HAL open science

Comparison between ontology distances (preliminary results)

Jérôme David, Jérôme Euzenat

To cite this version:

Jérôme David, Jérôme Euzenat. Comparison between ontology distances (preliminary results). Proc. 7th international semantic web conference (ISWC), Oct 2008, Karlsruhe, Germany. pp.245-260, 10.1007/978-3-540-88564-1_16 . hal-00793534

HAL Id: hal-00793534

https://inria.hal.science/hal-00793534v1

Submitted on 22 Feb 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Comparison between ontology distances (preliminary results)

Jérôme David and Jérôme Euzenat

INRIA Grenoble Rhône-Alpes & LIG
Grenoble, France
Jerome.{David|Euzenat}@inrialpes.fr

Abstract. There are many reasons for measuring a distance between ontologies. In particular, it is useful to know quickly if two ontologies are close or remote before deciding to match them. To that extent, a distance between ontologies must be quickly computable. We present constraints applying to such measures and several possible ontology distances. Then we evaluate experimentally some of them in order to assess their accuracy and speed.

1 Motivations

The semantic web aims at exploiting formal knowledge at the world scale. It is, in particular, based on ontologies: a structure defining concepts used to represent knowledge and their relationships. These concepts are used for specifying semantic web services, annotating web resources (pictures, web pages, music) or for describing data flows.

It is however likely that different information sources will use different ontologies. It is thus necessary to find correspondences between ontologies in order to communicate from one ontology to another. Finding correspondences is called matching ontologies and the resulting set of correspondences is called an alignment [Euzenat and Shvaiko, 2007].

Together with matching ontologies, there are many occasions where it is useful to know if two ontologies are close to each others or not, or what is the closest ontology to another one. In particular,

- when one wants to find the community of people with whom she will be more likely to communicate easily, finding if they use similar ontologies can be useful information [Jung and Euzenat, 2007]; This can also help identifying 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 other ones. 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;

– 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 be separated as they are distant to each others;
– 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. 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 the closest ontology.

In these various applications, there are different requirements for an ontology distance measure. In particular, 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.

The remainder of this paper is as follows: we first present and discuss previous work. Then, after recalling general definitions about distance measures, we introduce constraints applying to distances between ontologies. The next section introduces some ontology distances. Finally, we evaluate these measures and with regard to the criteria.

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.

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 a quite 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 §4.3 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 of 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. This is what we attempt to do here.

3 Distances properties

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

3.1 Ontology model

All measure will be based on a set of ontologies O which we refer to as the ontology space. For simplification purposes, an OWL ontology $o \in O$ is represented as a set of named entities E_o. These entities can be classes (C), properties (P) or individuals (I): $E = C \cup P \cup I$. Each entity is identified by a URI thanks to the function $uri : E \longrightarrow URI$. The function $l_{ln} : E \longrightarrow String$ returns the local name of the entity which is the specific part of the entity URI in the ontology. Each entity can be also described by annotations, i.e., labels, comments. The function $l_{annot} : E \longrightarrow \mathcal{P}(String)$ assigns a set of annotations to each entity.

3.2 Algebraic distance properties

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

Definition 1 (Dissimilarity). *Given a set O of ontologies, a dissimilarity $\delta : O \times O \rightarrow \mathbb{R}$ is a function from a pair of ontologies to a real number such that:*

$$\forall o, o' \in O, \delta(o, o') \geq 0 \qquad \textit{(non-negativeness)}$$
$$\forall o \in O, \delta(o, o) = 0 \qquad \textit{(minimality)}$$
$$\forall o, o' \in O, \delta(o, o') = \delta(o', o) \qquad \textit{(symmetry)}$$

Some authors consider a 'non symmetric (dis)similarity', [Tverski, 1977]; we then use the term non symmetric measure or pre-similarity. There are more constraining notions of dissimilarity, such as distances and ultrametrics.

Definition 2 (Distance). *A distance (or metric) $\delta : O \times O \rightarrow \mathbb{R}$ is a dissimilarity function satisfying the definiteness and triangular inequality:*

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

There are in fact 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.

We will see below that there are good reasons to avoid symmetry as well.

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 3 (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.

3.3 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 closest are corresponding entities;
- the easier (the faster) it is to answer queries;
- ...

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

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

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

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

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

4 Ontology distances

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 be computed faster than the actual alignment.

There are many possible ways to define a distance between ontologies. First of all, an ontology can be just viewed as a bag of terms. This approach is similar to those used in information retrieval based on the vector space model. These techniques relies on vector representations of ontologies and use distances 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 the ontologies will very often rely on:

- a distance (δ) or similarity (sim) measure between entities;
- a collection distance (Δ) which will use the distance between entities for computing a distance between ontologies.

We first present ontology distances based on the vector space model. Then, we consider some distances between entities before presenting various kinds of collection distances.

4.1 Ontology distances based on the vector space model

A distance can be computed by comparing the sets 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. This distance would certainly run faster than any serious matching algorithm but does not tell a lot about the matching process. However, more elaborate measures based on the vector space model (VSM) have been designed.

In the VSM, each ontology is represented by a vector of terms. 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}: $T_o = \bigcup_{e \in E_o} l_{te}\big(l_{annot}(e)\big) \cup l_{ln}(e)$.

Let O 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 $\overrightarrow{D_O} = (w_1, ..., w_n)$ 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] : $w_i = TF(t_i, o).\ln \frac{|O|}{|\{o|t_i \in l_{te}(o\}|}$

Then for comparing ontologies, it is possible to apply various similarity measures:

- Jaccard index with boolean weights;

– cosine index with frequency weights;
– cosine index with TF·IDF weights.

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

4.2 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 sort of 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.

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:

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

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

Structural distances There has 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].

OLA similarity One good candidate as structural similarity is the those defined 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 designate them, i.e., URIs, labels, names, etc.,
– the similarity of the pairs of neighbor 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 datavalue 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 $Sim_X : X^2 \rightarrow [0,1]$ 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 normalized, 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 1 illustrates the set of similarities used by OLA.

Funct.	Node	Factor	Measure
Sim_O	$o \in O$	$\lambda(o)$	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
		$o \in O, (a,o) \in \mathcal{U}$	$MSim_O$
		$v \in V, (a,v) \in \mathcal{U}$	$MSim_V$
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
		$c \in C, (r, \mathtt{domain}, c) \in \mathcal{R}$	$MSim_C$
		$c \in C, (r, \mathtt{range}, c) \in \mathcal{R}$	$MSim_C$
		$d \in D, (r, \mathtt{range}, d) \in \mathcal{R}$	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
		$c \in C, (p, \mathtt{all}, c) \in \mathcal{R}$	$MSim_C$
		$n \in \{0,1,\infty\}, (p, \mathtt{card}, n) \in \mathcal{R}$	equality

Table 1. Similarity function decomposition (`card = cardinality` and `all = allValuesFrom`).

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. Since this similarity is already computed as an optimization problem, it generates a match of maximal weight.

Triple-based iterative similarity measure We have also defined a new measure based on RDF triple similarity. This similarity is defined as a convergent sequence. Initially, some similarity values between nodes in the RDF graph are initialized 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,
- 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 JaroWinkler or Levenstein.

Then, 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:

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

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 4.3. $simT_i$ is a similarity between two triples defined as the average values of similarities between nodes of two triples at stage i:

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

In the evaluations, we instantiate this measure with JaroWinkler similarity as $simS$ and MWMGM similarity (presented Section 4.3) as Δ. We choose to stop iterations when $\sum_{n_1, n_2 \in Nodes^2} |simN_i(n_1, n_2) - simN_{i+1}(n_1, n_2)| \leq 1$.

4.3 Collection distances

Once one has a distance δ (or similarity sim) among concepts 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 4 (Average linkage). *Given a set of entities E and a dissimilarity function $\delta : E \times E \rightarrow [0\ 1]$, the average linkage measure between two ontologies is a dissimilarity function $\Delta_{alo} : 2^E \times 2^E \rightarrow [0\ 1]$ such that $\forall o, o' \subseteq E$,*

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

Definition 5 (Hausdorff distance). *Given a set of entities E and a dissimilarity function $\delta : E \times E \rightarrow [0\ 1]$, the Hausdorff distance between two sets is a dissimilarity function $\Delta_{Hausdorff} : 2^E \times 2^E \rightarrow [0\ 1]$ 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 in function of the distance between one pair of members of the sets. The average linkage, on the other hand, has its value function of the distance between all the possible comparisons. None of these are satisfactory. 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. The quality of such a measure is thus that closeness 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 6 (Minimum weight maximum graph matching distance). *Given a set of entities E and a dissimilarity function $\delta : E \times E \rightarrow [0\ 1]$, 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:

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

Computing the minimum weight maximum graph matching distance (MWMGM) from a similarity, involves two related steps: extracting an alignment between the ontology 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.

5 Experimental setting

The presented measures have to our knowledge, not been evaluated on ontology distances. We have emitted opinion on 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.

5.1 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 shows Table 2. 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. All other measures are entity based measures and then, they have been tested with the three collection measures presented Section 4.3: the Average Linkage, the Hausdorff distance, and the MWMGM distance. For normalization purpose, all measures evaluated here are similarity measures.

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

Table 2. Selected measures.

5.2 Evaluation on 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.
Specialization 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)latened.

Since, 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.

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

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

For each category of alteration $c \in \{Name, Comments, Specialization, Instances, Properties, Classes\}$ 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** $N \leq R \leq \emptyset$
Specialization 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 1 displays a transitive reduction of this partial order.

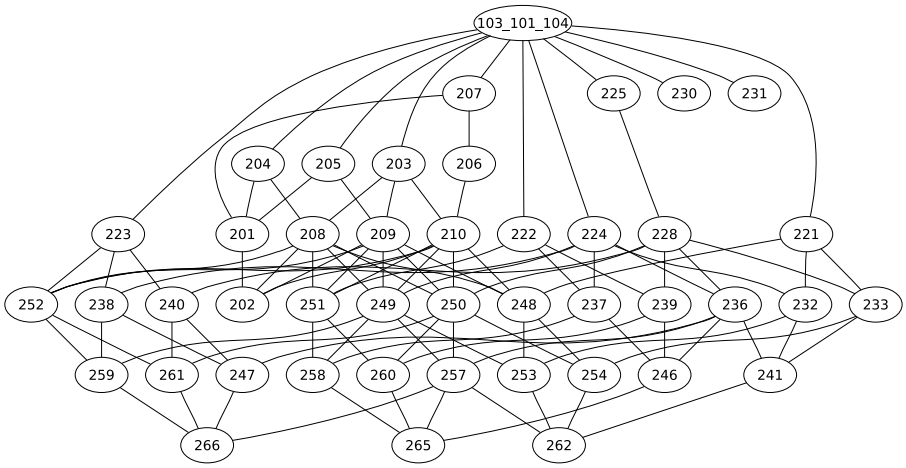

Fig. 1. Order lattice on benchmark ontologies set.

Initially we were considering all triples of ontologies related by a transformation path. But 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 given by this test set is that the transformations were 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 2.

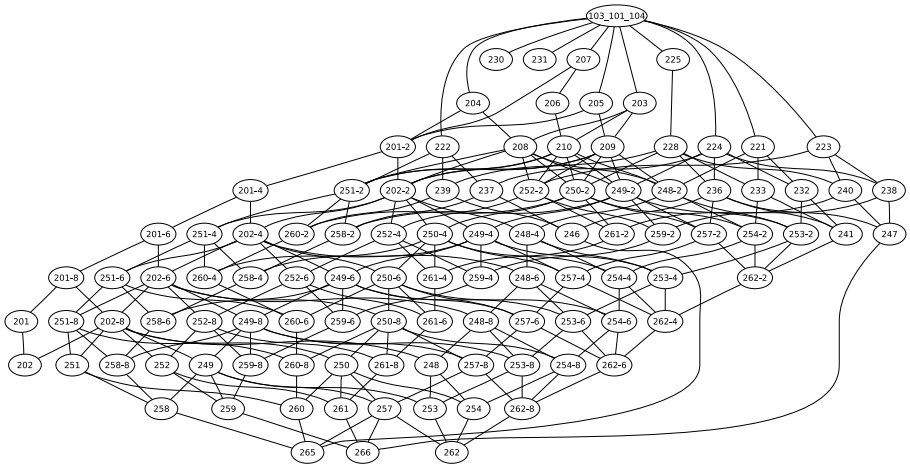

Fig. 2. Order lattice on the enhanced benchmark ontologies set.

6 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.

6.1 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 3 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 the 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 successfully passed only the 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, $sim(o, o'')$,

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 3. Results on the original and enhanced benchmark test sets.

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 it 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 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.

6.2 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 favored 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).

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 4 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 the best. Results given by Hausdorff measure combined with lexical and OLA similarities are not relevant since a lot of values are equals 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 penalize 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.

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 4. Ordering error between ontologies on a selection of ontologies.

6.3 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 two times with no significant differences in taken times. These results are those of the second round. Table 5 shows the CPU time spent to compute these 1225 similarities and the average time spent to compute one similarity value.

These results clearly shows 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 VSM 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 useable at large scale.

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

Table 5. CPU time consumption on the original benchmark.

7 Conclusion

Measuring distances between ontologies can be useful in various tasks for different purposes (finding an ontology to replace another, finding an ontology in which queries can be translated, finding people using similar ontologies). Hence there is no universal criterion for deciding if an ontology is close or far from another.

There exist many measures using different aspects of ontologies. In order to know the behaviour of these methods, we have evaluated them on specific test benches. Results shows that structural similarities tends to 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 their complexity, structural measures are not adapted for real-time applications or for measuring similarities between large ontologies. We can also notice that some basic measures such cosine on TF vector give quite accurate results. Such kind of measures can be useful for quickly select 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.

This paper only restricts the study to measures used for matching ontologies. It could be interesting to test others measures relying on some global ontology characteristics (size, graphs densities, etc.).

References

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.

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.

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.

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

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.

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.

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.

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.

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.

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.

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

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.

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

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.

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.