., Cupid, COMA)
Structure-level techniques: Taxonomy-based

Ontologies are viewed as graph-like structures containing terms and their inter-relationships.

▶ **Bounded path matching**
  These take two paths with links between classes defined by the hierarchical relations, compare terms and their positions along these paths, and identify similar terms (e.g., Anchor-Prompt, NOM, QOM)

---

**Upward cotopic distance**
Measures the ratio of common super-classes.

\[
\delta_H(c, c') = 1 - \frac{|UC(c, H) \cap UC(c', H)|}{|UC(c, H) \cup UC(c', H)|}
\]

where \( UC(c, H) = \{c' \in H; c \leq c'\} \) is the set of superclasses of \( c \).

\[
\begin{align*}
\delta(a, a) &= 1 - 1 = 0 & \delta(b, c) &= 1 - 5/7 \approx .286 \\
\delta(a, e) &= 1 - 3/5 = .4 & \delta(c, d) &= 1 - 4/8 = .5 \\
\delta(a, f) &= 1 - 2/5 = .6 & \delta(a, b) &= 1 - 3/8 \approx .625 \\
\delta(d, a) &= 1 - 3/8 \approx .625
\end{align*}
\]
Structure-level techniques: Tree-based

- **Children**
  - Two non-leaf schema elements are structurally similar if their immediate children sets are highly similar

- **Leaves**
  - Two non-leaf schema elements are structurally similar if their leaf sets are highly similar, even if their immediate children are not

(e.g., Cupid, COMA)
Structure-level techniques: Graph-based

- Iterative fix point computation
  - If the neighbors of two nodes of the two ontologies are similar, they will be more similar.

(e.g., SF, OLA)

Structure-level techniques: Model-based

- Propositional satisfiability (SAT)

\[
Axioms \rightarrow rel(\text{context}_1, \text{context}_2)
\]

\[
\begin{align*}
\text{Electronics} & \leftrightarrow \text{Electronics}_2 \\
\text{Personal Computers} & \leftrightarrow \text{PC} \\
\text{Microprocessors} & \leftrightarrow \text{ID} \\
\text{PID} & \leftrightarrow \text{PC board}
\end{align*}
\]

\[
Axioms = \begin{cases}
(Electronics_1 \leftrightarrow Electronics_2) \land (\text{Personal Computers}_1 \leftrightarrow \text{PC}_2) \\
\text{context}_1 \\
(Electronics_1 \land \text{Personal Computers}_1) \leftrightarrow (Electronics_2 \land \text{PC}_2) \\
\text{context}_2
\end{cases}
\]

(e.g., CtxMatch, S-Match)
Structure-level techniques: Model-based

Description logics (DL)-based

\[
\begin{align*}
\text{micro-company} & = \text{company} \\
\cap \leq 5 \text{ employee} & \quad = \\
\text{SME} & = \text{firm} \\
\cap \leq 10 \text{ associate} & \\
\text{company} & = \text{firm} ; \text{associate} \sqsubseteq \text{employee} \\
\text{micro-company} & \sqsubseteq \text{SME}
\end{align*}
\]
Sequential composition

Data integration as sequential composition
Parallel composition

Many algorithms are based on similarity or distance computation. A number of operations can be based on similarity/distance matrices.
Sequential composition through distance matrices

Parallel composition through distance matrices
Aggregation operations

There are many different ways to aggregate matcher results, usually depending on confidence/similarity:

- **Triangular norms** (min, weighted products) useful for selecting only the best results;
- **Multidimensional distances** (Euclidean distance, weighted sum) useful for taking into account all dimensions;
- **Fuzzy aggregation** (min, weighted average) useful for aggregating competing algorithms and averaging their results;
- Other specific measures (e.g., ordered weighted average).

Dealing with cycles: fix point computation

\[ \sigma_C(c, c') = w_C^A \cdot \frac{1}{\max(|A(c)|, |A(c')|)} \sum_{(a, a') \in \text{match}(A(c), A(c'))} \sigma_A(a, a') + w_C^N \cdot \sigma(N(c), N(c')) \]

\[ \sigma_A(a, a') = w_C^A \cdot \sigma_C(\text{domain}(a), \text{domain}(a')) + w_N^A \cdot \sigma(N(a), N(a')) \]
Dealing with cycles: fix point computation

\[
\sigma_C(c, c') = \frac{1}{\max(|A(c)|, |A(c')|)} \sum_{(a, a') \in \text{match}(A(c), A(c'))} \sigma_A(a, a') + 4.\sigma(N(c), N(c'))
\]

\[
\sigma_A(a, a') = 6.\sigma_C(\text{domain}(a), \text{domain}(a')) + 4.\sigma(N(a), N(a'))
\]
Dealing with cycles: fix point computation

\[ \sigma_C(c, c') = \frac{1}{\max(|A(c)|, |A(c')|)} \sum_{(a, a') \in \text{match}(A(c), A(c'))} \sigma_A(a, a') + 4.\sigma(N(c), N(c')) \]

\[ \sigma_A(a, a') = 6.\sigma_C(\text{domain}(a), \text{domain}(a')) + 4.\sigma(N(a), N(a')) \]

Threshold reached: no .1 variation
Learning matcher (parameter)s

Learning algorithms

- Bayes learning
- WHIRL learner
- Neural networks
- Decision trees
- Stacked generalisation
Filtering similarities: thresholding

- **Hard threshold** retains all the correspondence above threshold $n$;
- **Delta threshold** consists of using as a threshold the highest similarity value out of which a particular constant value $d$ is subtracted;
- **Proportional threshold** consists of using as a threshold the percentage of the highest similarity value;
- **Percentage** retains the $n\%$ correspondences above the others.

Filtering similarities: Softening and hardening

Applies a monotonous function $f : [0, 1] \rightarrow [0, 1]$

- **Hardening** all correspondences with non-1 confidence are assigned 0 confidence;
- **Smoothening** (e.g., sigmoid) consists of using as a threshold the highest similarity value out of which a particular constant value $d$ is subtracted;
- **Weakening** consists of using as a threshold the percentage of the highest similarity value;
Extracting alignments

<table>
<thead>
<tr>
<th></th>
<th>Book</th>
<th>Translator</th>
<th>Publisher</th>
<th>Writer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>.84</td>
<td>0.</td>
<td>.90</td>
<td>.12</td>
</tr>
<tr>
<td>Provider</td>
<td>.12</td>
<td>0.</td>
<td>.84</td>
<td>.60</td>
</tr>
<tr>
<td>Creator</td>
<td>.60</td>
<td>.05</td>
<td>.12</td>
<td>.84</td>
</tr>
</tbody>
</table>

▶ **Greedy algorithm**: 1.96;
▶ **Stable marriage**: 2.1;
Extracting alignments

<table>
<thead>
<tr>
<th></th>
<th>Book</th>
<th>Translator</th>
<th>Publisher</th>
<th>Writer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>.84</td>
<td>0.</td>
<td>.90</td>
<td>.12</td>
</tr>
<tr>
<td>Provider</td>
<td>.12</td>
<td>0.</td>
<td>.84</td>
<td>.60</td>
</tr>
<tr>
<td>Creator</td>
<td>.60</td>
<td>.05</td>
<td>.12</td>
<td>.84</td>
</tr>
</tbody>
</table>

▶ **Greedy algorithm**: 1.96;
▶ **Stable marriage**: 2.1;
▶ **Maximal weight match**: 2.52.

Outline

1. The ontology matching problem
2. Classification
3. Basic techniques
4. Matching process
5. Systems
6. Other topics
7. Conclusions
State of the art systems

50+ matching systems exist, . . . we consider some of them

- Cupid (U. of Washington, Microsoft Corporation and U. of Leipzig)
- Falcon-AO (China Southwest U.)
- OLA (INRIA Rhône-Alpes and U. de Montréal)
- S-Match (U. of Trento)
- . . .

Cupid

- Schema-based
- Computes similarity coefficients in the [0 1] range
- Performs linguistic and structure matching
- Sequential system
Cupid architecture

OLA

- Schema- and Instance-based
- Computes dissimilarities + extracts alignments (equivalences in the [0 1] range)
- Based on terminological (including linguistic) and structural (internal and relational) distances
- Neither sequential nor parallel
OLA architecture

Falcon-OA architecture
S-Match

- Schema-based
- Computes equivalence (=), more general (⊇), less general (⊆), disjointness (⊥)
- Analyzes the meaning (concepts, not labels) which is codified in the elements and the structures of ontologies
- Sequential system with a composition at the element level
## What is an alignment for?

- Processing them and generating processing output (transformations, axioms, rules);
- Evaluating and comparing them;
- Storing, finding, and floating around;
- Piping alignments algorithms (improving an existing alignment);
- Manipulating (thresholding and hardening);
Processing alignments: operations

- \( \text{Merge}(o, o', A) = o'' \)
- \( \text{Transform}(o, o', A) = o'' \)
- \( \text{Translate}(d, A) = d' \)
- \( \text{TransformQuery}(q, A) = q' \) and \( \text{Translate}(a', \text{Invert}(A)) = a \)
- \( \text{TransformAsRules}(A) = o \)

Evaluation of matching algorithms

Goal: improvement of matching algorithms through comparison, measure of the evolution of the field.

- Yearly campaign comparing algorithms on different test benches;
- Participants submit their alignments in a standard format;
- These are compared with available reference alignments;
- Deviation is measured by classic measures such as precision and recall;
- Test and results are published on our web site.

http://oaei.ontologymatching.org
Alignment API

Examples of API use

```java
OWLOntology O1 = loadOntology(...);
OWLOntology O2 = loadOntology(...);
Alignment A1 = new SubsDistNameAlignment(O1, O2);
Alignment A2 = new PropSubsDistAlignment(O1, O2);
Alignment A3 = new NameAndPropertyAlignment(O1, O2);
A1.align(); A1.threshold(.5);
A2.align(); A3.align(A2);
Evaluator E = new PRecEvaluator(A1, A3);
E.eval();
if ( E.getPrecision() > .6 )
    A3.render(..., SWRLRendererVisitor);
```
Outline

1 The ontology matching problem
2 Classification
3 Basic techniques
4 Matching process
5 Systems
6 Other topics
7 Conclusions

Summary

- Ontology heterogeneity is the nature of the semantic web;
- Ontology matching is part of the solution;
- It can be based on many different techniques;
- There already are numerous systems there;
- A relatively solid research field has emerged (tools, formats, evaluation, etc.) and is making progress;
- But there remains serious challenges ahead.
Challenges

- Using background knowledge
- Performance of systems
- Interactive approaches
- Explanations of matching
- Social aspects of ontology matching
- Large-scale evaluation
- Infrastructures
- ...

Acknowledgments

We thank all the participants of the Heterogeneity workpackage of the Knowledge Web network of excellence

In particular, we are grateful to Than-Le Bach, Jesus Barrasa, Paolo Bouquet, Jan De Bo, Jos De Bruijn, Rose Dieng-Kuntz, Enrico Franconi, Raúl García Castro, Manfred Hauswirth, Pascal Hitzler, Mustafa Jarrar, Markus Krötzh, Ruben Lara, Malgorzata Mochol, Amedeo Napoli, Luciano Serafini, François Sharfe, Giorgos Stamou, Heiner Stuckenschmidt, York Sure, Vojtěch Svátek, Valentina Tamma, Sergio Tessaris, Paolo Traverso, Raphaël Troncy, Sven van Acker, Frank van Harmelen, and Ilya Zaihrayeu.

And more specifically to Marc Ehrig, Fausto Giunchiglia, Loredana Laera, Diana Maynard, Deborah McGuinness, Petko Valchev, Mikalai Yatskevich, and Antoine Zimmermann for their support and insightful comments
Ontology matching the book

Pavel Shvaiko, Jérôme Euzenat

Ontology matching

1. Applications
2. Problem definition
3. Classification
4. Basic techniques
5. Strategies
6. Systems
7. Evaluation
8. Representation
9. Explanation
10. Processing

http://book.ontologymatching.org

Questions?

pavel@dit.unitn.it
Jerome.Euzenat@inrialpes.fr

http://www.ontologymatching.org