How to Design Better Ontology Metrics

Denny Vrandečić and York Sure

Institut AIFB, Universität Karlsruhe (TH), Germany
{vrandecic,sure}@aifb.uni-karlsruhe.de

Abstract. You can only control what you can measure. Measuring ontologies is necessary to evaluate ontologies both during engineering and application. Metrics allow the fast and simple assessment of an ontology and also to track their subsequent evolution. In the last few years, a growing number of ontology metrics and measures have been suggested and defined. But many of them suffer from a recurring set of problems, most importantly they do not take the semantics of the ontology language properly into account. The work presented here is a principal approach to facilitate the creation of ontology metrics with the clear goal to go beyond structural metrics to proper semantic-aware ontology metrics. We have developed guidelines and a set of methodological tools based on the notions of "normalization" and "stable metrics" for creating ontology metrics. These guidelines allow the metric author to decide which properties metrics need to fulfil and to appropriately design the desired metric. A discussion of an exemplary metric (taken from literature) illustrates and motivates the issues and suggested solutions.

1 Introduction

Did you ever dare to raise the issue of ontology quality assurance? How did you control the process of improvement? As in many other related fields, you can only control what you can measure [4]. Measuring ontologies is necessary to evaluate ontologies both during engineering and application and is a necessary precondition to perform quality assurance and control the process of improvement. Metrics allow the fast and simple assessment of an ontology and also to track their subsequent evolution. In the last years, many ontology metrics and measures have been suggested and some principal work has been done to study the nature of metrics and measures for ontologies in general. We are extending this work.

There is a recurring set of problems with existing ontology metrics and measures, whereby we focus on the W3C standardized ontology language OWL [12]. We argue that most metrics are based on *structural notions* without taking into account the *semantics* which leads to incomparable measurement results. First, most ontology metrics are defined over the RDF graph that represents an OWL DL ontology and thus are basically graph metrics which take only structural notions into account. Second, only a very small number of metrics is taking the semantics of OWL DL into account (subsumption etc.). Third, almost no metric is taking the open world assumption into account. We believe that foundational work addressing these issues will substantially facilitate the definition of proper ontology metrics in the future.

In this paper we will study these issues, describe how they can be avoided, and under what circumstances they have to be avoided, and under which they are acceptable, will outline the foundations for a novel set of metrics and measures, and discuss the advantages and problems of the given solutions. Our approach is based on two notions, first "normalization" of an ontology, and second "stable metric".

Normalization consists of the five steps (i) name anonymous classes, (ii) name anonymous individuals, (iii) materialize the subsumption hierarchy and unify names, (iv) propagate instances to deepest possible class or property within the hierarchy, and (v) normalize property instances. We argue that such a normalization is useful as a kind of pre-processing in order to apply known structural metrics *in a semantics-aware way*. For instance, a known structural metric is the depth of of the class-hierarchy. However, the current measures of ontology depth depend on a number of structural parameters such as whether subsumption reasoning has been performed and whether the results have been materialized before measurement. Performing the normalization steps before measuring ensures that the value for the maximum depth of an ontology is comparable to the maximum depth of another ontology.

Stable metrics are metrics that take the open world assumption properly into account, that means that they are stable with regards to possible additions of further axioms to the ontology. Stable metrics allow us to make statements about the behaviour of an ontology in the context of a dynamic and changing world wide web, where ontologies may frequently be merged together in order to answer questions over integrated knowledge. We give an exemplary extension of the depth metric towards a stable metric in order to demonstrate how a classic metric can be turned into a stable one.

In this paper we assume the term to include both axioms and facts (as well as annotations and ontology properties, although those are not taken into regard for normalization), i.e. the TBox and the ABox. Here, ontology does not mean only the axioms (as it is assumed in many other works), but also a knowledge base, and any of them could be empty. Thus we follow the definition of ontology in the OWL standard [12].

The paper is structured as follows. In Section 2 we will examine existing metrics and measures, and thus survey related work. Section 3 contrasts the underlying notions of semantic metrics with structural metrics, and discusses which scenario will require what kind of metric. Section 4 introduces the notion of "normalization" of an ontology which forms the heart of our approach. In Section 5 we illustrate the practical application of normalization on examples. Section 6 addresses the issue of stable metrics with regards to the open world assumption. We conclude in Section 7, where we also discuss future work.

2 Current metrics and measures

In this paper we will concentrate on some foundational aspects that form the base for automatically acquirable measures. Therefore we will not define a number of metrics and measures, but rather take a step back and discuss conditions that measures have to adhere to in order to be regarded as semantically aware ontology metrics. This also helps to understand clearly what it means for a metric to remain on a a structural level.

Thus the scope of this work compares best to other metric frameworks, like the QOOD (quality oriented ontology description) framework [7] and the O^2 and *oQual* models [6]. The authors created semiotic models for ontology evaluation and validation, and thus describe how measures should be built in order to actually assess quality. They also describe the relation between the ontology description, the ontology graph, and the conceptualization that is expressed within the graph, and they define measures for the structural, functional, and usability dimension. In [5] they introduce further measures that can be applied within that framework. We will take one of the measures introduced in [5] as an example in Section 5, and show some shortcomings of the actual descriptions of such a measure (not of the framework as a whole!). We think that the work described here fits well into the QOOD framework by making the assumptions underlying such measures explicit.

A framework for metrics in the wider area of ontology engineering is provided by OntoMetric [11]. The authors name and sort a long list of metrics into several different areas, like tools, languages, methodologies, costs, and content. They define the relations between the different metrics, their attributes, and the quality attributes they capture. Within the OntoMetric framework, the work presented in this paper is based solely in the area of content metrics. It extends the discussions around content metrics, and elaborates properties of such metrics in more detail. Whereas OntoMetric regards all kind of metrics, we gear the results described here towards automatically measurable metrics.

OntoQA is a tool that implements a number of metrics [14], and thus it allows for the automatic measurement of ontologies. They define metrics like richness, population, or cohesion. Whereas all these metrics are interesting, they fail to define if they are structurally or semantically defined – which is a common lapse. Most of the metrics in OntoQA actually can be applied both before and after normalization (as described in the following section). We suppose that comparing these two measures will yield further interesting results.

Often metrics are defined purely structural. An example is given by [1], where the authors describe metrics for ranking ontologies, like the class match or the density measure. Interestingly even the so called semantic similarity measure is not a semantic measure in the sense described here, since they apply all these measures on the graph that describes the ontology, not on the ontological model.

OntoClean [9], currently the most well-known ontology evaluation approach, is a philosophically inspired approach for the evaluation of formal properties of a taxonomy. Some tools offer support for the manual tagging with OntoClean properties (OntoEdit [13] and WebODE [2]), a recent work deals with the automation of OntoClean [15]. From a practical perspective OntoClean provides means to derive measurable mismatches of a taxonomy with respect to an ideal structure which takes into account the semantics of the "is-a" relationship. Such mismatches have a structural nature, e.g. one is able to derive that a certain concept should not be the subconcept of another concept. OntoClean provides an explanation of why mismatches occur which subsequently might help to improve the taxonomical structure. For many people the philosophical notions of OntoClean are subject of long discussions, however, strictly speaking, this is not part of the evaluation but of the ontology engineering because deciding the proper

nature of a class forces the ontology to commit itself to a more specified meaning, which in turn allows for a more object evaluation technique.

Measures applied to ontologies from the Semantic Web are usually still in a very simple state [17] (unsurprising due to the overall bad quality of ontologies in the wild, and the high costs on resources for providing reasoning on a big number of ontologies). We think that the most prevalent hurdle towards applying more semantic measures on ontologies on the web is an actual lack of some foundational work towards defining such measures, and a subsequent lack of implementation. The work presented here is a step towards such an implementation, that will allow to measure the web in several new dimensions.

3 Ontological metrics

As shown in the previous section, current metrics often measure structural properties of the ontology. In the case of OWL DL, this often means that they measure the structure of the RDF graph that describes the ontology with well-known graph measures. Another approach is to measure the explicitly stated facts and axioms. Within these paper, we regard both approaches as structural. Structural metrics are often useful, and this paper does not suggest to replace them. It rather offers a way to extend the possibilities available to the ontology engineer with truly ontological metrics.

We define ontological, or semantic, metrics to be those who do not measure the structure of the ontology, but rather the models that are described by that structure. In a naïve way, we could state that we base our metrics not on the explicit statements, but on every statement that is entailed by the ontology.

But measuring the entailments is much harder than measuring the structure, and we definitively need a reasoner to do that. We also need to make a difference between a statement X that is entailed by an ontology O to be true ($O \models X$), a statement that is not entailed by an ontology ($O \not\models X$), and a statement that is entailed not to be true ($O \models \neg X$). To properly regard this difference leads us to so called stable metrics that can deal with the open world assumption of OWL DL. We will return to them in Section 6.

Note that measuring the entailments is more an intuitive description of how to describe ontological metrics than the actual approach. In many cases – for example for a measure that simply counts the number of statements in an ontology – measuring all entailed statements instead of measuring all explicit statements often leads to an infinite number of statements. Just to give one example, the ontology $\exists R.\top \sqsubseteq C$ also entails the statements $\exists R.\exists R.\top \sqsubseteq C$, $\exists R.\exists R.\exists R.\top \sqsubseteq C$, and so on, an endless chain of existentials. But only terminating measures are of practical interest, and thus we need approaches that allow us to capture ontological metrics in a terminating way.

In order to gain the advantage of the simple and cheap measurement of structural features, we can transform the structure of the ontology. These transformation need to preserve the semantics of the ontology, that is, they need to describe the same models. But they also need to make certain semantic features of the ontology explicit in their structure – thus we can take structural measures of the transformed ontology and interpret them as ontological measures of the original ontology. We call this kind of

transformations normalization. The following section describes five steps of normalization.

With these tools we will be enabled to define ontological metrics in a simpler and less error prone way than in current practice. We will show this on an exemplary metric in Section 5.

4 Normalization of an ontology

This section describes several steps of normalization. Their goal is to explicate some features of the semantics of an ontology within its structure, so that the structural metrics actually capture the semantics they are supposed to capture.

The following normalization steps are defined here:

1. name all relevant classes, so no anonymous complex class descriptions are left
2. name anonymous individuals
3. materialize the subsumption hierarchy and normalize names
4. instantiate the deepest possible class or property
5. normalize property instances

Notice that if we speak of names, we mean, in the context of OWL DL, the URI of the class, property, or individual, not the human readable label.

In the **first normalization** our aim is to get rid of anonymous complex class descriptions. After the first normalization, the TBox will contain two kind of axioms: class definitions of the form $A \equiv C$, where A is a class name and C a class description (or class name), and subsumption axioms of the form $A \sqsubseteq B$, where both A and B are class names. The ABox will consist of property instantiations of the form $R(i, j)$, and of facts of the form $A(i)$, with A being a class name.

The first normalization can be done as follows:

1. in all axioms of the form $C \sqsubseteq D$ where C (or D) is a complex class description, add a new axiom $A \equiv C$ ($B \equiv D$) with A (B) being a new class name. Replace the axiom $C \sqsubseteq D$ with $A \sqsubseteq D$ ($C \sqsubseteq B$, or even $A \sqsubseteq B$)
2. in all axioms of the form $C \equiv D$ where both C and D are complex class descriptions, replace that axiom with the two axioms $A \equiv C$ and $A \equiv D$, with A being a new class name
3. in all axioms of the form $C \equiv A$ where C is a complex class descriptions and A an atomic class name, replace that axiom with $A \equiv C$
4. in all axioms of the form $C(i)$ where C is a complex class description, replace that axiom with the axioms $A(i)$ and $A \equiv C$ with A being a new class name

None of these structural changes change the possible models, that means, that they are semantically equivalent. They do introduce new class names to the ontology, which may not be desirable in all cases (for example for presentation purposes, for counting the classes, and so on).

Note that it is possible to introduce named classes that are unsatisfiable. This does not mean that the ontology becomes unsatisfiable, but solely these newly introduced

classes. Instead of introducing new names for unsatisfiable classes though, we could simply use the name $\bot$.

The **second normalization** gets rid of anonymous individuals. This means that every blank node that is of the (asserted or inferred) type individual needs to be replaced with an URI reference. Especially in FOAF [3] files this occurs regularly since, for some time, it was regarded as good practice *not* to define URIs for persons. Integration of data was not done via the URI, but with inverse functional properties. This practice is problematic, since the semantics of blank nodes in RDF are rather often not fully understood, and should thus be avoided. The second normalization as defined here captures the semantics most users wanted to express anyway.

It is possible that these newly introduced individual names give a further name to already existing (or other newly introduced) individuals. But since OWL DL does not adhere to a unique name assumptions, this is no problem. Furthermore, the next step of normalization will take care to resolve such synonyms.

The **third normalization** will materialize the subsumption hierarchy and normalize the names. The first step requires a reasoner.

1. for all pairs of simple class names (A, B) in the ontology, add the axiom $A \sqsubseteq B$ if the ontology entails that axiom (that is, materialize all subsumptions between simple named classes).
2. detect all cycles in the subsumption structure. For each set of classes $A_1 \ldots A_n$ that participate in a cycle, remove all subsumption axioms from the ontology where both classes are members of this set. In subsumption axioms where only one class is a member of this set, replace the class with B in the axioms. Add the axioms $B \equiv A_1 \ldots B \equiv A_n$ to the ontology. B is a new class name for each cycle. If B is unsatisfiable, take $\bot$ instead of B. If B is equal to $\top$, take $\top$.
3. regarding solely the subontology H_3 that consists of all subsumption axioms of an ontology O, remove all redundant ones (that is, remove all subsumption axioms that are redundant due to the transitivity of the subsumption relation alone).

The subsumption structure now forms a directed acyclic graph that represents the complete subsumption hierarchy of the original ontology. We define a set of normal classes of an ontology as follows: every class that participates in an subsumption axiom after the third normalization of an ontology is a normal class of that ontology.

Since we got rid of facts with complex class descriptions in the first normalization, we do not need a reasoner in order to take care of fact normalization. We still have to replace every class name that is not normal with its normal equivalent within the facts.

Note that instead of creating a new class name for each detected cycle, often it will make more sense to choose a name from the set of classes involved in that cycle, based on some criteria (like the class name belonging to a certain namespace, the popularity of the class name on the web, etc.). For many ontology metrics, this does not make any difference, so we disregard it for now, but we expect the normalizations to have beneficial effects in other scenarios as well, in which case some steps of the normalization need to be revisited in more detail. We will further discuss this in Section 7.

Since in OWL DL it is not possible to make complex property descriptions besides inverse properties, property subsumption, and transitivity, (extensions towards enabling

more complex property descriptions are suggested in the OWL 1.1 proposal [8]) no heavy reasoning is involved for property normalization in most cases. In case a property has more than one name, we choose one (or introduce a new name and state the equality). All normal property names have to be stated explicitly to be equivalent to all other property names they are equal to (that is, we materialize the equality relations between the normal property names and the non-normal ones). All occurrences of non-normal property names (besides within the axiom stating equality with the normal property name, and besides within annotation property instances) are replaced with the normal property name.

The same holds true for individuals. In case an individual has more than one name, we decide on or introduce a normal one and state explicitly equality to the normal name, and then replace all occurrences of the non-normal individual names with the normal one (besides within the axiom stating equality with the normal individual name, and besides within annotation property instances).

We disregard annotation property instances since they may be used to state annotations about the URI, and not about the actual concept, property, or individual. There could be annotations that describe when a certain URI was introduced, who created it, its deprecation state, or that point to a discussion related to the introduction of the URI. Some annotations on the other hand may be useful for the normal name as well – especially labels, or sometimes comments. Since annotations do not have impact on the DL semantics of the ontology anyway, they may be dropped for the purpose of measuring semantic metrics. Nevertheless, if the normalization is done for some other purpose, and it is planned to further use the normalized version of the ontology in some scenario, than the possible replacement of names within annotation property instances depends both on the scenario and the instantiated annotation property (for example, it may be useful to normalize the label when the ontology will be displayed on the user interface, but it may be bad to normalize versioning information that is captured within annotations).

The **fourth normalization** aims towards moving the instantiations to the deepest possible level, as this conveys the most information explicitly (and deriving instantiations of higher levels is very cheap because of the asserted explicitness of the hierarchy due to third normalization). This does not mean that every instance will belong to only one class, multiple instantiations will still be necessary in general.

Here is a possible (though not efficient) algorithm to perform the fourth normalization of an ontology O.

1. for each normal class C and each normal individual i in O, add $C(i)$ to O if it is entailed by the ontology.
2. for each normal object property instance $R(i, j)$ and each object property S so that $S \sqsubseteq R$ is an explicit axiom in O, add $S(i, j)$ if it is entailed by the ontology. Check this also for the property instances added this way (this step will terminate since the subsumption hierarchy is finite).
3. for each normal data property instance $T(i, d)$ and each data property U, proceed as in the previous step.

4. create a subontology H_4 out of O including only the facts (that is, the ABox), and the explicitly stated subsumption hierarchy of the classes and properties (after third normalization)
5. remove all facts from O that are redundant in H_4

We do not want to remove all redundant facts from the ontology at this step, since there may be some facts that are redundant due to an interplay of different other axioms in the TBox. For example, in the following ontology:

$Person(Adam)$.
$likes(Adam, Eve)$.
$Person \sqsubseteq \exists likes.\top$

the first statement is actually redundant, but would not be removed by the above algorithm (the third statement states that the domain of *likes* is *Person*). This is because we only remove axioms that are redundant within the subontology H_4, and the axiom stating the domain of *Person* would not be part of it. This is due to the fact that after first normalization, the ontology would look like this:

$Person(Adam)$.
$likes(Adam, Eve)$.
$Person \sqsubseteq A$
$A \equiv \exists likes.\top$

So H_4 would not include the last axiom, and thus the first axiom would not be redundant within H_4.

The **fifth normalization** finally normalizes the properties: we materialize property instances of symmetric and inverse properties, and we clean the transitivity relationship. This can be done similar to the creation of the subsumption hierarchy in the third normalization: after materializing all property instances, we remove all that are redundant in the subontology H_5, which contains only the property instances of all transitive properties, and the axioms stating the transitivity of these properties.

It is important to mention that normalization does not lead to a canonic normalized version. This means that there may be many different ontologies that result from the normalization of an ontology. Often normalizations do not result in canonical, unique results (think about conjunctive normal forms). The normalization as described here can be extended in order to result in canonic normalized forms, but the benefit of such an extension is not clear. Considering that common serializations, like the RDF/XML serialization of OWL ontologies [12], lack a canonic translation anyway, and thus ontologies cannot be compared on a character by character base, for example as some version control systems like CVS or SVN would require.

Also, normalization is not an applicable solution for every metric. For example, if we want to know the number of atomic classes in an ontology, first normalizing it and then calculating the number actually will return the wrong result in the general case. The goal of normalization is to actually provide the metric designer some tools in order to simplify the description of his metric. In the following section we describe an example of how to apply the normalization for the description of a metric.

5 Examples of normalization

The metric we will regard in this example is the *depth of the ontology*. What we want to measure is intuitively described as the length of the subsumption hierarchy, or else the number of levels the class hierarchy has. In [5], this is the measure (M3), called *Maximal depth*, and the definition is given as follows:

$$m = N_{j \in P}$$

$$\forall i \exists j (N_{j \in P} \geq N_{i \in P})$$

where $N_{j \in P}$ is the set of all nodes in the path j from the set of all paths through the digraph g that represents the ontology, that is, the definition is the length of the longest succession of explicitly stated subsumption relations.

Let us regard the following ontology:

$C \equiv \geq 1R.\top$
$D \equiv \geq 2R.\top$
$E \equiv \geq 3R.\top$

By the definition of (M3), the depth of the ontology is 1 (since there are no explicitly stated subsumption axioms, every path has one node). But after normalization the ontology gets transformed to this:

$C \equiv \geq 1R.\top$
$D \equiv \geq 2R.\top$
$E \equiv \geq 3R.\top$
$D \sqsubseteq C$
$E \sqsubseteq D$

Now the very same metric, applied to the normalized ontology, actually captures the intuition of the depth of the ontology and returns 3.

As discussed earlier, this example also shows us that some metrics will not work with normalization. In [5], metric (M30) is the *axiom/class ratio*. On the original ontology it is 1, but raises to 5/3 in the normalized version. In case the original ontology is being distributed and shared, (M30) – if stated as metadata of the ontology, for example in some kind of ontology repository [10] – should be 1, and not calculated on the normalized version.

Let us regard another example. In the following ontology

$D \sqsubseteq C$
$E \sqsubseteq D$
$D \sqsubseteq E$
$F \sqsubseteq E$

(M3) will be ∞ due to the subsumption cycle between D and E. The cycle can be resolved by rewriting the axioms in the following way:

$D \sqsubseteq C$
$D \equiv E$
$F \sqsubseteq E$

But due to the definition, (M3) would yield 2 here – there are two explicit subsumption paths, C, D and E, F, both having two nodes, and thus the longest path is 2. The structural measure again does not bring the expected result. After normalization, though, the ontology will look like this:

$A \sqsubseteq C$
$A \equiv D$
$A \equiv E$
$F \sqsubseteq A$

We have introduced a new class name A that replaces the members of the cycle, D, E. Now the depth of the ontology is 3, as we would have expected from the start, since the cycle is treated appropriately.

Existing structural metrics, as discussed in Section 2, often fail to capture what they are meant for. Normalization is a tool that is easy to apply and that can easily repair a number of such metrics. Even seemingly simple metrics, as demonstrated here with the ontology depth, are defined in a way that makes too many assumption with regards to the structure of the measured ontologies.

As we can see in this section, simple structural measures on the ontology do yield values, and often these values may be highly interesting. If we know that (M3) resolves to ∞, then this tells us that we have a cycle in the subsumption hierarchy. Also a high number of classes and complex axioms, but a low (M3) may indicate an expensive to reason about ontology, since the major part of the taxonomy seems to be implicitly stated (but such claims need to be evaluated appropriately). But both results do not capture what the measure was meant to express, that is, the depth of the class hierarchy.

But this leads us to the possibility of creating measures by combining structural metrics on the original ontology and on its normalized version, for example to calculate ratios like $M_3(O)/M_3(N(O))$ (with $M_3(O)$ returning measure (M3) as described above, and $N(O)$ being a function that returns the normalized version of the ontology O). This could describe the *explicitness of the subsumption hierarchy*. Further work needs to investigate and evaluate such measures, and to assess their usefulness for evaluating ontologies.

6 Stability of metrics

Often metrics intend to capture features of the ontology that are independent of the actual representation of the ontology. But as we have seen, structural transformations of the ontology description often lead to differences in the metrics even though the semantics remained untouched. Normalization offers a way to overcome these problems in many cases.

One aspect of metrics, that are not touched upon by normalization, is the issue of how stable the metrics are with regards to the open world assumption of OWL DL

ontologies. In order to illustrate this issue let's take a look at a simple example. Imagine an ontology with the following three facts:

$author(paper, York)$.
$author(paper, Denny)$.
$author(paper, Zdenko)$.

Now let us ask the simple question: how many authors does the $paper$ have? It seems that the answer should be 3. But now, if you knew that $Zdenko$ is just another name for $Denny$, and thus state $Zdenko \approx Denny$, then you suddenly would change your answer to 2, or even, becoming more careful, giving an answer like "I am not sure, it is either 1 or 2". So finally we can state that $York \not\approx Denny$ and thus arrive at the answer that the paper indeed has 2 authors (and even that is possibly wrong if we consider that we could add statements any time in an open world that add further authors to the paper – all we know *as of now* is that the paper has *at least* two authors).

When creating a metric, we have to ask ourselves the following, similar question: how does the metric behave when additions to the ontology happen? Since ontologies are meant to be smushed and integrated constantly and dynamically, can we predict how certain properties of the ontology will behave, that is, if $M(O_1)$ and $M(O_2)$ for a metric M and two ontologies O_1 and O_2 are known, what can we state about $M(O_1 \cup O_2)$? Or even, can we give a function f_M so that $f_M(M(O_1), M(O_2)) = M(O_1 \cup O_2)$ without having to calculate $M(O_1 \cup O_2)$ (which may be much more expensive)?

In the previous section we have discussed the simple example of ontology depth. Let us return to this example again. We define the function $M_3(O)$ that returns the measure (M3) as described in [5], and already described above. If we have an ontology O_1:

$D \sqsubseteq C$
$E \sqsubseteq D$

And a second ontology O_2:

$C \sqsubseteq D$
$E \sqsubseteq D$

In this case, $M_3(O_1) = 3, M_3(O_2) = 2$. We would expect $M_3(O_1 \cup O_2)$ to be 3, since M(3) is defined as the maximal depth, but since the union of both ontologies actually creates a cycle in the subsumption hierarchy, (M3) is ∞ – or, after normalization, just 2, and thus even smaller than the maximal depth before the union.

We can avoid such behaviour of the metrics by carefully taking the open world assumption into account when defining the metric. But this leads us to three possibilities for defining metrics,

1. to base the value on the ontology as it is,
2. to measure an upper bound, or
3. to measure a lower bound.

We need a more complicated example to fully demonstrate these metrics:

$C \equiv D \sqcup E$
$D \sqcap E \sqsubseteq \bot$

$F \sqsubseteq E$
$G \equiv \neg C$
$H \sqsubseteq C$
$F(i)$.
$D(j)$.
$G(k)$.

This ontology says that D and E form a complete partition of C (the first two axioms), that E has the subclass F, that there are elements that are not in C, and it states the existence of three individuals, i, j and k, and the classes they belong to.

The normalized version of this ontology looks like this (shortened slightly for readability):

$C \equiv D \sqcup E$
$\bot \equiv D \sqcap E$
$D \sqsubseteq C$
$E \sqsubseteq C$
$F \sqsubseteq E$
$G \equiv \neg C$
$H \sqsubseteq C$
$F(i)$.
$D(j)$.
$G(k)$.

(M3) of this ontology is 3 (C, E, F). But besides the actual depth, we can also calculate the minimal depth of this ontology, that is, no matter what axioms are added, what is the smallest number of levels the ontology will have (under the condition that the ontology remains satisfiable)?

In the given example, if we add the axiom $F \equiv E$, (M3) will decrease to 2. But on the other hand, no matter what axiom we further add, there is no way to let C collapse with D and E, therefore C is a proper superset of both (that is, it contains more individuals than D or E alone). And because C cannot become $\top$ (due to k being outside of C), the minimum depth of the ontology is 2.

The maximum depth of an ontology is usually ∞ (since we can always add axioms about an arbitrarily long class hierarchy). Only in the case of an ontology with a closed domain, that is, if we have an axiom like $\top \equiv \{a, b, c\}$, then the maximum depth is set (to $|\top| - 1$, since there may be a class C with one element, a class D with two elements that subsumes C, and then $\top$ with three elements, but since $\top$ is not counted, the longest path would be (C, D), and every further class in this path would become equivalent to an already existing class or be empty). But we expect such axioms to usually appear only in theoretical musings and hardly be of any practical relevance.

Therefore we need to define a maximum depth in a slightly different way in order to be of practical value. In the following, we will discuss two possible definitions.

Instead of allowing for arbitrary axioms that may be added, we only allow to add axioms of the form $A \sqsubseteq B$ with A and B being normal class names of the normalized ontology. In the above example, we may add the axiom $H \sqsubseteq F$ to the ontology in order to increase (M3) from 3 to 4. No longer subsumption path is possible, since all

the other named classes would become unsatisfiable when added to an existing path. So this metric will provide with a maximum depth of the ontology, assuming no new class names are added.

Another possibility to constrain the axioms to be added, is to allow only for axioms that do not relate to the existing ontology, that is, the intersection of the signatures of the two ontologies is empty. The signature of an ontology is the set of all names used in the ontology (besides the names from the OWL, RDF, RDFS, and XSD namespaces). In this case, (M3) of the merged ontology is the maximal (M3) of the single ontologies, since no interaction between the axioms happen that may increase or reduce (M3). We can thus define $f_{M_3}(M_3(O_1), M_3(O_2)) = \mathsf{max}(M_3(O_1), M_3(O_2))$, which is much cheaper to calculate than $M_3(O_1 \cup O_2)$.

Stable metrics are metrics that take the open world assumption into account. Stable metrics will help us to evaluate ontologies for the wide wild web. Since we expect ontologies to be merged on the web dynamically, stable metrics allow us to state conditions that the ontology will fulfil in any situation. The depth of an ontology may be a too simple example to demonstrate the advantages of stable metrics, but imagine a dynamic, ontology-based graphical user interface. Having certain guarantees with regards to the future development of the properties of the ontology may help the designer of the user interface tremendously, even if it is such a seemingly trivial statement like "the depth of the ontology is never less than 3".

There is no simple recipe to follow in order to turn a metric into a stable metric, but the question outlined at the beginning of this section, and then discussed throughout the rest – how does the ontology behave when axioms are added? – can be used as a guideline in achieving a stable metric.

We expect that the ready availability of metrics that take the open world assumption into account will lead to more robust ontologies. Since ontology engineers will have these numbers available at engineering and maintenance time, they will learn easier how to achieve their actual goals. For example, ontology engineers that want to create a class hierarchy that will not collapse to less levels can always check if the minimum depth as described above corresponds to the asserted depth. Tools could guide the ontology engineer towards achieving such goals. Ontology engineers get more aware of such problems, and at the same time get tools to measure, and thus potentially control them.

7 Conclusion and Future Work

We have discussed the properties of ontology metrics. Sometimes simple structural metrics are sufficient for the task at hand, and many structural metrics exist today. Our goal in this paper was to raise the awareness for the difference between structural and ontological metrics, and to provide principle means for the simple definition of metrics that take the semantics of the ontology appropriately into account.

Ontology normalization was introduced as a preprocessing step in order to align structural measures with intended semantic measures. Further properties, like the stability of a metric towards ontology extension and merges, and the non-dichotomous nature

of ontologies were discussed, and an approach towards encapsulating these problems was suggested by introducing stable metrics.

In addition to offering the theoretical tool of normalization, we are planning to implement it as an extension to the OWL tools[1]. This will allow to access these metrics both from the command line as well as from a Java API. Besides making normalization available to tools and metric suites, this will also allows us to evaluate if there are further benefits to normalized ontologies. We assume such benefits with regards to query answering performance, usability, and ontology maintenance. Some properties of normalization suggest advantages in these and other areas, but we expect some parts of the normalization process to be adapted based on differing requirements by these other use cases.

Based on the foundational work provided in this paper, we plan to adapt, extend, and implement several metrics already known in literature. We hope that a thorough evaluation of these metrics will allow to correlate quality attributes to these metrics, and thus to finally lead to viable sets of metrics and measures for the whole ontology life cycle. We don't think that there is one single such set, but the ideas presented here make several design decision when creating metrics more explicit and point to common problems and pitfalls when creating metrics and measures in this field. This will help in deciding which metrics to choose for a given scenario.

We expect that future work will continue on this basis in order to create a bigger tool set for everybody dealing with ontologies to allow them to evaluate ontologies during every step of the ontology life cycle. This will lead to an overall higher quality of ontologies, and thus to a stronger foundation on which the Semantic Web is being built.

Acknowledgments. Research reported in this paper has been partially financed by the EU in the IST project Knowledge Web (FP6-507482, `http://knowledgeweb.semanticweb.org/`). We want to thank our colleagues for the fruitful and interesting discussions about the ideas presented in this paper, especially Aldo Gangemi, Marta Sabou, Markus Krötzsch, Jos Lehmann, and Malvina Nissim.

References

1. H. Alani and C. Brewster. Metrics for ranking ontologies. In Vrandečić et al. [16].
2. J. C. Arpírez, O. Corcho, M. Fernández-López, and A. Gómez-Pérez. WebODE: a scalable workbench for ontological engineering. In *Proceedings of the First International Conference on Knowledge Capture (K-CAP) Oct. 21-23, 2001, Victoria, B.C., Canada*, 2001.
3. D. Brickley and L. Miller. The friend of a friend (FOAF) vocabulary specification, July 2005. Namespace Document 27 July 2005 ('Pages about Things' Edition).
4. T. DeMarco. *Controlling Software Projects: Management, Measurement & Estimation*. Yourdon Press, New York, 1982.
5. A. Gangemi, C. Catenacci, M. Ciaramita, and J. Lehmann. Ontology evaluation and validation: an integrated formal model for the quality diagnostic task. Technical report, Laboratory of Applied Ontologies – CNR, Rome, Italy, 2005. available at http://www.loa-cnr.it/Publications.html.

[1] `http://owltools.ontoware.org/`

6. A. Gangemi, C. Catenaccia, M. Ciaramita, and J. Lehmann. Modelling ontology evaluation and validation. In Y. Sure and J. Domingue, editors, *Proceedings of the 3rd European Semantic Web Conference (ESWC2006)*, number 4011 in LNCS, Budva, Montenegro, June 2006. Springer-Verlag.
7. A. Gangemi, C. Catenaccia, M. Ciaramita, and J. Lehmann. Qood grid: A metaontology-based framework for ontology evaluation and selection. In Vrandečić et al. [16].
8. B. C. Grau(ed.). OWL 1.1 web ontology language, November 2006. Available at http://owl1_1.cs.manchester.ac.uk/.
9. N. Guarino and C. Welty. Evaluating ontological decisions with OntoClean. *Communications of the ACM*, 45(2):61–65, February 2002.
10. J. Hartmann, Y. Sure, P. Haase, R. Palma, and M. del Carmen Suarez-Figueroa. OMV – ontology metadata vocabulary. In C. Welty and A. Gangemi, editors, *ISWC 2005 - In Ontology Patterns for the Semantic Web*, Galway, Ireland, November 2005.
11. A. Lozano-Tello and A. Gómez-Pérez. OntoMetric: A method to choose the appropriate ontology. *Journal of Database Management, Special Issue on Ontological analysis, Evaluation, and Engineering of Business Systems Analysis Methods*, 15(2), April-June 2004.
12. M. K. Smith, C. Welty, and D. McGuinness. OWL Web Ontology Language Guide, 2004. W3C Recommendation 10 February 2004.
13. Y. Sure, J. Angele, and S. Staab. OntoEdit: Multifaceted inferencing for ontology engineering. *Journal on Data Semantics*, 1(1):128–152, NOV 2003. LNCS 2800.
14. S. Tartir, I. B. Arpinar, M. Moore, A. P. Sheth, and B. Aleman-Meza. OntoQA: Metric-based ontology quality analysis. In *Proceedings of IEEE Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources*, 2005.
15. J. Völker, D. Vrandečić, and Y. Sure. Automatic evaluation of ontologies (AEON). In Y. Gil, E. Motta, V. R. Benjamins, and M. A. Musen, editors, *Proceedings of the 4th International Semantic Web Conference (ISWC2005)*, volume 3729 of *LNCS*, pages 716–731. Springer Verlag Berlin-Heidelberg, NOV 2005.
16. D. Vrandečić, M. del Carmen Surez-Figueroa, A. Gangemi, and Y. Sure, editors. *Proceedings of the 4th International Workshop on Evaluation of Ontologies for the Web (EON2006) at the 15th International World Wide Web Conference (WWW 2006)*, Edinburgh, Scotland, May 2006.
17. T. D. Wang. Gauging ontologies and schemas by numbers. In Vrandečić et al. [16].