Open Information Systems 2019-2020

Lecture 9: Ontology Matching

Christophe Debruyne

While ontologies and KBs are different, we will – in this lecture – use the term “ontology” for both unless otherwise explicitly specified.

Semantic Heterogeneity

Semantic heterogeneity are variations in meaning or ambiguity in interpretations [1], which remains an challenge in database research (and other CS disciplines).

Semantic heterogeneity can be caused by:

- Ontologies sharing concepts, yet built for
  - different tasks or
  - by different communities
  - resulting in variation in scope, structure, and granularity

Example from the GLAM sector:
Dublin Core: names of people are literals
CIDOC CRM: names are instances of *Appellation* identifying instances of *Person*, and relationships between appellations can be declared.

Semantic Heterogeneity

Semantic heterogeneity can be caused by:

- The use of different ontology (or representation) languages: OWL vs. SQL
- Using different terminologies
  - within the same language (e.g., jargon and synonyms) and
  - across languages (also note *language variation*).
- Ontologies evolving over time (versioning)
- …

The Ontology Matching Problem

Semantic heterogeneity is typically tackled in two steps:

1. detecting the correspondences between the different ontologies, which constitute an alignment, and
2. interpreting these correspondences to create an executable mapping with respect to the application needs.

Why Ontology Matching (OM)? OM techniques are useful for :

• Schema & data integration	[integrating data w/ common schema]
• Data interlinking	[linking instances]
• Data transformation	[from one representation to another]
• Query answering	[via a common ontology]
• …	

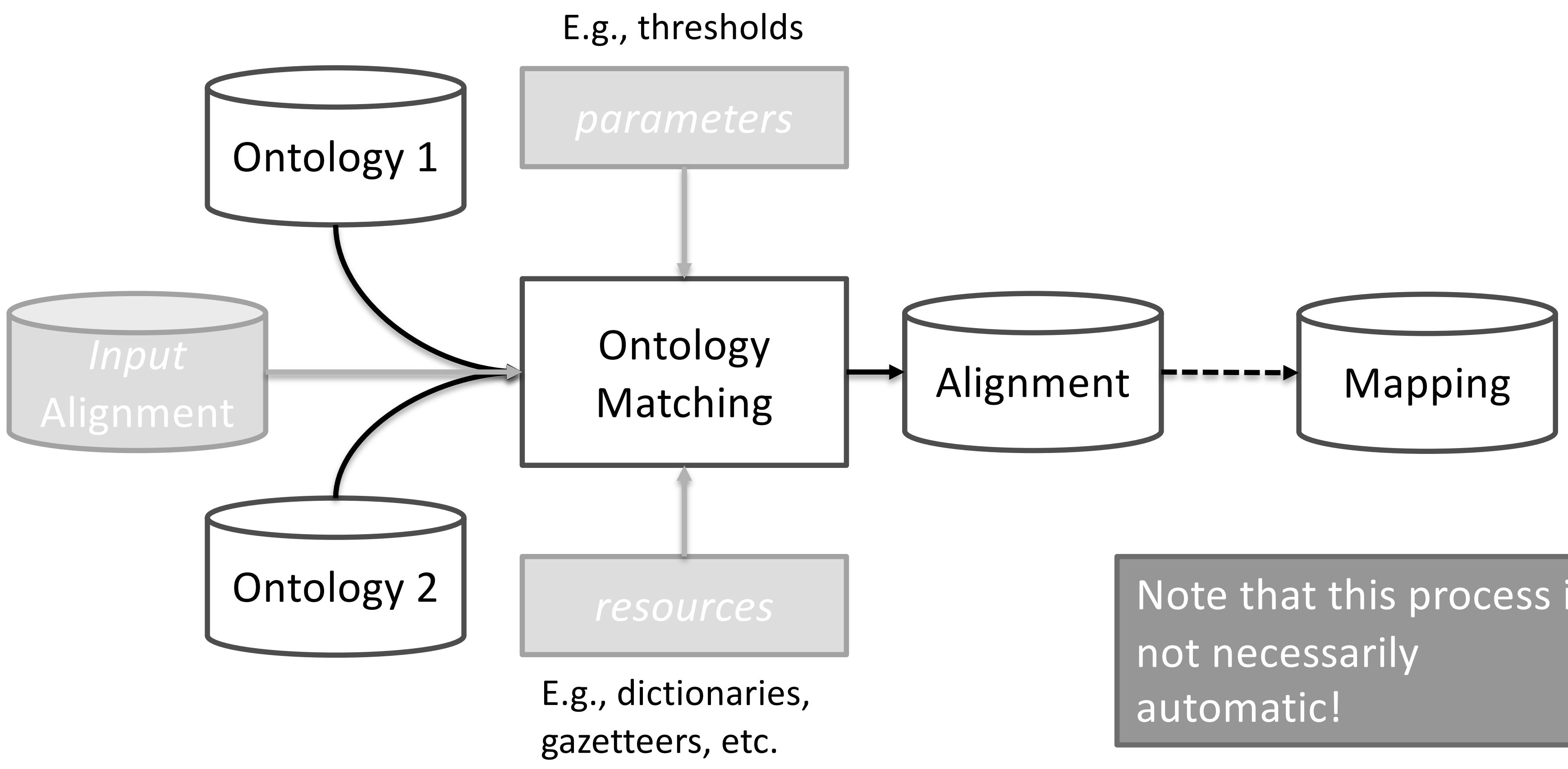
The Ontology Matching Problem

The Matching Problem is not new, it has been studies in the context of data warehousing and data integration.

Like ontologies, (database) schemas provide a vocabulary for describing a UoD. Both ontologies and schemas also provide ways that constrain the *use* of that vocabulary. There are some important differences, however:

Ontologies provide explicit semantics. The is-a relationship, for instance, is translated into a foreign key from one table to another. And an ontology is a logical system allowing you to reason over axioms.

The Ontology Matching Problem



Note that this process is not necessarily automatic!

Figure based on [1]

Definitions and Terminology

Given two ontologies O1 and O2, a correspondence between O1 and O2 is a tuple <e1, e2, r> such that: e1 and e2 are respectively entities of O1 and O2 (classes, properties,…), and r is a relation (equivalence, more specific/general, disjoint,…)

Given two ontologies O1 and O2, an alignment between O1 and O2 is a set of correspondences between O1 and O2 with additional metadata for each correspondence (cardinalities, provenance, confidence,…).

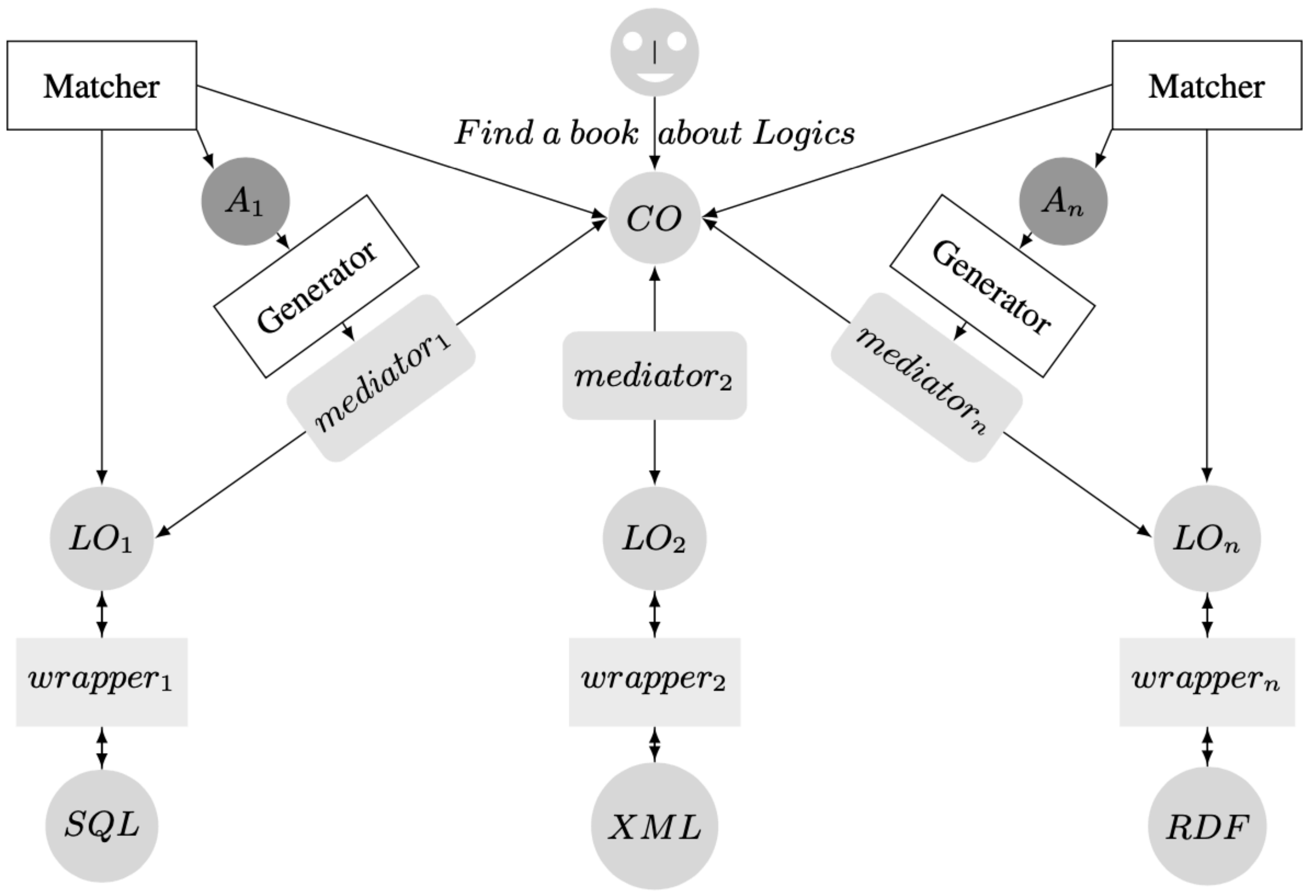
Ontology Matching is the process of finding correspondences between ontologies. An alignment is a set of correspondences between two ontologies and it the result of the matching process.

Scenarios: information integration and querying [3]

LO = local ontology
CO = common ontology

The alignment is used to generate a mediator that transform, in some way, queries against the CO into queries against a LO.

Whether matching happens at design time or at run time depends on the project.

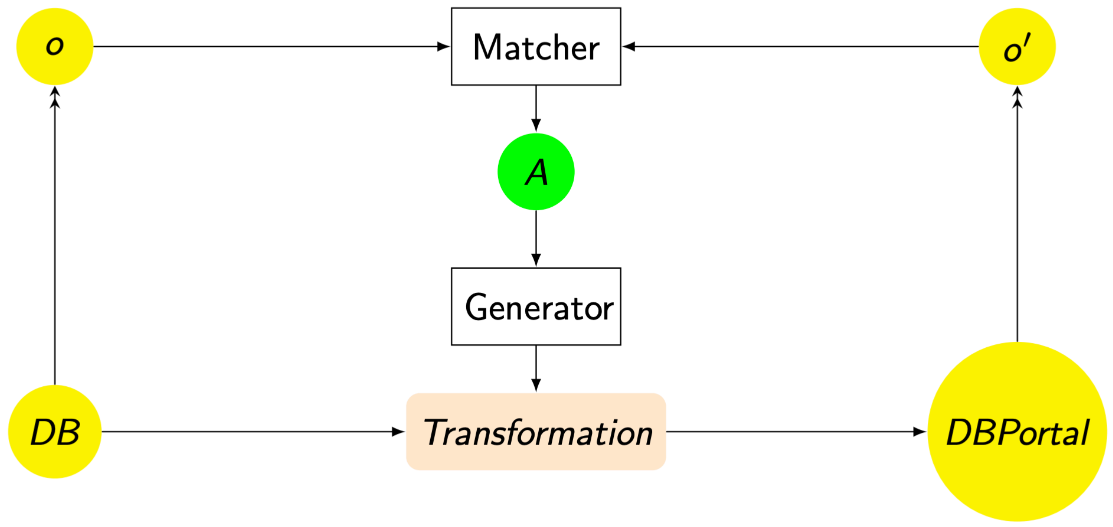


Source: [3]

Scenarios: Catalog Integration [3]

Note that this scenario is not only limited to market places.

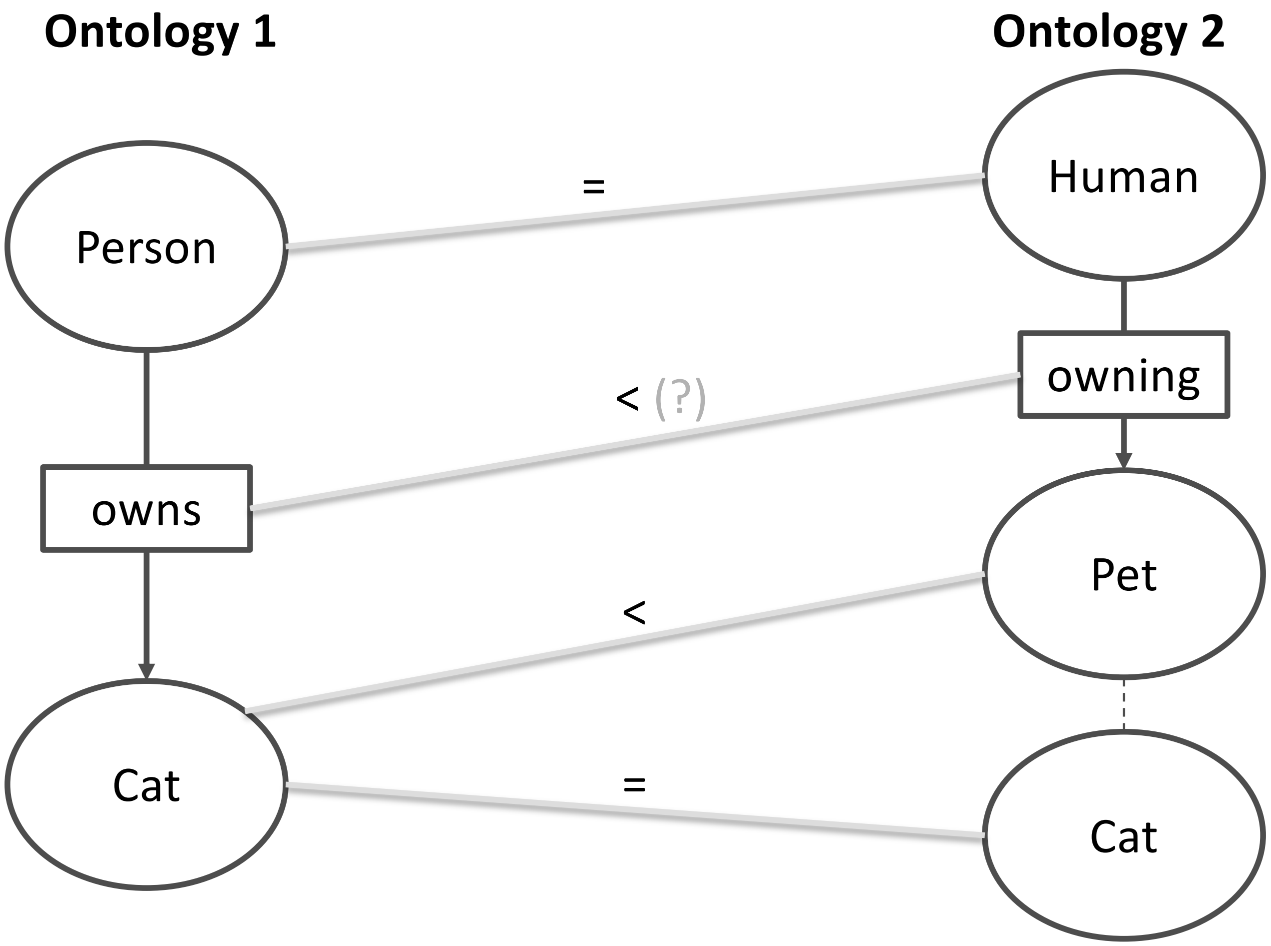
Merchant matches their catalog with that of a market place. The alignment is used to generate transformation to load the merchant's data into the market place (e.g., a portal).



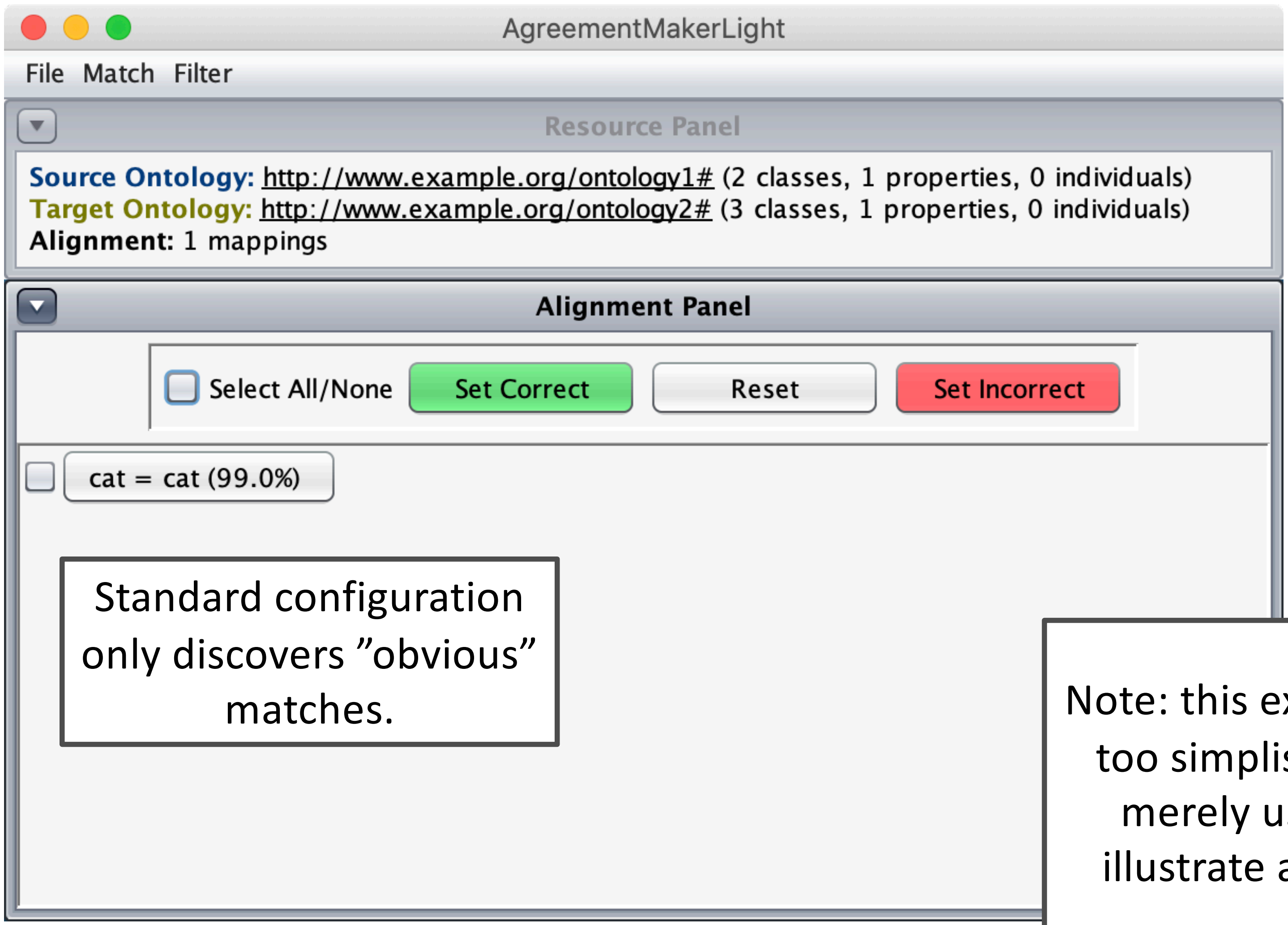
Source: [3]

A very simple example

What relations do we hope to discover?



AgreementMaker Light [2]



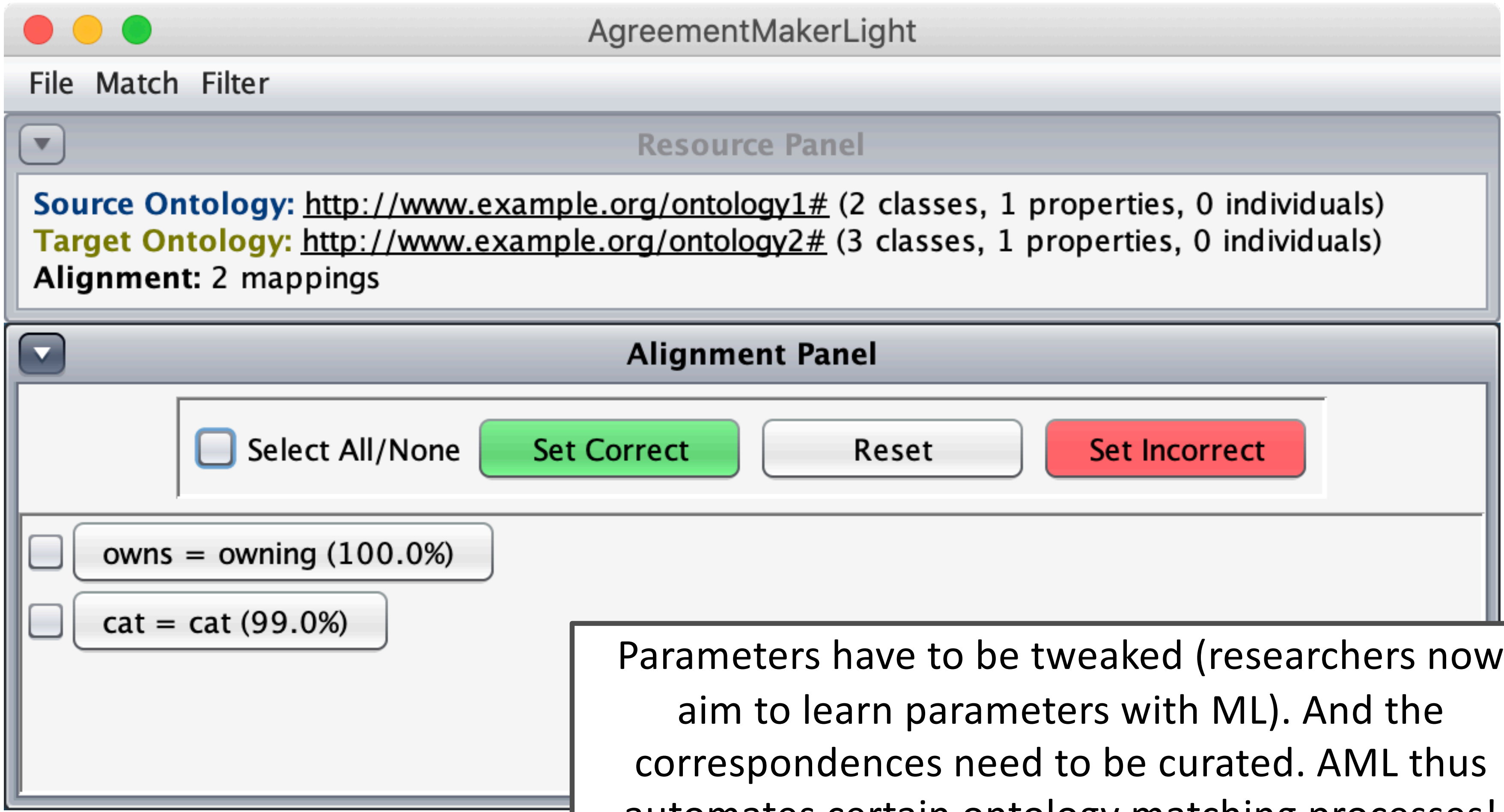
Source Ontology: http://www.example.org/ontology1# (2 classes, 1 properties, 0 individuals)
Target Ontology: http://www.example.org/ontology2# (3 classes, 1 properties, 0 individuals)
Alignment: 1 mappings

cat = cat (99.0%)

Standard configuration only discovers ”obvious” matches.

Note: this example is too simplistic and merely used to illustrate a point.

AgreementMaker Light [2]



Source Ontology: http://www.example.org/ontology1# (2 classes, 1 properties, 0 individuals)
Target Ontology: http://www.example.org/ontology2# (3 classes, 1 properties, 0 individuals)
Alignment: 2 mappings

owns = owning (100.0%)
cat = cat (99.0%)

Parameters have to be tweaked (researchers now aim to learn parameters with ML). And the correspondences need to be curated. AML thus automates certain ontology matching processes!

Ontology Matching with AML

- The matcher can be configured and finds candidate correspondences.
- The "Alignment Panel" contains the set of correspondences and is initially filled with those found by the matcher.
- Authors curate these correspondences by adding, removing, and editing correspondences in the alignment.

The alignment can be exported in a variety of formats.

The Alignment

- The matcher can be configured and finds candidate correspondences.
- The "Alignment Panel" contains the set of correspondences and is initially filled with those found by the matcher.
- Authors **curate** these correspondences by adding, removing, and editing correspondences in the alignment.
- The alignment can be stored in a variety of formats.
  - Alignment Format (AF) for simple correspondences and Expressive and Declarative Ontology Alignment Language (EDOAL) for complex correspondences
  - While not standardized, these formats are used within the community for benchmarking purposes.

The Alignment Format (for simple correspondences)

```
<?xml version='1.0' encoding='utf-8'?>
<rdf:RDF
xmlns='http://knowledgeweb.semanticweb.org/heterogeneity/alignment'
     xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
     xmlns:xsd='http://www.w3.org/2001/XMLSchema#'
     alignmentSource='AgreementMakerLight'>
<Alignment>
    <xml>yes</xml>
    <level>0</level>
    <type>11</type>
    <onto1>http://www.example.org/ontology1#</onto1>
    <onto2>http://www.example.org/ontology2#</onto2>
    <uri1>http://www.example.org/ontology1#</uri1>
    <uri2>http://www.example.org/ontology2#</uri2>
    <!– OMMITTED FOR BREVITY, SEE NEXT SLIDE -->
</Alignment>
</rdf:RDF>
```

The Alignment Format
(for simple correspondences)

```
<map>
    <Cell>
        <entity1 rdf:resource="http://www.example.org/ontology1#owns"/>
        <entity2 rdf:resource="http://www.example.org/ontology2#owning"/>
        <measure rdf:datatype="http://.../XMLSchema#float">1.0</measure>
        <relation>=</relation>
    </Cell>
</map>
<map>
    <Cell>
        <entity1 rdf:resource="http://www.example.org/ontology1#Cat"/>
        <entity2 rdf:resource="http://www.example.org/ontology2#Cat"/>
        <measure rdf:datatype="http://.../XMLSchema#float">0.99</measure>
        <relation>=</relation>
    </Cell>
</map>
```

Expressive and Declarative OAL

```
ont1:Cat a edoal:Class .
ont2:Pet a edoal:Class .
ont2:species a edoal:Relation .

[] a align:Cell ;
    align:entity1 ont1:Cat ;
    align:entity2 [
      a edoal:Class ;
      edoal:and ( ont2:Pet
                  [ a edoal:AttributeValueRestriction ;
                    edoal:onAttribute ont2:species ;
                    edoal:comparator edoal:equals ;
                    edoal:value [
                      a edoal:Literal ;
                      edoal:string "Cat" ;
                      edoal:type ”...XMLSchema#string" ] ] ) ] ;
    align:measure "1.0"^^xsd:float ;
    align:relation "=" .
```

Ontology Matching

- Now that we have seen an example, let’s consider what we can use for the purpose of ontology matching.
- There are broadly two categories: what’s inside an ontology, and what can be consulted outside the ontology.

Ontology Matching

Inside an ontology we have:

- (computer)labels, names, comments, natural language text, … for which we can use string metrics and NLP techniques.
- Class and property axioms constraining their intended use such as domain and range declarations, datatypes, …
- The data (extensional semantics) allowing us to avail of, for instance,
  - ML techniques to discover characteristics
  - Or compare populations with a set-theoretic approach
- The intensional semantics and the ontology language allowing us to use reasoners

Ontology Matching

Outside the ontology we have [remember the external resources in the image]:

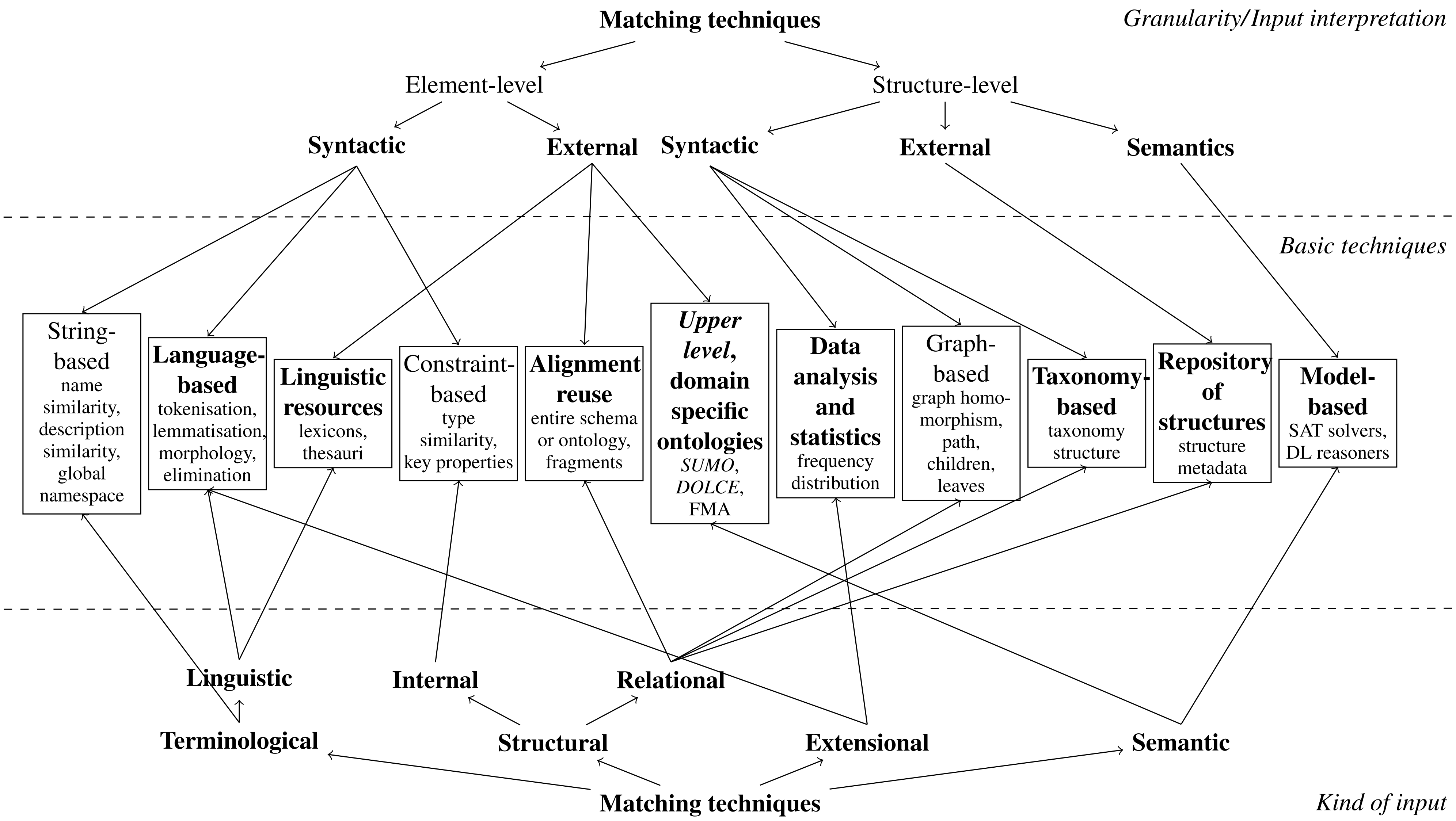
- Annotations, corpora, …
- External graphs such as DBpedia, YAGO, and WikiData
- External resources such as WordNet (an association base with word senses), dictionaries, thesauri, etc.
- Knowledge of subject matter experts or users
  - Mechanical Turk
  - Crowd and Niche Sourcing
  - Formalizing background knowledge
  - …

Matching Dimensions

- Input Dimensions
  - Underlying models (e.g., XML, OWL, combinations,…)
  - Are matching the schema (concepts and relations) or instances?
- Process Dimensions
  - Approximate or exact matches?
  - Feedback from subject matter experts
  - Interpretation of the input
- Output Dimensions
  - Cardinality
  - Equivalence, subsumption, or other relations?
  - Confidence interval? {0,1} vs. [0,1]

Source [3]

“You want to match literals in an XML files.”



Matching techniques
Granularity/Input interpretation
Element-level
Structure-level
Syntactic
External
Syntactic
External
Semantics
Basic techniques
String-based name similarity, description similarity, global namespace
Language-based tokenisation, lemmatisation, morphology, elimination
Linguistic resources lexicons, thesauri
Constraint-based type similarity, key properties
Alignment reuse entire schema or ontology, fragments
Upper level, domain specific ontologies SUMO, DOLCE, FMA
Data analysis and statistics frequency distribution
Graph-based graph homo-morphism, path, children, leaves
Taxonomy-based taxonomy structure
Repository of structures structure metadata
Model-based SAT solvers, DL reasoners
Linguistic
Internal
Relational
Terminological
Structural
Extensional
Semantic
Matching techniques
Kind of input

“You want to match the names of diseases.”

Element Level Syntactic Techniques

- String-based:
  - exact matches
  - string metrics
  - Soundex
  - edit distances
- Language-based:
  - Tokenization
  - Lemmatization (transforming into “basic” forms; Cats → Cat)
  - removing non-informative tokens (e.g., articles)
- Linguistic:
  - relationships between words (hypernym, hyponym, antonym, synonym, …)
  - Similarity between natural language definitions (glosses)
  - WordNet provides both and is a popular resource

Structure Level Syntactic Techniques

- Approaching ontologies as graphs
- Taxonomies and Trees
  - Given two entities from two different ontologies, take all the parents and children of their respective taxonomy (or other hierarchy) and compare the entities, their placement in their hierarchies, and the similarity of the terms of their taxonomies.
  - Given two entities, if their subclasses in their ontologies are similar, then it is likely the those entities are similar as well.
  - Given two entities, if their children in their ontologies are similar, then it is likely the those entities are similar as well.

Structure Level Semantic Techniques

Model-based: using OWL reasoners to infer relationships between entities.

$$\Omega_1 = \{MicroCompany \equiv Company \sqcap \leq 5 has.Associate\}$$
$$\Omega_2 = \{SME \equiv Firm \sqcap \leq 10 has.Employee\}$$

- Assuming we “discovered” that
  - Company and Firm are synonyms (and thus equivalent)
  - Associates are “junior” employees and thus a subclass of Employee
  - Both ”has” relations denote the meronomy relation
- Then we can infer that $MicroCompany \sqsubseteq SME$. Why?

The following exam is *based on* [3].

Creating the mapping

Let’s take one of the correspondences of our example:

```
<Cell>
  <entity1 rdf:resource="http://www.example.org/ontology1#Cat"/>
  <entity2 rdf:resource="http://www.example.org/ontology2#Cat"/>
  <relation>=</relation>
</Cell>
```

How can we implement this into a mapping?

A mapping can take many forms: OWL, SPARQL, rules, …

The appropriate form depends on the application.

Creating the mapping

Let’s take one of the correspondences of our example:

```
<Cell>
  <entity1 rdf:resource="http://www.example.org/ontology1#Cat"/>
  <entity2 rdf:resource="http://www.example.org/ontology2#Cat"/>
  <relation>=</relation>
</Cell>
```

Assume that we want all instances of ont2:Cat to be considered instances of ont2:Cat. How shall we go about doing this?

In OWL: $ont2{:}\,Cat \equiv ont1{:}\,Cat$

And in SPARQL?

Creating the mapping

Assume that we want all instances of ont2:Cat to be considered instances of ont2:Cat. How shall we go about doing this?

In SPARQL:

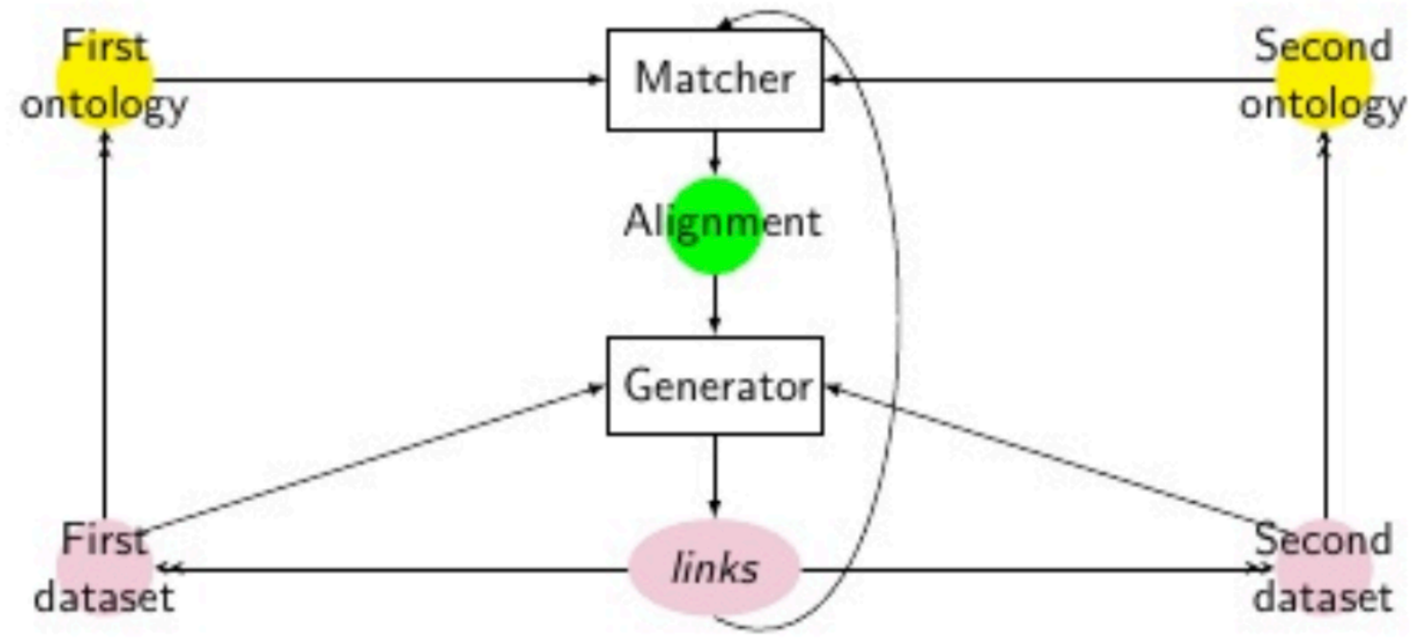
```
CONSTRUCT {
  ?x a ont1:Cat .
} WHERE {
 ?x a ont2:Cat .
}
```

Research Challenges in OM

- Large-Scale Matching Evaluation
- Efficiency of Matching Techniques
- Matching with Background Knowledge
- Selecting, Combining, and Parameterizing Matchers
- User Involvement
- Social and Collaborative Matching
- Alignment Management: Infrastructure and Support

Linking Linked Data Resources

We mainly focused on finding correspondences between ontologies. What if we want to detect correspondences between instances?



Linking Linked Data Resources

Discovering links is a challenging task. Scalability is an issue (there are billions of triples), data is heterogenous (both in representation and semantics), and data covers many domains.

Link Discovery tools allow one to create links between two datasets. While researcher look into the automatic detection of correspondences, most tools generate links based on user-defined rules (or mappings).

Those can, in some cases, take on the shape of a SPARQL CONSTRUCT query or a SWRL rule, but other tools exist such as SILK [4] and LIMES [8].

Linking Linked Data Resources

The Linked Data Integration Framework (SILK) framework [4,5] allows you to create an alignment that declares how different resources in two different Linked Data datasets are related.

http://silkframework.org/

That alignment can be created manually, though people are studying how to generate and refine those [6].

Let's assume that I have a dataset that I want to interlink with DBpedia. I have a dataset of boundary data for counties in Ireland. I know that those labels correspond with resources in DBpedia of a particular type and that their names should match.

```
<?xml version="1.0"?>
<Silk>
  <Prefixes>
    <!-- omitted, contains RDF, DCT, OWL,... -->
  </Prefixes>
  <DataSources>
    <Dataset id="dbpedia" type="sparqlEndpoint">
      <Param name="endpointURI" value="http://dbpedia.org/sparql"/>
      <!-- some parameters omitted -->
    </Dataset>
    <Dataset id="osi" type="sparqlEndpoint">
      <Param name="endpointURI" value="http://.../query"/>
      <!-- some parameters omitted -->
    </Dataset>
  </DataSources>
  <Interlinks>
    <Interlink id="link-counties">
```

First you select the datasets that you want to interlink and …

… restrict the ”types” of entities inside those datasets for the interlinking process. Note the use of source and target, this mapping has a direction.

```
      <SourceDataset
        dataSource="osi"
        var="b"
        typeUri="http://ontologies.geohive.ie/osi#County" />

      <TargetDataset
        dataSource="dbpedia"
        var="a"
        typeUri="http://dbpedia.org/ontology/Place">
        <RestrictTo>
?a
  &lt;http://purl.org/dc/terms/subject&gt;
    &lt;http://.../Category:Counties_of_the_Republic_of_Ireland&gt; .
        </RestrictTo>
      </TargetDataset>
```

```
<LinkageRule linkType="ov:similarTo">
  <Aggregate required="false" weight="1" type="min">
    <Compare required="false" weight="1"
      metric="levenshteinDistance" threshold="0.0" indexing="true">
      <TransformInput function="lowerCase">
        <Input path="?b/rdfs:label"/>
      </TransformInput>
      <TransformInput function="removeBlanks">
        <TransformInput function="replace">
          <TransformInput function="lowerCase">
            <Input path="?a/rdfs:label"/>
          </TransformInput>
          <Param name="search" value="county"/>
          <Param name="replace" value=""/>
        </TransformInput>
      </TransformInput>
      <Param name="minChar" value="0"/>
      <Param name="maxChar" value="z"/>
    </Compare>
  </Aggregate>
  <Filter/>
</LinkageRule>
```

Then you specify how those to entities are compared.

This is usually a manual process, but researchers are investigating way to learn those rules.

In OSI's dataset, the label "Dublin" for County Dublin becomes "dublin".

In DBpedia, the label "County Dublin" for County Dublin becomes "Dublin"

ov:similarTo was chosen over owl:sameAs

Linkage rules are a combination of RDF paths, transformations, similarity metrics, and aggregations.

```
    <Transforms>
  </Transforms>
  <Outputs>
    <Output type="file" minConfidence="0.95">
      <Param name="file" value="accepted-county-links.nt"/>
      <Param name="format" value="N-TRIPLES"/>
    </Output>
    <Output type="file" maxConfidence="0.95">
      <Param name="file" value="verify-county-links.nt"/>
      <Param name="format" value="N-TRIPLES"/>
    </Output>
  </Outputs>
</Silk>
```

Different output files for different confidence intervals.

Linking Counties of the ROI was easy (there were only 26) and we are pretty sure they represent the same "things", but townlands, on the other hand, are more challenging.

Linking Linked Data Resources

Many townlands in Ireland share a name, even within a county! Centroids in DBpedia were different (OSM), than those of OSi (legal).

What did we do? If there were part-of relations for a townland, then we could compare the parent. But, again, sometimes you had five townlands with a similar name in one county. If lat/longs of DBpedia and OSi were not too far off (<5km), we deem them close enough (as a third rule)

Here, the use of voc:similarTo made certainly sense as there are resources in DBpedia where it was impossible for us to be certain that they referred to the same thing.

Conclusions

- Ontology Matching is a two step process:
  - Detecting correspondences, and
  - Create an alignment

- Depending on the task, that alignment (created on the fly or not) is used to generate an artefact for
  - mapping, transforming, integrating…
  - data, queries,…
  - from one to the other

- We have discussed some of the various aspects that can be taken into account whilst matching ontologies and also looked at the problem of correspondence detection with SILK

References

1. Shvaiko, Pavel, and Jérôme Euzenat. "Ontology matching: state of the art and future challenges." IEEE Transactions on knowledge and data engineering 25.1 (2011): 158-176.
2. Daniel Faria, Catia Pesquita, Emanuel Santos, Matteo Palmonari, Isabel F. Cruz, and Francisco M. Couto. “The AgreementMakerLight Ontology Matching System.” OTM Conferences 2013 (2013): 527-541
3. Jérôme Euzenat, Pavel Shvaiko: Ontology Matching, Second Edition. Springer 2013, ISBN 978-3-642-38720-3, pp. I-XVII, 1-511
4. Julius Volz, Christian Bizer, Martin Gaedke, Georgi Kobilarov: Discovering and Maintaining Links on the Web of Data. International Semantic Web Conference 2009: 650-665
5. Julius Volz, Christian Bizer, Martin Gaedke, Georgi Kobilarov: Silk - A Link Discovery Framework for the Web of Data. LDOW 2009
6. Anna Primpeli, Christian Bizer: Robust Active Learning of Expressive Linkage Rules. WIMS 2019: 2:1-2:7
7. Axel-Cyrille Ngonga Ngomo, Sören Auer:
8. LIMES - A Time-Efficient Approach for Large-Scale Link Discovery on the Web of Data. IJCAI 2011: 2312-2317