Implementing probabilistic description logics: An application to image interpretation

Tobias H. Näth[1] **Ralf Möller**[1]

[1] Institute for Software Systems
Technische Universität Hamburg-Harburg,
Harburger Schlostr. 20
Hamburg D-21073, Germany,
Email: `tobias.naeth@tu-harburg.de`
Email: `r.f.moeller@tu-harburg.de`

Abstract

This paper presents an application of an optimized implementation of a probabilistic description logic defined by Giugno and Lukasiewicz [9] to the domain of image interpretation. This approach extends a description logic with so-called probabilistic constraints to allow for automated reasoning over formal ontologies in combination with probabilistic knowledge. We analyze the performance of current algorithms and investigate new optimization techniques.

Keywords: probabilistic description logics, image interpretation, probabilistic lexicographic entailment

1 Introduction

For at least ten years, modeling of uncertainty combined with description logics (DLs) has been a topic of research. Several theoretical approaches have been developed and discussed in the literature. Though there are implementations for fuzzy description logic available, e.g. [31], none of the probabilistic approaches has to our knowledge made its way into a DL reasoning system obtainable today. The reason is that various kinds of semantics were investigated, each of which turned out not to be easily realizable in a mature DL system. Recently, techniqes for so-called probabilistic lexicographic reasoning services were proposed by Giugno and Lukasiewicz [9]. Their approach, named P-$\mathcal{SHOQ}(D)$, reduces the probabilistic reasoning problems to solving linear programs and standard satisfiability tests with respect to the underlying description logic. This property allows for a modular implementation reusing mature software components, i.e., a DL reasoner and a linear program solver. Although this approach seems attractive, experience shows that, usually, severe performance problems can be expected for expressive logics. However, since the P-$\mathcal{SHOQ}(D)$ approach is well-suited for the application scenario we are currently investigating, namely image interpretation (see below). The use of description logic for image interpretation has also been explored in [26, 24]. We investigate the hypothesis that a modular implementation can be done in practice. As a DL reasoner RacerPro has been chosen due to its stability and maturity [3, 2]. As a solver for linear programs we used [4] and [1]. The combined probabilistic reasoning system is called ContraBovemRufum.[1] The results we obtained are indeed encouraging. From the experiences of the optimized, but modular implementation in our application scenario, we develop new optimization techniques for subsequent system implementations.

This work was partially supported by the EU-funded project BOEMIE (Bootstrapping Ontology Evolution with Multimedia Information Extraction, IST-FP6-027538).

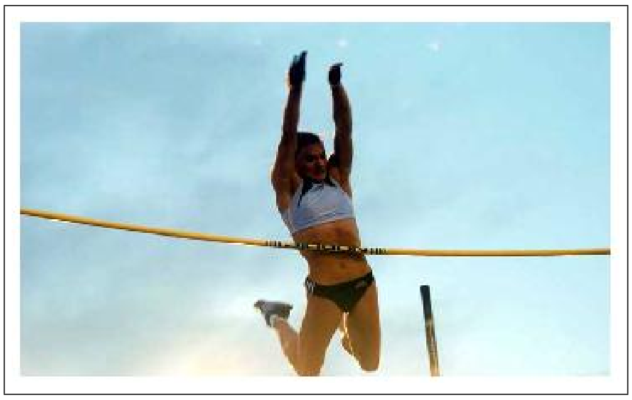

Figure 1: Pole-vault image. Or does it show a high-jump?

Automated high-level interpretation of images relies on low-level feature extraction of image data. Let us consider the image in Figure 1. The image is from the athletics domain. A low-level process might extract instances of the concepts athlete, pole and bar. These are visible objects. However, image interpretation should not stop here. The high-level interpretation of this image should probably be a more abstract objects, in this case perhaps a pole-vault event. But, can we be sure about this? If not, where does the uncertainty arise? What if the low level extraction fails to detect the pole and the image might be interpreted as a high-jump event? Naturally, both interpretations seem possible in this case. The pole-vault interpretation should be the more likely one, but we do not want to rule out the high-jump interpretation. With more evidence arriving it may also be the case that conclusions would be drawn differently, reconsidering the previously made interpretations. Usually the extraction process provides probability values of its certainty for each detected visible instance. These values could be used to guide the high-level interpretation process.
The main contributions of this paper can be summarised as follows:

1. We investigated the feasibility of the approach taken by Giugno and Lukasiewicz [9] in a practical implementation, showing that it can be implemented and that it is worth to continue in this line of research. The implementation already includes the optimised algorithms proposed in [21].

[1] See [25] for the genesis of the name.

2. The application of probabilistic description logics for image interpretation is examined and the benefits of the taken approach are shown.

3. New techniques for further optimisations of the reasoning algorithms are discussed.

The paper is organized as follows. Section 2 provides an overview over the related research. Section 3 introduces the syntax and semantic of the probabilistic reasoning extension. Within section 4 the previously described application scenario is modelled and the application of the new system is presented. Section 5 summarises the implementation. The algorithms and the implementation are analysed in section 6. Section 7 concludes this paper.

2 Related Research

The research related to the theory which is realized by the ContraBovemRufum system may be divided into three categories: *probabilistic logics*, *description logics* and *non-monotonic logics*. The first category deals with the combination of probability theory and logic, the second with decidable subsets of first order logic and the third with exceptions in the logic.

2.1 Probabilistic Logics

In “Boole's Logic and Probability”, Halperin [15] investigates the research done by George Boole (1815-1864) on the combination of probability theory and logic. He is describing how Boole's work may be applied to the subject of linear programming. Thus we may state that combining logics and probability is a rather old field of research.

Several probabilistic reasoning methods for knowledge based systems have been discussed and published in books by Pearl [28], Paass [27], Bacchus [6], Russell [30]. Scanning the related literature two approaches can be found in order to perform inferences with probabilities. The first approach uses Bayesian or belief networks to determine the inferred probabilities [28, 30], the second makes use of linear programming for this task [27]. Both approaches have been tried to be combined with description logics. The combination of Bayesian networks and description logics is first described in [19] (P-Classic), and is recently further investigated to some extend in [8]. An implementation of P-Classic is described in [18]. The disadvantage of P-Classic, namely the required upper bound for number restrictions led us to the investigation of another approach. Our system implements the algorithms proposed in [10] and its follow up [21], which use linear programming to achieve probabilistic inference.

2.2 Description Logics

Although the approach of [10] is based on the expressive description logic $\mathcal{SHOQ}$ [16], due to the modularity, any description logic could be considered. Our implementation is bound to $\mathcal{SHIQ}(D)$ [12, 17, 13] provided by RacerPro. For our drafted application scenario in this paper it is sufficient to introduce the description logic $\mathcal{ALC}$. A further overview on description logics is given in [5].

Let A be a concept name and R be a role name. Then, the set of $\mathcal{ALC}$concepts (denoted by C or D) is inductively defined as follows: $C, D \rightarrow A \,|\, \neg C \,|\, C \sqcap D \,|\, C \sqcup D \,|\, \forall R.C \,|\, \exists R.C$. Concepts can be written in parentheses. The semantics is defined in the standard form using a Tarskian interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ such that

$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}, R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$

$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$ (complement)

$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$ (conjunction)

$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$ (disjunction)

$(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \,|\, \exists y.\, (x, y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$ (value restriction)

$(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \,|\, \forall y.\, (x, y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$ (existential restriction)

A concept C is satisfiable if there exists an interpretation such that $C^{\mathcal{I}} \neq \emptyset$.

A (generalized) terminology (also called Tbox, $\mathcal{T}$) is a set of axioms of the form $C \sqsubseteq D$ (generalized concept inclusion, GCI). A GCI $C \sqsubseteq D$ is satisfied by an interpretation $\mathcal{I}$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. An interpretation which satisfies all axioms of a Tbox is called a model of the Tbox. A concept C is satisfiable w.r.t. a Tbox if there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $C^{\mathcal{I}} \neq \emptyset$. A concept D subsumes a concept C w.r.t. a Tbox if for all models $\mathcal{I}$ of the Tbox it holds that $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. D is called the subsumer, C is the subsumee. An ABox $\mathcal{A}$ contains a finite set of assertional axioms stating explicit knowledge concept membership ($a : C$ satisfied if $a^{\mathcal{I}} \in C^{\mathcal{I}}$) and role membership ($(a, b) : R$ satisfied if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$) on the elements of the domain. An interpretation $\mathcal{I}$ is a model of an ABox $\mathcal{A}$ with respect to a TBox $\mathcal{T}$ iff it is a model of $\mathcal{T}$ and satiesfies all assertions in $\mathcal{A}$.

2.3 Non-monotonic Logics

In the introduction we have seen that for image interpretation also non-monotonic logic are appropriate. The difference between monotonic and non-monotonic reasoning is summarised below.

Reasoning in Propositional Logic or First Order Logic is monotone with respect to the available knowledge. More precisely, adding new statements to the knowledge base ”can only increase the set of entailed sentences” [30] or in other words ”if something is known to be *true* (*false*), it will never become *false* (*true*)” [14]. In [30] this is formally stated as:

If $KB \models \phi$ then $KB \wedge \psi \models \phi$, iff all models $\mathcal{I}$ of KB and ψ are also models of ϕ. In the context of description logics a knowledge base KB is a Tbox.

By the observation of human reasoning one may recognise that it is not monotone at all. This can be described informally as ”changing one's mind”. Humans allow exceptions in their knowledge which can not be resembled neither in Propositional Logic nor in First Order Logic. To address these shortcomings several non-monotonic reasoning methods [29, 22, 23] have been developed.

Default Logic is a non-monotonic reasoning method developed by Reiter [29]. In this approach one differentiates between hard facts about a world and default rules. ”Since the set of all hard facts cannot completely specify the world, we are left with gaps in our knowledge; the purpose of the default rules is to help fill in those gaps with plausible conclusions” [28].

Further research in default logic developed several different kinds of entailments, e.g. 0-entailment, 1-entailment, z-entailment, lexicographic entailment and me-entailment to name a few (see [7] for an overview).

The probabilistic DL successors [9, 21] of lexicographic entailment [20] will be the object of the presented theory.

3 Probabilistic Lexicographic Entailment for Description Logics

In order to extend a description logic for dealing with probabilistic knowledge an additional syntactical and semantical construct is needed. This additional construct is called a *conditional constraint.* This extension of description logics has been first formalized in [10, 9].

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 constraint on that probability. It is written as follows:

$$(D|C)[l, u] \tag{1}$$

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 PKB here named probabilistic terminology $\mathcal{P}$. Additionally to the TBox T here classical terminology $\mathcal{T}_g$ of a description logic knowledge base we introduce the PTBox PT here generic probabilistic terminology $\mathcal{P}_g$, which consists of $\mathcal{T}_g$ and a set of conditional constraints $\mathcal{D}_g$, and a PABox P_o or assertional probabilistic terminology $\mathcal{P}_o$ holding conditional constraints for every probabilistic individual o. In [9] 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 P_o represents degrees of belief about the individuals 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 generic probabilistic terminology $\mathcal{P}_g = (\mathcal{T}_g, \mathcal{D}_g)$, it is said to be *generically consistent* or *g-consistent.* A probabilistic terminology $\mathcal{P} = (\mathcal{P}_g, (\mathcal{P}_o)_{o \in I_p})$ is *consistent* if and only if $\mathcal{P}_g$ is g-consistent and $\mathcal{P}r \models \mathcal{T}_g \cup \mathcal{T}_o \cup \mathcal{D}_o \cup \{(\{o\}|\top)[1, 1]\}$ for all $o \in I_p$. The z-partition groups the conditional constraints by specificity. A part of the z-partition is more specific than another if its index is higher. With possible conflicting conditional constraints and the inability to find a probabilistic interpretation $\mathcal{P}r$ that tolerates all of them, the goal is to select a preferred probabilistic interpretation $\mathcal{P}r$ to draw the conclusion from. Because of this according to [20] two criteria have to be taken into account:

1. A $\mathcal{P}r$ is preferred to another model if it satisfies more defaults.

2. Satisfying a more specific default is preferred to a less specific one.

Both criteria allow us to order the probabilistic interpretations $\mathcal{P}r$ according to them. Both orders should be combined into one, to determine the preferred model. A lexicographic order provides such an integration. The second criterion is chosen as the major criterion, since it is preferred to satisfy a more specific default than to satisfy less specific ones. 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 generic probabilistic terminology $\mathcal{P}_g$ 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 generic probabilistic terminology $\mathcal{P}_g$, written as $\mathcal{F} \Vdash\!\sim (D|C)[l, u]$ under $\mathcal{P}_g$, 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 $\mathcal{P}_g$ is defined as $\mathcal{F} \Vdash\!\sim_{tight} (D|C)[l, u]$ 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 terminology $\mathcal{P}$ for generic and assertional conditional constraints F.

If F is a generic conditional constraint, then it is said to be a lexicographic consequence of $\mathcal{P}$, written $\mathcal{P} \Vdash\!\sim F$ if and only if $\emptyset \Vdash\!\sim F$ under $\mathcal{P}_g$ and a tight lexicographic consequence of $\mathcal{P}$, written $\mathcal{P} \Vdash\!\sim_{tight} F$ if and only if $\emptyset \Vdash\!\sim_{tight} F$ under $\mathcal{P}_g$.

If F is an assertional conditional constraint for $o \in I_P$, then it is said to be a lexicographic consequence of $\mathcal{P}$, written $\mathcal{P} \Vdash\!\sim F$, if and only if $\mathcal{T}_o \cup \mathcal{D}_o \Vdash\!\sim F$ under $\mathcal{P}_g$ and a tight lexicographic consequence of $\mathcal{P}$, written $\mathcal{P} \Vdash\!\sim_{tight} F$ if and only if $\mathcal{T}_o \cup \mathcal{D}_o \Vdash\!\sim_{tight} F$ under $\mathcal{P}_g$.

4 Modelling the application scenario

Let us consider again the application scenario painted in the introduction. We will use modified elements of the athletics ontology from the BOEMIE project to demonstrate how the modelling with conditional constraints may be applied in this context. Concept definitions have been modelled as conditional constraints in the PTBox while the concept hierarchy remains in the TBox. Relevant axioms and conditional constraints which form the model of the application scenario are shown in Figure 2.

$\mathcal{T} = \{$
$High\_Jump \sqsubseteq Jumping\_Event$
$Pole\_Vault \sqsubseteq Jumping\_Event$
$High\_Jump \sqcap Pole\_Vault \sqsubseteq \bot\}$

$\mathcal{D}_g = \{$
$(High\_Jump|\exists hasPart.Bar)[0.4, 1.0]$
$(Pole\_Vault|\exists hasPart.Bar)[0.4, 1.0]$
$(Pole\_Vault|\exists hasPart.Pole)[0.7, 1.0]$
$(Pole\_Vault|\exists hasPart.Bar \sqcap \exists hasPart.Pole)[0.8, 1.0]\}$

Figure 2: Application scenario model

We assume that we gained the lower bound for our conditional constraints out of statistics over a reasonable amount of pictures from the athletics domain. With an upper bound of 1.0 we wish to express the default knowledge e.g. "If some hasPart.Bar is evident then this is a HighJump Event". More specific constraints having further evidences gain a higher lower bound probability and therefore their uncertainty interval becomes smaller.

The probabilistic lexicographic entailment may be used to decide the following two interesting probabilistic inference problems:

To compute a probabilistic subsumption for the concepts D, C the tight probabilistic lexicographic entailment is determined with respect to $\mathcal{P}_g$ and $\mathcal{F} = \emptyset$.

Probabilistic instance checking for an individual o and a concept E is done by computing the tight probabilistic lexicographic entailment with E set as conclusion and $\top$ as evidence and with respect to $\mathcal{P}_g$ and $\mathcal{F} = \mathcal{D}_o$.

Probabilistic instance checking is the reasoning service which we are using with respect to the generic probabilistic terminology in Figure 2 to determine the probability intervals of pictures showing specific jumping events. Out of a picture as shown in Fig. 1 image analysis and postprocessing might produce the following PAbox axioms :

$(\exists hasPart.Bar|\top)_{image1}[0.8, 1.0]$
$(\exists hasPart.Pole|\top)_{image1}[0.9, 1.0]$

Here the upper bound is set to 1.0 to express the optimism that the image extraction process identified the right concept. The lower bound is set to the probability provided by the image extraction process denoting the confidence in the identified concept. Notice that a concept which has been detected with a low confidence results in a larger interval of uncertainty.

We also investigated the consequences of a change of confidence in a detected concept with respect to probabilistic instance checking. The following values for $(PoleVault|\top)_{image1}[PVl, PVu]$ and $(HighJump|\top)_{image1}[HJl, HJu]$ have been computed while we varied the lower bound of $(\exists hasPart.Pole|\top)_{image1}[var, 1.0]$. The computed results are shown in Figure 3. How can we interpret these results in order to determine the preferred interpretation? If we interpret the results as points then we should consider the Euclidian distances to the two points $[0, 0]$ as negation point and $[0, 1]$ as ignorance point. Combining these two distances as a product we obtain a ranking for the interpretations. With this ranking the PoleVault results start to be preferred when the varied lower bound is between 0.4 and 0.5. Between 0.0 and 0.4 the results can not be discriminated.

var	0.0-0.4	0.5	0.6	0.7	0.8	0.9	1.0
HJu	0.68	0.65	0.58	0.51	0.44	0.37	0.3
HJl	0.32	0.32	0.32	0.32	0.32	0.0	0.0
PVu	0.68	0.68	0.68	0.68	0.68	1.0	1.0
PVl	0.32	0.35	0.42	0.49	0.56	0.63	0.7

Figure 3: Computed intervals

5 Implementation

The ContraBovemRufum system implements the algorithms presented by [10, 21] in the Java programming language. Figure 4 displays the generic systems architecture. For the reasoning tasks RacerPro with its JRacer interface is used. We are currently working on a version of JRacer which accesses RacerPro directly through a foreign function interface eliminating all network overhead. As solvers a native Java solver by Opsresearch and the Mosek linear program solver have been integrated.

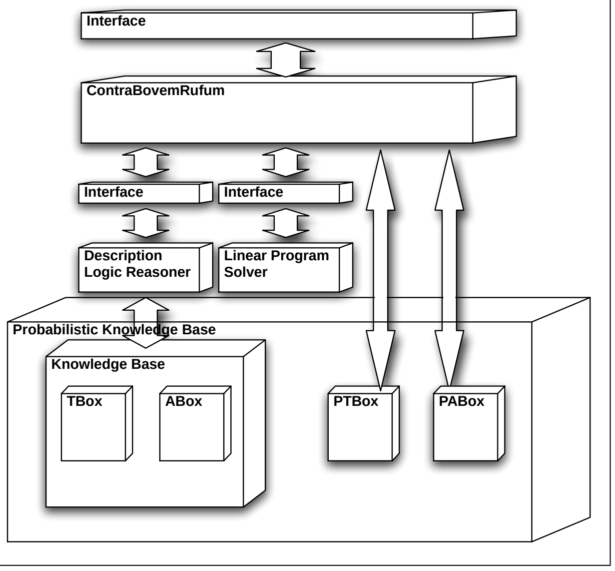

Figure 4: System architecture

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 PT and 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.

6 Analysis

Besides the successful implementation of the algorithm proposed in [10] a further objective was the analysis of average case behaviour. During the implementation it became obvious, that the algorithm used will perform in the average case like $O(2^n)$, where n is the number of conditional constraints within the ith part of the z-Partition.

In order to explain this conclusion, one has to take a closer look at the algorithm. The algorithm may be divided into three parts:

Part 1 tests, if the evidence concept is satisfiable against the TBox($\mathcal{T}_g$) and a set $\mathcal{F}$ of all ABox and PABox axioms bound to a certain individual. $\mathcal{F}$ is only relevant, if probabilistic concept membership is computed.

Part 2 computes lexicographic minimal sets of conditional constraints that are not in conflict with the verified evidence.

Part 3 computes the tightest entailed bounds using the previously computed sets and the conclusion concept.

The reason for poor average case performance is within the second part. Here the computation of the power set for each part $\mathcal{D}_i$ of the z-partition is required in order to iterate through all possible subsets $\mathcal{G}$. A power set has 2^n elements (see proof in [11]) and the algorithm visits all of them. Therefore the average complexity is determined to be $O(2^n)$ and it has been shown that the algorithm is intractable.

Here are some ideas on how to modify the algorithm to improve its average case performance:

As a first idea "lazy power set computation" is introduced. By this is meant, that one starts with $\mathcal{G}$ as the set containing all elements and then with all subsets, where we have $(n-1)$ elements and so on. The cardinality of these sets of subsets is given by $\binom{n}{n-i}$, where n is the number of elements and i the number of omitted elements.

If a set of subsets is found, where some sets are satisfiable, the process may be stopped since we are interested only in those kind of sets which satisfy as many conditional constraints as possible. Still for the worst case all subsets down to the level where the subsets only contain one element have to be computed. But on average the algorithm has to visit less subsets $\mathcal{G}$ than the implemented algorithm.

The second idea is to combine "lazy power set computation" with binary search. This works as follows:

We start with the set of subsets containing $\binom{n}{n-\lceil \frac{n}{2} \rceil}$ and test its elements for satisfiability. If at least one set is satisfied, the search continues within the upper half between n and $\lceil \frac{n}{2} \rceil$; otherwise the search continues in the lower half in the same manner.These two ideas have been already applied to the algorithm in [21] and implemented for ContraBovemRufum system.

A third idea is to introduce some heuristic method before performing binary search in order to reduce the search space. For example we could pick a random subset $\mathcal{G}$ of $\mathcal{D}_j$ and test it for satisfiability. If $\mathcal{G}$ is indeed satisfiable the cardinality of $\mathcal{G}$ is set as new lower bound for the binary search.

Further analysing the implemented software following observations were made. The time needed to compute one column of the application scenario in Figure 3 on a Pentium4@2,8Ghz with 1GB main memory was reduced to 7 seconds with binary search solving 44 linear programs coming from 8,6 seconds with full power set computation solving 62 linear programs. Still such a manageable scenario as ours already needs quite a bit of processing time. Therefore we also looked for opportunities to increase the performance without changing the implementation. Here both external software components come into view.

A change of the linear program solver did not improve the systems performance for the application scenario. Inspecting the log of the mosek solver showed that solving the linear problems with maximal number of 9 variables and 15 linear constraints required almost no cpu time. Therefore the time for our computations was spent in a different place.

Setting up one linear programs involves at most 2 reasoner calls per variable and conditional constraint, thus for the above mentioned case generating at most 126 calls. A previous naive implementation, which made 4 reasoner calls per variable and conditional constraint, spent 12,3 seconds and 15 seconds respectively. So reducing the costs to call the reasoner would achieve a significant performance gain. The communication with the reasoner involves significant overhead out of two sources, network communication and parsing of concept terms. The first source was eliminated by accessing RacerPro as a library via a foreign function interface. Here the computing times come down to 6 seconds and 7,3 seconds. To address the parsing of concept terms significant programming effort is necessary and would involve porting the system.

7 Conclusion

We presented the implementation of new probabilistic lexicographic reasoning services and discussed the underlying techniques. The complexity lies within the non-monotonic part of the logic, the computation of lexicographic minimal sets. It has been proven that the algorithm proposed [10] is intractable. The algorithm presented in [21] has significantly better performance. Still it is not expected to scale for large PTBoxes as the presented computing times indicate. These results are not unexpected because the algorithms were designed to prove sound and completeness.

The discussed application scenario shows that this probabilistic description logic is well suited to model this task. It also demonstrates that the non-monotonic part of the logic is needed for the proposed way of modelling. However if we find a way to model which avoids conflicting constraints then we only need tight logical entailment to determine the probabilistic inference problems. This means that only one linear program has to be set up, which has to be maximised and minimised to compute upper and lower bound. In this case it is expected that significant larger PT-Boxes can be supported.

The discussed optimisations show that computational time was halved and the presented optimisation ideas indicate that there is still room for further improvements.

In the future we will investigate further optimisations for the implemented algorithm in order to improve its average case behaviour. If the reader is interested in verifying the presented results on his own, the implantation can be obtained form the following site under the section software:
`http://www.sts.tu-harburg.de/%7Et.naeth/`

References

[1] Or-objects, 2000.

[2] Racerpro reference manual, 2005. Version 1.9.

[3] Racerpro user's guide, 2005. Version 1.9.

[4] The mosek java api manual version 4.0. MOSEK ApS, 2006.

[5] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[6] Fahiem Bacchus. *Representing and reasoning with probabilistic knowledge: a logical approach to probabilities.* MIT Press, Cambridge, MA, USA, 1990.

[7] Rachel A. Bourne and Simon Parsons. Connecting lexicographic with maximum entropy entailment. In *ESCQARU*, pages 80–91, 1999.

[8] Zhongli Ding and Yun Peng. A probabilistic extension to ontology language owl. *hicss*, 04:40111a, 2004.

[9] Rosalba Giugno and Thomas Lukasiewicz. P-$\mathcal{SHOQ}(\mathbf{D})$: A probabilistic extension of $\mathcal{SHOQ}(\mathbf{D})$ for probabilistic ontologies in the semantic web. In *JELIA*, pages 86–97, 2002.

[10] Rosalba Giugno and Thomas Lukasiewicz. P-$\mathcal{SHOQ}(\mathbf{D})$: A probabilistic extension of $\mathcal{SHOQ}(\mathbf{D})$ for probabilistic ontologies in the semantic web. Technical Report INFSYS RR-1843-02-06, TU Wien, 2002.

[11] Jr. H. F. Mattson. *Discrete Mathematics with Applications*, chapter 3, pages 120–121. John Wiley & Sons, Inc., 1993.

[12] V. Haarslev and R. Möller. Expressive abox reasoning with number restrictions, role hierarchies, and transitively closed roles. In Fausto Giunchiglia and Bart Selman, editors, *Proceedings of Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR2000), Breckenridge, Colorado, USA, 12-15 April*, pages 273–284. Morgan Kaufmann, 2000.

[13] V. Haarslev, R. Möller, and M. Wessel. The description logic alcnhr+ extended with concrete domains: A practically motivated approach. In R. Goré, A. Leitsch, and T. Nipkow, editors, *International Joint Conference on Automated Reasoning, IJCAR'2001, June 18-23, Siena, Italy*, pages 29–44. Springer-Verlag, 2001.

[14] Rolf Haenni. Towards a unifying theory of logical and probabilistic reasoning. In *ISIPTA*, pages 193–202, 2005.

[15] Theodore Hailperin. *Boole's Logic and Probability.* Elsevier Science Publishers B.V., second edition edition, 1986.

[16] I. Horrocks and U. Sattler. Ontology reasoning in the SHOQ(D) description logic. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, 2001.

[17] I. Horrocks, U. Sattler, and S. Tobies. Reasoning with individuals for the description logic SHIQ. In David MacAllester, editor, *Proceedings of the 17th International Conference on Automated Deduction (CADE-17)*, number 1831, Germany, 2000. Springer Verlag.

[18] Alissa Kaplunova and Ralf Möller. Probabilistic LCS in a P-Classic Implementation. Technical report, Institute for Software Systems (STS), Hamburg University of Technology, Germany, 2007. See http://www.sts.tu-harburg.de/tech-reports/papers.html.

[19] Daphne Koller, Alon Y. Levy, and Avi Pfeffer. P-classic: A tractable probabilistic description logic. In *AAAI/IAAI*, pages 390–397, 1997.

[20] Daniel Lehmann. Another perspective on default reasoning. *Annals of Mathematics and Artificial Intelligence*, 15:61–82, 1995.

[21] T. Lukasiewicz. Probabilistic description logics for the semantic web. Technical Report INFSYS RR-1843-06-05, TU Wien, 2006.

[22] John McCarthy. Circumscription - a form of non-monotonic reasoning. *Artif. Intell.*, 13(1-2):27–39, 1980.

[23] Drew V. McDermott and Jon Doyle. Non-monotonic logic i. *Artif. Intell.*, 13(1-2):41–72, 1980.

[24] R. Möller and B. Neumann. Ontology-based reasoning techniques for multimedia interpretation and retrieval. In *Semantic Multimedia and Ontologies : Theory and Applications.* 2007. to appear.

[25] Tobias H. Näth. Analysis of the average-case behavior of an inference algorithm for probabilistic description logics. Diplomarbeit, TU Hamburg-Harburg, February 2007.

[26] B. Neumann and R. Möller. On scene interpretation with description logics. *Image and Vision Computing, Special Issue on Cognitive Vision*, 2007. to appear.

[27] Gerhard Paass. *Non-Standard Logics for Automated Reasoning*, chapter 8, pages 213–251. Academic Press Limited, 1988.

[28] Judea Pearl. *Probabilistic Resoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, Inc., 1988.

[29] Raymond Reiter. A logic for default reasoning. *Artif. Intell.*, 13(1-2):81–132, 1980.

[30] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach.* Prentice-Hall, Englewood Cliffs, NJ, 2nd edition, 2003.

[31] Umberto Straccia. `fuzzydl`, 2007.