Combining Logic and Probabilities for Discovering Mappings between Taxonomies

Rémi Tournaire[1], Jean-Marc Petit[2], Marie-Christine Rousset[1], and Alexandre Termier[1]

[1] University of Grenoble, Laboratory of Informatics of Grenoble UMR 5217, 681, rue de la Passerelle, BP72, 38402 St-Martin d'Hères Cedex, France
Remi.Tournaire@imag.fr
[2] INSA Lyon, LIRIS UMR 5205, 69621 Villeurbanne Cedex, France

Abstract. In this paper, we investigate a principled approach for defining and discovering *probabilistic mappings* between two taxonomies. First, we compare two ways of modeling probabilistic mappings which are compatible with the logical constraints declared in each taxonomy. Then we describe a *generate and test* algorithm which minimizes the number of calls to the probability estimator for determining those mappings whose probability exceeds a certain threshold. Finally, we provide an experimental analysis of this approach.

1 Introduction

The decentralized nature of the development of Web data management systems makes inevitable the independent construction of a large amount of personalized taxonomies used for annotating data and resources at Web scale. Taxonomies are hierarchical structures appropriate for data categorization and semantic annotation of resources. They play a prominent role in the Semantic Web since they are central components of OWL [8] or RDF(S) [19] ontologies. A taxonomy constrains the vocabulary used to express metadata or semantic annotations to be classes that are related by structural relationships. Taxonomies are easy to create and understand by humans while being machine interpretable and processable thanks to a formal logical semantics supporting reasoning capabilities.

In this setting, establishing *semantic mappings* between taxonomies is the key to enable collaborative exchange of semantic data. Manually finding such mappings is clearly not possible at the Web scale. Therefore, the automatic discovery of semantic mappings is the bottleneck for scalability purposes.

Many techniques and prototypes have been developed to suggest candidate mappings between several knowledge representations including taxonomies, ontologies or schemas (see [25,14] for surveys). Most of the existing approaches rely on evaluating the degree of similarity between the elements (e.g., classes, properties, instances) of one ontology and the elements of another ontology. Many different similarity measures are proposed and often combined. Most of them are based on several syntactic, linguistic or structural criteria to measure the proximity of the terms used to denote the classes and/or their properties within the

Y. Bi and M.-A. Williams (Eds.): KSEM 2010, LNAI 6291, pp. 530–542, 2010.
© Springer-Verlag Berlin Heidelberg 2010

ontology. Some of them exploit characteristics of the data declared as instances of the classes (e.g. [12]).

As a result, most of the existing matching systems return for every candidate pair of elements a coefficient in the range [0,1] which denotes the strength of the semantic correspondence between those two elements [15,24,4]. A threshold is then used for keeping as *valid mappings* those pairs of elements for which the coefficient of similarity is greater than the threshold. Since most of the approaches are based on similarity functions that are symmetric, the mappings that are returned with high similarity scores are interpreted as *equivalence mappings*. Few approaches [17,18] handle *inclusion mappings* between classes. Yearly international evaluation campaigns[1] are organized to compare matching systems on different benchmarks, in terms of quality (recall and precision) of the mappings they return. Except until very recently, only equivalence mappings have been considered in the OAEI campaigns.

Our first claim is that *inclusion* mappings between classes of two pre-existing taxonomies are more likely to exist than *equivalence* mappings. When taxonomies are used as query interfaces between users and data, inclusion mappings between taxonomies can be used for query reformulation exactly like the subclass relationship within a taxonomy. For instance, a mapping $Opera \sqsubseteq Vocal$ between the class *Opera* of a taxonomy and the class $Vocal$ of a second taxonomy may be used to find additional answers to a query asking data about $Vocal$ by returning data categorized in the class *Opera* in the first taxonomy.

In contrast with logical approaches (e.g., [17]) for (inclusion) mapping discovery, we also claim that *uncertainty* is intrinsic to mapping discovery. Therefore, we advocate to consider *inclusion mappings with a probabilistic semantics*. Like the similarity scores, the probability coefficients can be compared to a threshold for filtering mappings. In addition, they can be the basis of a probabilistic reasoning and a probabilistic query answering through mapped taxonomies.

It is important to emphasize here that the similarity coefficients returned by most of the existing ontology or schema matching systems cannot be interpreted as *probabilities* of the associated mappings. The reason is that they do not take into account possible logical implications between mappings, which can be inferred from the inclusion axioms declared between classes within each ontology. Interpreting similarities between classes as probabilities of the corresponding mappings requires that the similarity between any subclass of a given class A_1 and any superclass of a given class A_2 is greater than the similarity between A_1 and A_2. Up to our knowledge, this monotony property is not satisfied in any of the existing similarity models.

In this paper, we propose an algorithm for automatic discovery of *probabilistic mappings* between taxonomies, which respects the above monotony property. First, we investigate and compare two ways of modeling probabilistic mappings which are compatible with the logical constraints declared in each taxonomy. In those two probabilistic models, the probability of a mapping relies on the joint probability distribution of the involved classes. They differ on the property of

[1] E.g., OAEI `http://oaei.ontologymatching.org/2009/`

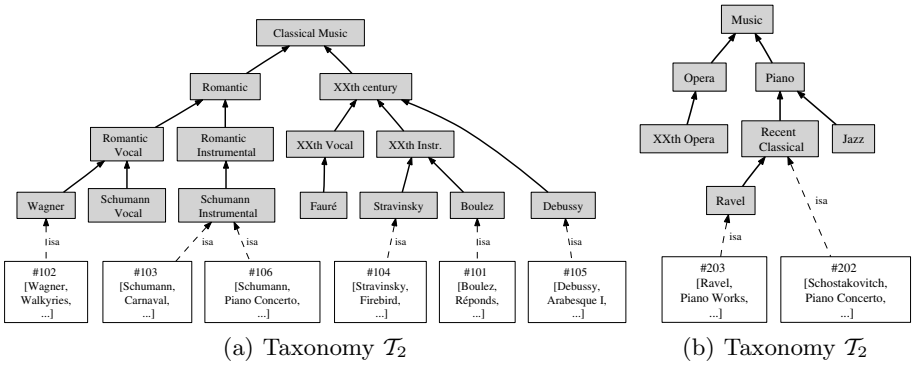
monotony of the corresponding probability function with respect to the logical implication. Based on the above probabilistic setting, we have designed, implemented and experimented a *generate and test* algorithm called ProbaMap for discovering the mappings whose probability is greater than a given threshold. In this algorithm, the monotony of the probability function is exploited for avoiding the probability estimation of as many mappings as possible. The paper is organized as follows. Section 2 presents the formal background and states the problem considered in this paper. Section 3 is dedicated to the definition and computation of mapping probabilities. In Section 4, we present the ProbaMap algorithm which discovers mappings with high probabilities (i.e., greater than a threshold). Section 5 surveys the quantitative and qualitative experiments that we have done. Finally, in Section 6, we compare our approach to existing works and we conclude.

2 Formal Background

We first define taxonomies as a graphical notation and its interpretation in standard first-order-logic semantics, on which the inheritance of instances is grounded. Then, we define *mappings* between taxonomies as inclusion statements between classes of two different taxonomies. Finally, we set the problem statement of matching taxonomies that we consider in this paper.

Taxonomies: classes and instances
Given a vocabulary $\mathcal{V}$ denoting a set of classes, a *taxonomy* $\mathcal{T}_\mathcal{V}$ is a Directed Acyclic Graph (DAG) where each node is labelled with a distinct *class* name of $\mathcal{V}$, and each arc between a node labelled with C and a node labelled by D represents a *specialization relation* between the classes C and D.

Each class in a taxonomy can be associated with a set of *instances* which have an *identifier* and a content *description*. In the following, we will abusively speak of the instance i to refer to the instance identified by i. Figure 1 shows



(a) Taxonomy $\mathcal{T}_2$ (b) Taxonomy $\mathcal{T}_2$

Fig. 1. 2 Taxonomies and associated instances

two samples of taxonomies related to the Music domain. Bold arrows are used for representing specialization relations between classes, and dashed arrows for membership relation between instances and classes. In both taxonomies, some instances, with description denoted between brackets, are associated to classes. For example, #102 is an instance identifier and [Wagner, Walkyries, ...] its associated description.

The instances that are in the scope of our data model can be web pages (which content description is a set of words) identified by their URLs, RDF resources (which content description is a set of RDF triples) identified by URIs, or audio or video files identified by a signature and whose content description may be attribute-value metadata that can be extracted from those files. Taxonomies have a logical semantics which provides the basis to define formally the extension of a class as the set of instances that are declared or can be *inferred* for that class.

Logical semantics

There are several graphical or textual notations for expressing the specialization relation between a class C and a class D in a taxonomy. For example, in RDF(S) [19] which is the first standard of the W3C concerning the Semantic Web, it is denoted by (C *rdfs:subclassOf* D). It corresponds to the inclusion statement $C \sqsubseteq D$ in the description logics notation.

Similarly, a membership statement denoted by an *isa* arc from an instance i to a class C corresponds in the RDF(S) notation to (i *rdf:type* C), and to $C(i)$ in the usual notation of description logics.

All those notations have a standard model-theoretic logical semantics based on interpreting classes as sets: an *interpretation* $\mathcal{I}$ consists of a non empty domain of interpretation $\Delta^{\mathcal{I}}$ and a function $.^{\mathcal{I}}$ that interprets each class as a non empty subset of $\Delta^{\mathcal{I}}$, and each instance identifier as an element of $\Delta^{\mathcal{I}}$. The classes declared in a taxonomy are interpreted as non empty subsets because they are object containers. According to the *unique name assumption*, two distinct identifiers a and b verify ($a^{\mathcal{I}} \neq b^{\mathcal{I}}$) in any interpretation $\mathcal{I}$.

$\mathcal{I}$ is a *model* of a taxonomy $\mathcal{T}$ if:

- for every inclusion statement $E \sqsubseteq F$ of $\mathcal{T}$: $E^{\mathcal{I}} \subseteq F^{\mathcal{I}}$,
- for every membership statement $C(a)$ of $\mathcal{T}$: $a^{\mathcal{I}} \in C^{\mathcal{I}}$.

An inclusion $G \sqsubseteq H$ is *inferred* by a taxonomy $\mathcal{T}$ (denoted by $\mathcal{T} \models G \sqsubseteq H$) iff in every model $\mathcal{I}$ of $\mathcal{T}$, $G^{\mathcal{I}} \subseteq H^{\mathcal{I}}$. A membership $C(e)$ is *inferred* by $\mathcal{T}$ (denoted by $\mathcal{T} \models C(e)$) iff in every model $\mathcal{I}$ of $\mathcal{T}$, $e^{\mathcal{I}} \in C^{\mathcal{I}}$.

Let $\mathcal{D}$ be the set of the instances associated with a taxonomy $\mathcal{T}$. The *extension* of a class C in $\mathcal{T}$, denoted by $Ext(C, \mathcal{T})$, is the set of instances for which it can be inferred from the membership and inclusion statements declared in the taxonomy that they are instances of C: $Ext(C, \mathcal{T}) = \{d \in \mathcal{D} / \ \mathcal{T} \models C(d)\}$.

Mappings

The mappings that we consider are inclusion statements involving classes of two different taxonomies $\mathcal{T}_1$ and $\mathcal{T}_2$. To avoid ambiguity and without loss of generality, we consider that each taxonomy has its own vocabulary: by convention

we index the names of the classes by the index of the taxonomy to which they belong. Mappings between $\mathcal{T}_1$ and $\mathcal{T}_2$ are consequently of the form $A_1 \sqsubseteq B_2$ or $A_2 \sqsubseteq B_1$. For a mapping m of the form $A_i \sqsubseteq B_j$, its left-hand side A_i will be denoted $lhs(m)$ and its right-hand side will be denoted $rhs(m)$.

A mapping $A_i \sqsubseteq B_j$ has the same meaning as a specialization relation between the classes A_i and B_j, and thus is interpreted in logic in the same way, as a set inclusion. The logical entailment between classes extends to logical entailment between mappings as follows.

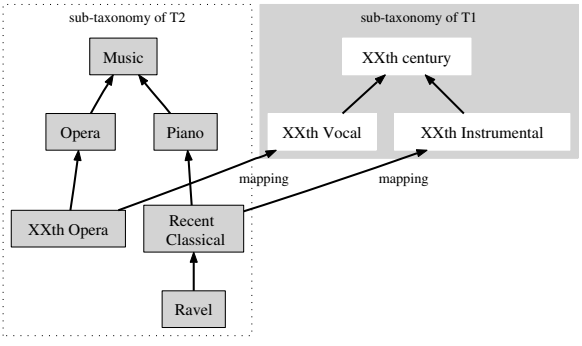


Fig. 2. 2 mappings between $\mathcal{T}_1$ and $\mathcal{T}_2$

Definition 1 (Entailment between mappings). *Let $\mathcal{T}_i$ and $\mathcal{T}_j$ be two taxonomies. Let m and m' be two mappings between $\mathcal{T}_i$ and $\mathcal{T}_j$: m entails m' (denoted $m \preceq m'$) iff every model of $\mathcal{T}_i$, $\mathcal{T}_j$ and m is also a model of m'.*

It is straightforward to show that $\preceq$ is a (partial) order relation on the set of mappings between the two taxonomies $\mathcal{T}_i$ and $\mathcal{T}_j$. If $m \preceq m'$, we will say that m is more specific than m' (also that m is an implicant of m') and that m' is more general than m (also that m' is an implicate of m).

The following proposition characterizes the logical entailment between mappings in function of the logical entailment between the classes of their left hand sides and right hand sides.

Proposition 1. *Let m and m' be two mappings between two taxonomies. Let $\mathcal{T}_i$ be the taxonomy of $lhs(m)$, $\mathcal{T}_j$ the taxonomy of $rhs(m)$.*
$m \preceq m'$ iff
- $lhs(m)$ and $lhs(m')$ are classes of the same taxonomy $\mathcal{T}_i$, and
- $\mathcal{T}_i \models lhs(m') \sqsubseteq lhs(m)$ and $\mathcal{T}_j \models rhs(m) \sqsubseteq rhs(m')$

For example, two mappings between taxonomies $\mathcal{T}_1$ and $\mathcal{T}_2$ of Figure 1 are illustrated in Figure 2. The mapping $XXth\ Opera_2 \sqsubseteq XXth\ Vocal_1$ is more specific than the mapping $XXth\ Opera_2 \sqsubseteq XXth\ Century_1$, and the mapping $RecentClassical_2 \sqsubseteq XXth\ Instrumental_1$ is more specific than the mapping $Ravel_2 \sqsubseteq XXth\ Century_1$.

3 Mapping Probabilities: Models and Estimation

We consider two probabilistic models for modeling uncertain mappings. They are both based on the discrete probability measure defined on subsets of the sample set representing the set of all possible instances of the two taxonomies. From now on, we will denote $Pr(E)$ the probability for an instance to be an element of the subset E.

The first model defines the probability of a mapping $A_i \sqsubseteq B_j$ as the conditional probability for an instance to be an instance of B_j knowing that it is an instance of A_i. It is the natural way to extend the logical semantics of entailment to probabilities.

The second model comes directly from viewing classes as subsets of the sample space: the probability of $A_i \sqsubseteq B_j$ is the probability for an element to belong to the set $\overline{A_i} \cup B_j$, where $\overline{A_i}$ denotes the complement set of A_i in the sample set. Both models are described below.

Definition 2 (Two probabilities for a mapping). *Let m be a mapping of the form $A_i \sqsubseteq B_j$.*

-Its conditional probability*, denoted $P_c(m)$, is defined as $P_c(m) = Pr(B_j|A_i)$.*
-Its union_set probability*, denoted $P_u(m)$, is defined as $P_u(m) = Pr(\overline{A_i} \cup B_j)$.*

Proposition 2 states the main (comparative) properties of those two probabilistic models. They both meet the logical semantics for mappings that are certain, and they can both be expressed using joint probabilities.

Proposition 2. *Let m be a mapping between two taxonomies $\mathcal{T}_i$ and $\mathcal{T}_j$. The following properties hold:*

1. $P_u(m) \geq P_c(m)$.
2. *If m is a* certain mapping*, $P_c(m) = P_u(m) = 1$*
3. $P_u(m) = 1 + Pr(lhs(m) \cap rhs(m)) - Pr(lhs(m))$
4. $P_c(m) = \frac{Pr(lhs(m) \cap rhs(m))}{Pr(lhs(m))}$

They differ on the monotony property w.r.t the (partial) order $\preceq$ corresponding to logical implication (cf. Definition 1): P_u is monotonous whereas P_c verifies a property of *weak* monotony only:

Theorem 1 (Property of monotony). *Let m and m' two mappings.*

1. *If $m \preceq m'$ then $P_u(m) \leq P_u(m')$*
2. *If $m \preceq m'$ and $lhs(m) = lhs(m')$, $P_c(m) \leq P_c(m')$*

The proof results from Proposition 1 and Proposition 2 which relate mappings with the classes of their left hand sides and right hand sides for logical entailment and probabilities respectively, and from considering (declared or inherited) inclusions of classes within each taxonomy as statements whose probability is equal to 1.

As shown in Proposition 2, the computation of $P_u(m)$ and $P_c(m)$ relies on computing the set probability $Pr(lhs(m))$ and the joint set probability $Pr(lhs(m)$

$\cap\, rhs(m)$). Those values are unknown and must be estimated. For doing so, we follow the Bayesian approach to statistics [9]: we model those (unknown) parameters as continuous random variables, and we use *observations* to infer their *posterior* distribution from their *prior* distribution. This is summarized in Definition 3.

Definition 3 (Bayesian estimator of $Pr(E)$). *Let E be a subset of the sample set Ω. Let $\mathcal{O}$ be a sample of observed elements for which it is known whether they belong or not to E. The Bayesian estimator of $Pr(E)$, denoted $\widehat{Pr}(E)$, is the* expected value *of the posterior distribution of $Pr(E)$ knowing the observations on the membership to E of each element in $\mathcal{O}$, and setting the prior probability of a random set to $\frac{1}{2}$, and of the intersection of two random sets to $\frac{1}{4}$.*

Setting the prior probabilities to $\frac{1}{2}$ and $\frac{1}{4}$ depending on whether E is a class or a conjunction of classes corresponds to the uniform distribution of instances among the classes. The following theorem provides a simple way to compute the Bayesian estimations $\widehat{P_u}(m)$ and $\widehat{P_c}(m)$ of the two probabilities $P_u(m)$ and $P_c(m)$ defined in Definition 2. It is a straightforward consequence of a basic theorem in probability theory (Theorem 1, page 160, [9]), stating that if the prior distribution of the random variable modeling $Pr(E)$ is a *Beta distribution* of parameters α and β, then its posterior distribution is also a Beta distribution the parameters of which are: $\alpha + |Ext(E, \mathcal{O})|$ and $\beta + |\mathcal{O}|$, where $Ext(E, \mathcal{O})$ is the set of observed instances of $\mathcal{O}$ that are recognized to belongs to E.

Theorem 2 (Estimation of probabilities)
Let $m : C_i \sqsubseteq D_j$ be a mapping between two taxonomies $\mathcal{T}_i$ and $\mathcal{T}_j$. Let $\mathcal{O}$ be the union of instances observed in $\mathcal{T}_i$ and $\mathcal{T}_j$. Let $N = |\mathcal{O}|$, $N_i = |Ext(C_i, \mathcal{O})|$, $N_j = |Ext(D_j, \mathcal{O})|$ and $N_{ij} = |Ext(C_i \cap D_j, \mathcal{O})|$.

$$\widehat{P_u}(m) = 1 + \frac{1+N_{ij}}{4+N} - \frac{1+N_i}{2+N} \qquad \widehat{P_c}(m) = \frac{1+N_{ij}}{4+N} \times \frac{2+N}{1+N_i}$$

Depending on the way the taxonomies are populated (manually or automatically), it is not always possible to obtain N_{ij} simply by counting the instances that are common to the two classes involved in the mapping. If the taxonomies are populated manually and independently by different users, it is indeed likeky that the intersection of the two taxonomies contains very few instances or even no instance at all. In that case, we apply existing *automatic classifiers* (e.g., Naive Bayes learning, decision trees, SVM) in order to compute $Ext(C_i \cap D_j, \mathcal{O})$, by following the same approach as [12] for training them on the description of the available instances in each taxonomy.

4 The ProbaMap Algorithm

Given two taxonomies $\mathcal{T}_i$ and $\mathcal{T}_j$ (and their associated instances), let $\mathcal{M}(\mathcal{T}_i, \mathcal{T}_j)$ be the set of all mappings from $\mathcal{T}_i$ to $\mathcal{T}_j$ (i.e., of the form $C_i \sqsubseteq D_j$). The ProbaMap algorithm determines all mappings m of $\mathcal{M}(\mathcal{T}_i, \mathcal{T}_j)$ verifying a probabilistic-based criterion of validity that will be denoted by $\widehat{P}(m) \geq S$.

$\widehat{P}(m) \geq S$ is a parameter in the algorithm, which can be one of the three following validity criteria, where S_u and S_c are two thresholds in $[0, 1]$:

- *Validity criterion 1*: $\widehat{P_u}(m) \geq S_u$
- *Validity criterion 2*: $\widehat{P_c}(m) \geq S_c$
- *Validity criterion 3*: $\widehat{P_c}(m) \geq S_c$ and $\widehat{P_u}(m) \geq S_u$.

Candidate mapping generation

The principle of ProbaMap algorithm is to generate mappings from the two sets of classes in the two taxonomies ordered according to a *topological sort* [6]. Namely, the nested loops (Line 2) in Algorithm 1 generate all the mappings $C_i \sqsubseteq D_j$ by enumerating the classes C_i of $\mathcal{T}_i$ following a *reverse* topological order and the classes D_j of $\mathcal{T}_j$ following a *direct* topological order. The following proposition is a corollary of Proposition 1.

Proposition 3. *Let $\mathcal{T}_i$ and $\mathcal{T}_j$ two taxonomies.*
Let $ReverseTopo(\mathcal{T}_i)$ be the sequence of classes of $\mathcal{T}_i$ resulting from a reverse topological sort of $\mathcal{T}_i$. Let $Topo(\mathcal{T}_j)$ be the sequence of classes of $\mathcal{T}_j$ resulting from a topological sort of $\mathcal{T}_i$. Let $m : C_i \sqsubseteq D_j$ and $m' : C'_i \sqsubseteq D'_j$ two mappings from $\mathcal{T}_i$ to $\mathcal{T}_j$. If m' is an implicant of m (i.e., $m' \preceq m$), then C_i is before C'_i in $ReverseTopo(\mathcal{T}_i)$ or $C_i = C'_i$ and D_j is before D'_j in $Topo(\mathcal{T}_j)$.

Pruning the candidate mappings to test

Based on the monotony property of the probability function P_u (Theorem 1), every mapping m' implicant of a mapping m such that $P_u(m) < S_u$ verifies $P_u(m') < S_u$. Therefore, in ProbaMap, if the validity criterion involves $\widehat{P_u}$, we prune the probability estimation of all the implicants of every m such that $\widehat{P_u}(m) < S_u$. We shall use the notation $Implicants(m)$ to denote the set of all mappings that are implicants of m. Similarly, based on the property of weak monotony of the probability function P_c (Theorem 1), if the validity criterion involves $\widehat{P_c}$, when a tested candidate mapping m is such that $\widehat{P_c}(m) < S_c$ we prune the probability estimation of all the implicants of m having the same left-hand side as m. We shall denote this set: $Implicants_c(m)$. Based on Proposition 1, $Implicants(m)$ and $Implicants_c(m)$ can be generated from $\mathcal{T}_i$ and $\mathcal{T}_j$.

Based on the order in which the mappings are generated, Proposition 3 shows that the validity test in Line 5 of the algorithm 1 maximizes the number of pruning. The resulting ProbaMap algorithm is described in Algorithm 1, in which:

- $\widehat{P}(m) \geq S$ in Line 6 denotes a generic validity criterion that can be instantiated either by $\widehat{P_u} \geq S_u$, or by $\widehat{P_c} \geq S_c$, or by ($\widehat{P_c} \geq S_c$ and $\widehat{P_u} \geq S_u$).
- In the case where the validity criteria involves $\widehat{P_c}$, $Implicants(m)$ in Line 9 must be replaced by $Implicants_c(m)$.
- In Line 4, $ReverseTopo_i$ and $Topo_j$ denote the respective sequences $ReverseTopo(\mathcal{T}_i)$ and $Topo(\mathcal{T}_j)$. $ReverseTopo_i[k]$ (resp. $Topo_j[l]$) denotes the class of $\mathcal{T}_i$ (resp. $\mathcal{T}_j$) ranked k (resp. l) in the sequence.

Algorithm 1 returns mappings directed from $\mathcal{T}_i$ to $\mathcal{T}_j$. In order to obtain *all* valid mappings, it must be applied again by swapping its inputs $\mathcal{T}_i$ and $\mathcal{T}_j$.

Algorithm 1. ProbaMap

Require: $\mathcal{T}_i, \mathcal{T}_j$,threshold S
Ensure: return $\{m \in \mathcal{M}(\mathcal{T}_i, \mathcal{T}_j) / \widehat{P}(m) \geq S\}$
$M_{Val} \leftarrow \emptyset$, $M_{NVal} \leftarrow \emptyset$
for $k = 1$ to $|\mathcal{T}_i|$ **do**
  for $l = 1$ to $|\mathcal{T}_j|$ **do**
    let $m = ReverseTopo_i[k] \sqsubseteq Topo_j[l]$
    if $m \notin M_{NVal}$ **then**
      if $\widehat{P}(m) \geq S$ **then**
        $M_{Val} \leftarrow M_{Val} \cup \{m\}$
      else
        $M_{NVal} \leftarrow M_{NVal} \cup Implicants(m)$
return M_{Val}

5 Experiments

In this section, we evaluate the quantitative and qualitative performances of ProbaMab (Algorithm 1) on large real-world taxonomies populated with instances. We focus our experiments on the Internet directories[2] from Yahoo! and Google (actually corresponding to Dmoz). This allows us to compare our approach to the SBI algorithm of Ichise et al. [20,21], which is dedicated to the discovery of mappings between Internet directories. Internet directories are huge trees of categories, which can be seen as taxonomies, categories being the classes. Each category contains a set of links (i.e. URLs to web sites), which can be seen as the instances of the class. Each link comes with a small text summary, whose words can be seen as instance attributes for classification.

Our datasets are corresponding locations in the Yahoo! and Google directories, that have also been used in the experiments of [20,21]:

- Yahoo! : Recreation / Automotive & Google : Recreation / Autos
- Yahoo! : Computers_and_Internet/Software & Google : Computers/Software
- Yahoo! : Arts / Visual_Arts / Photography & Google : Arts / Photography

The data from the directories was collected in the beginning of 2010, so is slightly different from the data of [21] and [20] which was collected in Fall 2001.

Table 1 shows for each dataset the number of classes and instances in each class, and the number of instances shared between the Yahoo! and the Google directories. Two instances are considered shared if they correspond to the same URL in both directories. For a fair comparison, we have implemented both ProbaMap and the SBI algorithm in Java.

The goal of our experiments is to compare the quality of Internet directories alignement for ProbaMap and SBI.

During the learning step, ProbaMap and SBI receive as training set a subset of the shared instances with their correct category in each of the directories. The test set is the remaining of the shared instances. When adding classification to

[2] `dir.yahoo.com,www.dmoz.org`

Table 1. Statistics on data collected from subdirectories on Yahoo! and Google

	Yahoo!		Google		shared
	classes	instances	classes	instances	instances
Autos	947	4406	967	6425	837
Software	323	2390	2395	30140	572
Photography	168	1851	321	3852	286

ProbaMap, the training set is extended with all the non shared instances. The classification is performed using the SVM implementation SMO[16] in Weka [27], where the classification attributes for an instance are the words of its summary.

We computed the accuracy (standard criteria used in [21]) of mapping prediction on the test set, and conducted a ten-fold cross validation. The results when varying the size of the training set are shown in Figure 3 for the Autos dataset.

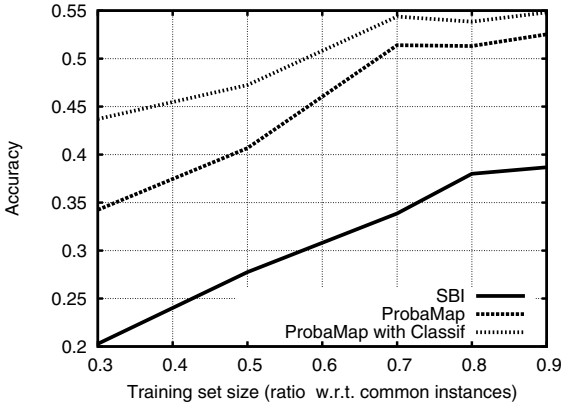


Fig. 3. Accuracy for alignment of Autos subdirectories (Yahoo to Google)

ProbaMap with or without classification significantly outperforms SBI, with a 15% better accuracy for ProbaMap alone and 15-25% better accuracy for ProbaMap with classification. Note that when using classification, the accuracy of ProbaMap with the smallest training set size (30% of shared instances) is better than the accuracy of SBI with a training set containing 90% of shared instances. We obtain similar results (and coherent with those in [21]) on Software and Photography subdirectories, with the same order for the three methods.

These results show that the probabilistic mapping discovery method that we propose gives excellent results on real-world datasets, and can take advantage of classification techniques to compensate small training set sizes. This is an important quality for real world taxonomies built by different people, that are unlikely to have many instances in common.

Thanks to the monotony property of our probabilistic mapping model (see Section 4), ProbaMap can prune large parts of the mapping search space and handle the alignement of very large taxonomies. We successfully conducted

scalability experiments on very large taxonomies from the OAEI[3] contest (directory dataset). We have compensated the lack of instances available for those taxonomies by automatically populating the classes with WordNet synsets. The taxonomies to align have 6628 and 2857 classes, leading to more than 15 millions potential mappings. Up to our knowledge, very few OAEI participants have taken the whole directory dataset as input, but instead a splitted version of it into branches which was provided by the OAEI organizers.

6 Related Work and Conclusion

As outlined in the introduction, semantic mappings are the glue for data integration systems. A wide range of methods of schema/ontology matching have been developed both in the database and the semantic web communities [14]. One of the principles widely exploited is terminological comparison of the labels of classes with string-based similarities or lexicon-based similarities (like WordNet) (e.g., TaxoMap [18], H-MATCH [4]) . Another widely used principle is structure comparison between labeled graphs representing ontologies (e.g., OLA [15]). In fact, most of the existing matchers combine these two approaches in different ways (e.g., COMA++ [1] and COMA [10], Cupid [24], H-MATCH [4]). Other approaches have been investigated with machine learning techniques using a corpus of schema matches (e.g., [23]), or a corpus of labelled instances (e.g., LSD [11], SemInt [22], GLUE [12], FCA-merge [26], SBI-NB[21]). SBI[20] computes the degree of agreement of each couple of classes based on instances statistics, but without machine learning. It is standard practice for ontology and schema matchers to associate numbers with the candidate mappings they propose. However, those numbers do not have a probabilistic meaning and are just used for ranking. In contrast, our approach promotes a probabilistic semantics for mappings and provides a method to compute mapping probabilities based on the descriptions of instances from in each ontology. It is important to note that even if we use similar classification techniques as [12], we use them for computing true probabilities and not similarity coefficients.

The most distinguishing feature of our approach is that it bridges the gap between logic and probabilities by providing probabilistic models that are consistent with the logical semantics underlying ontology languages. Therefore, our approach generalizes existing works based on algebraic or logical representation of mappings as a basis for reasoning (e.g., S-Match [17], Clio [5]). The mappings returned by ProbaMap can be exploited for mapping validation by probabilistic reasoning in the line of what is proposed in [3]. More generally, our approach is complementary of the recent work that has been flourishing on probabilistic databases [2,7]. It fits into the general framework set in [13] for handling uncertainty in data integration, for which it provides an effective way for computing mapping probabilities.

The experiments that we have conducted on both real-world and controlled data have shown the feasibility and the scalability of our approach.

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

References

1. Aumueller, D., Do, H.H., Massmann, S., Rahm, E.: Schema and ontology matching with COMA++. In: SIGMOD 2005, ACM, New York (2005)
2. Benjelloun, O., Sarma, A.D., Halevy, A.Y., Widom, J.: ULDBs: Databases with uncertainty and lineage. In: VLDB (2006)
3. Castano, S., Ferrara, A., Lorusso, D., Näth, T.H., Möller, R.: Mapping validation by probabilistic reasoning. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 170–184. Springer, Heidelberg (2008)
4. Castano, S., Ferrara, A., Montanelli, S.: H-MATCH: an algorithm for dynamically matching ontologies in peer-based systems. In: SWDB (2003)
5. Chiticariu, L., Hernández, M.A., Kolaitis, P.G., Popa, L.: Semi-automatic schema integration in clio. In: VLDB (2007)
6. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 2nd edn. The MIT Press, Cambridge (2001)
7. Dalvi, N.N., Suciu, D.: Answering queries from statistics and probabilistic views. In: VLDB (2005)
8. Dean, M., Schreiber, G.: OWL web ontology language reference. W3C recommendation, W3C (2004)
9. Degroot, M.H.: Optimal Statistical Decision. Wiley Classics Library (2004)
10. Do, H., Rahm, E.: COMA - a system for flexible combination of schema matching approaches. In: VLDB (2002)
11. Doan, A., Domingos, P., Levy, A.Y.: Learning mappings between data schemas. In: Proceedings of the AAAI 2000 Workshop on Learning Statistical Models from Relational Data (2000)
12. Doan, A., Madhavan, J., Domingos, P., Halevy, A.Y.: Learning to map between ontologies on the semantic web. In: WWW (2002)
13. Dong, X.L., Halevy, A.Y., Yu, C.: Data integration with uncertainty. VLDB Journal (2007)
14. Euzenat, J., Shvaiko, P.: Ontology matching. Springer, Heidelberg (2007)
15. Euzenat, J., Valtchev, P.: Similarity-based ontology alignment in OWL-lite. In: ECAI (2004)
16. Flake, G.W., Lawrence, S.: Efficient SVM regression training with SMO. Machine Learning (2002)
17. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: S-Match: an algorithm and an implementation of semantic matching. In: Bussler, C.J., Davies, J., Fensel, D., Studer, R. (eds.) ESWS 2004. LNCS, vol. 3053, pp. 61–75. Springer, Heidelberg (2004)
18. Hamdi, F., Zargayouna, H., Safar, B., Reynaud, C.: TaxoMap in the OAEI 2008 alignment contest. In: OAEI 2008 Campaign - Int. Workshop on Ontology Matching (2008)
19. Hayes, P. (ed.): RDF Semantics. World Wide Web Consortium (2004)
20. Ichise, R., Takeda, H., Honiden, S.: Integrating multiple internet directories by instance-based learning. In: IJCAI, vol. 18 (2003)
21. Ichise, R., Hamasaki, M., Takeda, H.: Discovering relationships among catalogs. In: Suzuki, E., Arikawa, S. (eds.) DS 2004. LNCS (LNAI), vol. 3245. Springer, Heidelberg (2004)
22. Li, W.S., Clifton, C.: Semint: a tool for identifying attribute correspondences in heterogeneous databases using neural networks. Data Knowl. Eng. 33(1) (2000)

23. Madhavan, J., Bernstein, P.A., Doan, A., Halevy, A.: Corpus-based schema matching. In: International Conference on Data Engineering (2005)
24. Madhavan, J., Bernstein, P.A., Rahm, E.: Generic schema matching with cupid. VLDB Journal (2001)
25. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB Journal (2001)
26. Stumme, G., Maedche, A.: FCA-MERGE: Bottom-Up Merging of Ontologies. In: Proc. of the 17th International Joint Conference on Artificial Intelligence (2001)
27. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)