Ontology distances for contextualisation

Jérôme Euzenat, Carlo Allocca, Jérôme David, Mathieu d'Aquin, Chan Le Duc, Ondrej Sváb-Zamazal

NeOn: Lifecycle Support for Networked Ontologies

Integrated Project (IST-2005-027595)

Priority: IST-2004-2.4.7 — "Semantic-based knowledge and content systems"

D3.3.4: Ontology distances for contextualisation

Deliverable Co-ordinator: **Jérôme Euzenat**

Deliverable Co-ordinating Institution: **INRIA**

Other Authors: **Carlo Allocca (Open university), Jérôme David (INRIA), Mathieu d'Aquin (Open university), Chan Le Duc (INRIA), Ondřej Zamazal (INRIA)**

Distances between ontologies are useful for searching, matching or visualising ontologies. We study the various distances that can be defined across ontologies and provide them in a NeOn toolkit plug-in, OntoSim, which is a library of distances that can be used for recontextualising.

Document Identifier:	NEON/2009/D3.3.4/v1.1	Date due:	August 31st, 2009
Class Deliverable:	NEON EU-IST-2005-027595	Submission date:	August 31st, 2009
Project start date	March 1, 2006	Version:	v1.1
Project duration:	4 years	State:	Final
		Distribution:	Public

NeOn Consortium

This document is part of the NeOn research project funded by the IST Programme of the Commission of the European Communities by the grant number IST-2005-027595. The following partners are involved in the project:

Open University (OU) – Coordinator Knowledge Media Institute – KMi Berrill Building, Walton Hall Milton Keynes, MK7 6AA United Kingdom Contact person: Martin Dzbor, Enrico Motta E-mail address: {m.dzbor, e.motta}@open.ac.uk	**Universität Karlsruhe – TH (UKARL)** Institut für Angewandte Informatik und Formale Beschreibungsverfahren – AIFB Englerstrasse 11 D-76128 Karlsruhe, Germany Contact person: Peter Haase E-mail address: pha@aifb.uni-karlsruhe.de
Universidad Politécnica de Madrid (UPM) Campus de Montegancedo 28660 Boadilla del Monte Spain Contact person: Asunción Gómez Pérez E-mail address: asun@fi.ump.es	**Software AG (SAG)** Uhlandstrasse 12 64297 Darmstadt Germany Contact person: Walter Waterfeld E-mail address: walter.waterfeld@softwareag.com
Intelligent Software Components S.A. (ISOCO) Calle de Pedro de Valdivia 10 28006 Madrid Spain Contact person: Jesús Contreras E-mail address: jcontreras@isoco.com	**Institut 'Jožef Stefan' (JSI)** Jamova 39 SL–1000 Ljubljana Slovenia Contact person: Marko Grobelnik E-mail address: marko.grobelnik@ijs.si
Institut National de Recherche en Informatique et en Automatique (INRIA) ZIRST – 665 avenue de l'Europe Montbonnot Saint Martin 38334 Saint-Ismier, France Contact person: Jérôme Euzenat E-mail address: jerome.euzenat@inrialpes.fr	**University of Sheffield (USFD)** Dept. of Computer Science Regent Court 211 Portobello street S14DP Sheffield, United Kingdom Contact person: Hamish Cunningham E-mail address: hamish@dcs.shef.ac.uk
Universität Kolenz-Landau (UKO-LD) Universitätsstrasse 1 56070 Koblenz Germany Contact person: Steffen Staab E-mail address: staab@uni-koblenz.de	**Consiglio Nazionale delle Ricerche (CNR)** Institute of cognitive sciences and technologies Via S. Marino della Battaglia 44 – 00185 Roma-Lazio Italy Contact person: Aldo Gangemi E-mail address: aldo.gangemi@istc.cnr.it
Ontoprise GmbH. (ONTO) Amalienbadstr. 36 (Raumfabrik 29) 76227 Karlsruhe Germany Contact person: Jürgen Angele E-mail address: angele@ontoprise.de	**Food and Agriculture Organization of the United Nations (FAO)** Viale delle Terme di Caracalla 00100 Rome Italy Contact person: Marta Iglesias E-mail address: marta.iglesias@fao.org
Atos Origin S.A. (ATOS) Calle de Albarracín, 25 28037 Madrid Spain Contact person: Tomás Pariente Lobo E-mail address: tomas.parientelobo@atosorigin.com	**Laboratorios KIN, S.A. (KIN)** C/Ciudad de Granada, 123 08018 Barcelona Spain Contact person: Antonio López E-mail address: alopez@kin.es

Work package participants

The following partners have taken an active part in the work leading to the elaboration of this document, even if they might not have directly contributed to the writing of this document or its parts:

- Jožef Stefan Institute
- Open university
- iSoCo
- Universität Karlsruhe
- INRIA
- Ontoprise
- CNR
- Universidad Politécnica de Madrid

Change Log

Version	Date	Author	Changes
0.1	15.12.2008	Jérôme Euzenat	initial version
0.2	24.02.2009	Jérôme Euzenat	changed title
0.3	03.04.2009	Jérôme Euzenat	included Section 4
0.4	30.05.2009	Jérôme Euzenat	cleaned-up descriptions, first version of Chapter 5 and 6
0.5	15.06.2009	Jérôme David	Finished Chapter 5
0.6	20.06.2009	Jérôme Euzenat	Overall revision and design of new experiments
0.7	21.08.2009	Carlo Alloca	Added measures descriptions coming from Kannel
0.8	28.08.2009	Ondřej Zamazal	Added experiment results in §6.3
0.9	31.08.2009	Jérôme David	Final revision
1.0	30.09.2009	Jérôme Euzenat	Integrated final experiments Implemented quality control comments
1.1	16.10.2009	Jérôme Euzenat	Integrated quality assessor comments

Executive Summary

Measuring distances between ontologies can be very important in various tasks for different purposes (finding an ontology to replace another, finding an ontology in which queries can be translated, finding people or businesses using similar ontologies).

In the framework of this work package, we have defined recontextualisation as the connection of ontologies to context ontologies through alignment. So this amounts to finding on the web (through Watson or the Alignment server) ontologies aligned with the ontology to contextualise or ontologies which are close with respect to a particular distance. Hence, context-based matching can rely on ontology distances or similarities.

Other uses of ontology measures within NeOn include finding related ontologies (like in Watson and Cupboard), visualising ontologies in metric space, measuring similarities between peers, or finding paths to answer queries. Hence there is no universal criterion for deciding if an ontology is close or far from another. There are also many possible ways to compute a similarity or a distance between ontologies and they may have different properties.

The goal of this deliverable is twofold: (1) evaluating the validity of some of these measures, and (2) providing a library of such measures which can be used within NeOn.

Concerning the first goal, we review a number of possible measures. We have separated these measures in two categories: measures using the content of the ontologies alone (the ontology space) and measures using existing alignment between ontologies (the alignment space).

Measures working in the ontology space are rather classic: they compare the content of both ontologies in order to find how they are similar or dissimilar. This comparison can be based on terminology, structure or be semantically grounded; they can consider the ontology as a whole or only salient features such as key concepts. Moreover, there are different methods for aggregating a measure between ontology entities into a measure between ontologies. We consider the latter independently from the former.

Measures working in the alignment space are introduced in this deliverable. They evaluate the similarity between two ontologies with regard to the available alignments between them. This reflects the possibility to link the ontologies and to perform actions such as instance import or query translation. They are based either on the availability of alignments or the entities they preserve.

These two sets of measures have been compared in two experimental settings and results are reported. The results show a wide variety of behaviour among measures concerning their speed, accuracy and robustness.

Moreover, these measures have all been implemented in a library, OntoSim, that is used as a NeOn toolkit plug-in. This plug-in has been used for performing the experiments and is available both in and out of the NeOn toolkit. We describe its architecture and provide examples of how to use it.

Contents

Chapter 1

Introduction

Measuring distances between ontologies is useful for several purposes in NeOn: in Watson and Cupboard for finding related ontologies, in Scarlet for recontextualising ontologies before matching and in visualising ontology clouds.

In this introduction, we first motivate the need for measuring proximity of ontologies in the context of networked ontologies (§1.1). We consider some related work (§1.2), then briefly describe the OntoSim library and NeOn toolkit plug-in (§1.3).

1.1 Motivations

Our goal is to investigate the possibility of measuring a distance or similarity between ontologies. There are many occasions in which it is useful to know if two ontologies are close to one anothers or not, or what is the closest ontology. In particular (retaining the categories identified in [David and Euzenat, 2008]):

- when one wants to find the community of people with whom they will be more likely to communicate easily, discovering whether or not they use similar ontologies can be particularly useful [Jung and Euzenat, 2007]; This can also help to identify communities in social networks [Jung *et al.*, 2007];
- in semantic peer-to-peer systems, it will be easier to find information if queries can be sent to nodes using similar ontologies because query transformation will miss less information [Ehrig *et al.*, 2005];
- in ontology engineering, it is useful to find similar ontologies that can be easily used in conjunction with others. For example, when developing an ontology for radiological diagnoses, it would be useful to find anatomy and pathology ontologies that can be used with each other; Ontologies for which an alignment is already available or which can be built quickly may be preferred.
- when modularising large ontologies into smaller parts [Stuckenschmidt and Klein, 2004], it is useful to consider the module candidates as sub-ontologies which will be more prone to being separated as they are distant from one another;
- in semantic search engines which return ontologies corresponding to a query [d'Aquin *et al.*, 2007], it would be useful to introduce a "Find similar ontologies" button. More generally, making relations between ontologies explicit, as investigated in Kannel, is a major challenge. Distances can also be used in this case for ordering answers to such a query (ontology ranking, [Alani and Brewster, 2005]) with regard to ontology proximity;
- in some ontology matching algorithms [Gracia *et al.*, 2007] when one wants to use an intermediate ontology between two ontologies, it may be useful to select those for which an alignment can easily be obtained. For example, if one wants to align two ontologies about music, it would be more accurate to use another ontology about music (which covers the two former ones) than an ontology about arts.

In particular, several, parts of the NeOn project are concerned with using distances between ontology:

- in work package 1, Watson is an ontology search engine; as such, it can take advantage of distances

for finding close ontologies;
- in work package 3, Scarlet is a context-based matcher which has to recontextualise ontologies before matching (see Deliverable 3.3.1 and 3.3.5). To that end, finding close ontologies allows for a more efficient recontextualisation;
- in work package 3, graphic visualisation of ontologies can take advantage of a measure between them that will support their display by projecting these ontologies on the implicit metric space.

Of course, each of these applications has different requirements for such a measure. This "proximity" requirement between ontologies should reflect different realities: ontologies may be similar because they have a lot of common concepts or they may be similar because they can easily be bridged, e.g., it is easy to obtain a mediator to transfer queries from one ontology to another. In turn, the easy creation of a mediator can have different faces: this can be easy because it can be computed quickly (or because it is already available) or it can be easy because it will yield more correct, more complete or more precise results. There is always a trade-off between speed and accuracy. We will review some of these possible measures and propose a first evaluation of their qualities.

So, our goal is to investigate what the criteria are for designing a distance measure between ontologies and to consider how candidate distances satisfy these criteria. We have designed a library of such measures and have evaluated them along these criteria. In doing so, we will consider already existing distances and introduce new ones.

1.2 Related works

Most of the work dealing with ontology distance [Mädche and Staab, 2002; Hu *et al.*, 2006; Vrandečić and Sure, 2007] is in reality concerned with concept distances. Such measures are widely used in ontology matching algorithms [Euzenat and Shvaiko, 2007]. They are quickly extended to ontologies without discussing the different ways to achieve this.

[Mädche and Staab, 2002] introduced a concept similarity based on terminological and structural aspects of ontologies. This very precise proposal combines an edit distance on strings and a structural distance on hierarchies (the cotopic distance). The ontology similarity strongly relies on the terminological similarity. This paper evaluates the ontology design process, but not ontology similarity. OLA [Euzenat and Valtchev, 2004] uses a concept similarity for ontology matching. This measure takes advantage of most of the ontological aspects (labels, structure, extension) and selects the maximum similarity. It should thus be a good candidate for ontology similarity.

The framework presented in [Ehrig *et al.*, 2005] aims at comparing concepts across ontologies instead of ontologies themselves. It provides a similarity combining string similarity, concept similarity – considered as sets – and similarity across usage traces.

There is also quite an elaborate framework in [Hu *et al.*, 2006]. This paper is mostly dedicated to the comparison of concepts but can be extended to ontologies. First, concepts are expanded so they are expressed in term of primitive concepts. Each concept is expressed as a disjunction of compound but conjunctive concepts. This works as long as no cycle occurs in the ontology. Then primitive concepts are considered as dimensions in a vector space and each concept is represented in this space. The weights used in this vector space are computed with TF·IDF. The distance between two concepts is the smallest cosine distance between vectors associated with disjuncts describing concepts. The way this is extended to ontology concepts is not clearly explained but the methods that will be explained in §3.3 would work.

Finally, [Vrandečić and Sure, 2007] more directly considered metrics evaluating ontology quality. This is nevertheless one step towards semantic measures since they introduce normal forms for ontologies which could be used for developing syntactically neutral measures.

A general comment about these works is that they rely on elaborate distance or similarity measures between concepts and they extend these measures to distance between ontologies. This extension is often considered as straightforward. However, they have barely been evaluated. We will evaluate some of these here. Table 1.1

Paper	*concept meas.*	*type*	*ontology meas.*	*evaluation*
[Mädche and Staab, 2002]	lex+str	-	no	no
[Euzenat and Valtchev, 2004]	lex+str	mwmgm	no	no
[Ehrig *et al.*, 2005]	lex+set	-	no	no
[Hu *et al.*, 2006]	str+vsm	-	lex+	no
[Vrandečić and Sure, 2007]	str	-	no	no
[Euzenat and Shvaiko, 2007]	*	*	no	no
[David and Euzenat, 2008]	*	*	*	yes

Table 1.1: Papers mentioning ontology distance or similarity ("*" means that many methods are considered; "-" means that none are considered).

summarises these measures briefly.

1.3 OntoSim

OntoSim is a library for computing distance or similarity measures between ontologies. It is integrated within the NeOn toolkit as a plug-in that can be exploited by other parts of the toolkit. It will also be available independently. The OntoSim project can be found at:

`http://ontosim.gforge.inria.fr`

OntoSim provides various measures. Some of them were developed before the NeOn project and integrated. Some of them, in particular those concerned with the alignment space, are original work.

1.4 Outline of the report

This report is organised as follows: We first provide general definitions about ontologies and similarities (§2). We will distinguish between measures in the ontology space, where the distance between ontologies is obtained by comparing them, and measures in the alignment space, where the distance is obtained with regard to how the ontologies are related by alignments. We provide definitions of distances and similarity in the ontology space (§3) and in the alignment space (§4). We present the implementation of such measures in the OntoSim plug-in (§5). Finally we provide an experimental evaluation of these measures (§6).

Chapter 2

Ontology distances and similarities

In this section, we first introduce the ontology model which we use and then review the general properties that distances between ontologies must satisfy.

2.1 Ontology model

All measures will be based on a set of ontologies Ω which we refer to as the ontology space. These ontologies are assumed to be written in a language like OWL. In the examples here, we will only consider classes, properties and subsumption relations.

An ontology $o \in \Omega$ expressed in a language L generates a set of ontology entities $Q_L(o)$ which are the objects that can appear in alignments. These entities can be simply made of all the formulas of the ontology language based on the ontology vocabulary. The entities can also be restricted to particular kinds of formulas from the language, such as atomic formulas, or even to terms of the language, like class expressions. They can as well be only named entities identified by a URI (classes, properties and individuals). The entity language can also be an extension of the ontology language. For instance, it can be a query language, such as SPARQL, adding operations for manipulating ontology entities that are not available in the ontology language itself, like concatenating strings or joining relations.

For simplification purposes, we will consider an OWL ontology $o \in \Omega$ represented as a set of named entities $Q_L(o)$. These entities can be classes (C), properties (P) or individuals (I): $Q_L(o) = C \cup P \cup I$. Each entity is identified by a URI which can be obtained by the function $uri : Q_L(o) \rightarrow URI$.

Because different versions of the same ontology may differ by their URI prefix only, very often, ontology names will be compared independently from this prefix. The function $l_{ln} : Q_L(o) \rightarrow String$ returns the local name of the entity which is the specific part of the entity URI in the ontology. We also consider a syntactical normalisation of the terms of the ontologies' vocabularies. For example, "MemberStaff", "Member-Staff", "Member_Staff" or "memberstaff" are all considered the same. The function $L : Q_L(o) \rightarrow String$ returns this normalised name. Each entity can be also described by annotations, i.e., labels, comments. The function $l_{annot} : Q_L(o) \rightarrow \mathcal{P}(String)$ assigns a set of annotations to each entity.

In addition, we will consider the set of axioms $ax(o)$ of an ontology and its set of triples $trp(o)$.

2.2 Algebraic distance properties

A dissimilarity is a real positive function δ of two ontologies which is as large as ontologies differ.

Definition 1 (Dissimilarity)**.** *Given a set Ω of ontologies, a dissimilarity $\delta : \Omega \times \Omega \rightarrow \mathbb{R}$ is a function from a*

pair of ontologies to a real number such that:

$$\forall o, o' \in \Omega, \delta(o, o') \geq 0 \quad \textit{(non-negativity)}$$
$$\forall o \in \Omega, \delta(o, o) = 0 \quad \textit{(minimality)}$$
$$\forall o, o' \in \Omega, \delta(o, o') = \delta(o', o) \quad \textit{(symmetry)}$$

The similarity is a dual operation. It is defined as follows.

Definition 2 (Similarity)**.** *A similarity $\sigma : \Omega \times \Omega \rightarrow \mathbb{R}$ is a function from a pair of entities to a real number expressing the similarity between two objects such that:*

$$\forall o, o' \in \Omega, \sigma(o, o') \geq 0 \quad \textit{(positiveness)}$$
$$\forall o \in \Omega, \forall o', o'' \in \Omega, \sigma(o, o) \geq \sigma(o', o'') \quad \textit{(maximality)}$$
$$\forall o, o' \in \Omega, \sigma(o, o') = \sigma(o', o) \quad \textit{(symmetry)}$$

Some authors consider a 'non symmetric (dis)similarity' [Tversky, 1977]; we then use the term non symmetric measure or pre-similarity. More stringent constraints can apply on dissimilarities (leading to distances, or metrics, and ultrametrics).

Definition 3 (Distance)**.** *A distance (or metric) $\delta : \Omega \times \Omega \rightarrow \mathbb{R}$ is a dissimilarity function satisfying the definiteness and triangular inequality:*

$$\forall o, o' \in \Omega, \delta(o, o') = 0 \textit{ if and only if } o = o' \quad \textit{(definiteness)}$$
$$\forall o, o', o'' \in \Omega, \delta(o, o') + \delta(o', o'') \geq \delta(o, o'') \quad \textit{(triangular inequality)}$$

There are many reasons why an ontology measure may not be a distance. In particular, if we want to consider the semantics of ontologies, a sheer semantic measure should be 0 when the two arguments are semantically equivalent, even if they are not the same. For the sake of finding a distance, we must work in the quotient space in which the congruence relation is semantic equivalence. However, given the cost of computing semantic equivalence we will try to avoid that.

There are good reasons to avoid symmetry as well (see end of §2.3 below).

Very often, the measures are normalised, especially if the dissimilarity of different kinds of entities must be compared. Reducing each value to the same scale in proportion to the size of the considered space is the common way to normalise.

Definition 4 (Normalised measure)**.** *A measure is said to be* normalised *if it ranges over the unit interval of real numbers $[0\ 1]$. A normalised version of a measure δ is denoted as $\overline{\delta}$.*

In the remainder, we will consider mostly normalised measures and assume that a dissimilarity function between two entities returns a real number between 0 and 1. It is easy to transform a normalised dissimilarity into a normalised similarity by using its complement to 1.

2.3 Application-specific distance properties

One could imagine some properties which are unrelated to the general notion of distance but are specific to its use. In addition to algebraic properties, we would like to express purpose-oriented constraints on the measure. Such constraints must ask that the smaller the distance:

- the faster it is to provide an alignment;
- the more entities correspond to entities of the other ontology;
- the more entities of the other ontology correspond to entities of this ontology;
- the closer are corresponding entities;

- the easier (the faster) it is to answer queries;
- etc.

For instance, we could take into account a property stating that the addition of independent information in one ontology implies an increase of the distance value:

$$\forall o, o', o'' \in \Omega, o'' \cap o = \emptyset \Rightarrow \delta(o, o') \leq \delta(o, o' \cup o'')$$

Conversely, the addition of information issued from the other ontology implies a decrease of the distance value:

$$\forall o, o', o'' \in \Omega, o'' \subseteq o - o', \Rightarrow \delta(o, o' \cup o'') \leq \delta(o, o')$$

These first properties show that the more ontologies share concepts, the smaller their distance. Nevertheless, they are useful only if we consider ontologies having entities which match perfectly. In actual cases, the ontologies are sufficiently heterogeneous and consequently, this property cannot be satisfied.

What is then needed is that the entities related by potential ontology alignments are preserved. Such properties are called coverage properties. They would ensure that the information from one ontology can be transformed in the second ontology or that queries can be transformed.

Such a property can be defined through $Dom_o(A) = \{e \in o; \exists \langle e, e', r \rangle \in A\}$.

The following formula provides a strict constraint for a measure comparing coverages:

$$\forall o, o', o'' \in \Omega, \forall A' \in \Lambda(o, o''), \exists A \in \Lambda(o, o'); Dom_o(A') \subseteq Dom_o(A) \Rightarrow \delta(o, o') \leq \delta(o, o'')$$

a weaker version compares cardinality of the domains instead of their inclusion:

$$\forall o, o', o'' \in \Omega, \exists A \in \Lambda(o, o'); \forall A' \in \Lambda(o, o''), |Dom_o(A')| \leq |Dom_o(A)| \Rightarrow \delta(o, o') \leq \delta(o, o'')$$

On the other side, it is useful that ontologies do not, if possible, lose information, i.e., that it is not related to another ontology by only projecting all entities into the same poorly informative entity. We call such properties injectivity properties. For rendering injectivity properties, we define $\widehat{Img}_o(A) = \{E \subseteq o; \exists e' \in o'; \forall e \in E, \langle e, e', = \rangle \in A\}$. $\widehat{Img}_o(A)$ is a partition of $Dom_o(A)$; if we add to it a class containing $o \backslash Dom_o(A)$, then it is a partition of o. This allows us to compare alignments on the same basis. For that purpose, these partitions can be compared with the usual partition inclusion $\ll$.

The following formula provides a strict constraint for a measure comparing injectivity:

$$\forall o, o', o'' \in \Omega, \forall A' \in \Lambda(o, o''), \exists A \in \Lambda(o, o'); \widehat{Img}_o(A') \ll \widehat{Img}_o(A) \Rightarrow \delta(o, o') \leq \delta(o, o'')$$

a weaker version compares cardinality of the domains instead of their inclusion:

$$\forall o, o', o'' \in \Omega, \exists A \in \Lambda(o, o'); \forall A' \in \Lambda(o, o''), |\widehat{Img}_o(A')| \leq |\widehat{Img}_o(A)| \Rightarrow \delta(o, o') \leq \delta(o, o'')$$

These two families of properties are expressed from the standpoint of one ontology and do not entail the symmetry of the measure. If one wants a symmetric measure, it will be necessary to require that the properties are satisfied on both directions

2.4 Conclusion

Our goal is to define measures between ontologies which can be used in various applications for assessing their proximity. We provide such measures in the two next sections. These measures may be either distances or similarities.

Chapter 3

Distances in the ontology space

When only two ontologies are available, ontology distances have to be computed by comparing them. On the basis of such measures, systems will decide between which ontologies to run a matching algorithm. They can measure the ease of producing an alignment (expected speed, expected quality). So naturally, one constraint is that the distance is computed faster than the actual alignment.

There are many possible ways to define a distance between ontologies. First of all, an ontology can just be viewed as a bag of objects (terms, concepts, axioms) on a common space and the measures be computed globally (§3.1). This approach encompasses those used in information retrieval based on the vector space model which rely on vector representations of ontologies and use distance measures between these vectors.

Another approach is to consider an ontology as a set of entities. These entities will depend on the techniques used for establishing the distance: they will generally be the classes or properties to be found within the ontologies. In this case, defining a distance between ontologies will very often rely on:

- a distance (δ) or similarity (σ) measure between entities (§3.2);
- a collection distance (Δ) which will use the measures between entities for computing a distance between ontologies (§3.3).

3.1 Global ontology distances

The global distances compare ontologies as wholes, i.e., they do not take into account the structure of the ontologies and their specific elements.They need that the entity they compare be directly comparable, i.e., that they can be part of both ontologies. Hence, they either consider labels or identify concepts by their local names. Otherwise, they would need the mediation of an entity measure (§3.2) to compare these entities.

3.1.1 Global measures of labels

A distance can be computed by comparing the sets (or bags) of labels appearing in both ontologies and using a measure such as the Hamming distance, i.e., the complement to 1 of the ratio of common terms over the whole set of terms used by any of the ontologies.

Definition 5 (Hamming distance on names)**.** *Let o and o' be two ontologies and $L(\cdot)$ a function returning the set of names of entities in these ontologies, the Hamming distance on names is characterised by:*

$$\delta_{hdcn}(o, o') = 1 - \frac{|L(o) \cap L(o')|}{|L(o) \cup L(o')|}$$

This is a normalised dissimilarity: it is obviously not definite. This measure is relatively easy to compute and experiments show that it is already accurate.

Another measure could also be defined as:

Definition 6 (Common name similarity)**.** *Let o and o' be two ontologies and $L(\cdot)$ a function returning the set of names of entities in these ontologies, the common name similarity is characterised by:*

$$\sigma_{cn}(o, o') = \frac{|L(o) \cap L(o')|}{max(|L(o)|, |L(o')|)}$$

3.1.2 Ontology distances based on the vector space model

A more elaborate proposal for comparing the labels in ontologies consists of using the vector space model (VSM) from information retrieval. In the vector space model, each ontology is represented by a vector in a space where the terms are the dimensions. These terms are extracted from the annotations of the ontology entities. The set of terms T_o contained in an ontology o is build with the help of a term extraction function l_{te}: $T_o = \bigcup_{e \in Q_L(o)} l_{te}\big(l_{annot}(e)\big) \cup l_{ln}(e)$[1]. So the ontology is in fact a point (or a vector) in this space and a distance between these points or vectors can be computed.

Definition 7 (Vector associated to an ontology)**.** *Let Ω be, a set of ontologies and $\mathcal{T}$ be the set of terms contained in these ontologies. The vector of terms representing an ontology o is $\overrightarrow{D_\Omega}(o) = \langle w_1, ..., w_n \rangle$ where each w_i represents the weight of term $t_i \in T$ for the ontology o.*

We have selected three types of weights :

- boolean weights: $w_i = 1$ if t_i occurs in o, $w_i = 0$ otherwise.
- frequency weights : $w_i = TF(t_i, o)$
- TF·IDF [Robertson and Spärck Jones, 1976] : $w_i = TF(t_i, o). \ln \frac{|O|}{|\{o|t_i \in l_{te}(o)\}|}$

TF·IDF [Robertson and Spärck Jones, 1976] measures the degree to which an ontology is relevant to another. This last approach is symmetric. However, it is based on global frequency computation, so each time a new ontology appears, then the measures change. This could be solved by using frozen IDF weights from a larger corpus, which would not change the model when adding ontologies. But the main problem is to find a corpus which is generic enough to contain all terms of the ontologies to compare. In addition, when we would measure distances between ontologies designed on a restricted domain, all these ontologies would be very similar and then difficult to compare.

For comparing ontologies, it is possible to apply various similarity measures between vectors associated to ontologies:

- Jaccard index;
- Euclidean distance;
- cosine index, which takes the cosine of the angle between the vectors.

The more basic measure (Jaccard index with boolean weights) is close to the complement to 1 of the Hamming distance on class names.

The measures presented so far can be used but they are very dependent on the language being used: if the ontology names are expressed in different natural languages, these measures will certainly not be helpful. However, it can be considered, that if the user is not able to understand the terms used in an ontology, these ontologies should not be considered as close.

3.1.3 Axiom-based similarity

The axiom-based similarity basically applies the same formula as in Definition 6 to the axioms of an ontology:

[1] Defined in Chapter 2

Definition 8 (Common axiom similarity)**.** *Let o and o' be two ontologies and $ax(\cdot)$ a function returning the set of axioms in these ontologies, the common axiom similarity is characterised by:*

$$\sigma_{cax}(o, o') = \frac{|ax(o) \cap ax(o')|}{max(|ax(o)|, |ax(o')|)}$$

3.1.4 Semantic measures

Finally, distances proposed so far offer no guarantee that they satisfy semantic constraints. What would be a semantic distance? This is a distance that is based on the meaning of the ontologies. We define this here as based on a consequence relation ($\models$).

Definition 9 (Semantic distance)**.** *Given two ontologies o and o' and a consequence relation $\models$ for the language in which they are expressed, a distance d on ontologies is a semantic distance if and only if:*

$$\begin{array}{ll} \forall o, o', o'', o \models o' \text{ and } o' \models o'' & \\ \quad \Rightarrow d(o, o') \leq d(o, o'') \text{ and } d(o', o'') \leq d(o, o'') & (\models\text{-compatibility}) \\ \forall o, o', o \models o' \text{ and } o' \models o, \text{ if and only if } \delta(o, o') = 0 & (\models\text{-definiteness}) \end{array}$$

We will try to define such distances based on the set of consequences of an ontology. Any kind of relations between sets can be used for comparing these sets. For instance, one can adapt the Hamming distance to consider consequences:

Definition 10 (Ideal semantic distance)**.** *Given two ontologies o and o' and a function Cn providing their sets of consequences, the distance given by:*

$$\delta_{is}(o, o') = 1 - \frac{|Cn(o) \cap Cn(o')|}{|Cn(o) \cup Cn(o')|}$$

is a semantic distance.

Unfortunately the set $Cn(o)$ of consequences of an ontology is usually an infinite set [Huang and Stuckenschmidt., 2005; Euzenat, 2007]. A classical alternative would be to use instead a reduction of these ontologies. However, this reduction is not unique in general, so the size of these reductions may vary. It will thus be difficult to use measures based on cardinality.

One can use the consequences of an ontology relative to another as in the common consequence similarity:

Definition 11 (Common consequence similarity)**.** *Let o and o' be two ontologies and $LCn(o, o') = o \cap Cn(o')$ a function returning the consequences of o' in o, the common consequence similarity is characterised by:*

$$\sigma_{ccsq}(o, o') = \frac{|LCn(o, o') \cap LCn(o', o)|}{max(|o|, |o'|)}$$

However, σ_{ccsq} is still depending on syntax: the form used for expressing o and o' may lead to different results. Figure 3.1 and Table 3.1 illustrate this point, which is clearer from the following table: the only part which does not depend on the syntax is $Cn(o)$ which is infinite. It could be compared with some finite set, but all the finite sets are dependent on the syntax used for o (i.e., non invariant). It is thus difficult to provide a measure which purely depends on semantics (thought it is possible to test for equivalence or entailment).

Of course, if the ontology language is not very expressive, i.e., if it accepts finite closure or unique reductions, then it is possible to compute semantic distances on closure or reduction. This is true of languages which only allow for the expression of taxonomies.

Moreover, we have always considered that the two ontologies where comparable. In reality, any semantic web ontology using URI for naming concepts will not be comparable to another as an alignment is required when comparing these ontologies. This kind of measure can therefore only be used when an alignment is not needed.

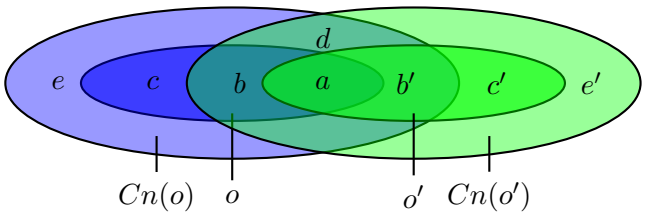


Figure 3.1: Two ontologies and their relations through the set of their consequences. Both set of statements in the ontologies (o and o') are displayed, as well as their sets of consequences ($Cn(o)$ and $Cn(o')$. The sets which can be algebraically computed from them are named by letters and referred to in Table 3.1

set	definition	unique	invariant	finite
	o	√		√
	$Cn(o)$	√	√	
a	$o \cap o'$	√		√
b	$o \cap Cn(o') - (o \cap o')$	√		√
c	$o - Cn(o')$	√		√
d	$(Cn(o) \cap Cn(o')) - (o \cup o')$	√		
e	$Cn(o) - (o \cup Cn(o')$	√		

Table 3.1:

3.1.5 Similarity based on key concepts

The *key concepts* of an ontology are the n concepts from the ontology that best describe it [Peroni *et al.*, 2008]. They are identified thanks to a set of measures taking into account various aspects of the ontology. We use the algorithm described in [Peroni *et al.*, 2008] to compute the key concepts of an ontology.

One of the important aspects concerning key concepts is that they are ranked. Thus we use:

- $KC(o, n)$ to indicate the first n key concepts of the ontology o;
- A_o to indicate a generic element belonging to $KC(o, n)$;

For example, given two ontologies o and o' with $KC(o, 3) = [A_o, B_o, C_o]$ and $KC(o', 3) = [C_{o'}, B_{o'}, A_{o'}]$ respectively, it can be seen that A_o represents the most important concept in o while the same concept in o', $A_{o'}$, represents a less important concept in o'.

We define below two measures of similarity:

σ_{ukc}**:** unranked key concept similarity, which considers the key concepts of an ontology without their level of importance;

σ_{rkc}**:** ranked key concept similarity, which takes the level of importance into account.

Definition 12 (Unranked key concept similarity)**.** *Given two ontologies o and o', an integer n and a function KC providing the n key concepts of an ontology, the* unranked key concept similarity *is defined by:*

$$\sigma_{ukc}(o, o') = \frac{|KC(o, n) \cap KC(o', n)|}{n}$$

This definition is similar to that already used in Definition 6. In fact, if we abstract from the concepts themselves and consider that they are equivalents when they have the same labels, then this measure is a refinement of σ_{cn} which only takes the key concepts into account.

The ranked key concept similarity considers the key concepts of an ontology with their level of importance. In this case, $KC(o,n)$ is a list of elements. The indicated order on the list express the importance of the concepts in the considered ontologies.

Definition 13 (Ranked key concept similarity)**.** *Given two ontologies o and o', an integer n and a function KC providing the n key concepts of an ontology, the* ranked key concept similarity *is defined by:*

$$\sigma_{rkc}(o,o',n) = \frac{\sum_{c \in KC(o,n)} \sigma_{kc}(KC(o,n), KC(o',n), c)}{n}$$

such that:

$$\sigma_{kc}(KC(o,n), KC(o',n), c) = \begin{cases} 1 - \frac{|rank(KC(o,n),c) - rank(KC(o',n),c)|}{n} & \text{if } c \in KC(o,n) \cap KC(o',n) \\ 0 & \text{otherwise.} \end{cases}$$

Here σ_{kc} calculates the inverse of the distance between the same key concept, c, in two different lists of key concepts and $rank(KC(o,n),c)$ indicates the level of importance of the key concept c in $KC(o,n)$, i.e., the rank of the concepts in the list.

Example 1. *Let o and o' be two ontologies, with $KC(o,3) = [\text{Person}_o, \text{Organization}_o, \text{Publication}_o]$ and $KC(o',3) = [\text{Publication}_{o'}, \text{Organization}_{o'}, \text{Person}_{o'}]$ respectively and their degree of importance be assigned as follows:*

1. $rank(KC(o,3), \text{Person}_o) = 3$;
2. $rank(KC(o,3), \text{Organization}_o) = 2$;
3. $rank(KC(o,3), \text{Publication}_o) = 1$;
4. $rank(KC(o',3), \text{Publication}_{o'}) = 3$;
5. $rank(KC(o',3), \text{Organization}_{o'}) = 2$;
6. $rank(KC(o',3), \text{Person}_{o'}) = 1$;

then

1. $\sigma_{kc}(KC(o,3), KC(o',3), \text{Person}) = 1 - \frac{|3-1|}{3} = 1/3$
2. $\sigma_{kc}(KC(o,3), KC(o',3), \text{Organization}) = 1 - \frac{|2-2|}{3} = 1$
3. $\sigma_{kc}(KC(o,3), KC(o',3), \text{Publication}) = 1 - \frac{|1-3|}{3} = 1/3$

Therefore, $\sigma_{rkc}(o,o',3) = 5/9$ meaning that, although the ontologies have the same key concepts, yielding $\sigma_{ukc}(o,o',3) = 1$, they have different importance in the two ontologies, certainly revealing difference in these ontologies. In fact, we have Publication as the most important concept in o' and Person in o. So, o' could be, for example, an ontology talking about the publications, while o could be, on the other hand, an ontology about publishers. Therefore, they are slightly different from each other conceptually.

3.2 Distances between ontology entities

The main way to measure a distance between ontologies is to compare their entities, e.g., their classes, properties, individuals. So any distance that has been developed for matching ontologies can be extended as a distance between ontologies. There are such entity distances mentioned in [Euzenat and Shvaiko, 2007] since they are the most common basis of ontology matching.

3.2.1 Label-based distances

Lexical aggregation-based similarity measure

This first local measure only relies on the lexical information coming from annotations and the local name. Given an entity e, $T(e) = \{l_{ln}(uri(e))\} \cup l_{annot}(e)$ represents the set containing the annotations and the

local name of e. The lexical similarity between two entities $e \in o$ and $e' \in o'$ is given by:

$$\sigma_l(e, e') = \frac{\sum_{(a,b)\in M(e,e')} \sigma_{jw}(a, b)}{min(|T(e)|, |T(e')|)}$$

such that $M(e, e')$ is a maximum weight matching of $T(e) \times T(e')$, and σ_{jw} is the Jaro Winkler similarity.

3.2.2 Structural distances

There have been many proposals for distances between ontology concepts. Indeed most of the proposed distances in the literature are based on concept distances [Mädche and Staab, 2002; Euzenat and Valtchev, 2004; Hu *et al.*, 2006; Vrandečić and Sure, 2007]. We present some representative measures below.

OLA similarity

One good candidate as structural similarity is the one designed for OLA [Euzenat and Valtchev, 2004] because it relies on every feature of ontologies. OLA first encodes the ontologies into a labelled graph called OL-graph. Then, given an OL-Graph node, the similarity between two OL-graph nodes depends on:

- the similarity of the terms used to denote them, i.e., URIs, labels, names, etc.,
- the similarity of the pairs of neighbour nodes in the respective OL-Graphs that are linked by edges expressing the same relationships, e.g., class node similarity depends on similarity of superclasses, of property restrictions and of member objects,
- the similarity of other local descriptive features depending on the specific category, e.g., cardinality intervals, property types.

Datatype and data value similarities are external and therefore they are either user-provided or measured by a standard function, e.g., string identity of values and datatype names/URIs.

Formally, given a category X together with the set of relationships it is involved in, $\mathcal{N}(X)$, the similarity measure $Sim_X : X^2 \to [0, 1]$ is a weighted sum aggregating the different features in $\mathcal{N}(X)$ with specific weights denoting the relative importance of each feature. It is defined as follows:

$$Sim_X(x, x') = \sum_{\mathcal{F}\in\mathcal{N}(X)} \pi^X_{\mathcal{F}} MSim_Y(\mathcal{F}(x), \mathcal{F}(x')).$$

The function is normalised, i.e., the weights $\pi^X_{\mathcal{F}}$ sum to one, $\sum_{\mathcal{F}\in\mathcal{N}(X)} \pi^X_{\mathcal{F}} = 1$. The set functions $MSim_Y$ compare two sets of nodes of the same category. Table 3.2 illustrates the similarities used by OLA.

The value of these similarities is computed as a fix-point of the set of equations defining the similarity. This process always converges towards a solution.

Triple-based iterative similarity measure

We have also defined a new measure based on RDF triple similarity [David and Euzenat, 2008]. This similarity is defined as a convergent sequence. Initially, some similarity values between nodes in the RDF graph are initialised via string similarity.

In a triple-based representation of an ontology, we consider 4 types of nodes:

- blank node: node having no URI,
- local node: named node defined in the ontology, i.e., node having the same namespace as the ontology,
- external node: named node not defined in the ontology : node imported from other ontologies, or node from the language,

Funct.	Node	Factor	Measure
Sim_O	$o \in \Omega$	$\lambda(o)$	sim_L
		$a \in A, (o,a) \in \mathcal{A}$	$MSim_A$
Sim_A	$a \in A$	$r \in R, (a,r) \in \mathcal{R}$	Sim_R
		$o \in \Omega, (a,o) \in \mathcal{U}$	$MSim_O$
		$v \in V, (a,v) \in \mathcal{U}$	$MSim_V$
Sim_V	$v \in V$	value literal	type dependent
Sim_C	$c \in C$	$\lambda(c)$	sim_L
		$p \in P, (c,p) \in \mathcal{A}$	$MSim_P$
		$c' \in C, (c,c') \in \mathcal{S}$	$MSim_C$
sim_D	$d \in D$	$\lambda(r)$	XML-Schema
Sim_R	$r \in R$	$\lambda(r)$	sim_L
		$c \in C, (r, \texttt{domain}, c) \in \mathcal{R}$	$MSim_C$
		$c \in C, (r, \texttt{range}, c) \in \mathcal{R}$	$MSim_C$
		$d \in D, (r, \texttt{range}, d) \in \mathcal{R}$	Sim_D
		$r' \in R, (r,r') \in \mathcal{S}$	$MSim_R$
Sim_P	$p \in P$	$r \in R, (p,r') \in \mathcal{S}$	Sim_R
		$c \in C, (p, \texttt{all}, c) \in \mathcal{R}$	$MSim_C$
		$n \in \{0,1,\infty\}, (p, \texttt{card}, n) \in \mathcal{R}$	equality

Table 3.2: Similarity function decomposition (`card` = `cardinality` and `all` = `allValuesFrom`).

– literal node.

$$simN_0(n_1,n_2) = \begin{cases} 1 \text{ if } n_1 = n_2, \\ simS(n_1,n_2) \text{ if } n_1 \text{ and } n_2 \text{ are literals}, \\ simS(n_1,n_2) \text{ if } n_1 \text{ and } n_2 \text{ are local nodes}, \\ 0 \text{ otherwise.} \end{cases}$$

where $simS$ is a syntactic similarity such as Jaro-Winkler or Levenstein (see [Euzenat and Shvaiko, 2007] for details if needed).

This measure is iteratively refined until the amount of change reaches a user-defined threshold. Given two nodes n_1 and n_2, the node similarity $simN_{i+1}$ between n_1 and n_2 is defined, for the stage $i+1$, by:

$$simN_{i+1}(n_1,n_2) = \frac{1}{N_{max}} \sum_{k \in \{sub,pred,obj\}} \underset{\substack{t_x \in T_x^k \\ t_y \in T_y^k}}{\Delta} \left(simT_i(t_x,t_y)\right) \times \max(|T_x^k|,|T_y^k|)$$

where $T_x^k = \{t_x | t_x.k = n_1\}$, $T_y^k = \{t_y | t_y.k = n_2\}$, and
$N_{max} = \sum_{k \in \{sub,pred,obj\}} \max(|T_x^k|,|T_y^k|)$

Δ is a collection similarity such as those introduced in Section 3.3. $simT_i$ is a similarity between two triples defined as the average values of similarities between nodes of two triples at stage i:

$$simT_i(t_1,t_2) = \frac{simN_i(t_1.pred, t_2.pred)}{3} \sum_{k \in \{sub,pred,obj\}} simN_i(t_1.k, t_2.k)$$

3.3 Collection distances

Once a distance δ (or similarity σ) among concepts is available, turning it into an ontology distance is not straightforward. There are different choices for extending measures at the concept level to the ontology level. This is achieved with the help of a collection measure Δ which computes the ontology measure value from the concept measures values. We present some of these below. These collection measures are defined as distance measures, but they can be turned into similarity measures easily.

Definition 14 (Average linkage)**.** *Given a set of entities E and a dissimilarity function $\delta : E \times E \to [0\ 1]$, the average linkage measure between two ontologies is a dissimilarity function $\Delta_{alo} : 2^E \times 2^E \to [0\ 1]$ such that $\forall o, o' \subseteq E$,*

$$\Delta_{alo}\langle o, o'\rangle = \frac{\sum_{(e,e')\in o\times o'} \delta(e, e')}{|o| \times |o'|}.$$

Definition 15 (Hausdorff distance)**.** *Given a set of entities E and a dissimilarity function $\delta : E \times E \to [0\ 1]$, the Hausdorff distance between two sets is a dissimilarity function $\Delta_{Hausdorff} : 2^E \times 2^E \to [0\ 1]$ such that $\forall o, o' \subseteq E$,*

$$\Delta_{Hausdorff}(o, o') = \max(\max_{e\in o}\min_{e'\in o'} \delta_K(e, e'), \max_{e'\in o'}\min_{e\in o} \delta_K(e, e'))$$

The problem with the Hausdorff distance, as with other linkage measures, is that its value depends on the distance between one pair of members of the sets. The average linkage, on the other hand, depends on the distance between all the possible comparisons. None of these are satisfactory because the distance should reflect the distance between each member of its counterpart if it exists. Matching-based dissimilarities [Valtchev, 1999] measure the dissimilarity between two ontologies by taking into account an alignment (matching) between these two ontologies. It can be defined independently of any alignment by using the minimum weight maximum matching. Closeness in such a measure depends on the actual correspondences between two ontologies (not an average). It will thus be possible to translate the knowledge of one ontology into another. However, these measures will be more difficult to compute.

Definition 16 (Minimum weight maximum graph matching distance)**.** *Given a set of entities E and a dissimilarity function $\delta : E \times E \to [0\ 1]$, for any ontologies $o, o' \subseteq E$, a minimum weight maximum graph matching is a one-to-one matching $M \subseteq o \times o'$, such that for any one-to-one alignment $M' \subseteq o \times o'$,*

$$\sum_{\langle p,q\rangle\in M} \delta(p, q) \leq \sum_{\langle p,q\rangle\in M'} \delta(p, q)$$

Then, one can define the distance between these two ontologies as:

$$\Delta_{mwmgm}(o, o') = 1 - \frac{\sum_{\langle p,q\rangle\in M}[1 - \delta(p, q)]}{\max(|o|, |o'|)}$$

Computing the minimum weight maximum graph matching distance (MWMGM) from a similarity involves two related steps: extracting an alignment between the ontologies and computing the distance value. The value depends on the extracted alignment and usual algorithms extract a matching, i.e., a one-to-one alignment. While this may be a reasonable choice mathematically, this may not be the needed alignment on which to ground such a distance. Hence, MWMGM leaves space for improvement.

Most of the measures of §3.1 can be used by considering that the "best match" is known (this is the one which maximises the similarity).

3.4 Conclusion

Already many measures are available for comparing ontologies.

We have divided the measures into those which assume that entities are found in both ontologies and compare them globally, and those which require individual comparison of the entities in order to evaluate their similarity. The latter requires measures for both comparing ontology entities and for aggregating these into a measure between ontologies. This allows for the generation of many measures.

In the next chapter, we will present another kind of measure that depends on the relations between the ontologies. They may be more suited for some uses but they rely on the availability of alignments between ontologies.

Chapter 4

Similarity in the alignment space

Aside from classical measures based on the content of ontologies, we designed measures based on their relations with other ontologies. For that purpose, we distinguish what we call the alignment space populated by ontologies and alignments and the ontology space which is populated by ontologies alone (§4.1). We design a similarity between ontologies which is based on the characteristics of this space. A first approach considers it as a graph and the similarity between ontologies is based on its topology (§4.2). We show that this does not account for the ability that alignments have to faithfully translate a query. We provide new measures (§4.3) that are based on the capacity of alignments to cover a large proportion of the ontology entities as well as to keep them distinct.

We will provide justifications for these measures with regard to the application to translate queries. We illustrate this by providing global and distributed algorithms for computing these similarities and show how they can be used for propagating queries (§4.4).

4.1 Alignment space and similarities

In this section, we first introduce the alignment model that we use and then review the general properties that similarities between ontologies must satisfy. We end by presenting precisely the considered problem.

4.1.1 Alignments

We provide the definition of alignments from [Euzenat and Shvaiko, 2007] where the interested reader will find further discussions.

Alignments express the correspondences between entities belonging to different ontologies.

Definition 17 (Correspondence)**.** *Given two ontologies o and o' with associated entity languages Q_L and $Q_{L'}$, a set of alignments relations Θ, and a confidence structure over Ξ, a* correspondence *is a quadruple:*

$$\langle e, e', r, n \rangle$$

such that

- $e \in Q_L(o)$ *and* $e' \in Q_{L'}(o')$*;*
- $r \in \Theta$*;*
- $n \in \Xi$*.*

The correspondence $\langle e, e', r, n \rangle$ asserts that the relation r holds between the ontology entities e and e' with confidence n.

A set of relations Θ is used for expressing the relations between the entities. Matching algorithms primarily use the equivalence relation ($=$) meaning that the matched objects are the same, or are equivalent if these

are formulas. It is possible to use relations from the ontology language within Θ. For instance, using OWL, it is possible to take advantage of the `owl:equivalentClass`, `owl:disjointWith` or `rdfs:subClassOf` relations. They can be used without reference to any ontology language.

For pragmatic reasons, the relationship between two entities is assigned a degree of confidence which can be viewed as a measure of trust in the fact that the correspondence holds – 'I trust 70% the fact that the correspondence is correct or reliable'. These values are taken from a bounded ordered set Ξ that we call a confidence structure.

Finally, an alignment is defined as a set of correspondences.

Definition 18 (Alignment)**.** *Given two ontologies o and o', an* alignment *is made up of a set of correspondences between pairs of entities belonging to $Q_L(o)$ and $Q_{L'}(o')$ respectively.*

Here, we will ignore completely the confidence measure and will only consider alignments in which the relations between entities are equivalence ($=$) or the subsumption ($\sqsubseteq, \sqsupseteq$).

Definition 19 (Simple alignment)**.** *Given two ontologies o and o', a* simple alignment *is a set of triples $\langle e, e', r\rangle$, such as:*

- *$e \in Q_L(o)$ and $e' \in Q_{L'}(o')$ are named entities issued from the ontologies;*
- *$r \in \{=, \sqsubseteq, \sqsupseteq\}$.*

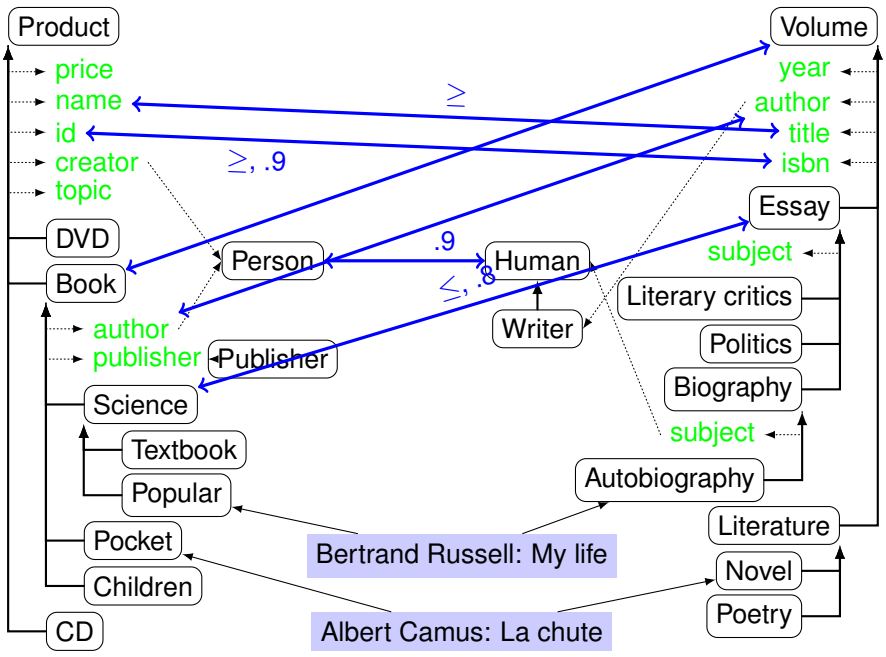


Figure 4.1: Alignment between two ontologies. Correspondences are expressed by two-headed (blue) arrows with one extremity in each ontology. By default their relation is $=$ and their confidence value is 1.0; otherwise, these are mentioned near the arrows.

In Figure 4.2, the alignments are as follows:

$$A_{1,2} \text{ is } \{\langle a_1, a_2, =\rangle, \langle b_1, b_2, =\rangle, \langle c_1, c_2, =\rangle\}$$
$$A_{1,3} \text{ is } \{\langle d_1, d_3, =\rangle, \langle e_1, e_3, =\rangle\}$$
$$A_{2,3} \text{ is } \{\langle c_2, c_3, =\rangle, \langle d_2, d_3, =\rangle, \langle e_2, e_3, =\rangle\}$$
$$A_{2,4} \text{ is } \{\langle a_2, a_4, =\rangle, \langle b_2, b_4, =\rangle, \langle c_2, c_4, =\rangle\}$$
$$A_{3,4} \text{ is } \{\langle c_3, c_4, =\rangle, \langle d_3, d_4, =\rangle, \langle e_3, e_4, =\rangle\}$$

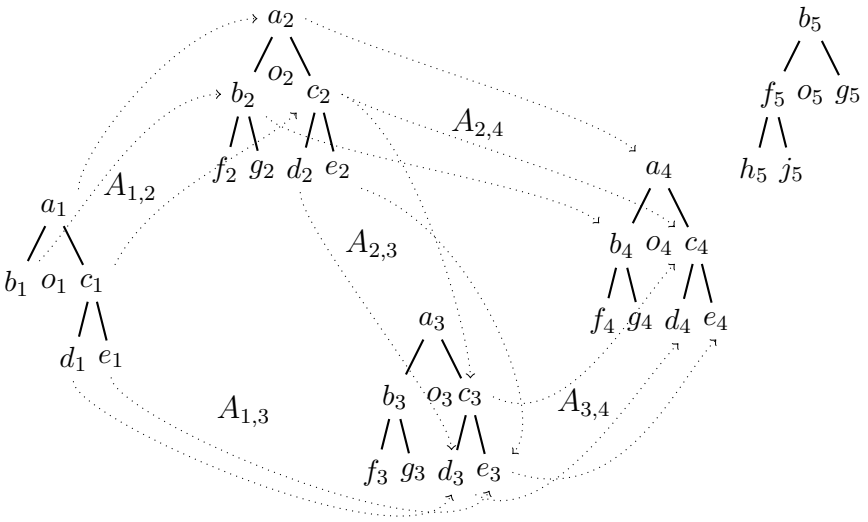


Figure 4.2: Five ontologies (o_1, o_2, o_3, o_4 and o_5) and five alignments ($A_{1,2}$, $A_{1,3}$, $A_{2,3}$, $A_{2,4}$ and $A_{3,4}$).

We will freely use the notation $A(s)$ for the action of replacing any ontology entity of s by the term it is in correspondence through A if any (otherwise, the entity is simply discarded). Of course, this assumes that there is only one such correspondence. When we restrict ourselves to equivalence relations in alignments, they can be used in both ways, i.e., $A_{1,2}^{-1} = \{\langle e', e, =\rangle; \langle e, e', =\rangle \in A_{1,2}\}$ can be used for converting queries from o_2 to o_1.

4.1.2 Alignment space

We call "alignment space" a set of ontologies and alignments between these ontologies. The assumption is that the set of alignments is already available. Hence the similarity will not measure the expected quality of a non existing alignment but the quality of using existing alignments.

Definition 20 (Alignment space)**.** *An* alignment space $\langle \Omega, \Lambda \rangle$ *is made of a set* Ω *of ontologies and a set* Λ *of simple alignments between ontologies in* Ω*. We denote as* $\Lambda(o, o')$ *the set of alignments in* Λ *between* o *and* o'*.*

The alignment space can be represented as a multigraph[1] $G_{\Omega,\Lambda}$ in which ontologies are vertices and alignments are edges. Figure 4.3 (left) represents the graph corresponding to the alignments and ontologies of Figure 4.2.

We call *complete alignment space* for a set of ontologies Ω, the alignment space in which $\Lambda = \mathbf{\Lambda}_\Omega$ is the set of all possible alignments over two ontologies in Ω, i.e., $\forall o, o' \in \Omega, \Lambda(o, o') \neq \emptyset$.

A path is simply defined as a non cyclic path in $G_{\Omega,\Lambda}$. We consider only non cyclic paths since applying alignments cannot increase information, hence looping through the same ontology can only be harmful.

Definition 21 (Path)**.** *Given a set of alignments* Λ*, a (non cyclic)* path π *in* Λ *is a sequence of alignments* $A_1, \ldots A_n$ *such that for each* $i \in [0, n-1]$*,* $A_i \in \Lambda(o_i, o_i')$ *and* $A_{i+1} \in \Lambda(o_{i+1}, o_{i+1}')$*,* $o_i' = o_{i+1}$ *and* $\forall i, j \in [0, n-1], i \neq j \Rightarrow o_i \neq o_j$*. The set of (non cyclic) paths in an alignment space is named* Π *and the set of (non cyclic) paths starting at an ontology* o *and ending at an ontology* o' *is identified by* $\Pi(o, o')$*.*

For instance, $\Pi(o_1, o_4)$ contains the four following non cyclic paths: $A_{1,2} \cdot A_{2,4}$, $A_{1,3} \cdot A_{3,4}$, $A_{1,2} \cdot A_{2,3} \cdot A_{3,4}$ and $A_{1,3} \cdot A_{2,3}^{-1} \cdot A_{2,4}$.

[1] A multigraph is needed, because there may be several alignments available between two ontologies

We extend the notation $A(s)$ to paths. If $\pi = A_1, \dots A_n$, then $|\pi| = n$ and $\pi(s) = A_n(\dots A_1(s)\dots)$. By convention, we will introduce the empty path ϵ from one ontology to itself, such that $\epsilon(s) = s$ and $|\epsilon| = 0$. We will note $o \dot{\in} \pi$ if o is one of the ontologies involved in an alignment of the path π.

Definition 22 (Alignment composition)**.** *Alignment composition is an operation $\cdot$ which takes two alignments $A \in \mathbf{\Lambda}_\Omega(o, o')$ and $A' \in \mathbf{\Lambda}_\Omega(o', o'')$ and returns an alignment $A \cdot A' \in \mathbf{\Lambda}_\Omega(o, o'')$ such that:*

$$A \cdot A' = \{\langle e, e', r \cdot r' \rangle; \exists \langle e, e''; r \rangle \in A \wedge \exists \langle e'', e', r' \rangle \in A'\}$$

This definition depends on a composition operation $\cdot$ over Θ as defined in [Euzenat, 2008]. When relations are restricted to equivalence, the result is equivalence. Hence $A_{1,2} \cdot 1_{2,3} = \{\langle c_1, c_3, = \rangle\}$.

Definition 23 (Compositional closure)**.** *Given a set of alignments Λ, its compositional closure $\overline{\Lambda}$ is a set of alignments such that:*

- $\Lambda \subseteq \overline{\Lambda}$;
- $\forall A \in \overline{\Lambda}(o, o''), A' \in \overline{\Lambda}(o'', o'), A \cdot A' \in \overline{\Lambda}(o, o')$.

By extension we will say that an alignment space is closed if and only if its set of alignments is closed. Of course, a closed alignment space is an alignment space.

The semantics of an alignment space should be the same as that of its closure (or reduction), and thus whatever property holds for a space holds for its closure.

4.2 Path-based similarities

In a first attempt at evaluating similarity in the alignment space, alignments can be considered as bridges between ontologies enabling their connection.

The first kind of similarity between two ontologies may be based on paths between these ontologies in the graph $G_{\Omega,\Lambda}$. In fact, the existence of a path guarantees that it will be possible to transform queries from one ontology to another. This can be refined by considering different values if the path is made of zero, one or several alignments:

Definition 24 (Alignment path similarity)**.**

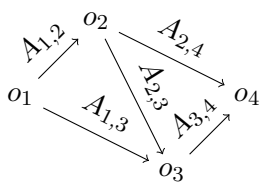
$$\sigma_{ap}(o, o') = \begin{cases} 1 & \text{if } o = o' \\ 2/3 & \text{if } o \neq o' \text{ and } \Lambda(o, o') \neq \emptyset \\ 1/3 & \text{if } o \neq o' \text{ and } \Lambda(o, o') = \emptyset \text{ and } \Pi(o, o') \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$$

Hence, such a similarity is minimal between two non connected ontologies and it is normalised. It is symmetric as long as alignments are considered symmetric. It is relatively easy to compute and provides information about the possibility to propagate information between two ontologies. However, it is not very precise in the number of transformations that may have to be performed to propagate this information.

So, a natural similarity measure depends on the shortest path in the graph $G_{\Omega,\Lambda}$. Indeed, the fewer alignments are applied to a query, the more it is expected that it is an accurate translation (we can assume information is lost each time a transformation is applied to a query). This measure is clearly a similarity.

Definition 25 (Shortest alignment path similarity)**.** *Given an alignment space $\langle \Omega, \Lambda \rangle$, the shortest alignment path similarity σ_{sap} between two ontologies $o, o' \in \Omega$ is the complement to 1 of the length of the shortest path between o and o' in $G_{\Omega,\Lambda}$:*

$$\sigma_{sap}(o, o') = \begin{cases} 1 - \frac{\min_{\pi \in \Pi(o,o')} |\pi|}{\varnothing_{\Omega,\Lambda}} & \text{if } \Pi(o, o') \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$$



σ_{ap}	o_1	o_2	o_3	o_4	o_5
o_1	1	$2/3$	$2/3$	$1/3$	0
o_2	0	1	2/3	2/3	0
o_3	0	0	1	2/3	0
o_4	0	0	0	1	0
o_5	0	0	0	0	1

σ_{sap}	o_1	o_2	o_3	o_4	o_5
o_1	1	$3/4$	$3/4$	$1/2$	0
o_2	0	1	3/4	3/4	0
o_3	0	0	1	3/4	0
o_4	0	0	0	1	0
o_5	0	0	0	0	1

Figure 4.3: Alignment space (left) corresponding to Figure 4.2 and the corresponding path-based measures (right). σ_{sap} is computed with $\varnothing_{\Omega,\Lambda}$ as the length of the longest path (3) plus 1.

In order to normalise the similarity, $\varnothing_{\Omega,\Lambda}$ can be either the size of Ω or the diameter of $G_{\Omega,\Lambda}$, i.e., the length of the longest shortest path, plus 1.

The computation of this similarity is not significantly more expensive than the computation of the alignment path similarity. This one is more precise because it will depend on how many transformations at least the information to propagate should support between two ontologies.

However, an alignment between two ontologies can be just empty: this does not mean that the ontologies are very close but rather that they are very different.

Even if alignments are not empty, this measure does not tell how much of an ontology is preserved through the translation. Indeed, considering the alignment space of Figure 4.3, it shows that for both measures, o_4 is farther from o_1 than o_3, however, if one looks at the alignments in Figure 4.2, the composition of $A_{1,2}$ and $A_{2,4}$ preserves more information than the alignment $A_{1,3}$.

4.3 Coverage-based similarities

If we want to go further in measuring the precise similarity for the querying application, another measure retains as the similarity between two ontologies the ratio of elements of the ontology which are covered by an alignment. In fact this measure can be applied to any set of elements, not just an ontology. Hence the coverage can be given with regard to an ontology entity (the ratio will be 1 or 0), to a query or to an ontology. It corresponds to the percentage of entities which have an image through the alignment.

Definition 26 (Alignment coverage)**.** *Given a set of ontology entities s over an ontology o and an alignment $A \in \Lambda(o, o')$, the* coverage *of s by A is given by:*

$$cov(s, A) = \frac{|\{e \in s | \exists \langle e, e', r\rangle \in A\}|}{|s|}$$

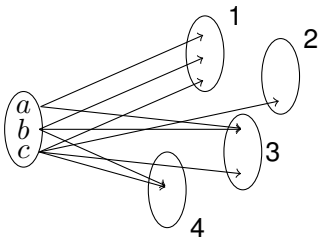
There is a second important notion which is the ability for the alignment to preserve the difference between entities which are deemed different in the source ontology. The alignment distinguishability measure is the proportion of matched entities which are kept distinct. This could be considered as preservation of information.

Definition 27 (Alignment distinguishability)**.** *Given a set of ontology entities s over an ontology o and an alignment $A \in \Lambda(o, o')$, the* distinguishability *(or separability) of s by A is given by:*

$$sep(s, A) = \frac{|\{e' | \exists \langle e, e', r\rangle \in A \wedge e \in s\}|}{|\{e \in s | \exists \langle e, e', r\rangle \in A\}|}$$

These two notions are obviously tied to the concepts of existence and injectivity of a function. Hence, cov does not have a strong relation with sep since one depends on $Q_L(o)$ and the other on $Q_{L''}(o'')$.

In the following, we will use a measure which accounts for both coverage and distinguishability at once: instead of making the count of ontology entities which have an image by the alignment, we will only count



measure	coverage	distinguishability	covdis
1	1	1	1
2	1/3	1	1/3
3	1	2/3	2/3
4	2/3	1/2	1/3

Figure 4.4: Simple alignments (left) and the corresponding coverage and distinguishability measures (right).

those distinct images. Hence the lack of distinguishability will automatically lower the returned value (this is presented in Figure 4.4).

Definition 28 (Alignment coverage distinguishability)**.** *Given a set of ontology entities s over an ontology o and an alignment $A \in \Lambda(o, o')$, the coverage distinguishability of s by A is given by:*

$$covdis(s, A) = cov(s, A) \times sep(s, A) = \frac{|A(s)|}{|s|}$$

This measure can easily be extended to path. Figure 4.4 shows the differences between the three measures.

4.3.1 Largest coverage

Hence, the natural measure for similarity is that of largest coverage.

Definition 29 (Largest covering similarity)**.** *Given an alignment space $\langle \Omega, \Lambda \rangle$, the* largest covering similarity *σ_{lc} between two ontologies $o, o' \in \Omega$ is*

$$\sigma_{lc}(o, o') = \max_{A \in \Lambda(o,o')} covdis(o, A)$$

Such a measure is clearly not symmetric anymore, even if the alignment is only made of equalities: the ratio depends on the size of the source ontology, independently of the target ontology. It is not definite either: if all information is preserved and distinguishable, the similarity will be 1 though the ontologies are not the same.

We have applied this measure to alignments and not to paths of alignments. Whilst there may be a direct alignment between two ontologies, there may be a path between them which better covers and preserves the ontology entities. This is what would happen if there were a direct alignment $\{\langle a_1, a_4 \rangle\}$ from o_1 to o_4. Then the coverage would be $.2$, while the coverage provided by the path $A_{1,2} \cdot A_{2,4}$ is $.6$. In that respect, o_4 is more similar to o_1 than o_3.

Hence, it is necessary to measure the similarity with regard to the paths which lead to an ontology. Composing the measures obtained by the alignments in order to get the measure for the path is not correct. Indeed, if two alignments have a similarity of 80%, the similarity of their compound alignment can be anything between 0% and 80%. We have computed the product of the similarity as the $\sigma_{\times lc}$ in Table 4.1.

It is thus necessary to evaluate path coverage distinguishability. In order to address this problem, we introduce measures which are based on path instead of simple alignments. This is the largest covering preservation similarity:

Definition 30 (Largest covering preservation similarity)**.** *Given an alignment space $\langle \Omega, \Lambda \rangle$, the* largest covering preservation similarity *σ_{lcp} between two ontologies $o, o' \in \Omega$ is:*

$$\sigma_{lcp}(o, o') = \max_{\pi \in \Pi(o,o')} covdis(o, \pi)$$

The complement of this similarity is only a dissimilarity and not a distance. Indeed, this measure is not absolute but relative to a particular set of entities ($Q_L(o)$).

measure	o_1	o_2	o_3	o_4	o_5
σ_{lc}	1	$3/5$	$2/5$	0	0
$\sigma_{\times lc}$	1	$3/5$	$2/5$	$9/35$	0
σ_{lcp}	1	$3/5$	$2/5$	$3/5$	0
σ_{upc}	1	1	$3/5$	1	0

Table 4.1: Coverage and distinguishability based similarities with regard to o_1 for the ontologies of Figure 4.2.

4.3.2 Union path coverage

So far, we only considered that a query would take one path at a time and that the query would be entirely evaluated through this path. In this case, the above measure is perfectly accurate. However, very often a query is split into parts which are sent to different peers and the results are composed through join or union depending on the query.

In this case, the measure above does not reflect the semantics of the alignment space and does not provide a measure of the similarity of the two ontologies for evaluating these queries. The meaning of the alignment space can basically be rendered by the transitive, symmetric and union closure of this alignment space[2]. In consequence, the coverage distinguishability should be computed not on the path that brings the maximal coverage but on the coverage provided by the combination of all the possible paths.

Definition 31 (Union path coverage similarity)**.** *Given an alignment space $\langle \Omega, \Lambda \rangle$, the union path coverage σ_{upc} between two ontologies $o, o' \in \Omega$ is:*

$$\sigma_{upc}(o, o') = \frac{|(\bigcup_{\pi \in \Pi(o,o')} \pi)(s)|}{|s|}$$

This measure indeed takes full advantage of all the alignments provided within the alignment space. In particular, it is able to account that, in the example of Figure 4.2, any query expressed with regard to entities of ontology o_1 can be evaluated in ontology o_4, yet through different paths depending on the considered entities.

4.4 Computing similarity and evaluating queries

In the context of peer-to-peer systems, the goal is to retrieve answers to queries and to do it in the most complete and efficient way. Usually the alignment space is not set once for all: peers can enter and quit the network very quickly and new alignments can be introduced dynamically. However, at any given moment, each peer must make its decision in a particular alignment space.

So the problem is to know, for the current space, the similarity between ontologies, sometimes with regard to a particular query. We first provide an efficient algorithm for computing all similarities at once, before considering an algorithm more adapted to compute similarity in a peer-to-peer manner. We then show how this can help to answer queries.

4.4.1 Centralised case: reasoning with alignments

The "centralised" case is relevant to the idea of reasoning with alignments only [Zimmermann, 2008]. Indeed, any peer can compute the similarity with regard to its own ontology o and the set of alignments Λ: it does not have to know anything about the other peer's ontologies.

Hence, computing the similarity between a peer and any other peer in the system can be achieved by:

[2]We assume here that this alignment space is consistent.

1. creating a $|\Omega| \times |\Omega|$ matrix, initialised with $M^0(o,o) = \{\langle\{\langle e, e\rangle; e \in o\}, \epsilon\rangle\}$ and $M^0(o,o') = \emptyset$ if $o \neq o'$;
2. updating it with the following formula:

$$M^{i+1}(o,o') = M^i(o,o') \cup \{\langle\{\langle e, e'\rangle; \exists \langle e, e''\rangle \in s \wedge e' = A(e'')\}, \pi \cdot A\rangle; \exists o''; \langle s, \pi\rangle \in M^i(o,o'') \wedge A \in \Lambda(o'', o') \wedge o' \dot{\notin} \pi\}$$

3. if $M^i \neq M^{i+1}$ then go to 2.

Lemma 1. *The above algorithm:*

1. *terminates, and*
2. $M^\infty(o,o')$ *contains the set of paths from* o *to* o' *with the set of images of entities of* o *by the path.*

Proof. For 1. The process can only iterate as long as there are unexplored paths. Since the number of paths in $G_{\Omega,\Lambda}$ is finite, then the algorithm finitely stops. In fact it will apply step (2) one more time than the length of the longest path in $G_{\Omega,\Lambda}$.
For 2., at the beginning the matrix accounts for all paths of length 0, i.e., using 0 alignments. At each iteration, it starts with a matrix containing all paths of length i and considers all possibilities to add an alignment to a path in this matrix without looping. It then adds in the matrix all paths of length $i+1$. As any path contains a valid subpath of inferior length, when the algorithm reaches the longest path, and then stop, it will have found all the paths. □

Theorem 2.

$$\sigma_{lcp}(o,o') = \frac{\max_{\langle s,\pi\rangle \in M^\infty(o,o')} |s|}{|o|} \text{ and } \sigma_{upc}(o,o') = \frac{|\bigcup_{\langle s,\pi\rangle \in M^\infty(o,o')} \{e'; \langle e, e'\rangle \in s\}|}{|o|}$$

Proof. Since the matrix contains, in each cell, all paths between the two ontologies concerned by the cell as well as the images of entities of the first ontology when applied all the alignments of the path, this is the information required for computing these two measures. □

The information in the matrix is in fact sufficient for computing the distance with regard to the terms of a query or to directly translate the query.

4.4.2 Distributed case: spreading activation

The computation of the similarity in the distributed case may be somewhat different. It may not be necessary to compute it globally but rather, to compute it depending on the goal which is to be accomplished: evaluating a query.
Algorithm 1 (TRAVERSE) computes the similarity between two ontologies o and o' in the alignment space by traversing $G_{\Omega,\Lambda}$. So, it only computes one cell of the previous matrix. It is expressed as a branch and bound traversal whose actual instructions depend on the measure to be computed (see Table 4.2).
Of course, the same result can be obtained by spreading the computation from one peer to another peer for which the similarity has to be computed. Therefore, this algorithm can be considered a distributed algorithm once each recursive call to itself is made on a different peer. However, branch-and-bound does not translate well into a peer-to-peer algorithm: (i) if all the calls are made in parallel, the branch-and-bound part is useless, and (ii) many computations may be carried out several times if the graph is dense. Hence, a caching strategy will be necessary.
The OntoSim implementation of the alignment space measures are similar to this last algorithm.

Algorithm 1 A branch and bound alignment space traversing algorithm for computing all similarities.

TRAVERSE (s, π, o, o') :
  if $o = o'$ **then**
    return EXTRACT(s); // SUCCESS
  end if
  if $o \dot{\in} \pi$ **then**
    return $\emptyset$; // LOOP
  end if
  if COND(s) **then**
    return $\emptyset$; // BOUNDING CONDITION=STOP
  else
    $s' \leftarrow s$; $m \leftarrow \emptyset$;
    for all $A \in \Lambda(o, o'')$ WHILE $\neg$COND(s') **do**
      $r \leftarrow$ TRAVERSE(APPLY(A, s'), $\pi \cdot A, o'', o'$)
      $s' \leftarrow$ DIFF(s', r)
      $m \leftarrow$ SUM(m, r)
    end for
    return m
  end if

measure	lc	lcp	upc
INIT(o)	o	$\langle o, 0\rangle$	$\{\langle e, e\rangle; e \in o\}$
EXTRACT(s)	$\lvert s\rvert$	$\langle s_o, \lvert s_o\rvert\rangle$	s
COND(s)	false	$\lvert s_o\rvert < s_n$	$s = \emptyset$
APPLY(A, s)	$A(s)$	$\langle A(s_o), n\rangle$	$\{\langle e, A(e')\rangle; \langle e, e'\rangle \in s\}$
DIFF(s, s')	s	$\langle s_o, \max(s_n, s'_n)\rangle$	$\{\langle e, e'\rangle \in s; \nexists \langle e, e''\rangle \in s'\}$
SUM(s, s')	$\max(s, s')$	$\langle s_o, \max(s_n, s'_n)\rangle$	$s \cup s'$
$\sigma(o, o')$	$\frac{\text{TRAVERSE}(o,\epsilon,o,o')}{\lvert o\rvert}$	$\frac{\text{TRAVERSE}(\langle o,0\rangle,\epsilon,o,o')_n}{\lvert o\rvert}$	$\frac{\lvert\text{TRAVERSE}(\{\langle e,e\rangle;e\in o\},\epsilon,o,o')\rvert}{\lvert o\rvert}$

Table 4.2: Actions performed by the TRAVERSE algorithm depending on the measure.

4.4.3 Answering queries

Consider a query $q_1 = b_1 \sqcap e_1$ to be evaluated in a large peer-to-peer network embedding that of Figure 4.2. The first thing that has to be done by the program is to compute the similarity of the query with the ontologies of the other peers. It can do it by any of the two methods above. In both cases, it will have the paths by which a query can be transformed in order to be transferred to the other peer.

This query can be evaluated in two ways:

- either by creating the combined alignment that allows a peer to reach the more similar peers in the most efficient way. In the present case, this would amount to compute the $A_{1,2} \cdot A_{2,4} \cup A_{1,3} \cdot A_{3,4}$ alignment which is $\{\langle a_1, a_4, = \rangle, \langle b_1, b_4, = \rangle, \langle c_1, c_4, = \rangle, \langle d_1, d_4, = \rangle, \langle e_1, e_4, = \rangle\}$. This trivially allows the peer to transform the query and send it to o_4. This method can be used when all alignments are available, the cost of combining them is negligible with regard to the cost of communication and we can consider that the similar peers are indeed those who will provide the information.
- or by executing the paths, i.e., by filtering the queries against the relevant paths and sending these queries to the first step of the path. Here, the query will be split in two: b_1 which will be translated through $A_{1,2}$ and sent to o_2 and e_1 which will be translated through $A_{1,3}$ and sent to o_3. These peers in turn will select relevant peers with regard to the similarity (depending on the query) and forward the query. This solution is closer to classical peer-to-peer behaviour but instead of broadcasting queries blindly, it will follow the paths which maximise the similarity (and thus coverage and distinguishability).

These techniques can be different from the classic strategy of sending the queries to the connected peers which maximise the covering of the query. Indeed, this may be the combination of non maximal subqueries which covers best the query and this cannot be found with direct coverage alone.

4.5 Conclusion

We have designed similarity measures between ontologies in the alignment space which take into account the coverage and distinguishability of alignments and can account for combined alignment paths for transforming queries. This allows global reasoning on alignments alone which is something less easy in local environments like those considered in C-OWL [Bouquet *et al.*, 2004] where alignment composition is limited.

The proposed measure has been designed with simplifying hypotheses that will require further investigation in order to relax them. This mostly concerns taking into account different alignment relations and alignment confidence, in the style of [Euzenat, 2008], as well as allowing m:n alignments.

Chapter 5

The OntoSim Plug-in

Most of the similarities discussed before have been made available in a library called OntoSim. We describe it in this chapter.

5.1 General description

OntoSim is a library of similarities and distances that can be used by other tools. It is integrated within the NeOn toolkit as a plug-in for other tools to use it.

The OntoSim package is a Java API allowing for computing similarities between ontologies. It relies on the Alignment API for ontology loading so it is quite independent of the ontology API used (JENA or OWL API).

The version of OntoSim described here and used for the evaluations is 1.1. It is available at `http://ontosim.gforge.inria.fr/`.

It contains 9 packages:

`fr.inrialpes.exmo.ontosim` Base package containing alignment, ontology and vectorial space measures. It also contains the interface `Measure`.

`fr.inrialpes.exmo.ontosim.align` Concrete alignment spaces measures.

`fr.inrialpes.exmo.ontosim.entity` Entity measure used by ontology space measures. It also contain 2 sub packages: `model` for entities and `triplebased` for the triple-based measure.

`fr.inrialpes.exmo.ontosim.set` `SetMeasure` interface and concrete implementations.

`fr.inrialpes.exmo.ontosim.string` Similarities and distances between strings. It contains a wrapper class for using second string API.

`fr.inrialpes.exmo.ontosim.util` Utility classes.

- **`fr.inrialpes.exmo.ontosim.util.matrix`** Matrix classes for storing efficiently large sparse matrix of unknown size.
- **`fr.inrialpes.exmo.ontosim.util.measures`** Utility measures such as measure cache for storing the results of similarities which computation takes a long time, measures where values are loaded from file, etc.

`fr.inrialpes.exmo.ontosim.vector` : `VectorMeasure` and concrete implementations; it also contains a sub-package for vector space `model`.

5.2 Interface

The `Measure<O>` interface that each measure has to implement contains three methods :

`double getSim(O o1, O o2)` returns the similarity between `o1` and `o2`.

`double getDissim(O o1, O o2)` returns the dissimilarity between `o1` and `o2`.

`double getMeasureValue(O o1, O o2)` returns the raw value which could be a similarity, a dissimilarity, a distance, etc. according its type obtained by `getMType()`

These methods could also be implemented with regard to a set of concepts instead of the whole ontology.

5.3 Distances in the ontology space and vector space

Distances in the ontology space are available in the package `fr.inrialpes.exmo.ontosim` and respectively implemented as `OntologySpaceMeasure` and `VectorSpaceMeasure`.

5.3.1 class OntologySpaceMeasure

It implements `Measure<LoadedOntology<?> >`, i.e., a measure which can compare instances of `LoadedOntology`. It is composed of two sub-measures: a local measure between ontology entities and a collection measure or global measure which aggregates the local measure values. In OntoSim, the global measure contains the local measure. So, when one wants to instantiate an `OntologySpaceMeasure`, the constructor only takes in parameter the global measure. The global measure implements interface `SetMeasure` described Section 5.3.2. Local measures are described Section 5.3.3.

Constructor: `public OntologySpaceMeasure(SetMeasure<Entity<?> > globalMeasure);`

From an instance of `OntologySpaceMeasure`, the instances of global and local measures are given by:

```
public Measure<Entity<?> > getLocalMeasure();
```

```
public SetMeasure<Entity<?> > getGlobalMeasure();
```

5.3.2 interface SetMeasure and implementing classes

`SetMeasure` is the interface that a collection measure has to implement. A `SetMeasure<S>` allows to compute a similarity or distance value between two sets of objects having the type `S`. This is done by collecting and aggregating the similarity or distance values between each single pair of objects contained in the two sets. The constructor of a `SetMeasure<S>` takes in parameter a local measure:

```
public SetMeasure(Measure<S> m).
```

The instance of the local measure used by the `SetMeasure` is given by:

```
public Measure<S> getLocalMeasure().
```

OntoSim provides 5 concrete set measures, described Section 3.3:

```
AverageLinkage<S>
```

```
FullLinkage<S>
```

```
Hausdorff<S>
```

`MaxCoupling<O>` (Maximum Weight Maximum Graph Matching similarity)

`SingleLinkage<S>`

5.3.3 Entity measures

Local measures used by `OntologySpaceMeasure` implement `Measure<Entity<?> >` where `Entity<?>` is an interface enclosing an ontology class, property or instance. This interface and its implementation are in the package `fr.inrialpes.exmo.ontosim.entity.model`.

OntoSim provides 4 entity measures described Section 3.2:

`AlignmentEntitySim` entity measure based on the confidence value contained in an alignment.

`EntityLexicalMeasure` by default, it is based on a MaxCoupling similarity between set of String (annotations), which are individually compared by JaroWinkler string distance.

`OLAEntitySim`

`TripleBasedEntitySim`

The two first measures are independent of the ontology API used while the latter two respectively depends on OWL API 1 and JENA.

5.3.4 class VectorSpaceMeasure

As `OntologySpaceMeasure`, `VectorSpaceMeasure` implements `Measure<LoadedOntology<?> >`. This kind of measure is described Section 3.1.2.

The default constructor instantiates a measure based on cosine similarity using term frequency weights. There are also constructors taking an initial collection of ontologies, or/and the vector measure used, i.e., cosine or jaccard and the term weighting scheme, i.e., TF or TFIDF:

```
public VectorSpaceMeasure(Collection<LoadedOntology<?> > ontologies)

public VectorSpaceMeasure(VectorMeasure m, DocumentCollection.WEIGHT vectorType)

public VectorSpaceMeasure(Collection<LoadedOntology<?> > ontologies,VectorMeasure m,
     DocumentCollection.WEIGHT vectorType)
```

We can add ontology to the measure:

```
public final boolean addOntology(LoadedOntology<?> ontology)
```

5.3.5 Example

```
OntologyFactory of=OntologyFactory.getFactory();

LoadedOntology<?> o2 = of.loadOntology((new File(args[1])).toURI());
LoadedOntology<?> o1 = of.loadOntology((new File(args[0])).toURI());

Vector<LoadedOntology<?>> ontos = new Vector<LoadedOntology<?>>();
ontos.add(o1);
ontos.add(o2);

VectorSpaceMeasure m = new VectorSpaceMeasure(ontos,new CosineVM(),DocumentCollection.WEIGHT.TFIDF);
System.out.println(m.getSim(o1, o2));

SetMeasure<Entity<?>> gm = new MaxCoupling(new OLAEntitySim());
OntologySpaceMeasure m2 = new OntologySpaceMeasure(gm);
System.out.println(m2.getSim(o1, o2));
```

```
Measure<Entity<?>> lm =  new EntityLexicalMeasure(new StringMeasureSS(new Levenstein()));
SetMeasure<Entity<?>> gm2 = new Hausdorff<Entity<?>>(lm);
OntologySpaceMeasure m3 = new OntologySpaceMeasure(gm2);
System.out.println(m3.getSim(o1, o2));
```

5.4 Distances in the alignment space

These measures are available in the package `fr.inrialpes.exmo.ontosim.align`.

5.4.1 interface AlignmentSpaceMeasure

It implements `Measure` and is defined by:

`public void setAlignmentSpace( OntologyNetwork on )` sets the ontology network on which the measure will be computed to `on`.

`public boolean addAlignment( Alignment al ) throws AlignmentException` adds the alignment `al` to the considered ontology network.

It has a basic implementation which provides the functions for manipulating the `OntologyNetwork`.
This interface is implemented in three classes corresponding to the methods provided before.

5.4.2 class ASAlignmentPathMeasure

This is the measure described in Definition 24. It implements `AlignmentSpaceMeasure` as a distance and offers the same additional interface as `ASShortestPathMeasure`.

5.4.3 class ASShortestPathMeasure

This is the measure described in Definition 25. It implements `AlignmentSpaceMeasure` as a distance and, in addition, provides:

`public static enum NORM { diameter, cardinal }` two values used for normalising the result: diameter is the length of the largest shortest path between two nodes; cardinal is the number of ontologies in the network.

`public NORM getNormModality()` returns the curently set normalisation modality.

`public void setNormModality( NORM mod )` sets the normalisation modality to `mod`.

5.4.4 class ASLargestCoverageMeasure

This is the measure described in Definition 29. It implements `AlignmentSpaceMeasure` as a similarity.

5.4.5 class ASUnionPathCoverageMeasure

This is the measure described in Definition 31. It implements `AlignmentSpaceMeasure` as a similarity.

5.4.6 Example

```
Alignment al1 = new AlignmentParser( 0 ).parse( aurl1 );
Alignment al2 = new AlignmentParser( 0 ).parse( aurl2 );

OntologyNetwork noo = new BasicOntologyNetwork();
noo.addAlignment(al1);
noo.addOntology(uri1);
noo.addOntology(uri2);
noo.addOntology(uri3);
noo.addAlignment(al2);

Measure<Ontology> m = new ASLargestCoverageMeasure( noo );
System.out.println( m.getSim(o1, o2,));

ASShortestPathMeasure mm = new ASShortestPathMeasure( noo );
mm.setNormModality( NORM.diameter );
System.out.println( m.getSim(o1, o2,) );
```

5.5 Conclusion

OntoSim provides a convenient set of measures between ontologies and between concepts. It is now used by the Alignment API (part of the NeOn toolkit alignment plug-in).

Chapter 6

Comparison of presented measures

In order, to better understand how these measures behave, we have performed two batches of experiments. The first experiments compare measures from the ontology space on the OAEI benchmark ontologies (§6.1). It especially offers a separated evaluation of entity similarity measures and set similarity measures. The second experiment compares measures in the ontology space and measures in the alignment space on the OAEI conference test set (§6.3).

6.1 OAEI experiments

The presented measures have, to our knowledge, not been evaluated on ontology distances. We have emitted opinion of their relevance only grounded on their mathematical form. It is necessary to enhance this judgement through evaluation. We want to evaluate both the speed of distance computation and the accuracy with regard to asserted similarity.

The ideal experimental setting comprises a corpus of ontologies with clear expectations about the distances that should be found between them. We do not have such a corpus annotated with distances values between ontologies, however, the most important thing is to know the proximity order between ontologies.

Finding a relevant corpus is not an easy task. For this reason we provide only preliminary results here. We first describe the tested methods, the test set and various tests performed with this test set.

6.1.1 Selected measures

In order to be representative, we selected both terminological and structural measures usually used in ontology matching. For each kind of measures, we chose to evaluate basic measures and more elaborated ones as Table 6.1 shows. The JaccardVM(TF) measure is the simplest one since it only represents the proportion of shared terms in two ontologies. CosineVM has been used with two types of weights: TF and TF·IDF (these measures are presented in §3.1.2). All other measures are entity based measures and then, they have been tested with the three collection measures presented in §3.3: the Average Linkage, the Hausdorff distance, and the MWMGM distance. For normalisation purpose, all measures evaluated here are similarity measures.

For the triple-based measure, we instantiate this measure with JaroWinkler similarity as $simS$ and MWMGM

	basic	**elaborate**
terminological	CosineVM(TF & TFIDF), JaccardVM(TF)	EntityLexicalMeasure
structural	TripleBasedEntitySim	OLAEntitySim

Table 6.1: Selected measures.

similarity (presented Section 3.3) as Δ). We choose to stop iterations when the sum of difference in similarity between each iteration is under a threshold, i.e., $\sum_{n_1,n_2 \in Nodes^2} |simN_i(n_1,n_2) - simN_{i+1}(n_1,n_2)| \leq 1$.

6.1.2 The OAEI benchmark suite

We have considered the Ontology Alignment Evaluation Initiative[1] benchmark test set because it offers a set of ontologies that are systematically altered from one particular ontology which will play the role of o. There are here 6 categories of alterations:

Name Name of entities that can be replaced by (R/N) random strings, (S)ynonyms, Name with different (C)onventions, (F) strings in another language than English.

Comments Comments can be (N) suppressed or (F) translated in another language.

Specialisation hierarchy can be (N) suppressed, (E)xpansed or (F)lattened.

Instances can be (N) suppressed

Properties can be (N) suppressed or (R) having the restrictions on classes discarded.

Classes can be (E)xpanded, i.e., replaced by several classes or (F)lattened.

As these ontologies are generated by applying successive transformations to o, we know that the ontologies resulting from applying less transformations (for inclusion) should be closer to o. This is this property that we have exploited in this first test set.

Order between ontologies of benchmark

We can build a partial order relation $\leq$ representing the alteration relation over all generated ontologies. $o \leq o'$ means that the ontology o is an alteration of o' (o can be obtained by altering some features of o').

For each category of alteration $c \in \{Name, Comments, Specialisation, Instances, Properties, Classes\}$ and each ontology o, $c(o)$ represents the type of alteration made on the reference ontology for the category c. For each category, these alterations are ordered in the following way:

Name $\{R,N\} \leq \{S,C,F\} \leq \emptyset$	**Instance** $N \leq \emptyset$
Comments $N \leq F \leq \emptyset$	**Property** $N \leq R \leq \emptyset$
Specialisation hierarchy $\{N,E,F\} \leq \emptyset$	**Classes** $\{E,F\} \leq \emptyset$

From these rules, an ontology o is an alteration of o', noted $o \leq o'$, if for each category of alterations c, we have $c(o) \leq c(o')$. Figure 6.1 displays the transitive reduction of this partial order.

Initially, we were considering all triples of ontologies related by a transformation path; however, we observed that this procedure was biased towards lexical measures since it compares only labels and thus only changes its value between equal labels and non equal ones. Since our test is based on $\leq$, very often the distances are the same and thus the property was satisfied. Given the huge proportion of such tests in our test set we restricted ourselves comparing two ontologies with the initial ontology.

Another bias in this test set is that the transformations are of the all-or-nothing kind: either all labels are changed, or they are preserved. For countering this bias, we produced a larger altered test set in which the label scrambling transformation is applied in (always the same) 20, 40, 60 and 80% of the labels. The new lattice is given in Figure 6.2.

[1] `http://oaei.ontologymatching.org`

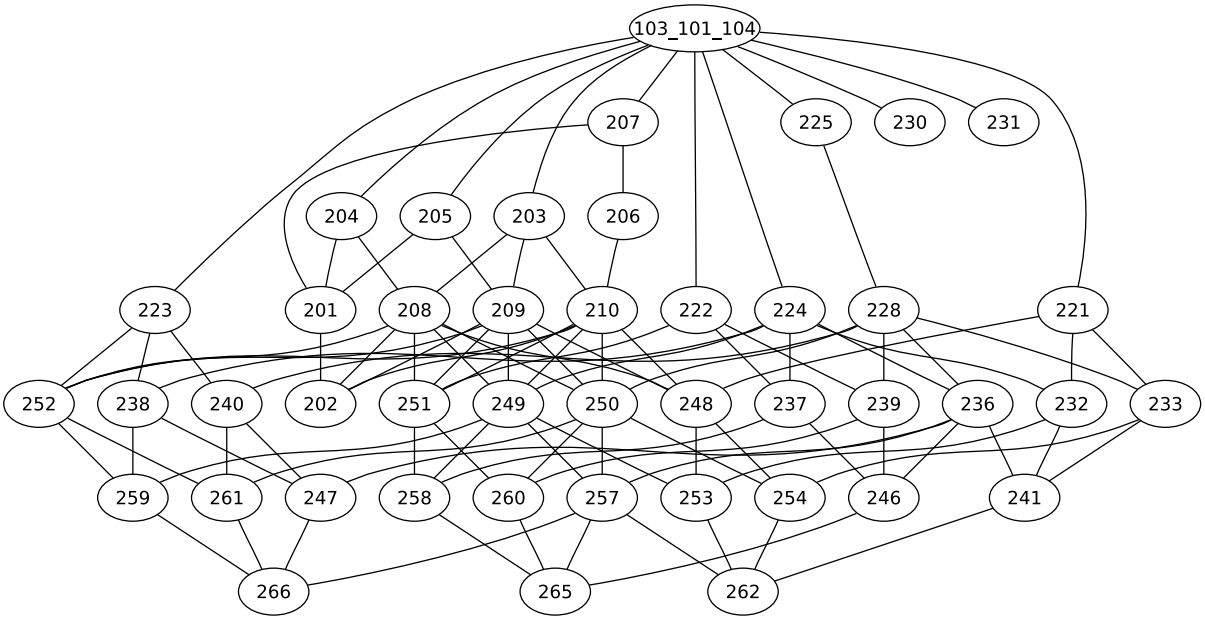


Figure 6.1: Order lattice on benchmark ontologies set.

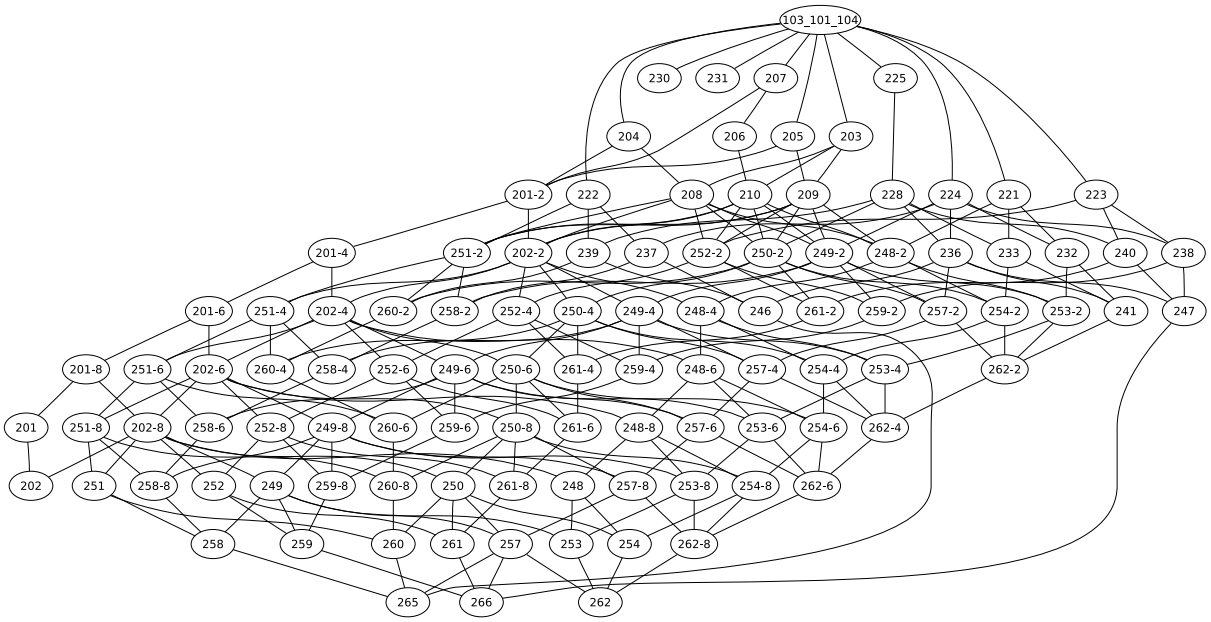


Figure 6.2: Order lattice on the enhanced benchmark ontologies set.

Measure	Tests Passed (ratio)	
	Original	Enhanced
MWMGM (EntityLexicalMeasure)	0.53	0.72
Hausdorff (EntityLexicalMeasure)	NaN	NaN
AverageLinkage (EntityLexicalMeasure)	0.44	0.31
MWMGM (OLAEntitySim)	0.75	0.78
Hausdorff (OLAEntitySim)	0.75	0.65
AverageLinkage (OLAEntitySim)	0.79	0.74
MWMGM (TripleBasedEntitySim)	0.86	0.92
Hausdorff (TripleBasedEntitySim)	0.86	0.89
AverageLinkage (TripleBasedEntitySim)	0.82	0.91
CosineVM (TF)	0.82	0.92
CosineVM (TFIDF)	0.57	0.81
JaccardVM (TF)	0.71	0.87

Table 6.2: Results on the original and enhanced benchmark test sets.

6.1.3 Results

According to the experimental settings introduced in the previous section, we have collected and analyzed 3 kinds of results. The first results concern the comparison of the orders induced by measures and those really observed on the benchmark suites. The second results show how measures behave when we introduce unrelated ontologies. The last results are about the time consumption of evaluated measures.

Order comparison on benchmark

This first experiment aims at checking if the tested similarities are compatible with the order induced by the alterations. It checks if the assertion $sim(o, o'') \leq sim(o, o')$ (or $\delta(o, o') \leq \delta(o, o'')$ for distances) is verified for any triple o, o', o'' such as $o'' < o' < o$. 1372 triples can be formed with the original benchmark suite and 15780 triples with the enhanced benchmark suite.

Table 6.2 shows results obtained respectively on the original benchmark and the enhanced one. This table presents, for each measure, the proportions of triples o, o', o'' (such as $o'' < o' < o$) verifying $sim(o, o'') \leq sim(o, o')$.

On the original benchmark, the triple-based similarity performs best with all structural measures. Cosine measure with TF weights also obtains good results and outperforms measures based on OLA similarity. Then, Jaccard similarity on TF weights gives satisfactory results. The measures based on lexical similarity and cosine with TF·IDF weights are the worst measures since they only passed successfully half of the tests. Concerning structural measures, MWMGM seems to be the best collection measure with the lexical and triple-based similarities. Hausdorff is only relevant with triple-based similarity because only 10% of tests have a reference similarity, $sim(o, o'')$, greater than 0 with OLA and none with lexical similarity (which explain the NaN value). Surprisingly, OLA similarity obtains its best result with AverageLinkage.

Measures tend to perform better on the enhanced benchmark than on the original benchmark. This improvement is especially noteworthy with lexical measures such as lexical entity, vector-based similarities. Triple-based similarity improves its results with all collection measures. OLA similarity almost obtain the same results. This owes to the fact that ontologies are, on average, more lexically similar in the enhanced benchmark. Results of MWMGM are better with all tested measures, but average linkage obtains worse results with lexical and OLA similarities. Results of Hausdorff are still not relevant with OLA and lexical similarities.

These results show that structural measures are more robust to the lexical alterations than lexical based measures. In cases where ontologies are lexically close, the use of vectorial measures seems to be relevant. The entity lexical measure is more sensitive to the collection measure used than structural measures. This can be explained by the fact that structural entity distances values are more dependent on each other than lexical entity distances values. The advantage of TF against TF.IDF is probably due to the nature of the benchmark: since the altered ontologies are generated from one reference, a term tends to appear in a lot of ontologies and then its IDF weight is around 0.

Tests with different cardinality matching

Finally, the benchmark test set is still biased towards 1-1 matching, so algorithms which are enforcing them (and in particular maximal matching algorithms) should be favoured by our tests. In order to counter this problem, we used two imperfect tests:

- we compared with similar but different ontologies: 301, 303, 304 to be compared with the higher level of the hierarchy where what has changed is added/suppressed classes (248, 251, 252, 221, 222, 223, 228, 250) and suppressed properties (228, 250);
- we compared with irrelevant ontologies (confious, iasted and paperdyne from the conference test; see §6.3).

The expected result here is that the slightly altered ontologies are closer than the 30x which are still better than the conference ontologies (2xx > 3xx > CONFERENCE).

Table 6.3 presents, for each measure, the ontologies which have not been correctly ordered and the observed order. For example, 250 < 304 means that the measure finds that 304 is closer to 101 than 250 ($sim(101, 250) < sim(101, 304)$). In this experiment MWMGM similarities perform best. Results given by the Hausdorff measure combined with lexical and OLA similarities are not relevant since a lot of values are equal to 0. Nevertheless, triple-based similarity with Hausdorff gives good results. Results obtained by average linkage with lexical and triple-based similarities are not satisfactory since a lot of ontologies are not correctly ordered. Vectorial measures fail on the same tests: they do not work very well when the names of classes have been removed (ontologies 248, 250, 251, 252).

In this experiment, ontologies having different cardinality do not penalise MWMGM in comparison with other measures. These results also show the limits of lexical measures on tests where structure is preserved but not the lexical data.

6.2 Conclusion on the benchmark experiment

This first experiment shows that basic measures such as Jaccard on shared terms sets, and cosine similarity, perform better than more advanced measures which have been successfully used in ontology matching. Concerning aggregation-based measures, the similarity using MWMGM seems to provide better results with all local measures tested than other aggregation schemes. However, it must be mentioned that this benchmark advantages such a measure because all entities in an ontology tend to have one (and only one) similar entity in the other ontologies.

Time consumption

We compared the CPU time used by each measure. The testing platform is powered by a quad-core 3GHz Xeon processor with a Linux OS. All evaluated measures (but OLA) have been implemented and evaluated using the same framework. For each measure, we computed all similarity values between ontologies of the original benchmark. This test was performed twice with no significant differences. These results are those of the second round. Table 6.4 shows the CPU time spent to compute these 1225 similarities and the average time spent to compute one similarity value.

Measure	Observed disorders
MWMGM(EntityLexicalMeasure)	250 < 304 250 < {CONFERENCE} {301, 303} < iasted < 304 < {confious ,paperdyne}
MWMGM(OLAEntitySim)	250 < 304 303 < {confious, paperdyne}
MWMGM(TripleBasedEntitySim)	250 < {301,303} < 228 < 304 250 < {CONFERENCE}
Hausdorff(EntityLexicalMeasure)	All values equals to 0
Hausdorff(OLAEntitySim)	{228, 248, 250, 251, 252} < 304 {228, 248, 250, 251, 252} < {CONFERENCE} {301, 302 }< iasted < 303 < {confious, paperdyne}
Hausdorff(TripleBasedEntitySim)	250 < {3xx} 250 < confious
AverageLinkage(EntityLexicalMeasure)	{2xx} < {301,304} {228, 248, 250, 251, 252} < {confious, iasted} < {221,222, 223} < paperdyne 303 < {CONFERENCE}
AverageLinkage(OLAEntitySim)	{223, 248, 250, 251, 252} <301 {303} < {confious, paperdyne}
AverageLinkage(TripleBasedEntitySim)	{ 248, 250, 251, 252} < 303 < {221, 222, 223, 228}< {301,304} 250 < {confious, iasted}
CosineVM(TF)	{248, 250, 251, 252}<{3xx}
CosineVM(TFIDF)	{248, 250, 251, 252}<{3xx}
JaccardVM(TF)	{248, 250, 251, 252}<{3xx}

Table 6.3: Ordering error between ontologies on a selection of ontologies.

Measure	Total time (s)	Average time per similarity value (s)
CosineVM (TF)	101	0.08
JaccardVM (TF)	101	0.08
CosineVM (TFIDF)	102	0.08
AverageLinkage (EntityLexicalMeasure)	444	0.36
Hausdorff (EntityLexicalMeasure)	451	0.37
MWMGM(EntityLexicalMeasure)	558	0.46
Hausdorff (TripleBasedEntitySim)	7 410	6.05
AverageLinkage (TripleBasedEntitySim)	7 671	6.26
MWMGM (TripleBasedEntitySim)	7 950	6.49
Hausdorff (OLAEntitySim)	38 912	31.76
AverageLinkage (OLAEntitySim)	38 995	31.83
MWMGM (OLAEntitySim)	39 074	31.9

Table 6.4: CPU time consumption on the original benchmark.

These results clearly show that measures based on OLA are runtime intensive. Measures using triple-based entity similarity are 5 times less expensive than the first family. Lexical Entity based similarities and those based on the vector space model measures are computed largely faster.

Globally, these results confirm that entity-based measures are more time intensive than vector space measures. Among the entity-based measures, structural measures (OLA and triple-based entity similarities) are more extensive than lexical ones since they rely on an iterative refinement process. The observed runtime is consistent with theoretical complexity of measures (and the computation of all but the more complex is deterministic), so we do not observe a significant effect from coding.

This experimentation shows that only lexical measures are usable on a large scale.

6.3 OntoFarm experiments

The previous experiments were run in the ontology space alone. In the OAEI benchmark data set, the alignment space is star-shaped: all ontologies are aligned only with ontology 101. The OntoFarm data set offers a more realistic set of ontologies, a subpart of which is given by hand-made reference alignments. This provides a more natural setting, but a less obvious reference for distances. We designed some experiments for evaluating the robustness of these distance measures anyway[2].

6.3.1 Dataset description

The OntoFarm dataset[3] [Šváb *et al.*, 2005] is made of a collection of 15 ontologies dealing with the conference organisation domain. Ontologies have been based upon three types of underlying resources:

- actual conference (series) and its web pages (indicated in column Type with mark Web),
- actual software tool for conference organisation support (indicated in column Type with mark Tool),
- experience of people with personal participation in organisation of actual conference (indicated in column Type with mark Insider).

The ontologies are described in Table 6.5. Their names are derived from the name of the conference or conference organisation tool. The DL expressivity was obtained from Pellet[4].

[2]These experiments are available at `http://nb.vse.cz/~svabo/NeOnExperimentD334/`
[3]`http://nb.vse.cz/~svatek/ontofarm.html`
[4]`http://clarkparsia.com/pellet/`

Name	Type	Number of classes	Number of Datatype Properties	Number of Object Properties	DL
Ekaw	Insider	77	0	33	SHIN
Sofsem	Insider	60	18	46	ALCHIF(D)
Sigkdd	Web	49	11	17	ALEI(D)
Iasted	Web	140	3	38	ALCIN(D)
Micro	Web	32	9	17	ALCOIN(D)
Confious	Tool	57	5	52	SHIN(D)
Pcs	Tool	23	14	24	ALCIF(D)
OpenConf	Tool	62	21	24	ALCOI(D)
ConfTool	Tool	38	23	13	SIN(D)
Crs	Tool	14	2	15	ALCIF(D)
Cmt	Tool	36	10	49	ALCIN(D)
Cocus	Tool	55	0	35	ALCIF
Paperdyne	Tool	47	21	61	ALCHIN(D)
Edas	Tool	104	20	30	ALCOIN(D)
MyReview	Tool	39	17	49	ALCOIN(D)

Table 6.5: The ontologies of the OntoFarm data set

This dataset has been used several times in the OAEI evaluation campaigns. We have used those of 2009[5]. It also comes[6] with reference alignments between five of these ontologies: Cmt, Ekaw, Sigkdd, Iasted and ConfTool.

These reference alignments contain correspondences with only equivalence relations and with confidence 1.0.

6.3.2 Measures agreement

This first experiment aims at comparing correlation between measures. The main idea is to observe how the measures in the alignment space are similar to those in the ontology space (and specially the better ones).

For each ontology of OntoFarm, we have retained the closest ontologies according each measure. Then these sets of closest ontologies are compared to each other according to a degree of agreement.

Let $\langle \Omega, \Lambda \rangle$ be an alignement space. Given an ontology $o \in \Omega$, the set of its closest ontologies in Ω according to distance measure m is defined as:

$$C(o, \langle \Omega, \Lambda \rangle, m) = \underset{o' \in \Omega - \{o\}}{\arg\min}\, m(o, o')$$

The degree of agreement is obtained by comparing the sets of closest ontologies given by two different measures. There are several strategies for doing that. We could get the number of equivalent sets and divide it by the number of ontologies in Ω. This first strategy is too strict when sets slightly differ. We then choose to compare sets according to the proportion of ontologies they share. From this, the degree of agreement between two distances m and m' is defined as:

$$A(m, m', \langle \Omega, \Lambda \rangle) = \frac{1}{|\Omega|} \sum_{o \in \Omega} \frac{|C(o, \Omega, m) \cap C(o, \Omega, m')|}{|C(o, \Omega, m) \cup C(o, \Omega, m')|}$$

[5] http://nb.vse.cz/~svabo/oaei2009/data/conference.zip
[6] http://oaei.ontologymatching.org/2008/results/conference/reference-alignment.zip

	1	2	3	4	5	6	7	8
1	-	1.0	0.08	0.08	0.08	0.08	0.08	0.08
2	-	-	0.08	0.08	0.08	0.08	0.08	0.08
3	-	-	-	0.91	0.41	0.47	0.0	0.14
4	-	-	-	-	0.34	0.41	0.0	0.14
5	-	-	-	-	-	0.74	0.07	0.27
6	-	-	-	-	-	-	0.07	0.21
7	-	-	-	-	-	-	-	0.34
8	-	-	-	-	-	-	-	-

Table 6.6: Agreement matrix for agreement experiment, Jaccard measure used.

Results of measure agreement

From this similarity matrix, we find clusters of similar measures by using hierarchical clustering[7]. The resulting dendrogram is given Figure 6.3.

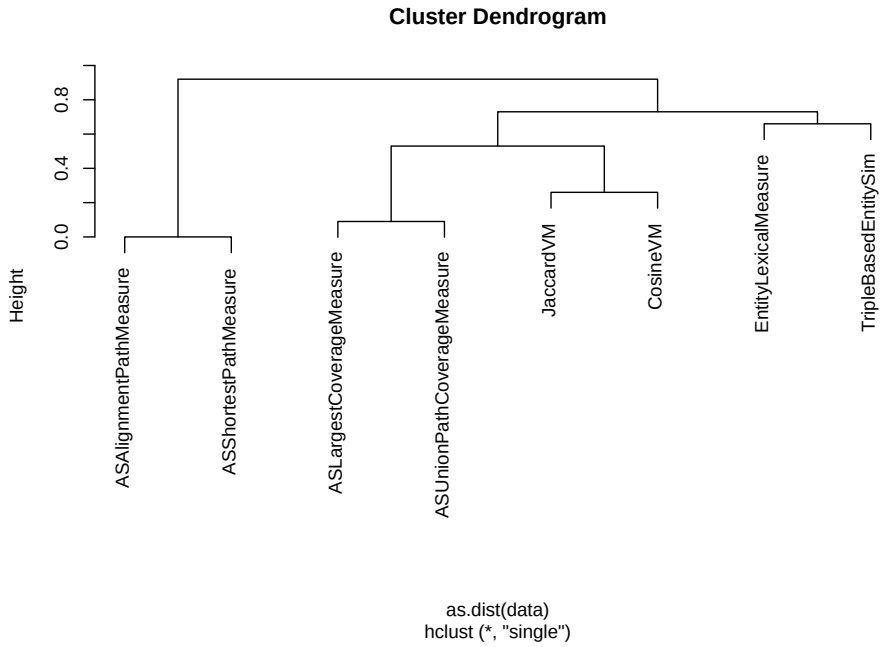


Figure 6.3: Cluster dendrogram for measures from alignment and ontology space measures.

According to the similarity matrix of Table 6.6 and the dendrogram of Figure 6.3, we obtain 3 clusters of measures:

1. ASAlignmentPathMeasure (1), ASShortestPathMeasure (2)
2. ASLargestCoverageMeasure (3), ASUnionPathCoverageMeasure (4), JaccardVM-TF·IDF (5), and CosineVM-TF·IDF (6)
3. EntityLexicalMeasure-MWMGM (7), TripleBasedEntitySim-MWMGM (8)

The two groups of alignment space measures (path-based and coverage-based) strongly agree inside both groups, but the two groups are not strongly correlated.

The two path measures, i.e., ASAlignmentPathMeasure and ASShortestPathMeasure, do not agree with other measures. This can be easily explained by the fact that, in this test, either the two ontologies are in the

[7]For this clustering, we use single linkage metric, but the other metrics give the same results

core set of ontologies with reference alignments or they are not. Only this will characterise their similarity. On one hand, this shows that these measures are very dependent on the topology of the alignment space. On the other hand, these measures are indeed quite naive.

The two other alignment space measures, i.e., ASLargestCoverageMeasure, ASUnionPathCoverageMeasure, are far more correlated with the content based measures since they are closer to them than to the path-based measures. They are even more correlated to the vector-space measures than the vector space measures agree with the entity-based measures. This is a very interesting result showing that indeed, these measures which do not have access to the content of ontologies are meaningful with regard to this content.

If we conclude from the results of the experiment on benchmark (§6.1.3), that the triple-based entity measure is a better measure than vector-based measures, then this result is less interesting. However, coverage-based measures are still closer to the triple-based measure than path based measure.

6.3.3 Robustness

This experiment has been focused on robustness of measures from Alignment Space. It has been done by degrading alignment space in a basic manner. We only retain two variants for this degradation:

variant 1: Randomly remove $n\%$ of correspondences in an alignment space

variant 2: Randomly remove $n\%$ of alignments in an alignment space

Basically, we expect that the degradation obtained with the second variant will have more negative impact on the robustness of measures.

The experiment here consisted in evaluating, for each measure, the degree of agreement between the sets of closest ontologies obtained with the ontology space without degradation and the sets obtained with an altered ontology space. This experiment has been done with several levels of degradation, from 10% to 100% with a step of 10%. Since this procedure is based on random degradation, we repeat it 10 times for each level.

The robustness of a measure m according a degraded alignment space $\langle \Omega, \Lambda' \rangle$ obtained from $\langle \Omega, \Lambda \rangle$ is given by:

$$A(m, \langle \Omega, \Lambda \rangle, \langle \Omega, \Lambda' \rangle) = \frac{1}{|\Omega|} \sum_{o \in \Omega} \frac{|C(o, \langle \Omega, \Lambda \rangle, m) \cap C(o, \langle \Omega, \Lambda' \rangle, m)|}{|C(o\langle \Omega, \Lambda \rangle, m) \cup C(o, \langle \Omega, \Lambda' \rangle, m)|}$$

For the first variant, we only compare the two coverage measures because this type of degradation has no impact on path measures since it preserves connectivity.

Results of variant 1

Results of this first variant are given in Figure 6.4. Results of ASLargestCoverageMeasure (degree of agreement with non-degraded variant) are higher, therefore we can conclude it is less dependent on particular correspondences. For LargestCoverage, the degree of similarity in the case of 50% degradation is almost the same as for 20% degradation (and we run the experiments several times with the same results). This seems strange and may be an artefact of the protocol because at each step (each n), the weakened network is generated from scratch.

However, in both cases, the measures are rather robust since they have their agreement decreasing less than the degradation. The measures are thus less dependent on correspondences.

Results of variant 2

Results of this second variant are given in Figure 6.5. We can see that all measures have very similar linearly decreasing curves with the LargestCoverage slightly less regular. This result shows the strong dependency of all these measures on available alignments.

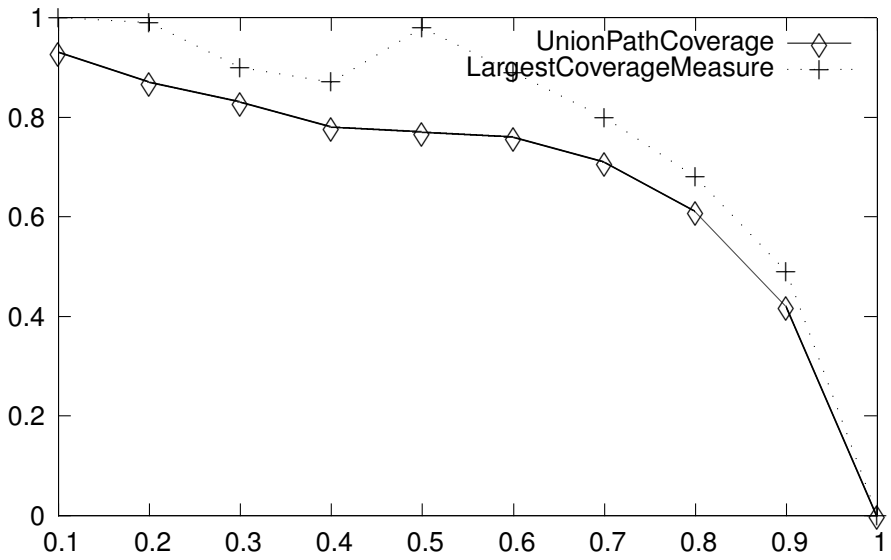


Figure 6.4: Robustness of measures in function of the degree of degradation, variant 1.

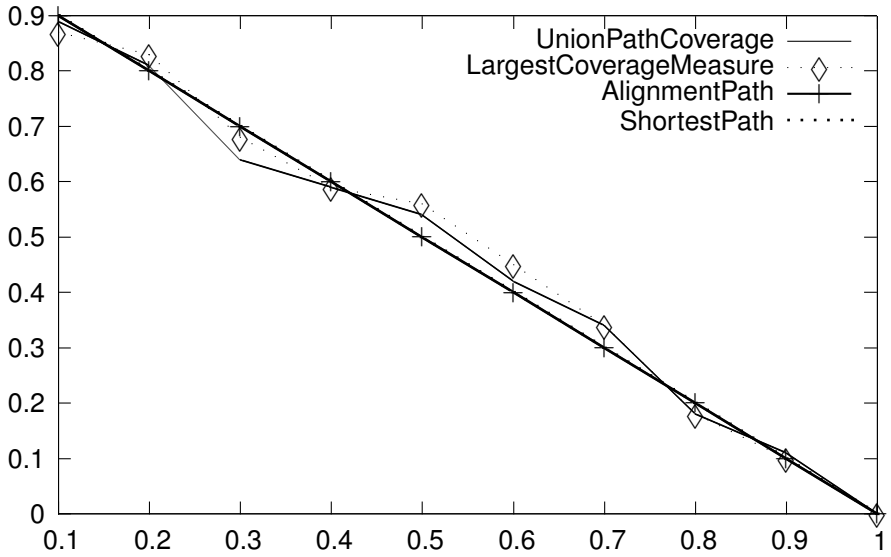


Figure 6.5: Robustness of measures in function of the degree of degradation, variant 2.

Conclusion on robustness

The robustness tests show that the alignment space measures are indeed correlated with the quality of the alignment space (so they are not random measures).

These measures are more robust to deletion of correspondences than to the deletion of alignments. The robustness of measures decreases linearly with the number of alignments deleted, but when only correspondences are deleted, the decrease is above the linear curve. This means that these measures can resist to some loss of quality in data.

6.3.4 Conclusion on the OntoFarm dataset

These experiments show promising results for coverage-based measures in terms of agreement with content-based measures and robustness.

Although these experiments show some interesting trends, they could be improved in several ways. First, it would be more appropriate to not only compare closest ontologies but to take the whole rankings into account by using some rank correlation coefficient. In fact, such correlation indexes should be more robust than those that have been used here and furthermore they would give more fine-grained results.

The robustness experiments could also be completed by proposing other degradation variants: Instead of deletion of alignments and correspondences, one could imagine to study the effects of noise in the alignment space. It could be interesting to also compare results obtained with measures by using alignment space provided by different matchers and confront them with quality measures used in ontology alignment evaluation.

6.4 Conclusion

Due to lack of common data involving “real” distance data in a corpus of aligned ontologies, it is difficult to obtain conclusive results comparing measures in the ontology space and in the alignment space. The agreement test shows that there is a continuity between these measures.

Further experimentation is required, but the measures which have been found to perform best during these tests are not always the same. Attention must therefore be paid to select measures of OntoSim depending on the expected application.

Chapter 7

Conclusion

There exist many similarity or dissimilarity measures between ontologies using different aspects of ontologies. We have sketched a wide panorama of possible measures which have been classified with regard to the space to which they apply (ontology space or alignment space) and the kind of resources (labels, concepts, triples, axioms) and techniques (vector-space, key concepts, entailment, path or coverage) they use. This generated many different measures satisfying different properties

In order to assess the behaviour of these methods, we have evaluated them on specific test benches. However, the lack of compelling test data prevents us from drawing definitive conclusions about these measures. Results show that structural similarities tend to be more reliable and robust than lexical similarities. This is especially true when the ontologies to compare do not share a lot of common vocabulary. Nevertheless, due to their complexity, structural measures are not adapted for real-time applications or for measuring similarities between large ontologies. Meanwhile, some basic measures, such as cosine on TF vectors, quickly provide quite accurate results. Such measures can be useful for quickly selecting a subset of close ontologies and thus allowing the use of structural measures in order to refine the proximity relation between the selected ontologies. Hence, more work must be developed for finding trade-offs between accuracy and efficiency.

However, it is clear that there is no measure covering all needs: the notion of distance between ontologies depends on the criterion for defining the distance (interoperability, equivalence, domain consistency).

Finally, we introduced OntoSim, a library of measures which contains all measures presented here. It is available as a stand alone library as well as a plug-in for the NeOn toolkit and can be used by various other plug-ins (this is the case for the alignment plug-in and soon for the Watson plug-in).

Having a common interface such as that provided by OntoSim, allows for the development and testing of such measures in applications without committing to any one particular measure.

Bibliography

[Alani and Brewster, 2005] Harith Alani and Christopher Brewster. Ontology ranking based on the analysis of concept structures. In *Proc. 3rd International conference on Knowledge Capture (K-Cap), Banff (CA)*, pages 51–58, 2005.

[Bouquet *et al.*, 2004] Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen, Luciano Serafini, and Heiner Stuckenschmidt. Contextualizing ontologies. *Journal of Web Semantics*, 1(1):325–343, 2004.

[d'Aquin *et al.*, 2007] Mathieu d'Aquin, Claudio Baldassarre, Laurian Gridinoc, Sofia Angeletou, Marta Sabou, and Enrico Motta. Watson: a gateway for next generation semantic web applications. In *Proc. Poster session of the International Semantic Web Conference (ISWC), Busan (KR)*, 2007.

[David and Euzenat, 2008] Jérôme David and Jérôme Euzenat. Comparison between ontology distances (preliminary results). In *Proc. 7th conference on international semantic web conference (ISWC), Karlsruhe (DE)*, pages 245–260, 2008.

[Ehrig *et al.*, 2005] Marc Ehrig, Peter Haase, Mark Hefke, and Nenad Stojanovic. Similarity for ontologies – a comprehensive framework. In *Proc. 13th European Conference on Information Systems, Information Systems in a Rapidly Changing Economy (ECIS), Regensburg (DE)*, 2005.

[Euzenat and Shvaiko, 2007] Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer, Heidelberg (DE), 2007.

[Euzenat and Valtchev, 2004] Jérôme Euzenat and Petko Valtchev. Similarity-based ontology alignment in OWL-lite. In *Proc. 16th European Conference on Artificial Intelligence (ECAI)*, pages 333–337, Valencia (ES), 2004.

[Euzenat, 2007] Jérôme Euzenat. Semantic precision and recall for ontology alignment evaluation. In *Proc. 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 348–353, Hyderabad (IN), 2007.

[Euzenat, 2008] Jérôme Euzenat. Algebras of ontology alignment relations. In *Proc. 7th conference on international semantic web conference (ISWC), Karlsruhe (DE)*, volume 5318 of *Lecture notes in computer science*, pages 387–402, 2008.

[Gracia *et al.*, 2007] Jorge Gracia, Vanessa Lopez, Mathieu d'Aquin, Marta Sabou, Enrico Motta, and Eduardo Mena. Solving semantic ambiguity to improve semantic web based ontology matching. In *Proc. 2nd ISWC Ontology matching workshop (OM), Busan (KR)*, pages 1–12, 2007.

[Hu *et al.*, 2006] Bo Hu, Yannis Kalfoglou, Harith Alani, David Dupplaw, Paul Lewis, and Nigel Shadbolt. Semantic metrics. In *Proc. 15th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*, volume 4248 of *Lecture notes in computer science*, pages 166–181, Praha (CZ), 2006.

[Huang and Stuckenschmidt., 2005] Zhisheng Huang and Heiner Stuckenschmidt. Reasoning with multi-version ontologies: a temporal logic approach. *Proc. 4th International Semantic Web Conference (ISWC2005), Galway (IE)*, pages 62–76, 2005.

[Jung and Euzenat, 2007] Jason Jung and Jérôme Euzenat. Towards semantic social networks. In *Proc. 4th European Semantic Web Conference, Innsbruck (AT)*, volume 4519 of *Lecture Notes in Computer Science*, pages 267–280, 2007.

[Jung *et al.*, 2007] Jason Jung, Antoine Zimmermann, and Jérôme Euzenat. Concept-based query transformation based on semantic centrality in semantic peer-to-peer environment. In *Proc. Advances in Data and Web Management, Joint 9th Asia-Pacific Web Conference (APWeb) and 8th International Conference, on Web-Age Information Management (WAIM), Huang Shan(CN)*, volume 4505 of *Lecture Notes in Computer Science*, pages 622–629, 2007.

[Mädche and Staab, 2002] Alexander Mädche and Steffen Staab. Measuring similarity between ontologies. In *Proc. 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*, volume 2473 of *Lecture notes in computer science*, pages 251–263, Siguenza (ES), 2002.

[Peroni *et al.*, 2008] Silvio Peroni, Enrico Motta, and Mathieu d'Aquin. Identifying key concepts in an ontology, through the integration of cognitive principles with statistical and topological measures. *Journal of Web Semantics*, ??:242–256, 2008.

[Robertson and Spärck Jones, 1976] Stephen Robertson and Karen Spärck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, 1976.

[Stuckenschmidt and Klein, 2004] Heiner Stuckenschmidt and Michel Klein. Structure-based partitioning of large concept hierarchies. In *Proc. 3rd International Semantic Web Conference (ISWC), Hiroshima (JP)*, volume 3298 of *Lecture Notes in Computer Science*, pages 289–303. Springer, 2004.

[Tverski, 1977] Amos Tverski. Features of similarity. *Psychological Review*, 84(2):327–352, 1977.

[Valtchev, 1999] Petko Valtchev. *Construction automatique de taxonomies pour l'aide à la représentation de connaissances par objets*. Thèse d'informatique, Université Grenoble 1, Grenoble (FR), 1999.

[Vrandečić and Sure, 2007] Denny Vrandečić and York Sure. How to design better ontology metrics. In *Proc. 4th European Semantic Web Conference, Innsbruck (AT)*, volume 4519 of *Lecture Notes in Computer Science*, pages 311–325, 2007.

[Šváb *et al.*, 2005] Ondřej Šváb, Vojtěch Svátek, Petr Berka, Dušan Rak, and Petr Tomášek. Ontofarm: Towards an experimental collection of parallel ontologies. In *Proc. 4th ISWC poster session, Galway (IE)*, 2005.

[Zimmermann, 2008] Antoine Zimmermann. *Sémantique des réseaux de connaissances: gestion de l'hétérogénéité fondée sur le principe de médiation*. Thèse d'informatique, Université Joseph Fourier, Grenoble (FR), 2008.

Index