Mapping Validation by Probabilistic Reasoning *

Silvana Castano[1], Alfio Ferrara[1], Davide Lorusso[1],
Tobias Henrik Näth[2], Ralf Möller[2]

[1] Università degli Studi di Milano,
DICo, 10235 Milano, Italy,
{castano, ferrara, lorusso}@dico.unimi.it
[2] Hamburg University of Technology,
Institute of Software Systems,
21079 Hamburg, Germany,
{r.f.moeller, tobias.naeth}@tu-harburg.de

Abstract. In the semantic web environment, where several independent ontologies are used in order to describe knowledge and data, ontologies have to be aligned by defining mappings among the elements of one ontology and the elements of another ontology. Very often, mappings are not derived from the semantics of the ontologies that are compared. Rather, mappings are computed by evaluating the similarity of the ontologies terminology and/or of their syntactic structure. In this paper, we propose a new mapping validation approach. The approach is based on the notion of probabilistic mappings and on the use of probabilistic reasoning techniques to enforce a semantic interpretation of similarity mappings as probabilistic and hypothetical relations among ontology elements.

1 Introduction

In the semantic web environment, as well as in any information system where two or more independent ontologies are used in order to describe knowledge and data, some kind of ontology interoperability is needed. It can be realized either by integrating the different ontologies into a new one or by aligning them with a mechanism for translating a query over an ontology A into a query over another ontology B. In both cases, two (or more) ontologies have to be aligned by defining correspondences among the elements (e.g., concepts, properties, instances) of one ontology and the elements of another ontology. Such correspondences are called *mappings*. Usually a mapping is featured by a relation holding between the two elements mapped and a measure in the range [0,1], which denotes the strength of the relation between the two elements. In literature, many ontology matching approaches and systems have been proposed for automatically discovering ontology mappings [1]. In most of the proposed approaches, mappings are not derived from the semantics of the ontologies that are compared, but, rather, from an evaluation of the similarity of the terminology used in the two ontologies

* This paper has been partially funded by the BOEMIE Project, FP6-027538, 6th EU Framework Programme.

or of their syntactic structure. The reason is that mapping discovery is not really a deductive problem, if the two ontologies are really independent. In fact, the information about the correspondence of an element of the first ontology with an element of the second one is not implicitly contained in any of the two. However, mappings derived from linguistic/syntactic approaches cannot be interpreted as semantic relations among the ontology elements as it should be in order to use them for query processing. On the other hand, some approaches for detecting mappings with the support of logical reasoning have been proposed [2]. In this case, a set of semantic relations is derived from mappings, but all the relations are interpreted as ontology axioms. However, a certain level of uncertainty is intrinsically due to the mechanism used for mapping discovery. In fact, ontology mapping tools derive such mappings from uncertain sources or techniques (e.g., string matching, graph matching) or they rely on external lexical sources (e.g., WordNet) that are not specifically tailored for the domain at hand. This uncertainty leads to mapping inaccuracy. A way to overcome inaccuracy is to manually validate each mapping not only per se (i.e., to state that the mapping is correct with respect to the meaning of the involved elements), but also with respect to all the other mappings in the set. Of course, this activity is time consuming and it would however be affected by a subjectivity component, as it is performed by human experts. In all these cases, a post-processing activity on ontology mappings is required in order to validate mappings with respect to the semantics of the ontologies involved and, at the same time, by maintaining the uncertain nature of mappings. In this paper, we propose a new mapping validation approach for interpreting similarity-based mappings as semantic relations, by coping also with inaccuracy situations. The idea is to see two independent ontologies as a unique distributed knowledge base and to assume a semantic interpretation of ontology mappings as probabilistic and hypothetical relations among ontology elements. We present and use a probabilistic reasoning tool in order to validate mappings and to possibly infer new relations among the ontologies.

The paper is organized as follows: in Section 2, we present our approach and we show an example of mapping acquisition. In Section 3, we present the P-$\mathcal{SHIQ}$(D) formalism and we show how mappings can be represented as probabilistic knowledge. In Section 4, we discuss probabilistic reasoning and its use for mapping validation. In Section 5, we discuss some related work concerning our approach. Finally, in Section 6, we give our concluding remarks.

2 Ontology Mappings

Our approach to mapping validation is articulated in four phases.

1. *Ontology mapping acquisition.* In this phase, we acquire mappings produced using an ontology matching system; the matching system can rely on syntactic, linguistic, structural, or even semantic matching techniques.
2. *Probabilistic interpretation of mappings.* In this phase, acquired mappings are interpreted as probabilistic constraints holding among the elements of the

two ontologies; in other words, the mapping is associated with a probability that the relation expressed by the mapping is true.

3. *Probabilistic reasoning over mappings.* In this phase, the merged ontology obtained by combining the two initial ontologies using the semantic relations derived from probabilistic mappings is validated by means of a probabilistic reasoning system.
4. *Mapping validation and revision.* In this phase, mappings are revised according to the reasoning results; mappings causing incoherences within the merged ontology are discarded while new mappings are inferred from the valid mappings.

In this paper, the main contribution regards the representation of mappings as probabilistic knowledge and the reasoning-driven mapping validation (phases 2, 3, and 4), which are addressed in more detail in Sections 3, and 4, respectively. In the remainder of this section, we first introduce the notion of probabilistic mapping and then we describe an example of acquisition of linguistic mappings generated by our ontology matching system HMatch 2.0 [3].

2.1 Probabilistic mappings

Given two ontologies O_1 and O_2, a mapping set $\mathcal{M}_{O_1}^{O_2}$ is a collection of mappings $m_{i,j,r}$ of the form:

$$m_{i,j,r} = \langle r(i,j), v \rangle$$

where $i \in O_1$ and $j \in O_2$ are two elements of O_1 and O_2, respectively, r is a binary relation holding between i and j, and $0 \leq v \leq 1$ is a value denoting the strength of the relation r. Such kind of mappings can be automatically defined by a wide range of available techniques and by using one of the ontology matchmaking tools proposed in literature. Goal of the mappings is to define which is the kind of semantic relation which holds between an element of O_1 and an element of O_2. The relation r associated with a mapping depends on the matching technique used. In most cases, it denotes a generic correspondence between two elements (e.g., similarity), while, in other cases, it represents a terminological relationship between the names of i and j (e.g., synonymy, hypenymy, hyponymy). The value v usually denotes the strength of the belief in the proposition $r(i,j)$. In order to take into account the intrinsic inaccurate nature of a mapping, we define a probabilistic mapping as follows.

Definition 1. *Probabilistic Mapping. Given two ontology elements i and j, a probabilistic mapping $pm_{i,j,r}$ between i and j is a mapping $m_{i,j,r}$ whose value v is the probability that the mapping relation $r(i,j)$ is true, that is:*

$$pm_{i,j,r} = \langle r(i,j), v \rangle, \textit{such that } Pr(r(i,j)) = v$$

In defining probabilistic mappings out of generic mappings, we introduce the hypothesis that the strength of the relation holding between two mapped elements can be expressed in terms of probabilities. In particular, we want to measure the probability of the assertion $r(i,j)$ to be true.

2.2 Mapping acquisition

In mapping acquisition, we work under the assumption that the mechanism used for ontology matching is compatible with a probabilistic interpretation of the resulting mappings. In this paper, we show how this can be achieved using a linguistic matching technique based on the WordNet lexical system. In particular, we rely on the linguistic matching technique of HMatch 2.0, which determines the linguistic mapping between two ontology elements (i.e., concepts, properties, individuals) on the basis of their names and of the terminological relationships holding between in WordNet. The mapping degree $v \in [0, 1]$ is calculated through a Linguistic Affinity function (LA), which returns a value proportional to the probability that a terminological relationship holds between two terms. Given two elements i and j of two ontologies O_1 and O_2, the interaction between HMatch 2.0 and the lexical system WordNet produces a set of linguistic mappings of the form:

$$m_{i,j,r} = \langle r(T_i, T_j), v \rangle$$

where:

- T_i and T_j are terms used as names for i and j, respectively.
- $r \in \{\mathtt{SYN}, \mathtt{BT}, \mathtt{NT}\}$ is a terminological relationship defined in WordNet for T_i and T_j, where SYN denotes synonymy, BT denotes hypernymy, and NT denotes hyponymy.
- v is the mapping value in the range [0,1] associated with r.

The LA function works on the synsets of WordNet. In WordNet, each possible meaning of a term is represented by a set of synonyms, called *synset*. Given two terms T_i and T_j, we search in WordNet the set R of terminological relationships holding between at least one synset of T_i and at least one synset of T_j. When linguistic mappings are interpreted as probabilistic mappings, many terms can have more than one meaning and we do not know which specific meaning has been used in the ontologies to be compared. If we find a terminological relationship $r_i(s_l, s_k) \in R$ between two synsets s_l of T_i and s_k of T_j, the probability $Pr(r_i(s_l, s_k))$ is equal to $Pr(Pr(s_l) \wedge Pr(s_k))$, where $Pr(s_l)$ is the probability that the intended meaning of T_i in its ontology is expressed by the synset s_l, while $Pr(s_k)$ is the probability that the intended meaning of T_j in its ontology is expressed by the synset s_k, respectively. In general, given a term T, the probability $Pr(s_t)$ that s_t is the synset which denotes the intended meaning of T depends on the total number N_T of synsets for T in WordNet, such that $Pr(s_t) = \frac{1}{N_T}$. Thus, the probability $Pr(r_i(s_l, s_k))$ of the terminological relationship r_i to be true is defined as:

$$Pr(r_i(s_l, s_k)) = Pr(Pr(s_l) \wedge Pr(s_k)) = \frac{1}{N_{T_i}} \cdot \frac{1}{N_{T_j}}$$

Ontology 1	**Ontology 2**
hotel $\sqsubseteq$ building $\sqcap$ $\exists$hosts.tourist	accommodation $\equiv$ $\exists$hosting.traveler
	motel $\sqsubseteq$ accommodation
	hostel $\sqsubseteq$ accommodation
	camping $\sqsubseteq$ accommodation

Fig. 1. Example of two ontologies in the touristic domain

Example. Let us consider the two simple ontologies of Figure 1. In order to define mappings between concepts described as classes, we consider their names. Consider the terms `hotel` and `hostel`. In WordNet, `hotel` is associated only with a synset $s(hotel)_1$, defining the hotel as "a building where travelers can pay for lodging and meals and other services"; `hostel` is associated with two synsets $s(hostel)_1$ and $s(hostel)_2$, stating that a hostel is a "hotel providing overnight lodging for travelers" and a "inexpensive supervised lodging", respectively. By means of the linguistic component of HMatch 2.0, we retrieve the relationship $BT(s(hotel)_1, s(hostel)_1)$. Our intuition is that such a relation is correct if the intended meaning of the term hotel is expressed by $s(hotel)_1$ and if the intended meaning of the term hostel is expressed by $s(hostel)_1$. In the first case, since hotel has only one meaning, the probability is equal to one. In the second case, since hostel has two possible meanings, the probability of $s(hostel)_1$ to be the correct synset is equal to $1/2 = 0.5$. Then, the comprehensive probability of the relation between hotel and hostel is calculated as follows:

$$Pr(BT(s(hotel)_1, s(hostel)_1) = \frac{1}{1} \cdot \frac{1}{2} = 0.5$$

In order to define linguistic mappings between concepts expressed by a property restriction of the form $(\exists \mid \forall \mid \geq n \mid \leq n \mid = n)R.C$, the quantifier must be equal and a linguistic mapping between the two properties R and the two concepts C must hold. In particular, given a terminological relationship r_1 between the properties of two restrictions and a terminological relationship r_2 between the concepts of two restrictions, we establish a terminological relationship r_3 between the two restrictions by choosing r_3 as the most restrictive terminological relationship between r_1 and r_2. The probability associated with r_3 is calculated as $Pr(r_3) = Pr(r_1) \cdot Pr(r_2)$. For example, take into account the concepts $\exists hosts.tourist$ and $\exists hosting.traveler$. Given the mappings $SYN(hosts, hosting)$ with probability 1.0 and $NT(tourist, traveler)$ with probability 1.0, we obtain the mapping:

$$Pr(NT(\exists hosts.tourist, \exists hosting.traveler)) = 1.0$$

When the two ontologies of Figure 1 are aligned, we collect all the mappings available among the concepts of the two ontologies, that are shown in Figure 2 together with their probabilities.

Mapping relation	Probability
BT(hotel, motel)	1.0
NT(∃hosts.tourist, ∃hosting.traveler)	1.0
NT(tourist, traveler)	1.0
BT(hotel, hostel)	0.5
BT(building, motel)	0.25
BT(building, hostel)	0.125

Fig. 2. Example of probabilistic mappings between the ontologies of Figure 1

3 Formalizing Probabilistic Mappings Using P-$\mathcal{SHIQ}$(D)

In this section, we present P-$\mathcal{SHIQ}$(D), the probabilistic description logics we use in order to formalize probabilistic mappings as semantic relations.

3.1 A Probabilistic Extension of $\mathcal{SHIQ}(D)$

In order to extend a description logic for dealing with probabilistic knowledge an additional syntactical and semantical construct is needed. The additional construct is called a *conditional constraint.* This extension of description logics has been first formalized in [4] for $\mathcal{SHOQ}(D)$. Later the extension was adapted to $\mathcal{SHIF}(D)$ and $\mathcal{SHOIN}(D)$ in [5], due to the fact that probabilistic reasoning problems are reduced to solving linear programs and standard satisfiability tests regarding the underling DL . We use $\mathcal{SHIQ}(D)$ as underling DL, hence the name P-$\mathcal{SHIQ}$(D). For Syntax and Semantic of $\mathcal{SHIQ}(D)$ the reader is referred to [6]. A conditional constraint consists of a statement of conditional probability for two concepts C, D as well as a lower bound l and an upper bound u on that probability. It is written as follows: $(D|C)[l, u]$ Where C can be called the *evidence* and D the *hypothesis.* To gain the ability to store such statements in a knowledge base it has to be extended to a probabilistic knowledge base $\mathcal{PKB}$. Additionally to the TBox $\mathcal{T}$ of a description logic knowledge base we introduce the PTBox $\mathcal{PT}$, which consists of $\mathcal{T}$ and a set of conditional constraints $\mathcal{D}_g$, and a PABox $\mathcal{PA}$ holding sets of conditional constraints $\mathcal{D}_o$ for each probabilistic individual o. We also define the set of probabilistic individuals I_p, which contains all individuals o for which some probabilistic knowledge is available and therefore a set $\mathcal{D}_o$. In [4] there is no ABox declared, knowledge about so called classical individuals is also stored inside the TBox using nominals. $\mathcal{D}_g$ therefore represents statistical knowledge about concepts and $\mathcal{D}_o$ represents degrees of belief about the individual o. To be able to define the semantics for a description logic with probabilistic extension the interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot)$ has to be extended by a probability function μ on the domain of interpretation $\Delta^{\mathcal{I}}$. The extended interpretation is called the *probabilistic interpretation* $\mathcal{P}r = (\mathcal{I}, \mu)$. Each individual o in the domain $\Delta^{\mathcal{I}}$ is mapped by the probability function μ to a value in the interval [0,1] and the values of all $\mu(o)$ have to sum up to 1 for any probabilistic interpretation $\mathcal{P}r$. With the probabilistic interpretation $\mathcal{P}r$ at hand the probability of a concept C, represented by $\mathcal{P}r(C)$, is defined as sum of all $\mu(o)$ where

$o \in C^{\mathcal{I}}$. The probabilistic interpretation of a conditional probability $\mathcal{P}r(D|C)$ is given as $\frac{\mathcal{P}r(C \sqcap D)}{\mathcal{P}r(C)}$ where $\mathcal{P}r(C) > 0$. A conditional constraint $(D|C)[l,u]$ is *satisfied* by $\mathcal{P}r$ or $\mathcal{P}r$ *models* $(D|C)[l,u]$ if and only if $\mathcal{P}r(D|C) \in [l,u]$. We will write this as $\mathcal{P}r \models (D|C)[l,u]$. A probabilistic interpretation $\mathcal{P}r$ is said to *satisfy* or *model* a terminology axiom T, written $\mathcal{P}r \models T$, if and only if $\mathcal{I} \models T$. A set $\mathcal{F}$ consisting of terminological axioms and conditional constraints, where F denotes the elements of $\mathcal{F}$, is satisfied or modeled by $\mathcal{P}r$ if and only if $\mathcal{P}r \models F$ for all $F \in \mathcal{F}$. The *verification* of a conditional constraint $(D|C)[l,u]$ is defined as $\mathcal{P}r(C) = 1$ and $\mathcal{P}r$ has to be a model of $(D|C)[l,u]$. We also may say $\mathcal{P}r$ *verifies* the conditional constraint $(D|C)[l,u]$. On the contrary the *falsification* of a conditional constraint $(D|C)[l,u]$ is given if and only if $\mathcal{P}r(C) = 1$ and $\mathcal{P}r$ does **not** satisfy $(D|C)[l,u]$. It is also said that $\mathcal{P}r$ *falsifies* $(D|C)[l,u]$. Further a conditional constraint F is said to be *tolerated* under a Terminology $\mathcal{T}$ and a set of conditional constraints $\mathcal{D}$ if and only if a probabilistic interpretation $\mathcal{P}r$ can be found that verifies F and $\mathcal{P}r \models \mathcal{T} \cup \mathcal{D}$. With all these definitions at hand we are now prepared to define the *z-partition* of a set of generic conditional constraints $\mathcal{D}_g$. The z-partition is build as ordered partition $(\mathcal{D}_0, \ldots, \mathcal{D}_k)$ of $\mathcal{D}_g$, where each part $\mathcal{D}_i$ with $i \in \{0, \ldots, k\}$ is the set of all conditional constraints $F \in \mathcal{D}_g \setminus (\mathcal{D}_0 \cup \cdots \cup \mathcal{D}_{i-1})$, that are tolerated under the generic terminology $\mathcal{T}_g$ and $\mathcal{D}_g \setminus (\mathcal{D}_0 \cup \cdots \cup \mathcal{D}_{i-1})$. If the z-partition can be build from a PABox $\mathcal{PT} = (\mathcal{T}, \mathcal{D}_g)$, it is said to be *generically consistent* or *g-consistent*. A probabilistic knowledge base $\mathcal{PKB} = (\mathcal{PT}, (\mathcal{P}_o)_{o \in I_p})$ is *consistent* if and only if $\mathcal{PT}$ is g-consistent and $\mathcal{P}r \models \mathcal{T} \cup \mathcal{D}_o$ for all $o \in I_p$.

We use the z-partition for the definition of the lexicographic order on the probabilistic interpretations $\mathcal{P}r$ as follows: A probabilistic interpretation $\mathcal{P}r$ is called *lexicographical preferred* to a probabilistic interpretation $\mathcal{P}r'$ if and only if some $i \in \{0, \ldots, k\}$ can be found, that $|\{F \in \mathrm{D}_i \mid \mathcal{P}r \models F\}| > |\{F \in \mathrm{D}_i \mid \mathcal{P}r' \models F\}|$ and $|\{F \in \mathrm{D}_j \mid \mathcal{P}r \models F\}| = |\{F \in \mathrm{D}_j \mid \mathcal{P}r' \models F\}|$ for all $i < j \leq k$.

We say a probabilistic interpretation $\mathcal{P}r$ of a set $\mathcal{F}$ of terminological axioms and conditional constraints is a *lexicographically minimal model* of $\mathcal{F}$ if and only if no probabilistic interpretation $\mathcal{P}r'$ is lexicographical preferred to $\mathcal{P}r$. By now the meaning of *lexicographic entailment* for conditional constraints from a set $\mathcal{F}$ of terminological axioms and conditional constraints under a PTBox $\mathcal{PT}$ is given as: A conditional constraint $(D|C)[l,u]$ is a *lexicographic consequence* of a set $\mathcal{F}$ of terminological axioms and conditional constraints under a PTBox $\mathcal{PT}$, written as $\mathcal{F} \Vert\!\!\sim (D|C)[l,u]$ under $\mathcal{PT}$, if and only if $\mathcal{P}r(D) \in [l,u]$ for every lexicographically minimal model $\mathcal{P}r$ of $\mathcal{F} \cup \{(C|\top)[1,1]\}$. *Tight lexicographic consequence* of $\mathcal{F}$ under $\mathcal{PT}$ is defined as $\mathcal{F} \Vert\!\!\sim_{tight} (D|C)[l,u]$ if and only if l is the infimum and u is the supremum of $\mathcal{P}r(D)$. We define $l = 1$ and $u = 0$ if **no** such probabilistic interpretation $\mathcal{P}r$ exists. Finally we define lexicographic entailment using a probabilistic knowledge base $\mathcal{PKB}$ for generic and assertional conditional constraints F. If F is a generic conditional constraint, then it is said to be a lexicographic consequence of $\mathcal{PKB}$, written $\mathcal{PKB} \Vert\!\!\sim F$ if and only if $\emptyset \Vert\!\!\sim F$ under $\mathcal{PT}$ and a tight lexicographic consequence of $\mathcal{PKB}$, written $\mathcal{PKB} \Vert\!\!\sim_{tight} F$ if and only if $\emptyset \Vert\!\!\sim_{tight} F$ under $\mathcal{PT}$. If F is an assertional conditional constraint for $o \in I_P$, then it is said to be a lexicographic consequence of $\mathcal{PKB}$, written $\mathcal{PKB} \;\|\!\!\sim F$, if and only if $\mathcal{D}_o \;\|\!\!\sim F$ under $\mathcal{PT}$ and a tight lexicographic consequence of $\mathcal{PKB}$, written $\mathcal{PKB} \;\|\!\!\sim_{tight} F$ if and only if $\mathcal{D}_o \;\|\!\!\sim_{tight} F$ under $\mathcal{PT}$.

3.2 Interpreting mappings as probabilistic knowledge

The basic idea behind the formalization of mappings as probabilistic knowledge is to transform probabilistic mappings into conditional constraints. In order to do that, we assume as hypothesis that a mapping relation r can be seen as a corresponding constraint over the ontology concepts. As an example, we say that, given a terminological relation $BT(A, B)$ with probability p, it can be interpreted as the probability that an instance of B is also an instance of A. In general, given the set R of relations produced by a matching system, we introduce a set R' of rules for the translation of each kind of relation in R into a corresponding conditional constraint. Taking into account the example of Section 2, we start from the set of relations $R = \{SYN, BT, NT\}$ and we define the following translation rules: i) $Pr(SYN(A, B)) = p \rightarrow (A|B)[p, p] \wedge (B|A)[p, p]$; ii) $Pr(BT(A, B)) = p \rightarrow (A|B)[p, p]$; iii) $Pr(NT(A, B)) = p \rightarrow (B|A)[p, p]$. The goal of the translation process is to assume a hypothetical representation of probabilistic mappings as semantic relations in P-$\mathcal{SHIQ}$(D). Given two $\mathcal{SHIQ}(D)$-compatible ontologies O_1 and O_2, we build a new P-$\mathcal{SHIQ}$(D) ontology $O_{1,2}$ by merging the axioms of O_1 and O_2 and by augmenting the resulting ontology with conditional constraints derived from mappings. Recalling the example of Section 2, the resulting ontology is shown in Figure 3.

TBox	**PTBox**
hotel$\sqsubseteq$building $\sqcap$ $\exists$hosts.tourist	$(hotel\|motel)[1.0, 1.0]$
accommodation$\equiv$$\exists$hosting.traveler	$(\exists hosting.traveler\|\exists hosts.tourist)[1.0, 1.0]$
motel$\sqsubseteq$accommodation	$(traveler\|tourist)[1.0, 1.0]$
hostel$\sqsubseteq$accommodation	$(hotel\|hostel)[0.5, 0.5]$
camping$\sqsubseteq$accommodation	$(building\|motel)[0.25, 0.25]$
	$(building\|hostel)[0.125, 0.125]$

Fig. 3. P-$\mathcal{SHIQ}$(D) representation of the ontologies of Figure 1

4 Probabilistic Reasoning Over Mappings

Now we will recap the techniques required to reason in P-$\mathcal{SHIQ}$(D). The motivation of this discussion is to present algorithmic issues on the one hand, and point to implementation issues on the other. Subsequently, we describe our approach to mapping validation by using the proposed ContraBovemRufum reasoning System (CBR).

4.1 Reasoning with P-$\mathcal{SHIQ}$(D)

As previously mentioned the proposed approach for solving probabilistic reasoning problems relies on use of standard reasoning services provided by a description logic reasoner in order to build linear programs which may be handed over to a solver in order to decide the solvability of the programs. Although this approach seems attractive usually severe performance problems can be expected for expressive logics. The complexity of the used algorithms lies within NP or worse depending on the underlying DL as it has been analyzed in [5]. Despite these results we think it is still worth investigating the approach to develop optimization techniques to bring down average case complexity.

In order to deciding satisfiability the first objective is to build a set $\mathcal{R}_{\mathcal{T}}(\mathcal{F})$. It contains the elements r, which map conditional constraints $F_i = (D_i|C_i)[l_i, u_i]$, elements of a set of conditional constraints $\mathcal{F}$, onto one of the following terms $D_i \sqcap C_i$, $\neg D_i \sqcap C_i$ or $\neg C_i$ under the condition, that the intersection of our r is not equal to the bottom concept given the terminology $\mathcal{T}$, written $\mathcal{T} \not\models r(F_1) \sqcap \cdots \sqcap r(F_n) \sqsubseteq \bot$. In the following we will write $\sqcap r$ instead of $r(F_1) \sqcap \cdots \sqcap r(F_n)$ as an abbreviation. With the set $\mathcal{R}_{\mathcal{T}}(\mathcal{F})$ at hand we are able to set up linear programs to decide the satisfiability of the terminology $\mathcal{T}$ and a finite set of conditional constraints $\mathcal{F}$. The constraints of the linear program are displayed in Figure 4. We say that $\mathcal{T} \cup \mathcal{F}$ is satisfiable if and only if the linear program with the constraints 1a-d is solvable for variables y_r, where $r \in \mathcal{R}_{\mathcal{T}}(\mathcal{F})$. This means, that in the objective function all coefficients preceding the variables y_r are set to 1. We further need to introduce the meaning of $r \models C$ which is used as index of the summation in 1a and 1b. It is an abbreviation for $\emptyset \models \sqcap r \sqsubseteq C$. So the coefficient preceding the variables y_r is set in linear constraints 1a and 1b if either $r \models \neg D \sqcap C$ or $r \models D \sqcap C$ may be proven.

$$\sum_{r \in \mathcal{R}_{\mathcal{T}}(\mathcal{F}), r \models \neg D \sqcap C} -l y_r + \sum_{r \in \mathcal{R}_{\mathcal{T}}(\mathcal{F}), r \models D \sqcap C} (1-l) y_r \geq 0 \quad \text{(for all } (D|C)[l,u] \in \mathcal{F}\text{)} \quad \text{(1a)}$$

$$\sum_{r \in \mathcal{R}_{\mathcal{T}}(\mathcal{F}), r \models \neg D \sqcap C} u y_r + \sum_{r \in \mathcal{R}_{\mathcal{T}}(\mathcal{F}), r \models D \sqcap C} (u-1) y_r \geq 0 \quad \text{(for all } (D|C)[l,u] \in \mathcal{F}\text{)} \quad \text{(1b)}$$

$$\sum_{r \in \mathcal{R}_{\mathcal{T}}(\mathcal{F})} y_r = 1 \quad \text{(1c)}$$

$$y_r \geq 0 \quad \text{(for all } r \in \mathcal{R}_{\mathcal{T}}(\mathcal{F})\text{)} \quad \text{(1d)}$$

Fig. 4. Constraints of the linear program

Why is the creation of linear programs reasonable? Consider the following: By definition a conditional constraint is satisfied if $u \geq \mathcal{P}r(D|C) \geq l \Leftrightarrow u\mathcal{P}r(C) \geq \mathcal{P}r(D \sqcap C) \geq l\mathcal{P}r(C)$. This may lead us to linear constraints 1a and 1b. Lets focus on the upper bound, whose derivation is displayed in Figure 5. The derivation

$$u \sum_{r\in\mathcal{R}_{\mathcal{T}}(\mathcal{F}),r\models C} y_r \geq \sum_{r\in\mathcal{R}_{\mathcal{T}}(\mathcal{F}),r\models D\sqcap C} y_r \Leftrightarrow \tag{2a}$$

$$u \sum_{r\in\mathcal{R}_{\mathcal{T}}(\mathcal{F}),r\models(\neg D\sqcap C)\sqcup(D\sqcap C)} y_r \geq \sum_{r\in\mathcal{R}_{\mathcal{T}}(\mathcal{F}),r\models D\sqcap C} y_r \Leftrightarrow \tag{2b}$$

$$u \sum_{r\in\mathcal{R}_{\mathcal{T}}(\mathcal{F}),r\models\neg D\sqcap C} y_r + u \sum_{r\in\mathcal{R}_{\mathcal{T}}(\mathcal{F}),r\models D\sqcap C} y_r \geq \sum_{r\in\mathcal{R}_{\mathcal{T}}(\mathcal{F}),r\models D\sqcap C} y_r \Leftrightarrow \tag{2c}$$

$$\sum_{r\in\mathcal{R}_{\mathcal{T}}(\mathcal{F}),r\models\neg D\sqcap C} u y_r + \sum_{r\in\mathcal{R}_{\mathcal{T}}(\mathcal{F}),r\models D\sqcap C} (u-1) y_r \geq 0 \tag{2d}$$

Fig. 5. Upper bound derivation

for the lower bound 1a follows analogously. The linear constraints 1c and 1d reflect that all $\mu(o)$ have to sum up to 1 and all $\mu(o) \in [0,1]$ Lets have a look at the number of subsumbtion tests which need to be performed to set up the system of linear constraints. At a first glance, finding a $\models$ under each sum, one might say four tests per variable and conditional constraint. Looking closer we discover that only two are required because they are identical for lower and upper bound. But even this may be optimised further. Considering that the $\sqcap r$ represents a map of all conditional constraints on $D_i \sqcap C_i$, $\neg D_i \sqcap C_i$ or $\neg C_i$ and they are tested on subsumbtion against $D \sqcap C$ and $\neg D \sqcap C$ we observe that only if the first subsumbtion test of $\sqcap r \sqsubseteq D \sqcap C$ failed the second one is necessary. Therefore significantly reducing the number of required tests per variable and conditional constraint. With the tool at hand to decide satisfiability, we may also decide if a conditional constraint may be tolerated by a set of conditional constraints $\mathcal{F}$. To verify a constraint we add a conditional constraint $(C|\top)[1,1]$. With the extended set the linear program is generated and solved. If an unfeasible solution is computed the conditional constraint is conflicting. If an optimal solution is found, the conditional constraint is tolerated. Now the z-partition of a set of conditional constraints is computable. How to compute tightest probability bounds for given evidence C and conclusion D in respect to a set of conditional constraints $\mathcal{F}$ under a terminology $\mathcal{T}$? The task is named *tight logical entailment* and denoted $\mathcal{T} \cup \mathcal{F} \models_{tight} (D|C)[l,u]$. Given that $\mathcal{T} \cup \mathcal{F}$ is satisfiable, a linear program is set up for $\mathcal{F} \cup (C|\top)[1,1] \cup (D|\top)[0,1]$. The objective function is set to $\sum\limits_{r\in R, r\models D} y_r$. So the coefficient in front of the variables y_r are set 1 if $r \models D$. The tight logical entailed lower bound l is computed by minimising, respectively the upper bound u by maximising the linear program. In order to compute tight probabilistic lexicographic entailment for given evidence C and conclusion D under a $\mathcal{PKB}$ the following steps have to be taken:

1. Compute the z-partition of $\mathcal{D}_g$ in order to be able to generate a lexicographic ordering

2. Compute lexicographic minimal sets $\mathcal{D}'$ of conditional constraints of $\mathcal{D}_g$ as elements of $\overline{\mathcal{D}}$.
3. Compute the tight logical entailment $\mathcal{T} \cup \mathcal{F} \cup \mathcal{D}' \models_{tight} (D|C)[l,u]$ for all $\mathcal{D}' \in \overline{\mathcal{D}}$.
4. Select the minimum of all computed lower bounds and the maximum of all upper bounds.

The 2. step needs some explanation since a new task "compute *lexicographic minimal sets*" is introduced. In order to define a lexicographic minimal set $\mathcal{D}'$, a preparatory definition is required. A set $\mathcal{D}' \subset \mathcal{D}_g$ *lexicographic preferable* to $\mathcal{D}'' \subset \mathcal{D}_g$ if and only if some $i \in \{0,\dots,k\}$ exists such that $|\mathcal{D}' \cap \mathcal{D}_i| > |\mathcal{D}'' \cap \mathcal{D}_i|$ and $|\mathcal{D}' \cap \mathcal{D}_i| > |\mathcal{D}'' \cap \mathcal{D}_i|$ for all $i < j \leq k$. With the lexicographic order introduced onto the sets $\mathcal{D}'$ the definition of lexicographic minimal is given as: $\mathcal{D}'$ is lexicographic minimal in $\mathcal{S} \subseteq \{S|S \subseteq \mathcal{D}_g\}$ if and only if $\mathcal{D}' \in \mathcal{S}$ and no $\mathcal{D}'' \in \mathcal{S}$ is lexicographic preferable to $\mathcal{D}'$.

We implemented the algorithms presented by [4, 5] in the Java programming language into the ContraBovemRufum System. For the reasoning tasks RacerPro with its JRacer interface is used. As solvers a native Java solver by Opsresearch and the Mosek linear program solver have been integrated. For application programmers two different sets of interfaces are provided. The first set contains the ProbabilisticKBInterface, which provides all operations related to setting up and modifying PTBox and PABox, and the ProbabilisticEntailmentInterface, which offers the probabilistic inference operations to decide consistency for $\mathcal{PT}$ and $\mathcal{PKB}$ as well as probabilistic subsumption and probabilistic instance checking. The second set of interfaces handles the configuration of the services. Using the SolverConfiguration interface the selection of the solver may be changed at runtime. The ReasonerConfiguration interface makes the settings for the reasoner. With the EntailmentConfiguration interface the algorithm used to compute the entailment may be chosen at runtime. Currently tight logical entailment and the two available algorithms for tight lexicographic entailment are supported.

4.2 Mapping validation and revision

After mapping acquisition and representation as probabilistic constraints, we want to use tight lexicographic entailment in order to check if one (or more) constraints used to represent mappings causes incoherence within the PTBox of the ontology obtained by merging the two ontologies. A mapping can produce contradictions in two ways: i) the constraint itself is contradictory with respect to some of the axioms already present in the two original ontologies; ii) the constraint is correct, but the probability associated with it is not compatible with the probability constraints already present in the ontologies. Moreover, incoherence can be caused by two different situations: i) the mapping is not compatible with one or more axioms already present in the merged ontology; ii) the mapping is not compatible with other mappings introduced in the merged ontology. In the first case, incompatible mappings are discarded in order to preserve the original knowledge provided by the ontologies at hand. In the second case, we need a heuristic for choosing the incompatible mapping(s) to be discarded. In our approach, we propose to keep the mappings featured by higher probabilities to be true under the assumption that relations which are more probable are also more reliable. In order to implement this approach, we adopt the mapping validation procedure depicted in Figure 6.

```
mapping_validation($M$,$O$):
 input: a mapping set $M$, a merged ontology $O$
 output: a validated mapping set $M'$
 begin
   $M$ := sort $M$ w.r.t. the probability
         associated with each mapping $m_i \in M$;
   for $m_i \in M$:
     $c_i$ := translate $m_i$ into a conditional constraint;
     add $c_i$ to $O$;
     check coherency of $O$ with CBR;
     if $O$ is coherent then:
       continue;
     else:
       remove $c_i$ from $O$;
     end if
 end
```

Fig. 6. Mapping validation and revision procedure

The procedure takes as input a mapping set together with a merged ontology. As a first step, the mapping set M is ordered by descending ordering from the most probable to the less probable mapping. In such a way, we first insert the more probable mappings and, in case of incoherence, we discard always the latter (i.e., less probable) mapping(s). Then, for each mapping, we translate the mapping into a conditional constraint as shown in Section 3 and we insert such a constraint into the P-$\mathcal{SHIQ}$(D) ontology. We use the CBR reasoning system in order to check the coherence of the ontology augmented with the conditional constraint. In the coherence check fails, we discard the mapping and we delete the conditional constraint from the ontology. The final result is that the ontology is augmented only with mappings which do not cause incoherence problems into the PTBox.

Example. As an example, we consider mappings of Figure 2 and the merged ontology of Figure 3. The first four mappings do not cause any incoherence within the PTBox, then the corresponding conditional constraints are added to the ontology. The mapping $BT(building, motel)$ with probability 0.25 is not compatible with other mappings. In particular, it happens that an instance of motel is also an instance of hotel with probability 1.0, due to the first mapping. Moreover, the concept hotel is subsumed by the concept building. Then, an instance of motel must be an instance of building with probability $p \geq 1.0$. According to the mapping validation procedure, this mapping is discarded. We note that such a mapping is also useless, since the relation between motel and building is already implicitly expressed by the axioms of the first ontology and by the first mapping. Analogously, we discard also the mapping $BT(building, hostel)$ with probability 0.125. Finally, we can check which conditional constraints can be inferred by the CBR system by considering the validated mappings. The final result of our example is summarized in Table 1.

Table 1. Mappings after validation and revision

Discarded constraints	**Revised constraints**
$(building\|motel)[0.25, 0.25]$	$(\exists hosts.tourist\|motel)[1.0, 1.0]$
$(building\|hostel)[0.125, 0.125]$	$(\exists hosts.tourist\|hostel)[0.5, 1.0]$
	$(building\|motel)[1.0, 1.0]$
	$(building\|hostel)[0.5, 1.0]$
	$(accommodation\|hotel)[1.0, 1.0]$
	$(accommodation\|\exists hosts.tourist)[1.0, 1.0]$
	$(\exists hosts.tourist\|accommodation)[1.0, 1.0]$

5 Related Work

Related work concerns both ontology mapping validation and probabilistic reasoning techniques. The problem of automatically discovering ontology/schema mappings has been studied both in the field of ontologies [1] and in the field of databases [7]. In the literature, some approaches have been proposed for validating mappings with respect to the semantics of the involved ontologies, but in most of the cases these approaches do not deal with uncertain mappings. An example of work in this direction is the *S-Match* system [2]. The key idea behind S-Match is to compute an initial set of relations between concepts of two taxonomies by means of external sources (i.e. WordNet); then, the problem of matching is translated into a validity check problem on the concept taxonomy. However, the validation approach of S-Match does not take into account the uncertain nature of mappings which are instead translated into crisp relations. Another important step towards mapping validation is represented by the theoretical study proposed in [8], which differs from previous works in that logical reasoning about mappings is performed. Mappings are translated into logical constraints and represented by means of Distributed Description Logics [9], an extension of classical DLs for distributed knowledge bases. An implementation of this approach has been discussed in [10]. However, also in this case, mapping uncertainty is not supported. Other relevant work has been done in using fuzzy reasoning for handling uncertainty in ontologies [11]. A fuzzy approach is different from our proposal in that a different understanding of the semantics of mappings is applied. However, especially for instance matching, the use of fuzzy reasoning can be integrated in our approach. In [12] the authors propose a mapping validation approach based on the idea of representing mappings as probabilistic rules. With respect to this work, we present a linguistic matching procedure which is compatible with the probabilistic interpretation of mappings instead of applying probabilities to every kind of results produced by matching tools. In recent years, the problem of modeling uncertainty has become one focus of interest also in description logic research. First probabilistic extensions started by modelling a degree of overlap between concepts via probabilities at the TBox level [13] as well as believes about certain individuals at the ABox level [14]. In P-CLASSIC [15] an approach was presented which integrates Bayesian networks as underlying reasoning formalism. Further work on integration of description logic and Bayesian networks has been presented in [16]. Efforts to integrate several approaches of uncertainty representation into one coherent framework have been made in [17]. The presented extension differs from the aforementioned ones, due to its ties to default logic, a non-monotonic reasoning method developed by Reiter [18]. Here lexicographic entailment for defaults [19] was adapted to probabilistic description logics which provides non-monotonic inheritance properties along subclass relationships.

6 Concluding Remarks

In this paper, we have proposed a mapping validation approach with the following original contributions: i) *separation of concern*: the mapping validation process is independent from the mapping discovery process, in order to enable interoperability with existing matching systems; *uncertainty-compliance*: a notion of probabilistic mapping is introduced to capture the uncertain nature of the correspondences stated by mappings; *well-founded approach*: the probabilistic extension P-$\mathcal{SHIQ}$(D) of $\mathcal{SHIQ}(D)$ is used to represent probabilistic mappings and a sound reasoning system for P-$\mathcal{SHIQ}$(D) is presented and used for working with probabilistic mappings and formally validate them.

Our approach to mapping validation is adopted in the European BOEMIE project, where ontology mappings are used as a support for ontology evolution. In the context of BOEMIE, an evaluation of this validation approach is being performed on linguistic mappings between the athletics domain ontology of BOEMIE and a set of external knowledge sources. Goal of the evaluation is to compare the quality of mappings in terms of precision and accuracy before and after the validation process. Our ongoing and future work is mainly devoted to: i) use of validated mappings for ontology evolution; ii) investigation on optimization techniques for the probabilistic reasoning algorithms implemented in the CBR System, in order to improve the systems average case performance; iii) investigation on the problem of automatically detecting the minimal set of mappings causing incoherence during validation, in order to substitute the current heuristic-based approach to mapping revision with a reasoning-based approach.

References

1. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer-Verlag (2007)
2. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: S-Match: an Algorithm and an Implementation of Semantic Matching. In: Proc. of the First European Semantic Web Symposium (ESWS). (2004) 61–75
3. Castano, S., Ferrara, A., Montanelli, S.: Matching ontologies in open networked systems: Techniques and applications. Journal on Data Semantics (JoDS) **V** (2006)
4. Giugno, R., Lukasiewicz, T.: P-$\mathcal{SHOQ}(\mathbf{D})$: A probabilistic extension of $\mathcal{SHOQ}(\mathbf{D})$ for probabilistic ontologies in the semantic web. In: JELIA. (2002) 86–97
5. Lukasiewicz, T.: Expressive probabilistic description logics. Artif. Intell. **172** (2008) 852–883
6. Haarslev, V., Möller, R.: Expressive abox reasoning with number restrictions, role hierarchies, and transitively closed roles. In: Proc. of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR2000), Breckenridge, Colorado, USA (2000) 273–284
7. Rahm, E., Bernstein, P.A.: A survey of approaches to automatic schema matching. VLDB Journal **10** (2001) 334–350
8. Stuckenschmidt, H., Serafini, L., Wache, H.: Reasoning about ontology mappings. Technical report, ITC-IRST, Trento (2005)
9. Borgida, A., Serafini, L.: Distributed description logics: Assimilating information from peer sources. Journal on Data Semantics **1** (2003) 153–184
10. Serafini, L., Tamilin, A.: Drago: Distributed reasoning architecture for the semantic web. In: Proc. of the 2nd European Semantic Web Conference (ESWC05), Heraklion, Greece (2005) 361–376
11. Stoilos, G., Stamou, G., Pan, J., Tzouvaras, V., Horrocks, I.: Reasoning with very expressive fuzzy description logics. Journal of Artificial Intelligence Research **30** (2007) 133–179
12. Cali, A., Lukasiewicz, T., Predoiu, L., Stuckenschmidt, H.: A framework for representing ontology mappings under probabilities and inconsistencies. In: Proc. of the Workshop for Uncertainty Reasoning on the Semantic Web (URSW) in conjunction with the ISWC 2007, Busan, Korea (2007)
13. Heinsohn, J.: Probabilistic description logics. In: Proc. of the 10th Conf. on Uncertainty in Artificial Intelligence, Seattle, Washington (1994) 311–318
14. Jaeger, M.: Probabilistic reasoning in terminological logics. In: Proceedings of the Fourth International Conference on Knowledge Representation and Reasoning (KR94). (1994) 305–316
15. Koller, D., Levy, A.Y., Pfeffer, A.: P-classic: A tractable probabilistic description logic. In: AAAI/IAAI. (1997) 390–397
16. Ding, Z., Peng, Y.: A Probabilistic Extension to Ontology Language OWL. In: HICSS '04: Proc. of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04), Washington, DC, USA (2004)
17. Haarslev, V., Pai, H.I., Shiri, N.: A generic framework for description logics with uncertainty. In: ISWC-URSW. (2005) 77–86
18. Reiter, R.: A logic for default reasoning. Artif. Intell. **13** (1980) 81–132
19. Lehmann, D.: Another perspective on default reasoning. Annals of Mathematics and Artificial Intelligence **15** (1995) 61–82