

# Using Google Distance to weight approximate ontology matches

Risto Gligorov  
Zharko Aleksovski  
Warner ten Kate  
Philips Research, Eindhoven  
[zharko@few.vu.nl](mailto:zharko@few.vu.nl)  
[warner.ten.kate@philips.com](mailto:warner.ten.kate@philips.com)

Frank van Harmelen  
Vrije Universiteit, Amsterdam  
[Frank.van.Harmelen@cs.vu.nl](mailto:Frank.van.Harmelen@cs.vu.nl)

## ABSTRACT

Discovering mappings between concept hierarchies is widely regarded as one of the hardest and most urgent problems facing the Semantic Web. The problem is even harder in domains where concepts are inherently vague and ill-defined, and cannot be given a crisp definition. A notion of approximate concept mapping is required in such domains, but until now, no such notion is available.

The first contribution of this paper is a definition for *approximate mappings between concepts*. Roughly, a mapping between two concepts is decomposed into a number of submappings, and a *sloppiness value* determines the fraction of these submappings that can be ignored when establishing the mapping.

A potential problem of such a definition is that with an increasing sloppiness value, it will gradually allow mappings between any two arbitrary concepts. To improve on this trivial behaviour, we need to design a heuristic weighting which minimises the sloppiness required to conclude desirable matches, but at the same time maximises the sloppiness required to conclude undesirable matches. The second contribution of this paper is to show that a *Google-based similarity measure* has exactly these desirable properties.

We establish these results by *experimental validation in the domain of musical genres*. We show that this domain does suffer from ill-defined concepts. We take two real-life genre hierarchies from the Web, we compute approximate mappings between them at varying levels of sloppiness, and we validate our results against a hand-crafted Gold Standard.

Our method makes use of the huge amount of knowledge that is implicit in the current Web, and exploits this knowledge as a heuristic for establishing approximate mappings between ill-defined concepts.

## Categories and Subject Descriptors

I.2.4 [ARTIFICIAL INTELLIGENCE]: Knowledge Representation Formalisms and Methods

## General Terms

information integration, semantic integration, ontology mapping

## Keywords

approximation, Google distance

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.

WWW 2007, May 8–12, 2007, Banff, Alberta, Canada.  
ACM 978-1-59593-654-7/07/0005.

## 1. INTRODUCTION & MOTIVATION

### 1.1 Introduction

The progress of information technology has made it possible to store and access large amounts of data. However, since people think in different ways and use different terminologies to store information, it becomes hard to search each other's data stores. With the advent of the Internet, which has enabled the integrated access of an ever-increasing number of such data stores, the problem becomes even more serious.

The Semantic Web aims to use semantics in the retrieval process, where the semantics is captured in ontologies or at the very least in concept hierarchies. The task then is to find pairs of concepts from different meta-data schemas that have an equivalent meaning, a problem known as ontology matching. This problem has been extensively studied in the Semantic Web and elsewhere, see [1, 2, 3, 4, 5] for recent survey papers.

However, in many realistic domains, it is impossible to give precise concept definitions, and consequently no crisp notion of concept equivalence exists. Below we will illustrate this in the music-domain (an important commercial domain on the Web), where musical genres are inherently imprecise. Such imprecision is a fundamental aspect of many other domains as well. Ontology matching must then be redefined to finding a concept with the closest meaning in the other schema when an equivalent one does not exist. We then require mechanisms that are able to find approximate correspondences rather than exact ones.

The first contribution of this paper is to define a notion of *approximate ontology matching* between inherently imprecise domain concepts (section 2). In section 3 we refine this definition with a weighting function to ensure that the approximation method does not simply allow any mappings, but that correct approximations are favoured over incorrect ones. As the second main contribution of this paper, in section 3.3, we instantiate this weighting function with a Google-based scheme, and show in section 4 through experiments in the music domain that this weighting scheme has indeed the desired behaviour of increasing recall without losing precision.

Before moving to the technical part of the paper, we first briefly introduce the domain of musical genres, and will argue why this is an appropriate domain for investigating techniques for approximate ontology mapping.

### 1.2 Internet Music Schemas

The variety and size of music content on the Internet (even when restricted to legal distributions) make it difficult to find music of interest. It is often cumbersome to retrieve even a known piece of

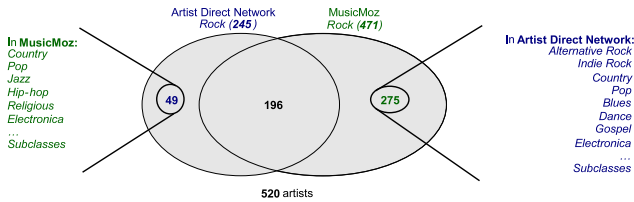


Figure 1: Low semantic agreement between ADN and MM.

music, given the large number of music providers, each with their own music schema. Finding the right music that fits a user’s preferences is dependent on semantic integration over different music provider’s schemas.

Music content providers usually classify the music they offer into classes for easy access. These classes are organised in a hierarchy. We refer to the classified entities as instances. Usually these are artists, albums, compilations, other kinds of releases, songs, etc. The majority of the terms used to identify a class of music entities are music genres and styles, for example *Blues*, *Jazz*,... with further distinctions such as *American Blues*, *Electronic Jazz*,...

**Imprecision in Music Schemas.** Music genres and styles are intrinsically ill-defined, see [6, 7]. There is no single authority that can decide for a music entity which genre or style it belongs to. For example, it’s becoming increasingly difficult to categorise the newly emerging musical styles that incorporate features from multiple genres. Also, the attempts to classify particular musicians in a single genre are sometimes ill-founded as they may produce music in a variety of genres over time or even within a single piece.

There are no objective criteria that sharply define music classes. Genre is not precisely defined. When asked, people will classify the same music entity in different genres, with an agreement of only in the 30-40% region. As a result, different providers often classify the same music entities (artists, albums, songs...) differently. Widely used terms like Pop and Rock do not denote the same sets of artists at different portals, [6]. That is also the case for even more specific styles of music like Speed Metal.

In our experiments when testing with instance data, we restricted to the artists shared by MusicMoz and Artist Direct Network, i.e. artists that are present and classified in both portals. In the sequel we refer to them as MM and ADN, respectively. As an example, from the class named *Rock* (including its subclasses) in MM there are 471 shared classified artists, in ADN there are 245, and 196 shared artists are classified under *Rock* in both of them. Hence, from all the artists classified under *Rock* in at least one of the two portals, only about 38% (196 out of 520) is classified under *Rock* in both portals, see figure 1.

This example shows that there is a high degree of imprecision in the music domain. Therefore we expect that reasoning methods that look for exact matches are not useful, and approximate methods are more appropriate.

**Representation of hierarchical Music Schema.** As with any semantic concept, musical genres can in principle be defined either intensionally (as a set of rules that define when an entity belongs to the genre or not), or extensionally (as the set of all music entities that belong to the genre). With rare exceptions such as Music Genome<sup>1</sup> and Wikipedia<sup>2</sup>, who aim to provide intensional

<sup>1</sup><http://www.pandora.com/mgp.shtml>

<sup>2</sup><http://en.wikipedia.org/wiki/Music.genre>

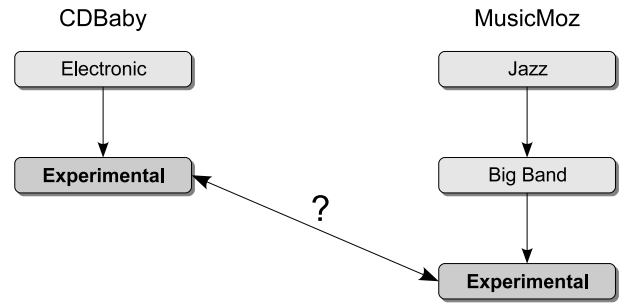


Figure 2: Two music genres: Although the labels are equivalent, *Experimental*, they represent different classes.

definitions of musical genres, the extensional approach is the most widely used in practice. Hence, for our purpose, we assume an extensional treatment for the genres and styles as sets of music entities. Consequently, a music classification is a collection of music concepts described with English language terms, where instances are being classified. It can be modelled as a concept hierarchy.

However, it is not sufficient to use only the concept labels to identify the concepts, since, their position in the schemas influences their meaning as well. Figure 2 illustrates this with an example from existing music schemas. Although the labels are equivalent (namely *Experimental*), they represent different classes. We adopt the approach proposed in [8] to make the meanings explicit by conjoining the concepts with their superconcepts in the hierarchy. This then makes the meaning of the two concepts in the example explicit as

$$\begin{aligned} & \text{Electronic} \cap \text{Experimental} \\ & \text{Jazz} \cap \text{Big Band} \cap \text{Experimental} \end{aligned}$$

**Data Selection.** Music metadata schemas on the Internet mostly exist in the form of a navigation path through the music offered. A meta-data schema isn’t always offered next to the music, but a visitor can interactively navigate through different pages that list the music. We consider this structure of navigation paths together with the labelling on the links and pages as the meta-data schema of that provider. After considering several music provision sites, we selected seven of them and extracted the schema (navigation path), as shown in figure 3. Also, we have extracted the music schema present in the free online encyclopaedia Wikipedia.

## 2. AN APPROXIMATION METHOD FOR ONTOLOGY MATCHING

Ontology matching is often phrased as finding equivalences  $A \equiv B$  between concepts  $A$  and  $B$  from separate hierarchies. We take a slightly more general view, namely that of finding subsumption relations  $A \subseteq B$  between concepts from separate hierarchies (with finding equivalences as an obvious special case of mutual subsumptions).

The representation of concepts as a conjunction of propositional symbols described in the previous section implies that concepts have the form  $B = B_1 \cap \dots \cap B_k$ <sup>3</sup>. This means that the subsumption check

$$A \subseteq B \tag{1}$$

<sup>3</sup> Remember that we use the interpretation of the concepts as sets and consequently we replace conjunction by intersection relation, and implication by a subset relation.

Schema	# concepts	max. depth
CDNOW <a href="http://www.cdnw.com/">http://www.cdnw.com/</a>	2410	5
MusicMoz <a href="http://musicmoz.org/">http://musicmoz.org/</a>	1073	7
Artist Direct Network <a href="http://artistdirect.com/">http://artistdirect.com/</a>	465	2
All Music Guide <a href="http://www.allmusic.com/">http://www.allmusic.com/</a>	403	3
Artist Gigs <a href="http://www.artistgigs.com/">http://www.artistgigs.com/</a>	382	4
CD Baby <a href="http://www.cdbaby.com/">http://www.cdbaby.com/</a>	222	2
Yahoo LaunchCast <a href="http://launch.yahoo.com/">http://launch.yahoo.com/</a>	96	2

Figure 3: The extracted music schemas.

can be split into a set of *subproblems*:

$$\bigwedge_i (A \subseteq B_i) \quad (2)$$

with each subproblem checking if  $A$  is a subclass of one of the conjuncts  $B_i$ . The original subsumption problem (1) is satisfied if and only if all the subproblems from (2) are satisfied.

This approach is independent from the choice of solver to use for solving the individual subproblems  $A \subseteq B_i$ . This might be an intensional reasoner (using axioms for  $A$  and  $B_i$ ), or an extensional checker (checking the instance sets of  $A$  and  $B_i$ ), or a simple lexical approach (using the textual descriptive labels of  $A$  and  $B_i$ ). In our experiments later in this paper, we will in fact use the lexical approach, but our approach to approximate ontology matching is independent of this choice.

Now we introduce the idea of approximation. In our approximation, we allow a few of the subproblems from (2) to be unsatisfiable, while still declaring the original problem (1) satisfiable. The (relative) number of satisfiable subproblems is a measure of how strongly the subclass relation between the two given formulas hold. If for only a few of the subproblems the relation (2) doesn't hold, we may say that the relation (1) *almost holds*. We can express the strength at which the relation (1) holds as the ratio between the number of false subproblems (subproblems for which the subclass relation doesn't hold) and the total number of subproblems. We call this ratio the *sloppiness* and we use the letter  $s$  to denote its value:

$$s(A \subseteq B) = \frac{|\{i : A \not\subseteq B_i\}|}{k} \quad (3)$$

Here  $|\{i : A \not\subseteq B_i\}|$  denotes the number of unsatisfied subproblems that are ignored, and  $k$  is the total number of subproblems. We will later refer to this sloppiness value as the *uniform weighting*.

In the example of figure 4, the subsumption  $A \subseteq B_1 \cap B_2 \cap B_3$  is not classically valid, because  $A \not\subseteq B_3$ , but it is valid at a sloppiness level of 1/3.

Some properties that can be easily seen are:

PROPERTY 1. *Only classically valid subsumptions hold at  $s = 0$ .*

PROPERTY 2. *Any subsumption holds at  $s = 1$ .*

<sup>4</sup>We will often omit the argument to  $s$  if this is clear from the context, and simply speak of the sloppiness value  $s$ .

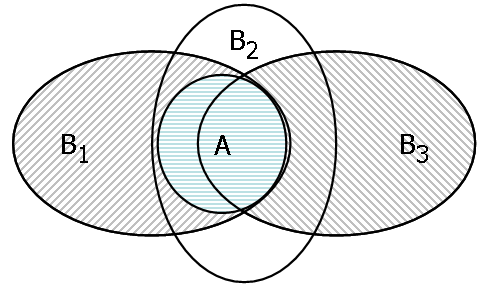


Figure 4: Example of an approximate subsumption relation:  $A \subseteq B$  with  $B = B_1 \cap B_2 \cap B_3$ .

From these two properties it follows that our approximation only makes sense for conjunctive formulae  $B_1 \dots B_k$  with  $k > 1$ . A trivial conjunctive formula with only one conjunct is not approximable: the only sloppiness values for such a formula are either 0 or 1, limiting us to either classical subsumption or trivial acceptance of the subsumption. But such a formula would correspond to a hierarchy of depth 1, which is a case not very interesting for our problem anyway.

PROPERTY 3. *The set of subsumptions that hold at a given sloppiness value grows monotonically with increasing sloppiness value.*

Properties 1-3 together tell us that at small  $s$ -values, the approximation will be too strict, and no (actually: only classically valid) mappings will be found; while at large  $s$ -values, the approximation will be too loose, and many invalid mappings (actually: all combinations) will be found.

In the next subsection, we will discuss how we can influence the approximation level (the  $s$ -values) at which valid and invalid mappings are found.

### 3. HEURISTIC WEIGHTING

#### 3.1 Defining weights

We can influence the sloppiness level at which a mapping is discovered with a heuristic *weighting function*: each subproblem  $A \subseteq B_i$  is given a weight  $w_i$ , and the *heuristic sloppiness* value  $\tilde{s}(A \subseteq B)$  is then the summed weight of all the subproblems that needed to be ignored for the subsumption to be accepted:

$$\tilde{s}(A \subseteq B) = \sum_{\{i : A \not\subseteq B_i\}} w_i.$$

Notice that the non-heuristic  $s$ -value (the uniform weighting) is a special case of the heuristic  $\tilde{s}$ -value, namely with equal values for all  $w_i$ .

#### 3.2 Choosing Weights

How should we make a heuristic weighting function, and how should we judge if one weighting heuristic is better than another? As before, let  $A \subseteq B$  with  $B = B_1 \cap B_2 \cap B_3$  be a potential ontology match (= cross-hierarchy subsumption relation) that does not hold classically. To judge the quality of a weighting heuristic, we must distinguish two cases:

**desirable matches:** although  $A \subseteq B$  does not hold classically, it might nevertheless be a useful mapping. (This might be known from intended (informal) meaning of  $A$  and  $B$ , or because we have a gold standard that contains this match).

In this case, we would want  $A \subseteq B$  to be derived at a *low* level of sloppiness (i.e. by ignoring only a small number of well-chosen subproblems)

**undesirable matches:** conversely, we might know from the intended meaning of  $A$  and  $B$  that indeed  $A \subseteq B$  should not be derived. In this case, we would want  $A \subseteq B$  to be derived only at a very high level of sloppiness (i.e. only after ignoring many subproblems).

Put in other words, a heuristic weighting function should minimise the sloppiness required to derive desirable matches, and maximise the sloppiness required to derive undesirable matches. This would ensure that when gradually increasing the allowed sloppiness level, we begin to discover desirable matches quickly, while only including undesirable matches very late in the process. This would have the effect that *when increasing the  $s$ -value, we have an early increase of recall, but a late decrease of precision.*

It is easily seen that the uniform weighting scheme from formula (3) amounts to choosing equal values for all  $w_i$  (namely  $1/k$ ), and hence to making a random choice for subproblems to ignore. In general, however, the subterms  $B_i$  have different significance in capturing the meaning of  $B$ . We would expect any reasonable heuristic to improve over a random choice, and ignoring more significant subterms should go with a penalty, i.e. should require a higher  $w_i$  for such significant subterms. A requirement on an informed heuristic form choosing the weights  $w_i$  is therefore:

REQUIREMENT 1. *The heuristic weights  $w_i$  should be chosen in such a way that:*

- if  $A \subseteq B$  is a desirable match, then  $\tilde{s}(A \subseteq B) < s(A \subseteq B)$
- if  $A \subseteq B$  is an undesirable match, then  $\tilde{s}(A \subseteq B) > s(A \subseteq B)$ .

Consider again the example of figure 4, and let  $A \subseteq B$  be a desirable match. Suppose that we heuristically choose the following weights:  $w_1 = 0.7, w_2 = 0.2, w_3 = 0.1$ . Then this heuristic accepts  $A \subseteq B$  already at sloppiness level 0.1, i.e.  $\tilde{s}(A \subseteq B) = 0.1$ , since only  $A \subseteq B_3$  with weight 0.1 needs to be non-classically assumed to hold (even though classically it doesn't).

The uniform weighting scheme would result in  $s(A \subseteq B) = 1/3$ . If, as assumed,  $A \subseteq B$  is a desirable match, then the uniform weighting performs less well than the example heuristic values, since the required increase in recall (i.e. accepting  $A \subseteq B$ ) is only achieved at a higher sloppiness value (1/3 instead of 0.1). If on the other hand  $A \subseteq B$  would be an undesirable match, the uniform weighting would be preferred.

Clearly, the recall and precision of the the above approximations mapping approach relies on choosing the right weights  $w_i$ .

The main intuition behind the choice for a good weighting function is that the weight assigned to the subproblem  $A \subseteq B_i$  should reflect how much information the concept  $B_i$  provides about the concept  $B$ . The level of *informativeness* can be observed as a “semantic closeness” between the concepts  $B_i$  and  $B$ . Intuitively, a concept  $B_i$  that is semantically close to  $B$  should be more relevant, and have a higher weight, than a concept that is semantically more distant. In the next section, we will consider a weighting heuristic based on such a notion of semantic distance.

### 3.3 Normalised Google Distance

The weighting scheme we consider in this section takes advantage of the vast knowledge available on the web by using a Google-based dissimilarity measure.

We utilise a dissimilarity measure, called Normalised Google Distance (NGD), introduced in [9]. NGD takes advantage of the number of hits returned by Google to compute the semantic distance between concepts. The concepts are represented with their labels which are fed to the Google search engine as search terms. Given two search terms  $x$  and  $y$ , the *the normalised Google distance* between  $x$  and  $y$ ,  $NGD(x, y)$ , can be obtained as follows

$$NGD(x, y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x, y)}{\log M - \min\{\log f(x), \log f(y)\}} \quad (4)$$

where

$f(x)$  is the number of Google hits for the search term  $x$ ,  
 $f(y)$  is the number of Google hits for the search term  $y$ ,  
 $f(x, y)$  is the number of Google hits for the tuple of search terms  $x y$  and

$M$  is the number of web pages indexed by Google<sup>5</sup>.

Intuitively,  $NGD(x, y)$  is a measure for the symmetric conditional probability of co-occurrence of the terms  $x$  and  $y$ : given a web-page containing one of the terms  $x$  or  $y$ ,  $NGD(x, y)$  measures the probability of that web-page also containing the other term<sup>6</sup>.

**Example:** We will determine the normalised Google distance between the search terms “jazz” and “rock” that correspond to concepts  $B_{jazz}$  and  $B_{rock}$ , respectively. The number of hits for the search term “jazz” is given by  $f(jazz) = 196\,000\,000$  and for the search term “rock” by  $f(rock) = 723\,000\,000$ . Furthermore, Google returns  $f(jazz, rock) = 119\,000\,000$  web pages in which both “jazz” and “rock” occur. Therefore,  $NGD(jazz, rock) = 0.458846$ .<sup>7</sup>

### 3.4 Google-based weighting

For a subsumption check  $A \subseteq B$  with  $B = B_1 \cap \dots \cap B_k$ , the Google-based weighting scheme is defined as follows:

First, we compute the *normalised Google distances*.

$$d_i = NGD(B_i, B)$$

We normalise these values to the  $[0, 1]$  interval

$$d'_i = \frac{d_i}{\sum_{j=1}^k d_j} \quad (5)$$

Subsequently, the normalised distance values are converted into similarities

$$s_i = 1 - d'_i$$

Finally, from the similarity values the weights for the subproblems are derived.

$$w_i = \frac{s_i}{\sum_{j=1}^m s_j}$$

<sup>5</sup>Currently, the Google search engine indexes approximately ten billion pages ( $M \approx 10^{10}$ ).

<sup>6</sup>The NGD measure assumes monotonicity of Google. In reality Google is known to show non-monotonic behaviour, i.e. adding more words in the search query may increase the number of hits instead of decrease it. Yet, such cases are exceptions and did not affect the results of our experiments

<sup>7</sup> The values for these queries were obtained by conjoining the search terms (“jazz” and “rock”) with the general scope-term “music”, in order to avoid homonym problems such as “rock” in its geological sense.

### 3.5 Modified Google Distance

The general NGD formula (4) (taken from [9]) can be slightly simplified because of the special form of our queries. As said, we compute the NGD values using (4)

$$\begin{aligned} d_i &= \text{NGD}(B_i, B) \\ &= \frac{\max\{\log f(B_i), \log f(B)\} - \log f(B_i, B)}{\log M - \min\{\log f(B_i), \log f(B)\}} \end{aligned} \quad (6)$$

where  $f(B_i)$ ,  $f(B)$  and  $f(B_i, B)$  give the number of hits for  $B_i$ ,  $B$  and  $(B_i, B)$ , respectively.

The Google query for  $B_i$  is comprised of search terms that are also present in the Google query for  $B$ . Therefore,  $f(B_i) \geq f(B)$ . This inequality implies that

$$\begin{aligned} \max\{\log f(B_i), \log f(B)\} &= \log f(B_i) \text{ and} \\ \min\{\log f(B_i), \log f(B)\} &= \log f(B). \end{aligned}$$

For the same reason (namely that all terms from  $B_i$  also occur in  $B$ ), the Google query for  $B$  coincides with the Google query for the tuple  $(B_i, B)$  which means  $f(B_i, B) = f(B)$ .

If we consider the last few deductions we can rewrite (6) in the following manner

$$d_i = \text{NGD}(B_i, B) = \frac{\log f(B_i) - \log f(B)}{\log M - \log f(B)}$$

or

$$d_i = \text{NGD}(B_i, B) = \frac{N_i}{N} \quad (7)$$

where  $N = \log M - \log f(B)$  and  $N_i = \log f(B_i) - \log f(B)$

If we consider (7) we can rewrite the expressions given with (5) as follows:

$$d'_i = \frac{\frac{N_i}{N}}{\sum_{j=1}^m \frac{N_j}{N}} = \frac{N_i}{\sum_{j=1}^m N_j}$$

As we can see the value of  $d'_i$  does not depend on  $N$ . Therefore, we can use the following expression to compute the Normalised Google Distance

$$d_i = m\text{NGD}(B_i, B) = \log f(B_i) - \log f(B) = \log \frac{f(B_i)}{f(B)}$$

where  $m\text{NGD}$  stands for modified Normalised Google Distance. The advantage of NGD over the NGD is that  $m\text{NGD}$  no longer depends on  $M$ , the size of the Google index, for which we would have to guesstimate a value.

### 3.6 Examples

In this section we illustrate the process of approximate ontology matching with weighting. In our examples we consider the subsumption relations between two pairs of styles from MusicMoz and ArtistDirectNetwork portals, as shown in Figures 5 and 6.

*Example 1: Country and Bluegrass gospel.* The first step is to transform the concepts into formulas. The individual concepts are represented with their labels, and conjoined with the representation of their parents, as discussed in section 1.2. This yields the following formulas representing the meaning of *Country* from ADN and *Bluegrass Gospel* from MM:

$$\begin{aligned} \text{Country}_{ADN} &= \text{Country} \\ \text{Bluegrass Gospel}_{MM} &= \text{Country} \cap \text{Bluegrass} \cap \\ &\quad \text{Bluegrass Gospel} \end{aligned}$$

We use these formulas to test for the subsumption relation

$$\text{Country}_{ADN} \subseteq \text{Bluegrass Gospel}_{MM}.^8 \quad (8)$$

As described in section 2, we have to solve the following subproblems:

$$\begin{aligned} \text{Country} &\subseteq \text{Country} \\ \text{Country} &\subseteq \text{Bluegrass} \\ \text{Country} &\subseteq \text{Bluegrass Gospel} \end{aligned}$$

In order to solve the individual subproblems, we apply the following method:

- Concepts with the same label have the same meaning
- If one label is derived from another by adding extra words, we assume that the first is a more specific concept than the second, hence the second concept subsumes the first. This is a reasonable assumption because additional words usually restrict an expression's meaning.

Given this, we obtain the following results for the subproblems

$$\begin{aligned} \text{Country} \subseteq \text{Country} &: \text{true (Country on both sides)} \\ \text{Country} \subseteq \text{Bluegrass} &: \text{false} \\ \text{Country} \subseteq \text{Bluegrass Gospel} &: \text{false} \end{aligned}$$

Using the uniform weighting scheme, subsumption (8) is acceptable at a sloppiness-level of 0.66, since 2 out of 3 subproblems are found not to hold, and must be ignored in order for the subsumption to go through

Next, we apply the Google-based weighting scheme on the same set of subproblems. We compute the NGD values as described in section 3.4:

$$\begin{aligned} d_1 &= \text{NGD}(\text{Country}, \\ &\quad \text{Country} \cap \text{Bluegrass} \cap \text{Bluegrass Gospel}) \\ d_2 &= \text{NGD}(\text{Bluegrass}, \\ &\quad \text{Country} \cap \text{Bluegrass} \cap \text{Bluegrass Gospel}) \\ d_3 &= \text{NGD}(\text{Bluegrass Gospel}, \\ &\quad \text{Country} \cap \text{Bluegrass} \cap \text{Bluegrass Gospel}) \end{aligned}$$

After providing each of the terms with the scope-term "music", the Google queries "Country" music, "Bluegrass" music, "Bluegrass Gospel" music and "Country" "Bluegrass" "Bluegrass Gospel" music return 467 000 000, 24 000 000, 338 000 and 261 000 hits, respectively. Consequently, we find the following  $m\text{NGD}$  distances:

$$\begin{aligned} d_1 &= 7.44147 \\ d_2 &= 4.39942 \\ d_3 &= 0.164622. \end{aligned}$$

Using these values we derive the weights for the subproblems, as described in section 3.4:

$$\begin{aligned} w_1 &= 0.190081 \\ w_2 &= 0.336629 \\ w_3 &= 0.493144 \end{aligned}$$

Since the last two of the three subproblems must be ignored for the match to go through, the heuristic  $\tilde{s}$ -value is 0.81. Hence, in this example,  $\tilde{s}$ -value  $>$   $s$ -value, which, since the match is actually undesirable, is an improvement (of 15 %-points) over the uniform weighting, in line with the requirement stated in section 3.1.

<sup>8</sup>As any music-lover will know, this subsumption is actually false, and hence constitutes an undesirable mapping.

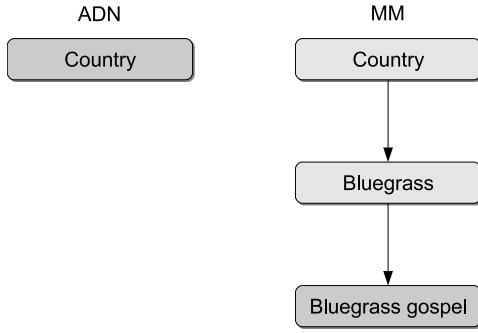


Figure 5: Matching concepts *Country* and *Bluegrass gospel*

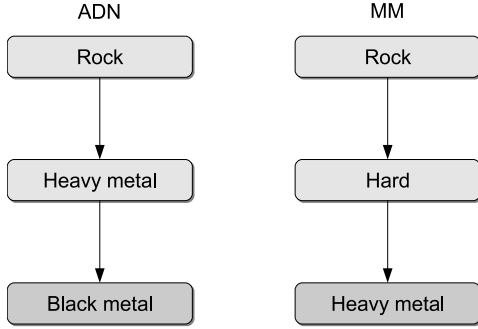


Figure 6: Matching concepts *Black metal* and *Heavy metal*

*Example 2: Black metal and Heavy metal.* The previous example showed the behaviour of the weighting on an undesired match. In this example we examine the weighting behaviour on a desired match. Figure 6 shows the following matching problem:

$$Black\ metal_{ADN} \subseteq Heavy\ metal_{MM}.^9 \quad (9)$$

The separate subproblems in this case are:

$$\begin{aligned} Rock \cap Heavy\ metal \cap Black\ metal &\subseteq Rock \\ Rock \cap Heavy\ metal \cap Black\ metal &\subseteq Hard \\ Rock \cap Heavy\ metal \cap Black\ metal &\subseteq Black\ metal \end{aligned}$$

The first and the third subsumptions are correct established lexically, but the second is found false. As a consequence the uniform weighting scheme would need a sloppiness-level of  $s = 0.33$  to establish (9). Using NGD we obtain the following weight values for each of the subproblems:

$$\begin{aligned} w_1 &= 0.25462 \\ w_2 &= 0.25588 \\ w_3 &= 0.48950. \end{aligned}$$

This allows (9) to be established at a heuristic sloppiness-level of  $\tilde{s} = 0.25588$ . This is 8 %-points lower than the required  $s$ -value, and hence an improvement, since (9) is a desired mapping.

## 4. EXPERIMENTS

To validate our approach of improving approximation by weighting, we conducted experiments using data from the music domain. In this section, we describe the experiments and summarise and discuss the results.

<sup>9</sup>*Black metal* is widely accepted as being a substyle of *Heavy metal*

## 4.1 Experimental setup

**Goal of the experiment:** In our experiment, we want to measure both if approximation improves over simple classical matching (first goal), and if approximation with heuristic weighting improves over uniform weighting (second goal).

**Data acquisition:** We used genre hierarchies that we extracted from the meta-data schemas underlying the classifications of the Artist Direct Network (ADN), MusicMoz (MM) and Wikipedia music portals, as discussed in section 1.2.<sup>10</sup> One of the concept was always from MM, and the other concept in 30% of the cases from ADN and 70% of the cases from Wikipedia

**Construction of a Gold Standard:** We randomly selected 50 pairs of concepts, and manually assessed whether a mapping (= a cross-hierarchy subsumption relation) between the concepts within each selected pair holds or not. In other words, we classified each of the 50 pairs as either a *desirable* or an *undesirable* matching relation, as defined in section 3.2. This yielded 9 desirable mappings and 41 undesirable mappings.<sup>11</sup>

**Evaluation criteria:** For measuring the first goal (improvement of approximate matching over classical matching), we use standard recall and precision measures: heuristic matching is better than classical matching if recall improves (more desirable matches are found), while precision does not decrease much (not many undesirable matches are found). For measuring the second goal (improvement of heuristic weighting over uniform weighting), we use Requirement 1 from section 3.2. Heuristic weighting is better than uniform weighting if desirable matches are found with a lower heuristic  $\tilde{s}$ -value than the uniform  $s$ -value, and conversely for undesirable matches.

**Choice of subproblem solver:** Remember that we regard ontology matches as cross-hierarchy subsumptions, and that our approximation method reduces subsumption-queries  $A \subseteq B$  to a set of subproblems  $A \subseteq B_i$ . Of course, each of these atomic subproblems still needs to be solved. As in section 3.6, we apply a simple lexical method that establishes whether  $A$  is (strictly) subsumed by  $B_i$  by checking whether the set of words in label of  $A$  is (properly) contained in the set of words of the label of  $B_i$  (since additional words usually restrict an expression's meaning).

## 4.2 Comparing approximate and exact matching

When comparing approximate matching with exact matching, we cannot simply compare two scores. After all, the behaviour of approximate matching is a function of the sloppiness-level  $s$ .

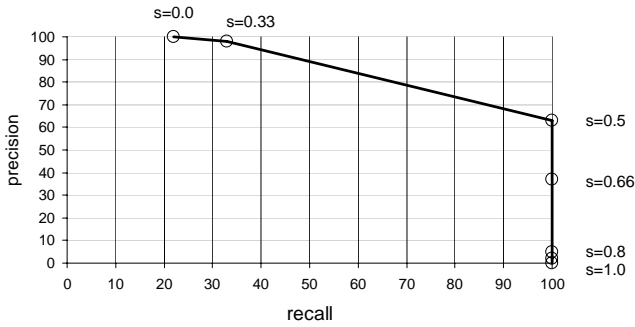
Figure 7 shows a recall-precision plot of uniformly weighted approximate matching. Since exact matching corresponds to approximate matching at  $s = 0$ , the graph also shows the precision and recall of exact matching: 22% recall at 100% precision. This score is to be expected of the simple lexical matcher that we employ to solve subproblems: lexical matching will fail to find many semantically desirable matches (hence the low recall of 22%), but when it finds a match, it is indeed a correct one (precision of 100%).

As expected of approximate matching, increasing values of  $s$  increase recall and decrease precision. (Figure 7 quantitatively shows what was already qualitatively predicted by properties 1 and 2 from section 2).

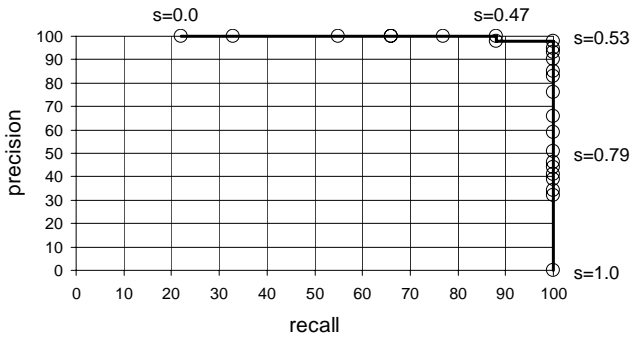
Concerning the comparison with exact matching, figure 7 shows that a slight increase in  $s$ -levels is indeed useful (a jump in recall

<sup>10</sup> We selected these particular genre hierarchies because they contained enough additional information for us to construct a Gold Standard.

<sup>11</sup> It is probabilistically plausible that of set of randomly drawn cross-hierarchy pairs, the large majority are undesirable matches.



**Figure 7: Recall/precision graph of approximate matching with uniform weighting at increasing sloppiness-levels. Exact matching happens at  $s=0$ .**



**Figure 8: Recall/precision graph of approximate matching with Google weighting at increasing sloppiness-levels.**

from 22% to 33% at a negligible loss of precision when increase  $s$  from 0.2 to 0.33). The graph also shows that there is no point in increasing the uniform  $s$ -value beyond 0.5, since at that point full recall is achieved. However, this has happened at the cost of a significant loss in precision (down to 63% at  $s = 0.5$ ).

Overall, figure 7 tells us that approximate matching at small  $s$ -values does indeed usefully improve over exact matching, even when randomly choosing the subproblems to be approximated (= the uniform weighting).

In the next section, we will evaluate to what extent an informed weighting heuristic can improve over this.

### 4.3 Google-based weighting experiments

The behaviour of the Google-weighted approximate matching is shown in the recall-precision plot of figure 8. This is near perfect recall-precision plot, which shows that we can increase the  $\tilde{s}$ -values (and hence increase the recall) until we have achieved perfect recall, all at the cost of only a negligible loss in precision of 2%, at  $\tilde{s} = 0.53$ . The precision only starts to drop significantly after a perfect recall has been achieved. This plot compares very favourably with that of the uniform weighting scheme in figure 7, where a perfect recall could only be achieved at a disappointing 63%.

Interestingly enough, both the uniform and the Google weighting achieve 100% recall around the  $\tilde{s} = s = 0.5$  level (i.e. ignoring about half the subproblems), but apparently the Google weighting is very successful at maintaining exactly those subproblems which prevent a drop in precision (while the uniform weighting simply ignores random subproblems).

One observation which questions the stability of our results is

	Total	Better	Equal	Worse
Undesirable mappings	41	39	1	1
Desirable mappings	9	5	3	1
Total	50	44(88%)	4(8%)	2(4%)

**Figure 9: NGD weighting compared to uniform weighting. “Better” and “Worse” mean NGD weighting is better (worse) than uniform weighting.**

that after the  $\tilde{s} = 0.5$ -level, the precision starts to drop very sharply (to 83% at  $\tilde{s} = 0.75$  and even to 51% at  $\tilde{s} = 0.79$ ). This suggests that on other data-sets, the near perfect score at  $\tilde{s} = 0.53$  may not be repeated. Further experiments on other data-sets are required to verify this.

An overall summary of the results of our experiments is given in figure 9. Informed weighting yielded an improvement over uninformed weighting in almost all cases. The main gain is in a better performance on undesirable matches (i.e. improving precision).

The most detailed analysis of our experimental data is shown in figures 10 and 11. There we plot for all 41 undesirable mappings (figure 10) and for all 9 desirable mappings (figure 11) the minimal  $s$ - and  $\tilde{s}$ -values at which these mappings are found. Figure 10 shows that for almost all undesirable mappings, the informed  $\tilde{s}$ -values are higher than the  $s$ -value, meaning that the informed weighting scheme is more resistant to accepting such undesirable mappings (explaining its dramatically better precision scores in the preceding recall/precision plots). Figure 11 shows that for the desirable mappings, both weighting schemes behave roughly equally well.

Taken together, we can conclude that the main gain of the informed weighting scheme is that it manages to avoid a drop in precision while maintaining the high recall-levels of the uninformed scheme.

Figure 10 also shows that the lion-share of the gains by the informed weighting are made at the  $s = 0.5$  level, where the uninformed weighting accepts an incorrect mapping. These are often cases with labels of length 2 (such as “hard rock”  $\subseteq$  “gospel rock”). In this example, “gospel” is the most informative term, but since the uninformed weighting randomly chooses a subproblem to ignore, it may choose to ignore “gospel”, ending up with “hard rock”  $\subseteq$  “rock” as the only subproblem, which does indeed hold. The informed Google weighting would instead realise that “gospel” is the most informative term, hence choose to ignore “rock” instead, ending up with “hard rock”  $\subseteq$  “gospel” as the remaining subproblem, and correctly refusing to accept this mapping. Note that this is not simply a matter of preferring adjectives over nouns, consider for example the case “Country Gospel”  $\subseteq$  “Christian Gospel”, which is an intended mapping, and can only be established by preferring the noun “Gospel” over the adjective “Christian”. A truly semantically informed weighting scheme is required, no simple lexical fix will do.

## 5. RELATED WORK

Although we claim that our approach is entirely novel (both the idea of approximating ontology mappings by ignoring subproblems, and the idea of using the Google distance as a weighting heuristic for this problem), the different components of our approach can be found elsewhere.

Existing ontology mapping methods typically use one or a combination of the following approaches:

**terminological:** use the labels of the entities to derive matches be-

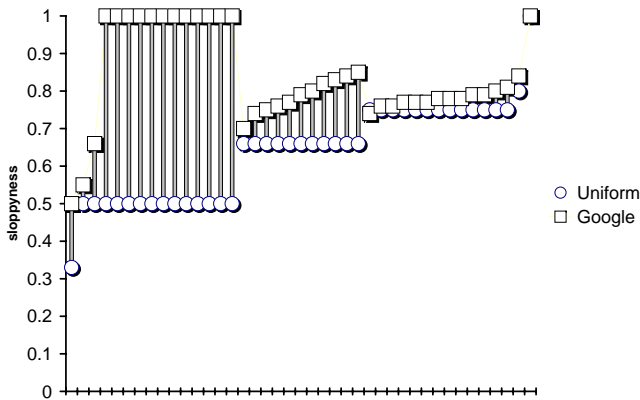


Figure 10: Google-based versus uniform weighting on undesirable matches

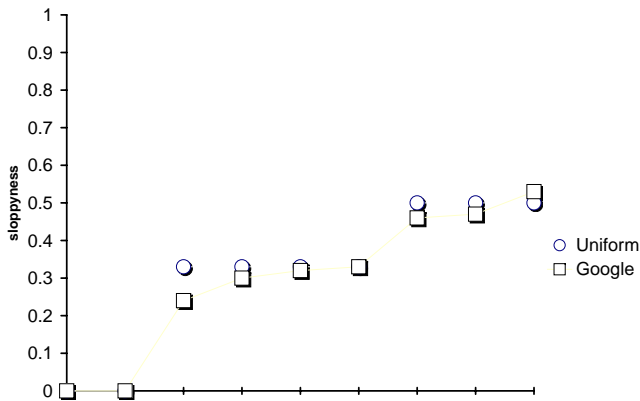


Figure 11: Google-based versus uniform weighting on desirable matches

tween the elements from different ontologies, see [10, 11, 12, 13]. The method that we employ to solve the subproblems is a very simple version of these methods.

**structural:** use the structure of the ontology, i.e. the relatedness among entities within the matching ontologies, see [14, 10, 11, 12]. Our method of splitting up  $A \subseteq B$  into  $A \subseteq B_i$  can be seen as a simple way of using logical structure of the ontology.

**instance-based:** use the overlap or relatedness between classified instances, see [15].

**background knowledge:** use external resources such as thesaurus, dictionaries or more complicated ontologies to perform the task, see [16, 17, 18, 19].

As mentioned in section 2, any of the above approaches to ontology matching could be deployed in solving our  $A \subseteq B_i$  subproblems.

Our Google-based weighting method resembles recent work in the field of *Knowledge Acquisition* which exploits the Web as a source for discovering new knowledge, for example to discover relations between concepts in various ways: using the number of hits when querying a search engine like Google, by analysing co-occurrence of concepts within large text corpora, or by exploiting patterns to construct queries to check for a relation (see [20, 21, 22, 9] for recent work).

Other attempts exist to deal with vaguely defined domain concepts, most notably by using fuzzy-logic representations (e.g. [23]), and more recently by using rough sets [24]. An inherent price of these approaches is that they require significant extra modelling effort to capture the imprecision of the concepts, either in terms of fuzzy membership functions or in terms of rough-set upper- and lower-bounds. The advantage of our approximation approach is that it applies directly to semantically lightweight hierarchies typically found in imprecise domains without any further need for complicated and expensive domain modelling.

Finally, we point out that the “confidence-factor” which is part of the format used by the Ontology Alignment Evaluation Initiative<sup>12</sup> only expresses the degree of confidence in the truth of a *crisp* matching relation, and does not represent an approximation of such a matching relation.

## 6. CONCLUSION

In this study, we have addressed the problem of discovering approximate matching relations between concepts from different concept hierarchies. Such approximate matching relations are required in many domains where concepts are ill-defined, and any attempt to find precise equivalences (or even precise subsumptions) will fail because of the imprecise nature of the concepts. Our method is directly applicable to the lightweight hierarchies found in practice in imprecise domains.

We have given a declarative definition for approximate ontology-mapping with a variable degree of approximation. We have shown how this approximation degree (for which we used the term sloppiness level) can be influenced by a semantic similarity measure that we derive from the Normalised Google Distance.

In order to validate our theoretical proposal, we harvested from the Web realistic concept hierarchies that represented musical genres, we constructed a small Gold Standard corpus of mapping candidates, and we performed experiments to test the precision and recall of our approximation method. Our results show that the Google-based semantic measure significantly outperforms an un-informed measure.

<sup>12</sup><http://oaei.ontologymatching.org/>

Our approach is entirely independent from the algorithm used to establish the submappings that together build up an approximate mapping. In the music domain, we have used simple lexical techniques to establish these submappings, but this can be replaced with more complicated mapping techniques, while the essential idea of our sloppiness value and the weighting function can still be applied unchanged.

In very general terms, our method makes use of the huge amount of knowledge that is implicit in the current Web, and exploits this knowledge as a heuristic for establishing approximate mappings between ill-defined concepts.

## 7. REFERENCES

- [1] Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. *Journal on Data Semantics* **IV** (2005) 146–171
- [2] Giunchiglia, F., Shvaiko, P.: *Semantic matching*. The Knowledge Engineering Review **18:3** (2004, Cambridge University Press) 265–280
- [3] Shvaiko, P.: A classification of schema-based matching approaches. In: *Proceedings of the Meaning Coordination and Negotiation workshop at ISWC'04*. (2004)
- [4] de Bruijn, J., Martin-Recuerda, F., Manov, D., Ehrig, M.: D4.2.1 state-of-the-art-survey on ontology merging and aligning v1. SEKT Project deliverable D4.2.1 (2004)
- [5] Rahm, E., Bernstein, P.A.: A Survey of Approaches to Automatic Schema Matching. *VLDB Journal* **10(4)** (2001)
- [6] Aucouturier, J.J., Pachet, F.: Representing musical genre: A state of the art. *Journal of New Music Research* 2003, Vol. 32, No. 1, pp. 83-93 (2003)
- [7] Pachet, F., Cazaly, D.: A taxonomy of musical genres. In: *RIAO'00: Proc. Content-Based Multimedia Information Access*. (2000)
- [8] Bouquet, P., Serafini, L., Zanobini, S.: *Semantic coordination: a new approach and an application*. Technical report, University of Trento (2003)
- [9] Cilibrasi, R., Vitanyi, P.: The google similarity distance. *IEEE Transactions on knowledge and data engineering* **19(3)** (2007) 370–383
- [10] Noy, N.F., Musen, M.A.: *PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment*. In: *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)*, Austin, TX, AAAI/MIT Press (2000)
- [11] McGuinness, D.L., Fikes, R., Rice, J., Wilder, S.: *The Chimaera Ontology Environment*. In: *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, AAAI Press / The MIT Press (2000) 1123–1124
- [12] Jian, N., Hu, W., Cheng, G., Qu, Y.: *Falcon-ao: Aligning ontologies with falcon*. In: *K-Cap 2005 Workshop on Integrating Ontologies*. (2005)
- [13] Gusfield, D.: *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press; 1st edition (1997)
- [14] Ehrig, M., Sure, Y.: *Foam - framework for ontology alignment and mapping; results of the ontology alignment initiative*. In Ashpole, B., Ehrig, M., Euzenat, J., Stuckenschmidt, H., eds.: *Proceedings of the Workshop on Integrating Ontologies*. Volume 156., CEUR-WS.org (2005) 72–76
- [15] Ichise, R., Takeda, H., Honiden, S.: *Integrating multiple internet directories by instance-based learning*. In: *Proceedings of the eighteenth International Joint Conference on Artificial Intelligence*. (2003)
- [16] Aleksovski, Z., Klein, M., ten Kate, W., van Harmelen, F.: *Matching unstructured vocabularies using a background ontology*. In: *Proceedings of Knowledge Engineering and Knowledge Management (EKAW)*. (2006) 182–197
- [17] Aleksovski, Z., ten Kate, W., van Harmelen, F.: *Exploiting the structure of background knowledge used in ontology matching*. In: *Ontology Matching Workshop at International Semantic Web Conference (ISWC)*. (2006)
- [18] Zhang, S., Bodenreider, O.: *Alignment of multiple ontologies of anatomy: Deriving indirect mappings from direct mappings to a reference*. In: *AMIA Symposium Proceedings*. (2005) 864–868
- [19] Giunchiglia, F., Shvaiko, P., Yatskevich, M.: *S-match: an algorithm and an implementation of semantic matching*. In: *Proceedings of the European Semantic Web Symposium (ESWC)*. (2004) 61–75
- [20] Hearst, M.A.: *Automatic acquisition of hyponyms from large text corpora*. In: *Proceedings of the 14th conference on Computational linguistics*. (1992) 539–545
- [21] Agichtein, Y.E.: *Extracting relations from large text collections*. PhD thesis, Columbia University (2005)
- [22] Crescenzi, V., Mecca, G.: *Automatic information extraction from large websites*. *J. ACM* **51(5)** (2004) 731–779
- [23] Straccia, U.: *A fuzzy description logic for the semantic web*. In Sanchez, E., ed.: *Fuzzy Logic and the Semantic Web. Capturing Intelligence*. Elsevier (2006) 73–90
- [24] Schlobach, S., Klein, M., Peelen, L.: *Description logics with approximate definitions: Precise modeling of vague concepts*. In Veloso, M., ed.: *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI 07, Hyderabad, India* (2007)