A Formal Framework for Theory Learning using Description Logics

Jordi Alvarez

TALP Research Center
Universitat Politècnica de Catalunya
Mòdul C6, Campus Nord, 08034 Barcelona
jalvarez@talp.upc.es

Abstract. This paper introduces a formal base in order to talk about theory learning from a Description Logics (DL) perspective. A probabilistic Description Logics is introduced; and its need is intuitively justified. Theory learning is defined using information theory concepts and the probabilistic DL framework. Once this has been done, a general theory learning environment is constructed on this theory. This environment is based on the application of a set of induction rules. New rules can be defined easily using a set of syntactic manipulators that modify concept expressions.

1 Introduction

Theory learning has been a main goal of machine learning from its beginning. Inductive Logic Programming (ILP) is the only area in machine learning that uses a representational formalism rich enough to make possible to learn an expressive theory.

Some work has been done also in the DL community. DL is an evolution of semantic networks and frame system that uses model theory to give a formal base to the representation and reasoning system [6]. Most DL learning stuff is related with the computation of the *Least Common Subsumer* (LCS), introduced in [3] as an adaptation of *Relative Least General Generalization* to the DL field. See for example [4, 7]. The work of [11] by one hand, and [5] by another, try to acquire a whole theory (using the LCS computation as a subtask).

All this DL work, and most ILP one have been done from a concept learning perspective. In this paper I establish a formal framework that addresses the problem of theory learning as a whole, from a quite general point of view. In order to achieve that, model theory has been used in the reverse direction in which is used in DL formalization. Information theory measures are used in order to give a concrete definition for theory learning. This definition provides a way to devise algorithms that deal with theory learning as a whole.

The formalization introduced in this paper is based in probabilistic DL. Probabilistic DL has been boarded before in [10, 8, 12]. In section 3, a new and simple approach to probabilistic DL is introduced.

A learning algorithm based on the introduced theory has been designed and a preliminary implementation performed as the YAYA system.

Although the theory introduced in this paper differs from ILP theory in some basic things (specially in the underlying representation being used), the YAYA learning system described later has a lot of commonalities with ILP systems. And the usage of DL representation formalism in combination with ILP techniques can be very interesting since one of the main goals of DL systems is to effectively and efficiently represent hierarchical knowledge and other knowledge patterns that usually appear in domain theories. Nevertheless, this has to be further explored.

2 Description Logics

DL was born from the need to provide a formal and homogeneous base to semantic network and frame systems. This need is achieved by one hand by the clear distinction between intensional and extensional knowledge; and by other hand by the definition of the language used to express intensional language, commonly called *Concept Language*, and the definition of its semantics in a model theoretic way.

2.1 Concept Language

The concept language is usually a subset of first order logics with equality; although its syntax is a bit different. Figure 1 shows the syntax and semantics for YAYA concept language concept-forming constructors, and also for YAYA role constructors. The reader can refer to [6] for a deep description of DL topics.

There exist many DL systems with different concept languages. In this paper we will restrict to YAYA concept language; which is constituted by the core language $\mathcal{ALCN}(\top, \bot, A, \sqcap, \sqcup, \exists, \forall, \leq_n, \geq_n$, and P constructors), plus the loop concept constructor. Definitions in section 3 and section 4 are valid for any DL concept language having at least concept name negation (negation restricted to concept names).

From now on, we will use $\mathcal{L}$ to refer to the YAYA concept language and the set of terms in the concept language indistinctly. We will also use $\mathcal{LR}$ as the set of role expressions that can be constructed in $\mathcal{L}$.

A *conceptual interpretation* $\mathcal{I} = < \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} >$ consists of a non-empty set $\Delta^{\mathcal{I}}$ (the domain); and a function $\cdot^{\mathcal{I}}$ that maps every concept in $\mathcal{L}$ into a subset of $\Delta^{\mathcal{I}}$, and every role expression in $\mathcal{LR}$ into a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$[1].

[1] We have introduced the term *conceptual interpretation* in order to distinguish it from what will later be called *specific interpretation.* DL literature defines an only notion of interpretation. Our motivation to distinguish two slightly different notions of interpretation will be clear later, when the process of learning a theory from a set of models is defined. From now on, the term interpretation will be used when there is no ambiguity or when it is not important to distinguish which notion of interpretation are we using.

Constructor name	Syntax	Semantics
Concept name	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
Top	$\top$	$\Delta^{\mathcal{I}}$
Bottom	$\bot$	$\varnothing$
Negation ($\mathcal{C}$)	$\neg\ C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
Conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Disjunction ($\mathcal{U}$)	$C \sqcup D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Universal quantification	$\forall\ R.C$	$\{d_1 \mid \forall d_2 : (d_1, d_2) \in R^{\mathcal{I}} \rightarrow d_2 \in C^{\mathcal{I}}\}$
Existential quantification ($\mathcal{E}$)	$\exists\ R.C$	$\{d_1 \mid \exists d_2 : (d_1, d_2) \in R^{\mathcal{I}} \wedge d_2 \in C^{\mathcal{I}}\}$
Number restrictions ($\mathcal{N}$)	$(\geq\ n\ S)$	$\{d_1 \mid \#\{d_2 : (d_1, d_2) \in S^{\mathcal{I}}\} \geq n\}$
	$(\leq\ n\ S)$	$\{d_1 \mid \#\{d_2 : (d_1, d_2) \in S^{\mathcal{I}}\} \leq n\}$
Loop	$[R]$	$\{d \mid (d, d)^{\mathcal{I}} \in R^{\mathcal{I}}\}$

Constructor name	Syntax	Semantics
Role name	P	$P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
Inverse role	R^-	$\{(d_1, d_2) \mid (d_2, d_1) \in R^{\mathcal{I}}\}$
Role composition	$R_1 \circ R_2$	$\{(d_1, d_2) \mid \exists e \in \Delta \wedge (d_1, e)^{\mathcal{I}} \in R_1^{\mathcal{I}} \wedge (e, d_2)^{\mathcal{I}} \in R_2^{\mathcal{I}}\}$

Table 1. Syntax and semantics of YAYA concept language constructs (first table), and YAYA role language constructs (second one). A is a concept name, C and D are general concept expressions, R (and R_i) can be any kind of role expressions, S must be a simple (non-composite) role, P is a role name, and d_i, e are ABox individuals.

Then, an interpretation $\mathcal{I}$ is a *model* for a concept C if $C^{\mathcal{I}}$ is nonempty. A concept is *satisfiable* if it has a model. A concept C *is subsumed* by another concept D if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for every interpretation $\mathcal{I}$. Two concepts C and D are *equivalent* if for every interpretation $\mathcal{I}$ $C^{\mathcal{I}} = D^{\mathcal{I}}$.

2.2 Knowledge Bases

A knowledge base Σ is defined as a pair constituted by a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$ ($\Sigma = <\mathcal{T}, \mathcal{A}>$).

Intensional Knowledge $\mathcal{T}$ contains the intensional knowledge and is constituted by a set of axioms. Different types of axioms do appear in the DL literature (see [6] for details). Depending on the type of axioms allowed by a DL system, it will be more or less expressive. YAYA only deals with general *concept inclusion axioms*, that have the following form:

$$C \leq D \tag{1}$$

where C and D are $\mathcal{L}$ terms.

All other types of axiom can be reformulated as general concept inclusion axioms. Different semantics can be used in order to define the semantics for

TBoxes with general concept inclusions[2]. We use the so-called *descriptive semantics*, following the approach of [2] (instead of the more restrictive least and greatest fix-point semantics [14]).

Then, an interpretation $\mathcal{I}$ satisfies the axiom $C \leq D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. And an interpretation $\mathcal{I}$ is a *model* for a TBox $\mathcal{T}$ if it satisfies all the axioms in $\mathcal{T}$.

For our convenience, we define:

$$Models(\mathcal{T}) \doteq \{\mathcal{I} | \mathcal{I} \; is \; a \; model \; for \; \mathcal{T}\} \tag{2}$$

Extensional Knowledge The ABox $\mathcal{A}$ provides the extensional knowledge of a knowledge base Σ. The ABox provides information about a set of concrete objects called *individuals.*

The notion of conceptual interpretation $\mathcal{I} =< \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} >$ is extended to that of *specific interpretation* in order to cope with individuals. In a specific interpretation, the function $\cdot^{\mathcal{I}}$ also maps every individual for which $\mathcal{A}$ provides information to an element of $\Delta^{\mathcal{I}}$.

The extensional knowledge is provided as a set of assertions. Every assertion is of the form:

C(a) concept membership assertion
R(a,b) role membership assertion

where $C \in \mathcal{L}$, $R \in \mathcal{LR}$, and a, b are individuals.

We define a *simple assertion* as a concept membership assertion of the form $A(a)$, or a role membership assertion of the form $S(a, b)$, where A is a concept name, S is a role name, and a, b are individuals.

An interpretation $\mathcal{I} =< \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} >$ satisfies $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$. And $\mathcal{I}$ satisfies $R(a, b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$.

An ABox $\mathcal{A}$ is defined to be a set of assertions. Then, an interpretation $\mathcal{I}$ is a *model* of $\mathcal{A}$ if it satisfies all the assertions in $\mathcal{A}$. $\mathcal{A}$ is *satisfiable* if it has a model.

The algorithm used in order to check concept subsumption, concept satisfiability, and Σ-satisfiability is based on the tableaux calculus and is a mixture of the algorithms in [2, 9] in order to fit YAYA concept language.

3 Probabilistic Description Logics

YAYA TBox allows probabilistic axioms of the form:

$$C \leq_{\alpha,\sigma} D \tag{3}$$

[2] General concept inclusions allow to have axioms stating for example that all mammals are son of another mammal ($mammal \subseteq \exists parent.mammal$) ...

Where C and D are $\mathcal{L}$ concept expressions, α and σ are real numbers such that $\alpha \in [0,1]$, and $\sigma \geq 0$.

Axioms of this form will be called *probabilistic concept inclusions*. Axiom $C \leq_{\alpha,\sigma} D$ states that the conditional probability $P(D|C)$ follows a normal distribution centered in α and with a standard deviation of σ in the set of interpretations satisfying it.

Unfortunately, it is impossible to give a criterium in order to state when a given interpretation satisfies a probabilistic concept inclusion. Instead, the notion of *β-satisfiability* is introduced.

Given an interpretation $\mathcal{I} =< \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} >$, we say that $\mathcal{I}$ *β-satisfies* the axiom $C \leq_{\alpha,\sigma} D$ if $P(D^{\mathcal{I}}|C^{\mathcal{I}}) \in N(\alpha, \sigma)$ with probability $\geq 1 - \beta$.

The notion of $\mathcal{T}$-model is also extended to *β-plausible $\mathcal{T}$-model*. A conceptual interpretation $\mathcal{I}$ is a β-plausible model of $\mathcal{T}$ if $\mathcal{I}$ β-satisfies all the axioms in $\mathcal{T}$.

Then, an important property about TBoxes with probabilistic axioms (probabilistic TBoxes) is that they *preserve monotonicity from a model point of view*; that is, given two probabilistic TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$, if $\mathcal{T}_1 \subseteq \mathcal{T}_2$ then $Models_\beta(\mathcal{T}_1) \subseteq Models_\beta(\mathcal{T}_2)$[3].

From now on, we will refer to β-plausible models of a given $\mathcal{T}$ for a fixed β simply as models of $\mathcal{T}$.

3.1 Axiom Estimation

Probabilistic tboxes allow to easily "extract" axioms from a set of interpretations. And that is what theory learning is about. In our probabilistic environment, given a theory $\mathcal{T}$; then, for every two concepts C and D we can find an α and a σ for which we can build an axiom $\Phi = C \leq_{\alpha,\sigma} D$ such that $Models(\mathcal{T} \cup \Phi) = Models(\mathcal{T})$.

For every interpretation $\mathcal{I}$ and every pair of concepts C and D, we have a set of boolean variables, one for every element $d \in C^{\mathcal{I}}$, stating whether $d \in D^{\mathcal{I}}$ or not. Using the *central limit theorem*, for sufficiently large values of $|C^{\mathcal{I}}|$, we can approximate $P(D^{\mathcal{I}}|C^{\mathcal{I}})$ by a *normal distribution*.

Then, given a set of interpretations $\mathcal{M} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_n\}$ constituted by randomly chosen models of a TBox $\mathcal{T}$; and given two concepts C and D, we can *always* compute α and σ such that $Models(\mathcal{T}) \subseteq Models(\{C \leq_{\alpha,\sigma} D\})$. This fact is quite important, and will refereed later in the description of the learning algorithm.

An estimation for α can be computed as the mean if $P(D^{\mathcal{I}}|C^{\mathcal{I}})$ in $\mathcal{M}$. And a confidence interval for σ can be obtained by performing a χ^2 statistics test. Details about axiom estimation can be found at [1].

4 Theory Learning

In the previous section, the representation system that will be used to represent the learned theory has been explained, and its semantics has been defined. In

[3] $Models_\beta(\mathcal{T})$ is the set of β-plausible models of $\mathcal{T}$.

this section we will formally define the process of theory learning from a pair $< \mathcal{T}_0, \mathcal{M} >$; where $\mathcal{T}_0$ is an initial theory we have, and $\mathcal{M} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots \mathcal{I}_n\}$ is a set of conceptual interpretations. $\mathcal{T}_0$ is a subset of the theory to be learned $\mathcal{T}^*$ and is commonly referred as *background knowledge*, and the conceptual interpretations are supposed to be models of $\mathcal{T}^*$.

The reasoning procedures performed by DL systems can be seen as the process of determining the set of models of a given knowledge base; and then check some properties over those models (that corresponds to the obtained result). In the same way, theory learning can be seen as the process of determining a theory $\mathcal{T}^*$ from a set of models of it.

4.1 Model Representation

We will represent interpretations through a particular type of ABoxes that are defined below. In order to do so, we first define an *equivalence relation* between conceptual interpretations. We say two conceptual interpretations $\mathcal{I}_1$ and $\mathcal{I}_2$ are *equivalent* if there exists a bijection τ between $\Delta^{\mathcal{I}_1}$ and $\Delta^{\mathcal{I}_2}$ such that:

1. $C^{\mathcal{I}_1} = \{t | \tau(t) \in C^{\mathcal{I}_2}\}$ for every $C \in \mathcal{L}$, and
2. $R^{\mathcal{I}_1} = \{(s,t) | (\tau(s), \tau(t)) \in R^{\mathcal{I}_2}\}$ for every $R \in \mathcal{LR}$

The representation $\mathcal{A}$ of an interpretation $\mathcal{I}$ should be such that $\Sigma =< \varnothing, \mathcal{A} >$ will have as models all *specific interpretations* that are $\mathcal{A}$-extensions of *conceptual interpretations equivalent* to $\mathcal{I}$[4]; and no other interpretation should be a model of $\mathcal{A}$. This assures us that only the domain objects may change in $\mathcal{A}$ models; but the behaviour of objects in the domain and the relations among those objects are completely defined by $\mathcal{A}$. τ relates objects with the same behavior.

A new kind of ABox assertion is defined: *role non-membership assertion* $\neg\, R(a,b)$, where $R \in \mathcal{LR}$, and a, b are individuals[5]. We extend also the notion of simple assertion to this new kind of assertions. An interpretation $\mathcal{I}$ satisfies the assertion $\neg\, R(a,b)$ iff $(a,b) \notin R^{\mathcal{I}}$. Then, we define an *extended ABox* as an ABox allowing role-nonmembership assertions; and an extended simple ABox as an extended ABox containing only simple assertions.
The representation of a conceptual interpretation $\mathcal{I}$ can be computed in the following way:

1. For every object o_i in $\Delta^{\mathcal{I}}$ we define a new individual a_i. We also define a function $\cdot^{\mathcal{R}}$ that maps every object o_i to its corresponding individual a_i.
2. For every object $o \in \Delta^{\mathcal{I}}$ and every concept name $A \in \mathcal{L}$ add the assertion $A(o^{\mathcal{R}})$ if $o \in A^{\mathcal{I}}$, and $\neg A(o^{\mathcal{R}})$ if $o \notin A^{\mathcal{I}}$.

[4] The term $\mathcal{A}$-extension is used here in the sense that a conceptual interpretation is converted to a concrete interpretation by adding a mapping from $\mathcal{A}$ individuals to $\Delta^{\mathcal{I}}$.

[5] This is done in order to keep $\mathcal{L}$ free from expressivity restrictions such as having role negation.

3. For every pair of objects $(o_1, o_2) \in \Delta^{\mathcal{I}}$ and every role name $S \in \mathcal{LR}$ add the assertion $R(o_1^{\mathcal{R}}, o_2^{\mathcal{R}})$ if $(o_1, o_2) \in R^{\mathcal{I}}$, and $\neg R(o_1^{\mathcal{R}}, o_2^{\mathcal{R}})$ if $(o_1, o_2) \notin R^{\mathcal{I}}$.

We will refer to the extended ABox computed from a conceptual interpretation $\mathcal{I}$ using this procedure as: *ExtendedRepresentation($\mathcal{I}$)*. Note that *ExtendedRepresentation($\mathcal{I}$)* is constituted only by simple assertions.

4.2 Quantifying Description Logics Information

The *entropy* of a conceptual interpretation $\mathcal{I}$, *entropy*$(\mathcal{I})$, is defined as the number of assertions in *ExtendedRepresentation*$(\mathcal{I})$.

The entropy of $\mathcal{I}$ has a direct interpretation as the number of boolean variables necessary to represent $\mathcal{I}$. If no knowledge is available, for every concept name A, and every element $d \in \Delta^{\mathcal{I}}$ we need a boolean random variable stating whether $d \in A^{\mathcal{I}}$. Additionally, for every role name R, and every element pair $(d_1, d_2) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ we need a boolean random variable stating whether $(d_1, d_2) \in R^{\mathcal{I}}$. When no knowledge is available (and then we suppose that the probability for true is the same that the probability for false, and is equal to $\frac{1}{2}$), the entropy for any of these boolean random variables is $-\frac{1}{2} * log\frac{1}{2} - \frac{1}{2} * log\frac{1}{2} = 1$. Every assertion in $\mathcal{A}_E$ represents the value of one of these boolean variables.

So, the notion of entropy that has been introduced here is in fact a direct adaptation of Shannon measure.

The minimum expression of a given extended simple ABox $\mathcal{A}_E$ with respect to a TBox $\mathcal{T}$, $Minimum_{\mathcal{T}}(\mathcal{A})$, is defined as an extended simple ABox $\mathcal{A}_0$ such that $Models(< \mathcal{T}, \mathcal{A}_0 >) = Models(< \mathcal{T}, \mathcal{A} >)$ and $|\mathcal{A}_0 = minimum(\{|\mathcal{A}_E| : Models(\mathcal{A}) = Models(\mathcal{A}_E)\}$[6].

Then, we can define the amount of *redundant information* in an extended simple ABox $\mathcal{A}$ with respect to a TBox $\mathcal{T}$, $Redundant_{\mathcal{T}}(\mathcal{A})$, as $|\mathcal{A}| - |Minimum_{\mathcal{T}}(\mathcal{A})|$.

Now we are able to talk about the amount of information encoded in a TBox $\mathcal{T}$ from a model theoretic point of view. For this purpose, we define:

- The amount of information provided by $\mathcal{T}$ with respect to a model $\mathcal{I}$: $Info_{\mathcal{I}}(\mathcal{T}) = Redundant_{\mathcal{T}}(ExtendedRepresentation(\mathcal{I}))$
- The amount of information provided by $\mathcal{T}$ with respect to a set of conceptual interpretations $\mathcal{M} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots \mathcal{I}_n\}$ that are models of $\mathcal{T}$: $Info_{\mathcal{M}}(\mathcal{T}) = \frac{\sum_{\mathcal{I} \in \mathcal{M}} Info_{\mathcal{I}}(\mathcal{T})}{n}$

4.3 Learning a TBox

Let's go back to our problem. We want to learn a TBox $\mathcal{T}^*$ from a pair $< \mathcal{T}_0, \mathcal{M} >$ constituted by an initial theory $\mathcal{T}_0$ such that $\mathcal{T}_0 \subseteq \mathcal{T}^*$, and a set of $\mathcal{T}^*$-models $\mathcal{M} = \{\mathcal{I}_1, \mathcal{I}_2, \ldots \mathcal{I}_n\}$.

Then, we define the process of learning a TBox $\mathcal{T}^*$ from a pair $< \mathcal{T}_0, \mathcal{M} >$ as the process of acquiring a set of axioms $\{\Phi_1, \Phi_2, \ldots \Phi_m\}$ such that:

[6] Note that there can be several $\mathcal{A}_E$ that minimize $|\mathcal{A}_E|$. $Minimum_{\mathcal{T}}(\mathcal{A})$ is one of them.

1. $\mathcal{T}^* = \mathcal{T}_0 \cup \{\Phi_1, \Phi_2, \dots \Phi_m\}$
2. $M \subseteq Models(\mathcal{T}^*)$
3. $\forall\, TBox\ \mathcal{T} : \mathcal{T}^* \subset \mathcal{T} \wedge \mathcal{M} \subseteq Models(\mathcal{T}) \Rightarrow Info_{\mathcal{M}}(\mathcal{T}) = Info_{\mathcal{M}}(\mathcal{T}^*)$
4. $\forall\, TBox\ \mathcal{T} : \mathcal{T}_0 \subseteq \mathcal{T} \wedge \mathcal{T} \subset \mathcal{T}^* \Rightarrow Info_{\mathcal{M}}(\mathcal{T}) < Info_{\mathcal{M}}(\mathcal{T}^*)$

The first condition defines $\mathcal{T}^*$ as the union of $\mathcal{T}_0$ and the set of acquired axioms. The second one states that the set of conceptual interpretations from which we learn should be models of $\mathcal{T}^*$. The third point states that all TBoxes including $\mathcal{T}^*$ and having $\mathcal{M}$ as models provide no new information. And finally the fourth point states that no subset of $\mathcal{T}^*$ that has $\mathcal{T}_0$ as a subset provides as much information as $\mathcal{T}^*$ provides.

Before going on, we need some additional simple definitions:

– $\mathcal{T}$ is *consistent* with respect to $\mathcal{M} = \{\mathcal{I}_1, \mathcal{I}_2, \dots \mathcal{I}_n\}$ iff $\mathcal{M} \subseteq Models(\mathcal{T})$.
– An axiom Φ is *consistent* with respect to a pair $< \mathcal{T}, \mathcal{M} >$ iff $\mathcal{T} \cup \{\Phi\}$ is consistent with respect to $\mathcal{M}$.
– An axiom Φ is *interesting* with respect to a pair $< \mathcal{T}, \mathcal{M} >$ iff Φ is consistent with respect to $< \mathcal{T}, \mathcal{M} >$, and $Info_{\mathcal{M}}(\mathcal{T} \cup \{\Phi\}) > Info_{\mathcal{M}}(\mathcal{T})$.

The two latter definitions can also be extended to sets of axioms $\{\Phi_1, \Phi_2, \dots \Phi_m\}$.

Now, in order to provide a method for theory learning, we can benefit from the fact that our reasoning system is model-monotonic in order to convert the search in the space of TBoxes into a search in the space of $\mathcal{L}$ concept inclusion axioms. Then, given a TBox $\mathcal{T}$ such that $\mathcal{T}_0 \subseteq \mathcal{T} \subseteq \mathcal{T}^*$:

1. If an axiom Φ is found interesting for $\mathcal{T}$, it is because $\Phi \in \mathcal{T}^*$, or because $\mathcal{T}^*$ provides the information in Φ through other axioms not in $\mathcal{T}$.
2. If an axiom Φ is found to be not interesting for $\mathcal{T}$, we have that $\Phi \in \mathcal{T}$ or $\Phi \notin \mathcal{T}^*$.

This provides evidence for the known fact that the difficulty of learning a theory strongly depends on the theory itself, and on the set of examples used to learn it.

The problem of learning a theory for any pair $< \mathcal{T}, \mathcal{M} >$ is still intractable. So, we must perform some kind of tractable heuristic search. This search will not provide the exact solution but a TBox consistent with respect to $< \mathcal{T}, \mathcal{M} >$ that is *as close as possible* to $\mathcal{T}^*$.

5 The YAYA learning procedure

This section sketches YAYA, a learning system based on the theory described above.

A *YAYA learning state* is defined as $< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} >$. Where $\mathcal{T}$ is a TBox such that $\mathcal{T}_0 \subseteq \mathcal{T}$, $\mathcal{CN}$ is a set of concept names plus $\top$, $\mathcal{RN}$ is a set of role names, and $\mathcal{FC}$ is a set of forbidden and redundant $\mathcal{L}$ expressions (they help to importantly reduce the search space in some situations). None of the axioms in $\mathcal{T}$ contains a concept name that is not in $\mathcal{CN}$ nor a role name that is not in $\mathcal{RN}$. Concept expressions in $\mathcal{FC}$ cannot appear in a $\mathcal{T}$-axiom as a subconcept; and they can only appear in the left hand side of $\mathcal{T}$ axioms. The sets $\mathcal{CN}$ and $\mathcal{RN}$ restrict the set of concept names and role names that can take part in $\mathcal{T}$ axioms. cn and rn are subsets of the set of concept and role names appearing in $\mathcal{M}$.

The initial learning state is: $< \mathcal{T}_0, \{\top\}, \{S | S$ is a role name in $\mathcal{M}$ or its inverse $\}, \varnothing >$. So, YAYA learning system implements a *top-down* approach. As has been stated above, for any two concepts, and a set of models $\mathcal{M}$, we are able to find a probabilistic axiom $\Phi_1 = C \leq_{\alpha,\sigma} D$ such that $\mathcal{M} \sqsubset Models(\{\Phi_1\})$. This means that due to the use of probabilistic knowledge, YAYA learning system does not need any specialization step, as intermediate theories will never be overly specific.

YAYA learning algorithm performs a greedy search. The search is performed by a set of *induction rules* that modify the search state. Induction rules have been designed in order to overcome the drawbacks of greedy algorithms. They are applied according to a set of heuristics based information theory measures that can be computed efficiently and allow to compute an approximation to the measures defined in 4.3. The description of these measures can be found in [1].

5.1 Concept Generation

The set of concepts that can take part in axioms generated by the YAYA learning procedure are restricted by the set of induction rules provided by YAYA. In order to distinguish between the language supported by the reasoning system and the language the learning system is able to generate we will refer to the second one as $\mathcal{L}_G$. It holds that $\mathcal{L}_G \subset \mathcal{L}$. The $\mathcal{L}_G$ constructors are: $A,\ \top,\ \sqcap,\ \exists,\ \leq_1,\ \geq_n$ $, [R]\ , \circ$ and role inversion. Moreover, the way in which these constructors can be combined is also restricted by the set of induction rules.

The YAYA learner explores new axioms from the set of current axioms in $\mathcal{T}$. New axioms are generated by manipulating the concepts that take part in axioms in $\mathcal{T}$. There are several different concept manipulation procedures that are used by YAYA to obtain a concept from another one.
Before explaining them, we need some additional definitions:

- We define a *simple concept* as a concept containing only constructors in $\{A,\ \top,\ \bot\ , \sqcap,\ \sqcup,\ \leq_n,\ \geq_n\}$.
- $parent_{\mathcal{T},\mathcal{CN}}(A, B) \doteq \forall \mathcal{I} \in Models(\mathcal{T}) : A^{\mathcal{I}} \subseteq B^{\mathcal{I}}$, where $A, B \in \mathcal{CN}$
- $direct\_parent_{\mathcal{T},\mathcal{CN}}(A, B) \doteq parent(A, B) \wedge \forall A' \in \mathcal{CN} : \exists \mathcal{I} \in Models(\mathcal{T}) : A'^{\mathcal{I}} \subset A^{\mathcal{I}} \vee B^{\mathcal{I}} \subset A'^{\mathcal{I}}$, where $A, B \in \mathcal{CN}$.
- We will say a concept name A is the son (direct-son) of another concept name B if B is the parent (direct-parent) of A.

Now we are able to define concept manipulators:

- $append : \mathcal{L}_G \times \mathcal{L}_G \longrightarrow \mathcal{L}_G$
  $append(C, D)$ computes a concept subsumed by C in which D has been situated at the end of the chain of existentials of C.
  $append(G, C) \;=\; G \sqcap C$
  $append(G \sqcap \exists R.C, D) \;=\; G \sqcap \exists R.infix(C) \sqcap append(C, D)$
- $concept\_names : \mathcal{L}_G \longrightarrow 2^{\mathcal{CN}}$
  $concept\_names(C)$ returns a set with the set of concept names appearing in C.
  $concept\_names(A) \;=\; \{A\}$
  $concept\_names(\top) \;=\; \top$
  $concept\_names(G \sqcap C) \;= concept\_names(G) \cup concept\_names(C)$
  $concept\_names(\exists R.C) \;=\; concept\_names(C)$
  $concept\_names(\leq_m R) \;=\; concept\_names(\geq_m R) \;=\; \varnothing$
- $contained\_concepts : \mathcal{L}_G \longrightarrow 2^{\mathcal{L}_G}$
  $contained\_concepts(C)$ computes all the concept expressions that appear in C.
  $contained\_concepts(C) \;=\; \{D_1 | \exists D_2 | D_1 \in prefix(D_2) \wedge D_2 \in suffix(C)\}$
- $direct\_generalizations : \mathcal{L}_G \longrightarrow 2^{\mathcal{L}_G}$
  $direct\_generalizations(C)$ corresponds, for a given $\mathcal{T}$ and a given $\mathcal{CN}$, to the set of concepts in which only one concept name in C has been substituted by one of its direct parents.
  $direct\_generalizations(A) \;=\; \{A\} \cup \{B | direct\_parent_{\mathcal{T},\mathcal{CN}}(A, B)\}$
  $direct\_generalizations(G^-) \;=\; \{G^-\}$
  $direct\_generalizations(C \sqcap D) \;=\; \{E \sqcap D | E \in direct\_generalizations(D)\}$
  $\cup \{C \sqcap E | E \in direct\_generalizations(C)\}$
  $direct\_generalizations(\exists R.C) \;=\; \{\exists R.D | D \in direct\_generalizations(C)\}$
- $direct\_specializations : \mathcal{L}_G \longrightarrow 2^{\mathcal{L}_G}$
  $direct\_specializations(C)$ corresponds, for a given $\mathcal{T}$ and a given $\mathcal{CN}$, to the set of concepts in which only one concept name in C has been substituted by one of its direct parents.
  $direct\_specializations(A) \;=\; \{A\} \cup \{B | direct\_parent_{\mathcal{T},\mathcal{CN}}(A, B)\}$
  $direct\_specializations(G^-) \;=\; \{G^-\}$
  $direct\_specializations(C \sqcap D) \;=\; \{E \sqcap D | E \in direct\_specializations(D)\}$
  $\cup \{C \sqcap E | E \in direct\_specializations(C)\}$
  $direct\_specializations(\exists R.C) \;=\; \{\exists R.D | D \in direct\_specializations(C)\}$
- $existential\_chain : \mathcal{LR} \longrightarrow \mathcal{L}_G$
  $existential\_chain(CR)$ converts the sequence of simple roles in R to a chain of existentials. So, $existential\_chain(R_1 \circ R_2 \circ R_3 \ldots R_n) = \exists R_1.\exists R_2.\exists R_3 \ldots \exists R_n$
  $existential\_chain(R) \;=\; \exists R.\top$
  $existential\_chain(R \circ CR) \;=\; \exists R.existential\_chain(CR)$
- $generalizations : \mathcal{L}_G \longrightarrow 2^{\mathcal{L}_G}$
  $generalizations(C)$ corresponds, for a given $\mathcal{T}$ and a given $\mathcal{CN}$, to the set of concepts in which none, one or more of the concept names in C have been substituted by one of its parents.
  $generalizations(A) \;=\; \{A\} \cup \{B | parent_{\mathcal{T},\mathcal{CN}}(A, B)\}$
  $generalizations(G^-) \;=\; \{G^-\}$
  $generalizations(C \sqcap D) \;=\; \{E \sqcap D | E \in generalizations(D)\}$
  $\cup \{C \sqcap E | E \in generalizations(C)\}$
  $generalizations(\exists R.C) \;=\; \{\exists R.D | D \in generalizations(C)\}$

- $infix : \mathcal{L}_G \longrightarrow \mathcal{L}_G$
  $infix(C)$ is the most specific concept subsuming C in which there does not appear the $\exists$ constructor.
  $infix(G) \ = \ G$
  $infix(C \sqcap D) \ = \ infix(C) \sqcap infix(D)$
  $infix(\exists R.C) \ = \ \top$

- $invert : \mathcal{L}_G \longrightarrow \mathcal{L}_G$
  $invert(C)$ is the concept C seen from the end of the chain of existentials. For example, $invert(G_1 \sqcap \exists R_1.G_2 \sqcap \exists R_2.G_3 \sqcap \ldots \exists R_{n-1}.G_n) = G_n \sqcap \exists R^-_{n-1}.G_{n-1} \sqcap \exists R^-_{n-2}.G_{n-2} \ldots G_1$.
  $invert(G) \ = \ G$
  $invert(G \sqcap \exists R.C) \ = \ G_1 \sqcap \exists R_1.G_2 \sqcap \exists R_2.G_3 \ldots G_n \sqcap \exists R^-.G,$
  $where \ G_1 \sqcap \exists R_1.G_2 \sqcap \exists R_2.G_3 \ldots G_n \ is \ invert(C)$

- $obtain\_role : \mathcal{L}_G \longrightarrow \mathcal{LR}$
  $obtain\_role(C)$ Gets the structure of the concept (stated in the existentials in C) as a composite role. For example $obtain\_role(G_1 \sqcap \exists R_1.G_2 \sqcap \exists R_2.G_3 \ldots G_n) = R_1 \circ R_2 \circ \cdots \circ R_{n-1}$. $obtain\_role(\exists R_1.C_1 \sqcap \exists R_2.C_2)$ is undefined (as it would require role conjunction, not present in $\mathcal{LR}$). $obtain\_role(G)$ is also undefined.
  $obtain\_role(G \sqcap \exists R.C) \ = \ R \circ obtain\_role(C)$

- $prefixes : \mathcal{L}_G \longrightarrow 2^{\mathcal{L}_G}$
  $prefixes(C)$ is the set of concepts that are prefixes of C. In general, if $C = G_1 \sqcap \exists R_1.G_2 \sqcap \exists R_2.G_3 \ldots \exists G_n$, then $prefixes(C) = \{G_1, G_1 \sqcap \exists R_1.G_2, \ldots C\}$.
  $prefixes(G) \ = \ \{G\}$
  $prefixes(C \sqcap D) \ = \ specific(\{E|E \in generalizations(prefixes(C))\}$
  $\cup\{E|E \in generalizations(prefixes(D))\})$
  $prefixes(\exists R.C) \ = \ \{\top\} \cup \{\exists R.D|D \in prefixes(C)\}$

- $same\_as : \mathcal{L}_G \times \mathcal{L}_G \longrightarrow \mathcal{L}_G$
  Given two concepts C and D that should correspond to existential chains, $same\_as(C, D)$ compute a concept stating that the last individual in the existential chain of C is the same than the last individual in the existential chain of D.
  $same\_as(G_1, G_2) \ = \ \top$
  $same\_as(G_1, G_2 \sqcap \exists R.C) \ = \ [obtain\_role(\exists R.C)]$
  $same\_as(G_1 \sqcap \exists R_1.C_1, G_2 \sqcap \exists R_2.C_2) \ = \ [obtain\_role(\exists R_1.C_1) \circ obtain\_role(\exists R_2.C_2)^-]$

- $specializations : \mathcal{L}_G \longrightarrow 2^{\mathcal{L}_G}$
  $specializations(C)$ corresponds, for a given $\mathcal{T}$ and a given $\mathcal{CN}$, to the set of concepts in which none, one or more of the concept names in C have been substituted by one of its sons.
  $specializations(A) \ = \ \{A\} \cup \{B|parent_{\mathcal{T},\mathcal{CN}}(B, A)\}$
  $specializations(G^-) \ = \ \{G^-\}$
  $specializations(C \sqcap D) \ = \ \{E \sqcap D|E \in specializations(D)\}$
  $\cup\{C \sqcap E|E \in specializations(C)\}$
  $specializations(\exists R.C) \ = \ \{\exists R.D|D \in specializations(C)\}$

- $suffixes : \mathcal{L}_G \longrightarrow 2^{\mathcal{L}_G}$
  $suffixes(C)$ is the set of concepts that are suffixes of C. In general, if $C = G_1 \sqcap \exists R_1.G_2 \sqcap \exists R_2.G_3 \ldots \exists G_n$, then $suffixes(C) = \{G_n, G_{n-1} \sqcap \exists R_{n-1}.G_n, \ldots C\}$.
  $suffixes(G) \ = \ \{G\}$
  $suffixes(C \sqcap D) \ = \ specific(\{E|E \in generalizations(suffixes(C))\}$

$\cup\{E|E \in generalizations(suffixes(D))\})$
$suffixes(\exists R.C) \ = \ \{\exists R.C\} \cup suffixes(C)$

where A and B are concept names in $\mathcal{CN}$, $G, G_1 \dots G_n$ are simple concepts in $\mathcal{L}_G$, G^- is a simple concept with no concept names, C, D, E and F are $\mathcal{L}_G$ concepts, R is a role name or its inverse, and CR is a role expression.

In addition, we say that C is $\mathcal{T}$*-subsumed* by D iff $(C \sqsubseteq D) \vee (C \leq_{1,0} D \in \mathcal{T})$.

5.2 Induction Rules

The set of induction rules provided by YAYA learning system has been designed to reach as much as possible interesting axioms from an initially reduced set of axioms (if $\mathcal{T}_0 = \varnothing$, the starting point is axiom $\top \leq_{1,0} \top$).

As it has been said before, there is no need to generalize the theory in any moment. Then, all induction rules correspond to *ILP specialization operators.*

Due to the big amount of rules[7], I have preferred to maintain them classified into different classes. In this classification the terms *specialization* and *generalization* have been used in a different sense of that used for ILP operators.

Specialization rules specialize the *antecedent* or the *consequent* of an axiom. They can be justified intuitively from an information theoretic point of view: given axiom Φ in $\mathcal{T}$, it is probable that Φ is interesting because $\mathcal{T}^*$ contains a specialization of it, say Φ^*. Then, information in Φ^* is what makes Φ interesting.

Generalization rules generalize the *antecedent* or the *consequent* of an axiom. Given an axiom $\Phi^* \in \mathcal{T}^*$; we know that it is probable that given a specialization of it, Φ, Φ is interesting for $\mathcal{T}^* \setminus \{\Phi^*\}$. So, when an axiom is found to be interesting, it is possible that a generalization of it is still more interesting.

Besides these rules, YAYA provides some *general exploration rules*, and some *environment updating rules.* Before describing YAYA induction rules we define some functions that will be used in the rule definitions:

- $forbidden\_concept(C) \doteq \exists D_1, D_2 | D_1 \in contained\_concepts(C) \wedge D_2 \in \mathcal{FC} \wedge D_1 \sqsubseteq D_2$
- $C \trianglelefteq_{\mathcal{T}} D$, iff $((C \sqsubset D) \wedge (\exists D' : D \leq_{\alpha,\sigma} D' \in \mathcal{T}) \vee (C \leq_{1,0} D \in \mathcal{T})$
- We say that concept C is *hierarchically related* to concept D in $\mathcal{T}$ iff $C \trianglelefteq_{\mathcal{T}} D \vee D \trianglelefteq_{\mathcal{T}} C$

In order to provide examples for a few rules, let's introduce one of the domains for which YAYA has been tested: that of semantic parsing uswed in [16] (although the experiment was different from that in [16]). The domain consists of a set of over 1400 sentences of the form: *agent action [patient] [accomp] [instrument]*; where *agent*, *patient*, *action*, *accomp*, and *instrument* correspond to

[7] Although small is beautiful, the number of rules is necessarily big due to the big amount of $\mathcal{L}$ constructors and the differentiation of the antecedent and the consequent of an axiom (that results in providing different rules to treat the antecedent and the consequent).

individuals instance of a word sense from the WordNet lexical ontology [13][8]. Sentence examples are: "the girl ate the chicken with the fork" (where fork is the instrument), or "the man ate the chicken with the pasta" (where pasta is the accomp). Figure 1 shows the part of the taxonomy of WordNet senses that appear in sentences corresponding to the verb eat.

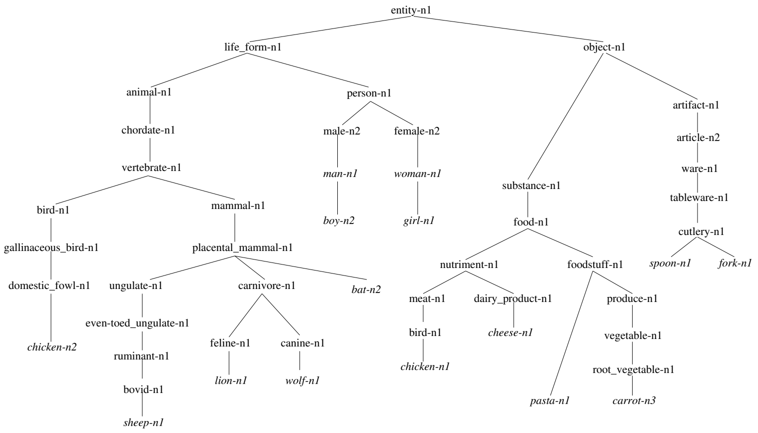

Fig. 1. A part of the WordNet sense taxonomy that is used in the semantic parsing exeperiment (italic names correspond to senses directly appearing in the sentences).

Now let's see the YAYA induction rules. For every rule, it is stated how the learning state is modified; and in which conditions the rule applies. The example domain is used only to provide examples for some of the rules.

cut consequent $(\longrightarrow_{\cdot\leq\cdot\setminus\exists})$ $<\mathcal{T},\mathcal{CN},\mathcal{RN},\mathcal{FC}> \longrightarrow <\mathcal{T}\cup\{\Phi\},\mathcal{CN},\mathcal{RN},\mathcal{FC}>$

1. $C \leq_{\alpha,\sigma} D \in \mathcal{T}$
2. $D = append(D', \exists R.G$, where G is a simple concept.
3. $\Phi = C \leq_{\mathcal{M}} D'$[9] is interesting with respect to $<\mathcal{T},\mathcal{M}>$[10].
4. $\neg\ forbidden\_concept(D')$

This is a consequent generalization rule.

equivalence rule $(\longrightarrow_{\equiv})$ $<\mathcal{T},\mathcal{CN},\mathcal{RN},\mathcal{FC}> \longrightarrow <\mathcal{T}\cup\{\Phi\},\mathcal{CN},\mathcal{RN},\mathcal{FC}>$

1. $C \leq_{1,0} D \in \mathcal{T}$
2. $\forall \mathcal{I} \in \mathcal{M} : D^{\mathcal{I}} = C^{\mathcal{I}}$
3. $\Phi = D \leq_{1,0} C$ is interesting for $<\mathcal{T},\mathcal{M}>$

expand antecedent $(\longrightarrow_{\cdot\exists\leq\cdot})$ $<\mathcal{T},\mathcal{CN},\mathcal{RN},\mathcal{FC}> \longrightarrow <\mathcal{T}\cup\{\Phi\},\mathcal{CN},\mathcal{RN},\mathcal{FC}>$

[8] WordNet senses are introduced as concept names.

[9] We will denote by $C \leq_{\mathcal{M}} D$ the axiom estimated from $\mathcal{M}$ for C and D.

[10] Computing whether an axiom is interesting for $\mathcal{T}$ with respect to $\mathcal{M}$ is approximated using the information theory measures described in [1].

1. $C \leq D' \in \mathcal{T}$ and D' is hierarchically related to D in $\mathcal{T}$.
2. $append(C_1, G) = C$, where G is a simple concept.
3. $C_2 = G \sqcap \exists R.A$, where $R \in \mathcal{RN}$ and $A \in \mathcal{CN}$.
4. $\Phi = append(C_1, C_2) \leq_{\mathcal{M}} D$ is interesting with respect to $< \mathcal{T}, \mathcal{M} >$.
5. $\neg\ forbidden\_concept(append(C_1, C_2))$

This rule is an antecedent specialization rule. Given axiom $C \leq D$, it specializes C by adding one more existential to the chain of existentials in D.

Example Imagine Φ = *eat-v1* $\leq_{0.2,0.7}$ $\exists$ *agent animal-n1*$\in \mathcal{T}$. The application of this rule would produce axiom Φ' = *eat-v1* $\sqcap$ $\exists$ *patient animal-n1* $\leq_{1,0}$ $\exists$ *agent animal-n1*[11] (taken that *animal-n1* $\in \mathcal{CN}$).

expand consequent $(\longrightarrow_{\cdot\leq\cdot\exists})$ $< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T} \cup \{\Phi\}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} >$

1. $C' \leq_{\alpha,\sigma} D \in \mathcal{T}$ and C' is hirarchically related to C in $\mathcal{T}$.
2. $append(D_1, G) = C$, where G is a simple concept.
3. $D_2 = G \sqcap \exists R.A$, where $R \in \mathcal{RN}$ and $A \in \mathcal{CN}$.
4. $\Phi = C \leq_{\mathcal{M}} append(D_1, D_2)$ is interesting with respect to $< \mathcal{T}, \mathcal{M} >$.
5. $\neg\ forbidden\_concept(append(D_1, D_2))$

This rule is a consequent specialization rule. Given axiom $C \leq D$, it tries to specialize D by adding one more existential to the chain of existentials in D.

feature rule $(\longrightarrow_{\leq_1})$ $< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T} \cup \{C \leq_{\alpha,\sigma} D'\}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} >$

1. $C \leq_{\alpha,\sigma} D \in \mathcal{T}$
2. $append(C_1, G \sqcap \exists R.C_2) = C$, where $\top \in suffixes(C_1) \wedge G \not\sqsubseteq \leq_1 R$
3. $D' = append(C_1, G \sqcap \leq_1 R \sqcap \exists R.C_2)$
4. $\forall \mathcal{I} \in \mathcal{M} : (C \sqcap D)^{\mathcal{I}} = (C \sqcap D')^{\mathcal{I}}$

This is a consequent specialization rule. Its use tries to specialize the consequent of an axiom Φ by adding $\leq_1$ restrictions to every one of the existentials taking part in Φ.

forbid same-as long paths $(\longrightarrow_{\mathcal{FC}\setminus[]})$
$< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T}', \mathcal{CN}, \mathcal{RN}, \mathcal{FC} \cup \{CR\} >$

1. $\{C \leq_{\alpha,\sigma} [R], [R] \leq_{\alpha,\sigma} C\} \cap \mathcal{T} \neq \varnothing$
2. $[R_1 \circ R_2]$ and R are equivalent, and the number of simple roles in R_1 is greater than the number of simple roles in R_2.
3. $CR = existential\_chain(R_1) \wedge CR \notin \mathcal{FC}$
4. $\Phi = CR \leq_{1,0} [R_1 \circ R_2]$ is consistent with respect to $< \mathcal{M}, \mathcal{T} >$.
5. $\mathcal{T}' = \mathcal{T} \cup \{\Phi\}$ if Φ is interesting with respect to $< \mathcal{T}, \mathcal{M} >$; and $\mathcal{T} = \mathcal{T}'$ otherwise.

This rule is an environment updating rule, and a general exploration rule at the same time. It checks whether the longer existential chain extracted from a loop construct is redundant. If it is, it adds the corresponding axiom to $\mathcal{T}$ and the existential chain to $\mathcal{FC}$.

generalize antecedent $(\longrightarrow_{\uparrow\leq\cdot})$ $< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T} \cup \{\Phi_2\}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} >$

[11] WordNet distinguishes between food senses and animal senses. Notice for example, that chicken-n2 in figure 1 corresponds to food; while chicken-n1 (not appearing in the figure) corresponds to animal.

1. $\Phi_1 = C \leq_{\alpha,\sigma} D' \in \mathcal{T}$ and D' is hierarchically related to D in $\mathcal{T}$.
2. $C = append(C_1, A \sqcap G \sqcap C_2)$
3. $C' = append(C_1, B \sqcap G \sqcap C_2)$, where $B \in direct\_parents(A)$
4. $\Phi_2 = C' \leq_{\mathcal{M}} D$, and Φ_2 is interesting for $< \mathcal{T}, \mathcal{M} >$.

This is a antecedent generalization rule. It is specially useful when large hierarchies of concept names are present in $\mathcal{T}_0$.

Example Imagine $\Phi = \textit{person-n1} \leq_{1,0} \exists\ agent^{-}\ eat \in \mathcal{T}$. As there are other sentences in which the agent is an animal, the antecedent would be generalized to *life_form-n1*, obtaining: $\Phi = \textit{life\_form-n1} \leq_{1,0} \exists\ agent^{-}\ eat$

generalize antecedent concept names $(\longrightarrow_{\uparrow_A^B \leq \cdot})$
$< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T}', \mathcal{CN}', \mathcal{RN}, \mathcal{FC} >$

1. $\Phi_1 = C \leq_{\alpha,\sigma} D' \in \mathcal{T}$ and D' is hierarchically related to D in $\mathcal{T}$.
2. $\Phi_2 = E \leq_{\alpha,\sigma} D$ such that $E \in generalizations(C) \wedge E \neq C \wedge \forall \mathcal{I} \in \mathcal{M} : (E \cap D)^{\mathcal{I}} = (C \cap D)^{\mathcal{I}}$
3. $\neg\ \exists \Phi' = F \leq_{\alpha,\sigma} D$ such that $F \in generalizations(E) \wedge F \neq E \wedge \forall \mathcal{I} \in \mathcal{M} : (F \cap D)^{\mathcal{I}} = (C \cap D)^{\mathcal{I}}$
4. $\Phi_3 = E \leq_{\mathcal{M}} C$
5. $\mathcal{T}' = \mathcal{T} \cup \{\Phi_2, \Phi_3\}$
6. $\mathcal{CN}' = \mathcal{CN} \cup concept\_names(E)$

This rule is an antecedent generalization rule and an environment updating rule at the same time (see the example in the specialize counterpart).

generalize consequent $(\longrightarrow_{\cdot \leq \uparrow})$ $< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T} \cup \{\Phi_2\}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} >$

1. $\Phi_1 = C' \leq_{\alpha,\sigma} D \in \mathcal{T}$ and C' is hierarchically related to C in $\mathcal{T}$.
2. $D = append(D_1, A \sqcap G \sqcap D_2)$
3. $D' = append(D_1, B \sqcap G \sqcap D_2)$, where $B \in direct\_parents(A)$
4. $\Phi_2 = C \leq_{\mathcal{M}} D'$, and Φ_2 is interesting for $< \mathcal{T}, \mathcal{M} >$.

This is a antecedent generalization rule. It is specially useful when large hierarchies of concept names are present in $\mathcal{T}_0$.

generalize consequent concept names $(\longrightarrow_{\cdot \leq \uparrow_A^B})$
$< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T}', \mathcal{CN}', \mathcal{RN}, \mathcal{FC} >$

1. $\Phi_1 = C' \leq_{\alpha,\sigma} D \in \mathcal{T}$ and C' is hierarchically related to C in $\mathcal{T}$.
2. $\Phi_2 = C \leq_{\alpha,\sigma} E$ such that $E \in generalizations(D) \wedge E \neq D \wedge \forall \mathcal{I} \in \mathcal{M} : (E \wedge C)^{\mathcal{I}} = (D \wedge C)^{\mathcal{I}}$
3. $\neg\ \exists \Phi' = C \leq_{\alpha,\sigma} F$ such that $F \in generalizations(E) \wedge F \neq E \wedge \forall \mathcal{I} \in \mathcal{M} : (F \wedge C)^{\mathcal{I}} = (D \wedge C)^{\mathcal{I}}$
4. $\Phi_3 = E \leq_{\mathcal{M}} D$
5. $\mathcal{T}' = \mathcal{T} \cup \{\Phi_2, \Phi_3\}$
6. $\mathcal{CN}' = \mathcal{CN} \cup concept\_names(E)$

This rule is a consequent generalization rule and an environment updating rule at the same time.

more than rule $(\longrightarrow_{\geq})$ $< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T} \cup \{C \leq_{\alpha,\sigma} D'\}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} >$

1. $C \leq_{\alpha,\sigma} D \in \mathcal{T}$
2. $append(C_1, G \sqcap \exists R.C_2) = C$, where $\top \in suffixes(C_1) \wedge G \not\sqsubseteq\ \leq_1 R$

3. $n > 1 \wedge n = max(m|\forall \mathcal{I} \in \mathcal{M} : (C \sqcap D)^{\mathcal{I}} = (C \sqcap append(C_1, G \sqcap \geq_m R \sqcap \exists R.C_2))^{\mathcal{I}}$
4. $D' = append(C_1, G \sqcap \geq_n R \sqcap \exists R.C_2)$

This is a consequent specialization rule. Its use tries to specialize the consequent of an axiom Φ by adding $\geq$ restrictions as restrictive as possible to every one of the existentials taking part in Φ.

pre-expand rule $(\longrightarrow_{\exists \cdot \leq \exists \cdot})$ $< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T} \cup \{\Phi\}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} >$

1. $C' \leq_{\alpha,\sigma} D \in \mathcal{T}$ and C' is hierarchically related to C in $\mathcal{T}$ or $C \leq D' \in \mathcal{T}$ and D' is hierarchically related to D in $\mathcal{T}$.
2. $E = A \sqcap \exists R.\top$, where $A \in \mathcal{CN}$ and $R \in \mathcal{RN}$.
3. $\Phi = append(E, C) \leq_{\mathcal{M}} append(E, D)$ is interesting with respect to $< \mathcal{T}, \mathcal{M} >$.
4. $\neg\ forbidden\_concept(append(E, C))$

This rule is a general exploration rule. Given axiom $\Phi^* \in \mathcal{T}^*$ such that $\Phi^* = C^* \leq_{\alpha^*,\sigma^*} D^*$; it is possible that axiom $\Phi = C \leq_{\alpha,\sigma} D$, where $C \in suffix(C^*)$ and $D \in suffix(D^*)$; is also interesting because of the subset of elements taking part in Φ that also take part in Φ^*. This rule tries to detect axioms in $\mathcal{T}^*$ from axioms that correspond to a "suffix" if it.

same-as consequent rule $(\longrightarrow_{\cdot \leq []})$ $< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T} \cup \{\Phi\}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} >$

1. $\{C \leq_{\alpha_1,\sigma_1} D_1, C' \leq_{\alpha_2,\sigma_2} D_2\} \in \mathcal{T} \wedge C' \trianglelefteq_{\mathcal{T}} C$
2. $\Phi = C' \leq same\_as(D_1, D_2)$ is interesting with respect to $< \mathcal{T}, \mathcal{M} >$.

This rule is a consequent specialization rule. It tries to find out relations between related objects. If two axioms have the same antecedent, it is possible that given an individual s holding both axioms, related individuals that make s satisfy the axioms are related among them.[12].

same-as axiom rule $(\longrightarrow_{[\leq]})$ $< \mathcal{T}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} > \longrightarrow < \mathcal{T} \cup \{\Phi\}, \mathcal{CN}, \mathcal{RN}, \mathcal{FC} >$

1. $C' \leq_{\alpha,\sigma} D \in \mathcal{T}$ and C' is hierarchically related to C in $\mathcal{T}$.
2. $\Phi = C \leq same\_as(C, D)$ is interesting with respect to $< \mathcal{T}, \mathcal{M} >$.

This is also a consequent specialization rule.

specialize antecedent $(\longrightarrow_{\downarrow \leq \cdot})$ This rule is equivalent to the *generalize antecedent*; but instead of generalizing concept names, it specializes them.

specialize antecedent concept names $(\longrightarrow_{\downarrow^{A}_{B} \leq \cdot})$ This rule is equivalent to the *generalize antecedent concept names*; but instead of generalizing concept names, it specializes them.

Example Let's take $\Phi =$ *eat-v1* $\sqcap \exists$ *patient animal-n1* $\leq_{1,0}$ $\exists$ *agent animal-n1*. As all eaten animals are eaten by carnivores, this rule will specialize the *animal-n1* concept name to *carnivore-n1*; adding axiom *eat-v1* $\sqcap \exists$ *patient animal-n1* $\leq_{1,0}$ $\exists$ *agent carnivore-n1*, and adding *carnivore-n1* to $\mathcal{CN}$.

[12] In fact, this rule is not a consequent specialization rule since D is not necessarily a specialization of D_1 or D_2. Its qualification as a specialization rule is because the conceptual idea behind the rule is that of a specialization one. The reason because it is not formally a specialization rule comes from the poor expressivity of the loop construct, that does not allow to qualify individuals taking part in the cycle. This also applies to the *same-as axiom rule*.

specialize axiom $(\longrightarrow_{\downarrow\leq\downarrow})$ $<\mathcal{T},\mathcal{CN},\mathcal{RN},\mathcal{FC}>\longrightarrow<\mathcal{T}\cup\{\Phi_2\},\mathcal{CN},\mathcal{RN},\mathcal{FC}>$

1. $\Phi_1 = C' \leq D \in \mathcal{T}$ and C' is hierarchically related to C in $\mathcal{T}$ or $\Phi_1 = C \leq D' \in \mathcal{T}$ and D' is hierarchically related to D in $\mathcal{T}$.
2. $C = append(C_1, A_C \sqcap G \sqcap C_2)$
3. $D = append(D_1, A_D \sqcap G \sqcap D_2)$
4. $C'' = append(C_1, B_C \sqcap G \sqcap C_2)$, where $B_C \in generalizations(A_C)$
5. $D'' = append(D_1, B_D \sqcap G \sqcap D_2)$, where $B_D \in generalizations(A_D)$
6. $\Phi_2 = C'' \leq_{\mathcal{M}} D''$ is interesting for $<\mathcal{T},\mathcal{M}>$

This is a specialization rule. Sometimes, specializing only the antecedent or the consequent in a given axiom is not enough to detect an interesting specialization of it. This rule tries to fix this problem by specializing both at the same time.

specialize consequent $(\longrightarrow_{\cdot\leq\downarrow})$ This rule is equivalent to the *generalize consequent*; but instead of generalizing concept names, it specializes them.

specialize consequent concept names $(\longrightarrow_{\cdot\leq\downarrow^A_B})$ This rule is equivalent to the *generalize consequent concept names*; but instead of generalizing concept names, it specializes them.

In order to try to minimize the search performed by the induction rules application; rule applications are performed in a given order that depends on the rule being applied:

1. *cut consequent* rule is applied if possible.
2. *forbid same-as long path* rule is applied if possible.
3. Applications involving generalization rules are applied.
4. The rest of applications are performed.

The order in which induction rule applications with the same priority are applied is established by a set of heuristics that combine several factors such as: minimizing the complexity of the axiom(s) that take part in the application; maximizing the information provided by the axiom(s) taking part in the application; and maximizing the amount of information that *possibly* could be provided by the axioms resulting from the rule application.

6 Experiments

A set of experiments have been done for several domains: the already mentioned semantic parsing [16], family relationships [15], and king-rook-king illegal chess positions [15].

The experiments have shown promising results, and helped to design and tune the set of induction rules. In the semantic parsing experiment the goal was to acquire restrictions between word senses (such as: if an animal is eaten, the agent should be a carnivore). The training set was constituted by 300 sentences with over 15 different patterns and 60 different senses. Word senses were mapped to WordNet 1.6 senses, and the WordNet taxonomy was used as background knowledge. Axioms for all the patterns were learned.

The classical two-family benchmark used in [15] has also been tested with YAYA. The set of role names used are: *mother-of*, *father-of*, *wife-of*, *son-of*, *daughter-of*, and their inverses. Axioms defining a role from the rest of them are learned for all the role names.

Respect to KRK benchmark, a set of 300 KRK random configurations has been automatically generated. The used representation is equivalent to the one used for ILP predicate-style representation. Axioms defining 99 per cent of illegal configurations were succesfully learned (the same of axioms for illegal configurations are equivalent to what is learned by [15]).

References

1. Jordi Alvarez. A formal framework for theory learning using description logics. Technical Report LSI-00-R, Universitat Politcnica de Catalunya, 2000.
2. Martin Buchheit, Francesco M. Donini, and Andrea Schaerf. Decidable reasoning in terminological knowledge representation systems. *Journal of Artificial Intelligence Research*, 1:109–138, 1993.
3. W. Cohen, A. Borgida, and H. Hirsh. Computing least common subsumers in description logics. In *Proceedings of AAAI-92*, pages 754–760, 1992.
4. W. Cohen and H. Hirsh. The learnability of description logics with equality constraints. *Machine Learning*, 17(2–3):169–199, 1994.
5. William W. Cohen and Haym Hirsh. Learning the CLASSIC description logic: Theoretical and experimental results. In John Doyle, E. Sandewall, and P. Torasso, editors, *Proceedings of 4th International Conference on Principles of Knowledge Representation and Reasoning*, pages 121–133, Bonn, 1994. Morgan Kaufmann.
6. F.M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. Reasoning in description logics. In G. Brewka, editor, *Studies in Logic, Language and Information*, pages 193–238. CLSI Publications, 1996.
7. M. Frazier and L. Pitt. CLASSIC learning. *Machine Learning*, 25:151–193, 1996.
8. Jochen Heinsohn. Probabilistic description logics. In *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence*, Seattle, Washington, 1994.
9. Ian Horrocks and Ulrike Sattler. A description logics with transitive and inverse roles and role hierarchies. *Journal of Logic and Computation*, 9(3):385–410, 1999.
10. Manfred Jaeger. Probabilistic reasoning in terminological logics. In *Proceedings of the 4th International Conference on the Principles of Knowledge Representation and Reasoning*, 1994.
11. J. Kietz and K. Morik. A polynomial approach to the constructive induction of structural knowledge. *Machine Learning*, 14(2):193–217, 1994.
12. Daphne Koller, Alon Levy, and Avi Pfeffer. P-CLASSIC: a tractable probabilistic description logics. In *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI'97)*, pages 390–397, aug 1997.
13. G.A. Miller. Wordnet: An online lexical database. *International Journal of Lexicography*, 3(4):235–312, 1990.
14. Bernhard Nebel. *Terminological Cycles: Semantics and Computational Properties*, chapter 11, pages 331–361. Morgan Kaufman Publishers, 1991.
15. J.R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.
16. J. Zelle and R. Mooney. Learning semantic grammars with constructive inductive logic programming. In *Proceedings of 11th AAAI*, pages 817–822, 1993.