TECHNISCHE UNIVERSITÄT DRESDEN

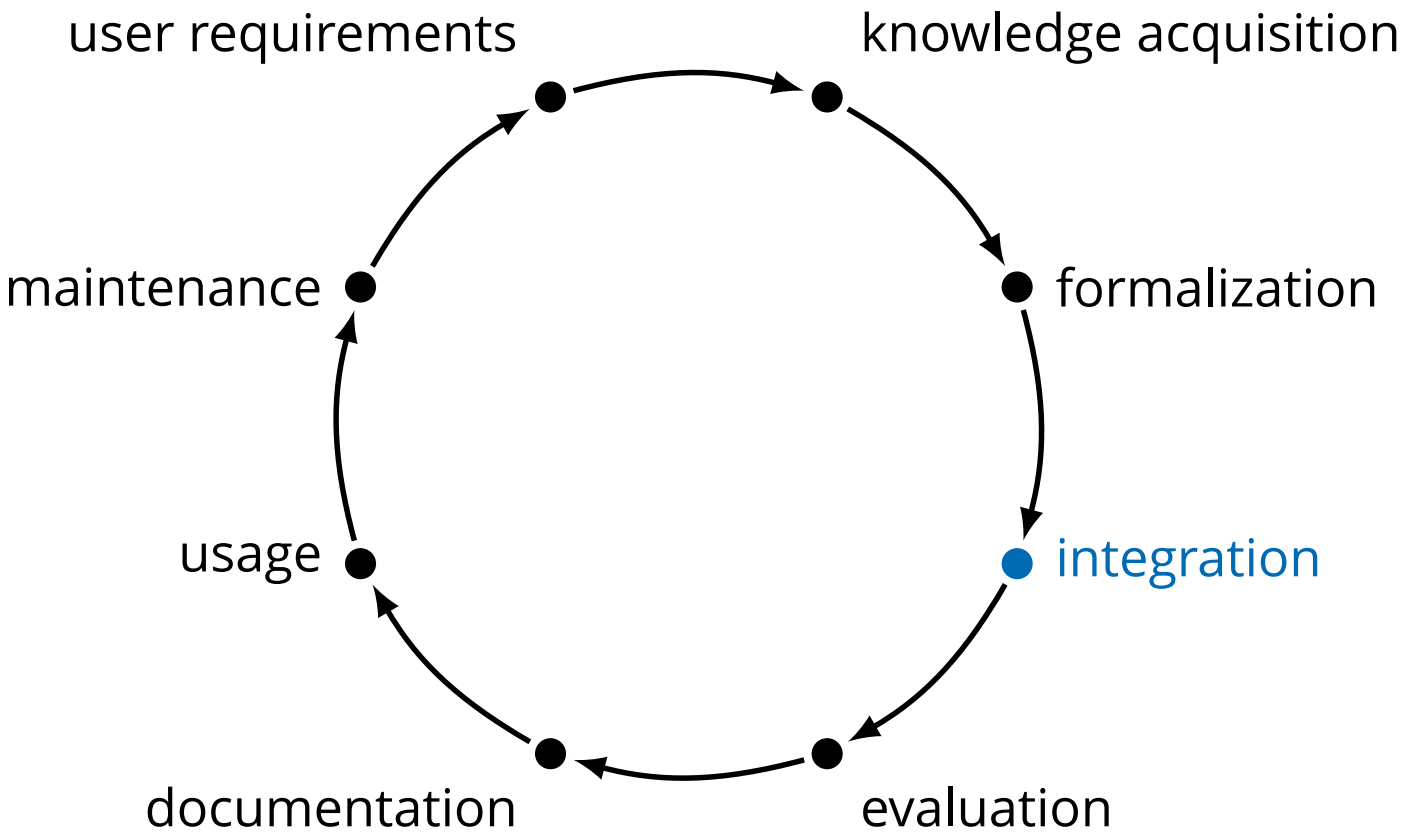
Stefan Borgwardt

Institute of Theoretical Computer Science, Chair of Automata Theory

Logic-Based Ontology Engineering

Part 3: Ontology Integration

The Ontology Life Cycle



Outline

3.1 Matching, Alignments, and Similarity

Problem Setting

NCI Thesaurus:

Class: http://ncicb.nci.nih.gov/xml/owl/EVS/Thesaurus.owl#C20480
Annotations: rdfs:label "Cellular Process"

Gene Ontology:

Class: http://purl.obolibrary.org/obo/GO_0009987
Annotations: rdfs:label "cellular process"
has_exact_synonym "cell physiology"

How can we find and represent correspondences between entities in different ontologies?

The Solution

UMLS Metathesaurus (not in OWL format):

C0007613 "Cell physiology"
  Related (other): C1325880
  Atom: http://ncicb.nci.nih.gov/xml/owl/EVS/Thesaurus.owl#C20480

C1325880: "cellular process"
  Atom: http://purl.obolibrary.org/obo/GO_0009987

The goal of ontology integration is to provide an automated way of finding such correspondences, to foster reuse and transfer of knowledge.

Applications of Ontology Integration

- Ontology creation
- Ontology evolution, versioning (see Part 4: Ontology Maintenance)
- Schema integration
  - several local information sources are linked to a single ontology
  - queries over the ontology are translated into queries over the local sources; the answers are translated back
- Schema merging
  - local information sources have to be merged into a single one
  - more than just linking: all entities have to be merged
  - much stronger requirements on correctness
- Web service composition, query answering on the Deep Web

> The main challenge is that the information sources (ontologies) are very heterogenous, due to various factors.

Types of Heterogeneity

Syntactic:

- different ontology languages
- use automatic translations where possible

Terminological:

- different naming conventions, synonyms
- different languages or technical jargon
- techniques from natural language processing

Semantic:

- differences in modeling
- different coverage (disease vs. medicine vs. anatomy)
- different granularity (disease names vs. detailed definitions)
- different perspective (diagnosis vs. disease classification)

Semiotic:

- different usage
- hard to handle automatically

Methods for Linking Ontologies

- Importing one ontology into the other: too strong
- Using prefixes: imports only the vocabulary, too weak
- Annotate entities with links to another ontology: no logical semantics
- Create a third ontology, called alignment

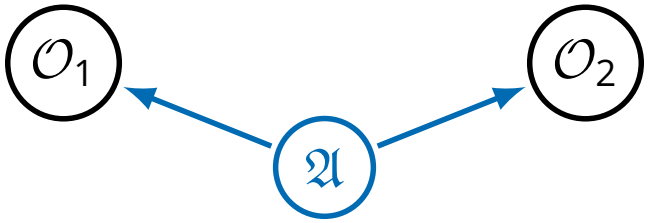
Matching:

> "The process of finding relationships or correspondences between entities of different ontologies" (Euzenat, Shvaiko, 2013)

Alignment:

> "a set of correspondences between two or more ontologies"; "the output of the matching process" (Euzenat, Shvaiko, 2013)

Matching: The Big Picture



Given two ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$, an alignment $\mathfrak{A}$ is a third ontology that shares the vocabularies of $\mathcal{O}_1$ and $\mathcal{O}_2$.

$\mathfrak{A}$ contains bridge axioms that relate one or more entities of the ontologies.

$\mathsf{C20480} \equiv \mathsf{GO\_0009987}$

$\mathsf{C20480} \equiv \exists \mathbf{relatedTo}.\mathsf{GO\_0009987}$

$\exists \mathbf{involvedIn}.\mathsf{C20480} \sqsubseteq \exists \mathbf{involvedIn}.\mathsf{GO\_0009987}$

In this lecture, $\mathcal{O}_1$, $\mathcal{O}_2$, and $\mathfrak{A}$ are OWL 2 DL ontologies.

Complex Bridge Axioms

Depending on the granularity and perspective, the same entity may be modeled as a concept in $\mathcal{O}_1$, and an individual in $\mathcal{O}_2$.

> In WordNet, the word **process** is an individual, an instance of the concept **Word**.
>
> In BFO, **Process** is a concept.

We cannot add an axiom to say that **process** and **Process** are equivalent. We need more complex axioms:

> $\mathbf{Process} \sqsubseteq \exists \mathbf{hasSyntacticRepresentation}.\{\mathbf{process}\}$

Complex correspondences are harder to find automatically.

> We focus on simple correspondences with entities of the same type.

Simple Alignments

Let $\mathcal{O}_1$ and $\mathcal{O}_2$ be two ontologies.

A correspondence between $\mathcal{O}_1$ and $\mathcal{O}_2$ is a tuple (e_1, r, e_2, c), where

- either $e_1, e_2 \in \mathbf{C}$, $e_1, e_2 \in \mathbf{R}$, or $e_1, e_2 \in \mathbf{I}$;
- e_1 occurs in $\mathcal{O}_1$ and e_2 occurs in $\mathcal{O}_2$;
- $r \in \{\equiv, \sqsubseteq, \sqsupseteq, \bot, \between\}$;
- $c \in [0, 1]$ is a confidence value.

A (simple) alignment of $\mathcal{O}_1$ and $\mathcal{O}_2$ is a set of correspondences between $\mathcal{O}_1$ and $\mathcal{O}_2$.

- The confidence values are due to the imprecise nature of automatic matching algorithms.

Simple Alignments as Ontologies

Ignoring the confidence values, correspondences can be seen as simple bridge axioms. In this way, a simple alignment can be used as an ontology:

e_1, e_2	$\equiv$	$\sqsubseteq$	$\bot$	$\not\!\!\lozenge$
$A_1, A_2 \in \mathbf{C}$	$A_1 \equiv A_2$	$A_1 \sqsubseteq A_2$	$\mathrm{Dis}(A_1, A_2)$	$a : A_1,\ a : A_2$
$r_1, r_2 \in \mathbf{R}$	$r_1 \equiv r_2$	$r_1 \sqsubseteq r_2$	$\mathrm{Dis}(r_1, r_2)$	$(c, d) : r_1,\ (c, d) : r_2$
$a_1, a_2 \in \mathbf{I}$	$a_1 \approx a_2$	$a_1 \approx a_2$	$a_1 \not\approx a_2$	$a_1 \approx a_2$

- Disjointness is crucial to detect inconsistency or incoherence in the ontology $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathfrak{A}$ (see also Part 4: Ontology Maintenance).
- Non-disjointness generalizes the coherence requirement: $(A_1, \not\!\!\lozenge, A_2, c)$ indicates that $A_1 \sqcap A_2$ should be satisfiable.

Alignments of Individuals and Roles

Individuals:

- Easy to identify based on names, titles, ID numbers
- Simple automated techniques result in high-quality alignments

Roles:

- Typically fewer roles than concept names
- Typically less information available (role hierarchy, characteristics)
- Manual matching is better suited

> Individual and role alignments are used to compute better alignments for concept names.

> In the following, we mainly consider correspondences between concept names.

Matching Techniques (I)

Matching techniques can be classified based on which types of information about entities are used to find correspondences.

Element-level techniques consider the entities of each ontology in isolation from their relations with other entities:

- String-based (string similarity measures, normalization)
- Language-based (lexical analysis of phrases)
- Linguistic resources (dictionaries or thesauri, e.g., WordNet)
- Constraint-based (types of outgoing roles, number restrictions)

Matching Techniques (II)

Structure-level techniques consider the relations of entities with other entities:

- Upper ontologies (alignment supported by third ontology)
- Graph-based (similarity of connected entities)
- Semantic (similarity of interpretations, reasoning)
- Extensional (statistical properties of instances)

Similarity Measures

Many matching techniques are based on computing similarity measures.

A similarity measure σ between two sets M_1 and M_2 is a function $\sigma : M_1 \times M_2 \to [0, 1]$.

A value of $\sigma(m_1, m_2) = 1$ indicates that m_1 and m_2 are very similar.

Our goal are concept name similarity measures, where $M_1 = \mathbf{C}(\mathcal{O}_1)$ and $M_2 = \mathbf{C}(\mathcal{O}_2)$. Often, concept name similarity measures are defined via other similarity measures, e.g., between strings.

Given a similarity measure σ between M_1 and M_2, and functions $f_1 : N_1 \to M_1$ and $f_2 : N_2 \to M_2$, the similarity measure induced by σ, f_1, and f_2 is $\sigma' : N_1 \times N_2 \to [0, 1]$, where $\sigma'(n_1, n_2) := \sigma(f_1(n_1), f_2(n_2))$.

For example, f_1/f_2 could assign string labels to concept names, and so induce a concept name similarity measure from a string similarity measure.

Defining Similarity Measures

- Many different ways to define similarity measures
- Different methods are better for different domains
- A string-based similarity measure ignores most of the information about the entities in the two ontologies
- Identify desirable properties of similarity measures

Properties of Similarity Measures

Let σ be a concept name similarity measure.

> σ is equivalence invariant if for all $A, B \in \mathbf{C}(\mathcal{O}_1)$ and $C, D \in \mathbf{C}(\mathcal{O}_2)$ with $A \equiv_{\mathcal{O}_1} B$ and $C \equiv_{\mathcal{O}_2} D$, we have $\sigma(A, C) = \sigma(B, C)$ and $\sigma(A, C) = \sigma(A, D)$.

Symmetry is important when looking for equivalences. However, a single similarity measure cannot be symmetric, since it is defined over two different domains ($\mathbf{C}(\mathcal{O}_1)$ and $\mathbf{C}(\mathcal{O}_2)$).

> A family of (concept name) similarity measures $(\sigma_{\mathcal{O}_1,\mathcal{O}_2})$ is a collection of similarity measures $\sigma_{\mathcal{O}_1,\mathcal{O}_2}$ that can be computed by an algorithm that takes arbitrary ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ as parameters.
>
> Such a family is symmetric if $\sigma_{\mathcal{O}_1,\mathcal{O}_2}(A, B) = \sigma_{\mathcal{O}_2,\mathcal{O}_1}(B, A)$ holds for all $\mathcal{O}_1, \mathcal{O}_2, A, B$.

Similarity vs. Distance

The distance measure δ induced by a similarity measure σ is also a function $\delta\colon M_1 \times M_2 \to [0,1]$, defined by $\delta(m_1, m_2) := 1 - \sigma(m_1, m_2)$.

Here, $\delta(m_1, m_2) = 1$ indicates the maximal distance between m_1 and m_2, i.e., total dissimilarity.

The two representations σ and δ are equivalent, but sometimes it is more convenient to use distance measures rather than similarity measures, or vice versa.

Similarity ≠ Equivalence

Caution: High similarity does not necessarily mean equivalence.

Close siblings in the concept hierarchy are similar, but often disjoint.

Cow **Bull** **Ox**

The interpretation of a similarity value also depends on the method that computes it.

(In the end, Matching, like Ontology Learning, can only support, not replace, human ontology engineers.)

Combining Similarity Measures: Sequential

Many techniques produce only “initial” similarity measures, which need to be refined further.

Sequential composition:

> The output σ of one algorithm is used as input for a second algorithm, which computes a new similarity measure σ'.

- Element-level similarities can be input for a structure-level method.
- The second algorithm needs to be able to use σ in the computation.
- Some methods use the same algorithm to iteratively refine a similarity measure several times. This raises the question of when a fixpoint is reached, or if the computation diverges.

Combining Similarity Measures: Parallel

Parallel composition:

Several similarity measures $\sigma_1, \ldots, \sigma_n$ are computed by different algorithms, and then combined by an aggregation operator $\otimes : [0,1]^n \to [0,1]$:

$$\sigma(m, m') := \sigma_1(m, m') \otimes \cdots \otimes \sigma_n(m, m')$$

- If $\sigma_1, \ldots, \sigma_n$ are equivalence invariant or symmetric, then so is σ.
- The same operators can also be applied to distance measures.

We present some aggregation operators for 2 similarity measures σ_1, σ_2, but they can be defined for any $n \geq 2$.

Aggregation: Triangular Norms and Conorms

A triangular norm is a binary operator $\otimes\colon [0,1] \times [0,1] \to [0,1]$ that is associative, commutative, monotone in both arguments, and has the neutral element 1.

- Triangular norms decrease the arguments: $x \otimes y \leq \min\{x, y\}$.
- We get high similarity values only if both inputs are high.
- If one argument is 0, the result is 0.
- Suitable when both σ_1 and σ_2 overestimate the "true" similarity.
- Examples: $\min\{x, y\}$ $x \cdot y$ $\max\{x + y - 1, 0\}$

A triangular conorm is a binary operator $\oplus\colon [0,1] \times [0,1] \to [0,1]$ that is associative, commutative, monotone in both arguments, and has the neutral element 0.

- Suitable when σ_1, σ_2 underestimate the similarity.
- Examples: $\max\{x, y\}$ $x + y - x \cdot y$ $\min\{x + y, 1\}$

Aggregation: Weighted Sum and Product

Given weights $w_1, w_2 \in [0,1]$ with $w_1 + w_2 = 1$,

- the weighted sum of $x_1, x_2 \in [0,1]$ is defined as $x_1 \oplus x_2 := w_1 \cdot x_1 + w_2 \cdot x_2$.
- the weighted product of $x_1, x_2 \in [0,1]$ is defined as $x_1 \otimes x_2 := x_1^{w_1} \cdot x_2^{w_2}$.

- Generalize arithmetic mean and geometric mean, respectively
- Monotone, but not associative or commutative
- Idempotent, i.e., $x \oplus x = x \otimes x = x$
- In general neither increasing nor decreasing the arguments
- Individual weight for each similarity measure

Aggregation: Minkowski Distance

For $p \geq 1$, the (normalized) Minkowski distance of $x_1, x_2 \in [0, 1]$ is defined as

$$\sqrt[p]{\frac{x_1^p + x_2^p}{2}}.$$

- Usually applied to distance measures δ_1, δ_2 in orthogonal dimensions, e.g., δ_1 compares concept labels and δ_2 compares role successors
- For $p = 1$: $(1 - x_1) + (1 - x_2)$ is the Manhattan distance.
- For $p = 2$: $\sqrt{(1 - x_1)^2 + (1 - x_2)^2}$ is the Euclidean distance.
- The factor $\sqrt[p]{\frac{1}{2}}$ normalizes the measure to the interval $[0, 1]$.

From Similarity to Alignment

Given a concept name similarity measure σ, each concept name $A \in \mathbf{C}(\mathcal{O}_1)$ may be similar to several concept names from $\mathbf{C}(\mathcal{O}_2)$, but for an alignment we keep only the best correspondences.

Threshold $\tau \in [0, 1]$:

To obtain an alignment $\mathfrak{A}$, add each correspondence $(A, \equiv, B, \sigma(A, B))$ such that

- $\sigma(A, B) \geq \tau$ (hard threshold).
- $\sigma(A, B) \geq \max_\sigma - \tau$ with $\max_\sigma := \max\{\sigma(A', B')\}$ (delta threshold).
- $\sigma(A, B) \geq \tau \cdot \max_\sigma$ (proportional threshold).
- it is among the $\tau \cdot |\mathbf{C}(\mathcal{O}_1)| \cdot |\mathbf{C}(\mathcal{O}_2)|$ correspondences with the highest similarity (percentage threshold).
- $\frac{\sigma(A,B)}{\max\{\sigma(A,B')\}} \geq \tau$ and $\frac{\sigma(A,B)}{\max\{\sigma(A',B)\}} \geq \tau$ (normalized threshold).

From Similarity to Bijective Alignment (I)

Some applications require a bijective alignment, where each concept name has exactly one $\equiv$-correspondence to a concept name from the other ontology.

(We assume without loss of generality that both ontologies have the same number of concept names.)

This requires to find a globally optimal alignment, rather than choose correspondences based on locally optimal similarity values.

Greedy Algorithm:

1. add some $(A, \equiv, B, \sigma(A,B))$ with $\sigma(A,B) = \max_\sigma$ to $\mathfrak{A}$;
2. set $\sigma(A, B') := 0$ and $\sigma(A', B) := 0$ for all A', B';
3. repeat until finished.

From Similarity to Bijective Alignment (II)

Stable Marriage Problem:

Find bijective alignment $\mathfrak{A}$ such that, for all $(A, \equiv, B, \sigma(A, B))$ and $(C, \equiv, D, \sigma(C, D))$ in $\mathfrak{A}$, it holds that

$$\sigma(A, B) \geq \sigma(A, D) \quad \text{or} \quad \sigma(C, D) \geq \sigma(C, B),$$

i.e., $\mathfrak{A}$ cannot be improved by permutations.

$\rightarrow$ Gale-Shapley algorithm

Maximum Weight Graph Matching:

Find bijective alignment $\mathfrak{A}$ such that, for all bijective alignments $\mathfrak{A}'$,

$$\sum_{(A,\equiv,B,\sigma(A,B))\in\mathfrak{A}} \sigma(A, B) \geq \sum_{(A,\equiv,B,\sigma(A,B))\in\mathfrak{A}'} \sigma(A, B),$$

i.e., $\mathfrak{A}$ maximizes the sum of all used similarity values.

$\rightarrow$ Hungarian method

Outline

3.2 Element-Level Matching

Element-Level Techniques

Element-level techniques consider the entities of each ontology in isolation from their relations with other entities:

- String-based (string similarity measures, normalization)
- Language-based (lexical analysis of phrases)
- Linguistic resources (dictionaries or thesauri, e.g., WordNet)
- Constraint-based (types of outgoing roles, number restrictions)

String-Based Techniques

- Compute similarity of entities based on similarity of the IRIs or rdfs:labels
- First normalize: remove upper case, spaces, punctuation, (diacritics)
- More informative measures are harder to compute

Hamming distance: $\delta(v, w) := \frac{1}{n} \cdot |\{i \in \{1, \ldots, n\} \mid v_i \neq w_i\}|$, where we assume that $v = v_1 \ldots v_n$ and $w = w_1 \ldots w_n$.

Substring similarity: $\sigma(v, w) := \frac{2|u|}{|v|+|w|}$, where u is the longest common substring of v and w.

n-gram similarity: $\sigma(v, w) := \frac{|n\text{-gram}(v) \cap n\text{-gram}(w)|}{|v|-n+1}$, where n-gram(v) is the set of n-letter substrings of v.

Levenshtein (edit) distance: minimal number of operations (insert, delete, replace a letter) that produce w from v (divided by $|v|$).

Bags of Words

Instead of single words, one can compare bags (multisets) of words, e.g., from rdfs:comment annotations or other texts associated with the entities.

A multiset over a domain D is a function $M\colon D \to \mathbb{N}$.

$M(d)$ is the multiplicity of an element $d \in D$ in M.

The size of M is $|M| := \sum_{d \in D} M(d)$

Here, D is the set of all words, and M is usually finite, i.e., $|M| < \infty$.

Let V, W be two multisets of words.

Jaccard measure: $\sigma(V, W) := \frac{|V \cap W|}{|V \cup W|}$

The TF-IDF Measure

Given a collection $W_1, \ldots, W_n$ of multisets (each represents one document):

$$\begin{aligned} \text{tf}(v, W) &:= \frac{W(v)}{|W|} && \text{term frequency} \\ \text{idf}(v) &:= \log \frac{n}{|\{W_i \mid v \in W_i\}|} && \text{inverse document frequency} \\ \text{tf-idf}(v, W) &:= \min\{1, \text{tf}(v, W) \cdot \text{idf}(v)\} \\ \sigma(V, W) &:= \max\{\text{tf-idf}(v, W), \text{tf-idf}(w, V) \mid v \in V, w \in W\} \end{aligned}$$

- Term frequency: How often does the word v appear in the document W (relative to the size of W)?
- Inverse document frequency: How rare is the word v?
  - 0 if v appears in every W_i
  - 1 if v appears in half of the W_i's
  - 2 if v appears in a quarter of the W_i's
- The tf-idf value indicates how "central" v is to the meaning of the document W.

Language-Based Techniques

- Improve string-based (and other) techniques
- Normalize phrases, e.g., **Warm-Blooded Animal** according to their grammatical structure (not just bags of words)
- Linguistic analysis:
  - Tokenization: Split phrase into words: **Warm Blooded Animal**
  - Lemmatization: Use word stems: **Warm Blood Animal**
  - Term extraction: Normalize word order based on lexical types (noun phrases, verb phrases): **Animal with Warm Blood** $\rightarrow$ **Warm Blood Animal**
  - Stopword elimination: Remove non-functional words (e.g., **with**)

String- and language-based techniques are not accurate when dealing with similar-looking words and synonyms:

> σ("article", "particle") vs. σ("article", "news story") ?

Linguistic Resources

String-based techniques are not designed to deal with

- synonyms
- hyponyms (more specific words)
- hypernyms (more general words)
- different languages
- abbreviations

(Multi-lingual) dictionaries or thesauri provide this information.

Different resources are appropriate for different ontologies; here we discuss WordNet.

Caution: Using synonyms to find correspondences increases the chance of false positives, in case the word senses do not match.

Word sense disambiguation tries to identify the word sense based on the context.

The Linguistic Resource WordNet

Uses of WordNet for matching:

- compute similarity based on synsets
- compute distance based on hypernym (superclass) structure
- compute distance based on gloss entries
- direct translation of semantic relations

> We do not use WordNet as an ontology, but as a repository of information about the (English) language.
>
> Hypernymy in WordNet is not modeled as subsumption, but via the transitive role hyponymOf.

Let $\Sigma(v)$ be the set of all synsets containing a word v (via some word sense),
$\leq$ be the "hypernym" relation between synsets, and
$\Gamma(v)$ be the bag containing all words of all glosses of synsets in $\Sigma(v)$.

Synonyms and Hypernyms in WordNet

synonymy: $\sigma(v,w) := \begin{cases} 1 & \text{if } \Sigma(v) \cap \Sigma(w) \neq \emptyset, \\ 0 & \text{otherwise.} \end{cases}$

cosynonymy (Jaccard): $\sigma(v,w) := \frac{|\Sigma(v) \cap \Sigma(w)|}{|\Sigma(v) \cup \Sigma(w)|}$

Reznik semantic similarity:

$\sigma(v,w) := \max\{-\log \pi(s) \mid s_1 \in \Sigma(v),\ s_2 \in \Sigma(w),\ s_1 \leq s,\ s_2 \leq s\}$

- s is a synset summarizing the information shared between v and w.
- $\pi(s)$ is the probability of s, as determined by some external measure, e.g., the term frequency in some text corpus.
- $-\log \pi(s)$ is the information content (or entropy) of s
- has to be normalized

Glosses in Wordnet

Gloss overlap (Jaccard): $\sigma(v,w) := \frac{|\Gamma(v) \cap \Gamma(w)|}{|\Gamma(v) \cup \Gamma(w)|}$

Derive $\sqsubseteq$-correspondences from glosses:

"article (nonfictional prose forming an independent part of a publication)" $\rightarrow$ (**Article**, $\sqsubseteq$, **Prose**, c)

c is derived from the similarity of the label of **Article** to the word "article", and the label of **Prose** to "prose", e.g., via some aggregation operator.

Direct Translation of WordNet Relations

We can add a correspondence

- $(A, \equiv, B, c)$ if (the labels of) A, B belong to the same synset.
- $(A, \sqsubseteq, B, c)$ if B is a hypernym of A.
- $(A, \perp, B, c)$ if A, B are antonyms or siblings in the partonomy of WordNet.

WordNet contains three part-of relations of types Member-Collection, Material-Object, and Component-Object

We can directly form complex bridge axioms:
$A \sqsubseteq \exists \mathsf{memberOf}.B$
$A \sqsubseteq \exists \mathsf{substanceOf}.B$
$A \sqsubseteq \exists \mathsf{partOf}.B$

Constraint-Based Techniques

- Similar to structure-level techniques, but only consider the "immediate surrounding structure" of the entities, e.g., outgoing roles
- Mainly used as pre-processing step
- Can identify incompatible entities early
- Criteria:
  - Number and range of (concrete) roles
  - Cardinality restrictions: Let $[n, m]$ and $[n', m']$ be (tight) lower and upper bounds on the cardinality of two roles, i.e., $A \sqsubseteq_{\mathcal{O}_1} \geq n\, r.\top \quad A \sqsubseteq_{\mathcal{O}_1} \leq m\, r.\top \quad B \sqsubseteq_{\mathcal{O}_2} \geq n'\, r'.\top \quad B \sqsubseteq_{\mathcal{O}_2} \leq m'\, r'.\top$:

$$\sigma([n, m], [n', m']) := \max\{0, \frac{\min(m,m') - \max(n,n')}{\max(m,m') - \min(n,n')}\} \qquad \text{("Jaccard similarity")}$$

Summary on Element-level Matching

String-based and constraint-based techniques generate first alignments of low quality.

Language-based techniques and linguistic resources can improve the quality, but suffer from increased number of false positives, due to multiple word senses or different translations.

Outline

3.3 Structure-Level Matching

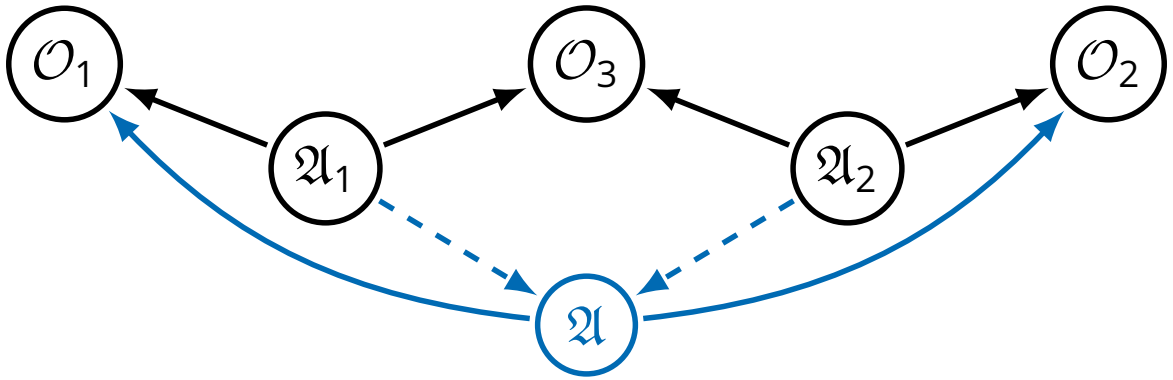
Structure-Level Techniques

Structure-level techniques consider the relations of entities with other entities:

- Upper ontologies (alignment supported by third ontology)
- Graph-based (similarity of connected entities)
- Semantic (similarity of interpretations, reasoning)
- Extensional (statistical properties of instances)

Upper Ontologies

Use (existing) alignments to a third ontology, e.g., the UMLS Metathesaurus:



Compute $\mathfrak{A}$ as composition of $\mathfrak{A}_1$ and $\mathfrak{A}_2$, for example:

$\mathfrak{A}_1$	$\mathfrak{A}_2$	$\rightsquigarrow$	$\mathfrak{A}$
$(A_1, \equiv, A_3, c_1)$	$(A_3, \sqsubseteq, A_2, c_2)$	$\rightsquigarrow$	$(A_1, \sqsubseteq, A_2, c_1 \otimes c_2)$
$(A_1, \sqsubseteq, A_3, c_1)$	$(A_3, \bot, A_2, c_2)$	$\rightsquigarrow$	$(A_1, \bot, A_2, c_1 \otimes c_2)$
$(A_1, \between, A_3, c_1)$	$(A_3, \sqsubseteq, A_2, c_2)$	$\rightsquigarrow$	$(A_1, \between, A_2, c_1 \otimes c_2)$

... where $\otimes$ is an aggregation operator.

Local Graph-Based Techniques

- Consider each ontology as a graph, where nodes are concept names
- Edges labels: $\sqsubseteq$, existential/universal restrictions on roles, disjointness
- Find similarities between the two graphs G_1, G_2

> Assuming that the roles have already been aligned, we can consider one relation at a time.
>
> We thus consider graphs G_1, G_2 with only one unlabeled edge relation.
>
> We also assume that a preliminary similarity measure is available.

Similarity of entities induced by the similarity of the sets of

- direct successors
- reachable entities, i.e., direct and indirect descendants
- "ultimate" descendants
- direct/indirect predecessors

> How to compute the similarity of sets?

Similarity Aggregation for Sets

Given a similarity measure σ' for entities, compute a similarity measure for sets of entities:

Single linkage: $\sigma(M,N) := \max_{(A,B)\in M\times N} \sigma'(A,B)$

Full linkage: $\sigma(M,N) := \min_{(A,B)\in M\times N} \sigma'(A,B)$

Average linkage: $\sigma(M,N) := \operatorname{avg}_{(A,B)\in M\times N} \sigma'(A,B)$

Hausdorff similarity:

$$\sigma(M,N) := \min\left\{ \min_{A\in M}\max_{B\in N} \sigma'(A,B),\ \min_{B\in N}\max_{A\in M} \sigma'(A,B) \right\}$$

Global Graph-Based Techniques

For two directed graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, a common subgraph is a triple (W_1, W_2, f), where $W_1 \subseteq V_1$, $W_2 \subseteq V_2$, and $f : W_1 \to W_2$ is an isomorphism between the subgraphs induced by W_1 and W_2, respectively.

A maximum common subgraph is a common subgraph, where

- the sets W_1, W_2 are maximal w.r.t. $\subseteq$, and
- $\operatorname{avg}_{v_1 \in W_1} \sigma'(v_1, f(v_1)) \geq \tau$

- The isomorphism f can be interpreted as a bijective alignment between (parts of) $\mathcal{O}_1$ and $\mathcal{O}_2$.
- Similar to maximum weight graph matching (see Section 3.1), but also considers role connections.
- Different maximum common subgraphs may not be compatible.

Semantic Techniques

Use semantics to evaluate similarity:

$A \equiv_{\mathcal{O}_1} \exists r.B \quad C \equiv_{\mathcal{O}_2} \exists s.D \quad \sigma'(B, D) = 0.8 \quad \sigma'(r, s) = 0.9$

$\rightarrow \sigma(A, C) = \sigma(\exists r.B, \exists s.D) = ?$

Again, we assume that a preliminary similarity measure σ' on concept and role names is available.

Let $\mathbf{CC}(\mathcal{O})$ be the set of (complex) concepts that can be formulated over the vocabulary of $\mathcal{O}$, i.e., $\mathbf{C}(\mathcal{O})$, $\mathbf{R}(\mathcal{O})$, and $\mathbf{I}(\mathcal{O})$.

A concept similarity measure σ is a similarity measure between $\mathbf{CC}(\mathcal{O}_1)$ and $\mathbf{CC}(\mathcal{O}_2)$.

σ is equivalence invariant if for all $C, D \in \mathbf{CC}(\mathcal{O}_1)$ and $E, F \in \mathbf{CC}(\mathcal{O}_2)$ with $C \equiv_{\mathcal{O}_1} D$ and $E \equiv_{\mathcal{O}_2} F$, we have $\sigma(C, E) = \sigma(D, E)$ and $\sigma(C, E) = \sigma(C, F)$.

A Simple Concept Similarity Measure

In general, many different concepts can be equivalent.

> We consider a logic where any concept can be reduced to a unique equivalent concept in reduced form, such that any two equivalent concepts have the same reduced form.

When designing $\sigma(\exists r.B, \exists s.D)$ as a function depending on $\sigma'(B, D)$, then σ' must already be equivalence invariant: If $B \equiv C$, then $\sigma(\exists r.B, \exists s.D)$ should be equal to $\sigma(\exists r.C, \exists s.D)$.

> We use σ itself to define the similarity of (some) concept names.

Cyclic equivalences like $B \equiv \exists r.B$ and $D \equiv \exists s.D$ mean that $\sigma(\exists r.B, \exists s.D)$ depends on $\sigma(B, D) = \sigma(\exists r.B, \exists s.D)$.

> We allow only acyclic axioms.

Acyclic TBoxes in $\mathcal{ELH}$

$\mathcal{EL}$ is the fragment of $\mathcal{ALC}$ where concepts can be built only from concept names, $\top$, $\sqcap$, and $\exists$.

$\mathcal{ELH}$ extends $\mathcal{EL}$ by allowing (non-complex) role inclusions in addition to GCIs and assertions.

An acyclic TBox $\mathcal{T}$ is a set of concept definitions $A \equiv C_A$ such that

- each concept name A has at most one definition C_A in $\mathcal{T}$, and
- the "depends on" relation between concept names is acyclic:

  A depends on B (w.r.t. $\mathcal{T}$) if B occurs in the definition C_A of A in $\mathcal{T}$.

Concept names with definitions in $\mathcal{T}$ are called defined, all others are called primitive.

Concept Expansion

Lemma (Expansion)

Let $\mathcal{O} = (\emptyset, \mathcal{T}, \mathcal{R})$ be an $\mathcal{ELH}$ ontology with an acyclic TBox $\mathcal{T}$.

Every $\mathcal{ELH}$ concept C can be expanded into a concept $\widehat{C} \equiv_{\mathcal{O}} C$ that does not contain defined concept names.

For two $\mathcal{ELH}$ concepts C, D, we have $C \sqsubseteq_{\mathcal{O}} D$ iff $\widehat{C} \sqsubseteq_{(\emptyset,\emptyset,\mathcal{R})} \widehat{D}$.

Proof: (Baader, Lutz, Horrocks, Sattler, 2017) □

- The ABox is irrelevant for checking subsumptions (Exercise).
- We can also assume that the TBox is empty, which leaves only the RBox.
- We can assume that the RBox contains no cycles between role names:

Equivalent roles $r \equiv_{\mathcal{R}} s$ are replaced by a single role.

$r \sqsubseteq_{\mathcal{R}} s$ holds iff s can be reached from r by a sequence of role inclusions in $\mathcal{R}$. What about $C \sqsubseteq_{\mathcal{R}} D$?

The Structure of $\mathcal{ELH}$ Concepts

We say that C and D are equal modulo ACU ($C =_{ACU} D$) if D can be obtained from C by reordering concepts inside conjunctions and removing/adding $\top$ from/to conjunctions.

For example, $A \sqcap B \sqcap C =_{ACU} C \sqcap B \sqcap A$.

ACU stands for associativity and commutativity of $\sqcap$, and the unit property w.r.t. $\top$.

An atom is a concept name (except $\top$) or an existential restriction.

Every $\mathcal{ELH}$ concept C is equal (modulo ACU) to a concept of the form $C_1 \sqcap \cdots \sqcap C_n$, where $C_1, \ldots, C_n$ are atoms.

$C_1, \ldots, C_n$ are the top-level atoms of C; we set $\mathrm{At}(C) := \{C_1, \ldots, C_n\}$.

The case $C = \top$ can be seen as the empty conjunction, with $\mathrm{At}(\top) = \emptyset$.

Structural Subsumption

Let C, D be $\mathcal{ELH}$ concepts and $\mathcal{R}$ be an RBox.

C is structurally subsumed by D (written $C \sqsubseteq^{\mathrm{s}}_{\mathcal{R}} D$) w.r.t. $\mathcal{R}$ if, for every $D' \in \mathrm{At}(D)$ there is a $C' \in \mathrm{At}(C)$ such that

- $C', D' \in \mathbf{C}$ and $C' = D'$, or
- $C' = \exists r.C''$, $D' = \exists s.D''$, $r \sqsubseteq_{\mathcal{R}} s$, and $C'' \sqsubseteq^{\mathrm{s}}_{\mathcal{R}} D''$.

This is well-defined, because the role depth is decreased in the recursion.

Lemma (Structural Subsumption)

Let C, D be $\mathcal{ELH}$ concepts and $\mathcal{R}$ be an RBox. Then $C \sqsubseteq_{\mathcal{R}} D$ iff $C \sqsubseteq^{\mathrm{s}}_{\mathcal{R}} D$.

Proof: Blackboard. □

Reduced Form of $\mathcal{ELH}$ Concepts

Given an RBox $\mathcal{R}$, every $\mathcal{ELH}$ concept can be reduced (w.r.t. $\mathcal{R}$) by exhaustively applying the following rules to all subconcepts (modulo AC):

$$C \sqcap \top \longrightarrow C$$
$$C \sqcap C \longrightarrow C$$
$$\exists r.C \sqcap \exists s.D \longrightarrow \exists r.C \text{ if } r \sqsubseteq_{\mathcal{R}} s \text{ and } C \sqsubseteq_{\mathcal{R}} D$$

Lemma (Reduced Form)

Let C, D be $\mathcal{ELH}$ concepts and C', D' their reduced forms.

Then $C \equiv_{\mathcal{R}} D$ iff $C' =_{AC} D'$.

Proof: Blackboard. □

- A concept C and its reduced form C' are equivalent w.r.t. $\mathcal{R}$.
- The reduced form of a concept C is unique (modulo AC).
- We can now assume concepts to be in reduced form.

A Concept Similarity Measure for $\mathcal{ELH}$

Given an RBox $\mathcal{R}$, two expanded $\mathcal{ELH}$ concepts C, D in reduced form, and $\beta \in [0,1]$, the directed similarity of C and D is

$$\sigma_d(C,D) := \begin{cases} \min\limits_{D' \in \mathrm{At}(D)} \max\limits_{C' \in \mathrm{At}(C)} \sigma_d(C', D') & \text{if } |\mathrm{At}(C)| > 1 \text{ or } |\mathrm{At}(D)| > 1 \\ 1 & \text{if } \mathrm{At}(D) = \emptyset \\ \sigma'(C,D) & \text{if } C, D \in \mathbf{C} \\ \sigma'(r,s)\big(\beta + (1-\beta)\sigma_d(E,F)\big) & \text{if } C = \exists r.E \text{ and } D = \exists s.F \\ 0 & \text{otherwise} \end{cases}$$

- $\sigma_d(C,D)$ measures "how much" C is subsumed by D.
- If $D = \top$, then $\mathrm{At}(D) = \emptyset$, which yields $\sigma_d(C,D) = 1$.
- σ' is used to compare role names and primitive concept names.
- We could use other aggregation operators instead of min and max.

A Symmetric Concept Similarity Measure for $\mathcal{ELH}$

The undirected similarity of C and D is

$$\sigma_u(C, D) := \sigma_d(C, D) \otimes \sigma_d(D, C),$$

where $\otimes$ is a commutative aggregation operator.

Similarity of arbitrary concept names $A \in \mathbf{C}(\mathcal{O}_1)$ and $B \in \mathbf{C}(\mathcal{O}_2)$, where $\mathcal{O}_1 = (\emptyset, \mathcal{T}_1, \mathcal{R}_1)$, $\mathcal{O}_2 = (\emptyset, \mathcal{T}_2, \mathcal{R}_2)$ are $\mathcal{ELH}$ ontologies with acyclic TBoxes:

- Expand A using $\mathcal{T}_1$ to $\widehat{A}$, and expand B using $\mathcal{T}_2$ to $\widehat{B}$.
- Reduce $\widehat{A}$ using $\mathcal{R}_1$ to A', and reduce $\widehat{B}$ using $\mathcal{R}_2$ to B'.
- Evaluate $\sigma_u(A, B) := \sigma_u(A', B')$.

Lemma (Undirected Concept Similarity)

σ_u is symmetric and equivalence invariant.

Proof: Blackboard. □

Extensional Techniques

- Use instances to find correspondences between classes

> If an alignment $\mathfrak{A}_\mathbf{I}$ between individual names is available, one can compare sets of instances $\mathbf{I}_A := \mathbf{I}_A(\mathcal{O}_1 \cup \mathfrak{A}_\mathbf{I})$ and $\mathbf{I}_B := \mathbf{I}_B(\mathcal{O}_2 \cup \mathfrak{A}_\mathbf{I})$, where $A \in \mathbf{C}(\mathcal{O}_1)$ and $B \in \mathbf{C}(\mathcal{O}_2)$.

- $(A, \equiv, B, c)$ if $\mathbf{I}_A = \mathbf{I}_B$ or $c = \frac{|\mathbf{I}_A \cap \mathbf{I}_B|}{|\mathbf{I}_A \cup \mathbf{I}_B|} \geq \tau$ (Jaccard)
- $(A, \sqsubseteq, B, c)$ if $\mathbf{I}_A \subseteq \mathbf{I}_B$ or $c = \frac{|\mathbf{I}_A \cap \mathbf{I}_B|}{|\mathbf{I}_A|} \geq \tau$
- $(A, \sqsupseteq, B, c)$ if $\mathbf{I}_A \supseteq \mathbf{I}_B$ or $c = \frac{|\mathbf{I}_A \cap \mathbf{I}_B|}{|\mathbf{I}_B|} \geq \tau$
- $(A, \bot, B, c)$ if $\mathbf{I}_A \cap \mathbf{I}_B = \emptyset$ or $c = 1 - \frac{|\mathbf{I}_A \cap \mathbf{I}_B|}{|\mathbf{I}_A \cup \mathbf{I}_B|} \geq \tau$
- $(A, \between, B, c)$ otherwise

> If only a similarity measure σ' on individual names is available, one can extend it to a set similarity between $\mathbf{I}_A(\mathcal{O}_1)$ and $\mathbf{I}_B(\mathcal{O}_2)$.

Summary

- Most implementations combine several element- and structure-level techniques.
- None of the presented techniques are perfect, all require manual supervision.
- Interactive systems allow to correct alignments while they are constructed.
- Systems are evaluated every year at the Ontology Matching workshop against human-created alignments.
  `http://oaei.ontologymatching.org/`
- What happens if created alignments are inconsistent or incoherent?