Description Logic Unplugged

Uwe Küssner
FR 6-10, Technische Universität Berlin
Franklinstr. 28/29, D-10587 Berlin,Germany
uk@cs.tu-berlin.de

1 Introduction

In [1], Bos introduces a general scheme of underspecification for a wide range of logics. In his approach he breaks down a formula into its parts and introduces variables, called *holes*, on argument positions. Each part gets a unique handle, called *label*. The possible connections of holes to labeled parts are represented with a new type of scope constraint. A *plugging* is a function that assigns a label to every hole. A plugging is consistent if it does not contradict the set of scope constraints. The semantics of an underspecified formula is defined to be the set of tuples of pluggings and denotations. The main motivation for Bos' work is to represent the scope ambiguity of natural language(NL) expressions. His example utterance "Do not sleep and pay attention, please!" has two readings differing with respect to negation scope. In one reading, the scope of "not" is "sleep", and in a second reading, "not" has scope over "sleep and pay attention". An underspecified representation is an expression denoting the set of both possibilities. This form of underspecification is also realized in the interface data structure VIT [2] of the NL processing system VERBMOBIL [7]. For the task of disambiguation, each VIT is mapped to objects of the Description Logic (DL) FLEX [5]. This mapping and the disambiguation task are described in [4]. However, FLEX does not support representation and reasoning with scope ambiguity. In this paper we fill this gap and adopt the approach of [1] to DL. We only describe the task of representing scope ambiguity here; further investigation is necessary for disambiguation. In general, it would be possible to introduce holes in the TBox as well, resulting in an underspecified terminology. But for now we restrict occurrences of holes to the ABox. In contrast to [1], there is no need for labels, because the relevant target entities already have names, namely those of objects. In the next section we present the syntax and semantics of an Unplugged Description Logic (UDL), more precisely a DL with a partly unplugged ABox. Then we present an algorithm of a constraint solver for solving the consistency problem of UDL. Because all interesting inferences can be reduced to consistency, this yields algorithms for subsumption and instance checking.

2 Syntax of UDL

We assume four disjoint alphabets of symbols: concept names (A), role names (R) and holes (H) and object names(O). Because the proposed extension is independent of of the syntax of concept descriptions, we can use the standard syntax of the language $\mathcal{ALC}$ [6] :

C, D	$\rightarrow$	$A \mid$	(atomic concept
		$\top \mid$	(top concept)
		$\bot \mid$	(bottom concept)
		$\neg C \mid$	(complement)
		$C \sqcap D \mid$	(conjunction)
		$C \sqcup D \mid$	(disjunction)
		$\forall R.C \mid$	(value restriction)
		$\exists R.C$	(existential qualification)

The following production rules define UDL-formulae.

γ	$\rightarrow$	$C \sqsubseteq D \mid$	(concept subsumption)
		$C \doteq D \mid$	(concept equivalence)
		$O : C \mid$	(object description)
		$(O, O) : R \mid$	(relation description)
		$(O, H) : R \mid$	(underspecified relation description)
		$O \preceq H$	(scope constraint)

$C \sqsubseteq D$ means that C is more specific than D; $C \doteq D$ means that C and D are equal. In the following we use $o_1, o_2, \ldots$ for object names and $r_1, r_2 \ldots$ for role names and $h, h_1, \ldots$ for holes. The expression $(o_1, o_2) : r_1$ describes that the two given objects are in the relation r_1. $(o_1, h) : r_1$ introduces a relation description, where the second object is underspecified. Instead of an object there is a hole. In case of $(o_1, h) : r_1$ the information about the unplugged object can be found in the corresponding scope constraint for the hole h . For example $o_2 \preceq h$ asserts that o_2 is in the scope of the operator r_1. This means that either o_2 is the filler, i.e. $(o_1, o_2) : r_1$, or that there is another object o_3, such that $(o_1, o_3) : r_1$ and $(o_3, o_2) : r_2$ holds. In this case o_2 is indirect in the scope of r_1. This applies recursively. In the next section we define the semantics of UDL.

3 Semantic of UDL

An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, [\![\cdot]\!]^{\mathcal{I}}, [\![\cdot]\!]^{\mathcal{I}}_{\mathcal{P}})$ consists of a domain of the interpretation $\Delta^{\mathcal{I}}$, an interpretation function $[\![\cdot]\!]^{\mathcal{I}}$ and a plugging function $[\![\cdot]\!]^{\mathcal{I}}_{\mathcal{P}}$. A plugging is a total and injective mapping from holes to objects. The interpretation function maps every atomic concept to a subset of $\Delta^{\mathcal{I}}$ and every atomic role to a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ and object names injectively into $\Delta^{\mathcal{I}}$, in accordance with the following equations :

$$\begin{array}{rcl}
[\![\top]\!]^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \\
[\![\bot]\!]^{\mathcal{I}} & = & \emptyset \\
[\![\neg C]\!]^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \setminus [\![C]\!]^{\mathcal{I}} \\
[\![C \sqcap D]\!]^{\mathcal{I}} & = & [\![C]\!]^{\mathcal{I}} \cap [\![D]\!]^{\mathcal{I}} \\
[\![C \sqcup D]\!]^{\mathcal{I}} & = & [\![C]\!]^{\mathcal{I}} \cup [\![D]\!]^{\mathcal{I}} \\
[\![\forall R.C]\!]^{\mathcal{I}} & = & \{a \in \Delta^{\mathcal{I}} \mid \forall (a,b) \in [\![R]\!]^{\mathcal{I}} : b \in [\![C]\!]^{\mathcal{I}}\} \\
[\![\exists R.C]\!]^{\mathcal{I}} & = & \{a \in \Delta^{\mathcal{I}} \mid \exists (a,b) \in [\![R]\!]^{\mathcal{I}} : b \in [\![C]\!]^{\mathcal{I}}\}
\end{array}$$

Satisfaction of formulae is defined as follows :

$$\begin{array}{lll}
\mathcal{I} \models C \sqsubseteq D & \text{iff} & [\![C]\!]^{\mathcal{I}} \subseteq [\![D]\!]^{\mathcal{I}} \\
\mathcal{I} \models C \doteq D & \text{iff} & [\![C]\!]^{\mathcal{I}} = [\![D]\!]^{\mathcal{I}} \\
\mathcal{I} \models O : C & \text{iff} & [\![O]\!]^{\mathcal{I}} \in [\![C]\!]^{\mathcal{I}} \\
\mathcal{I} \models (O_1, O_2) : R & \text{iff} & ([\![O_1]\!]^{\mathcal{I}}, [\![O_2]\!]^{\mathcal{I}}) \in [\![R]\!]^{\mathcal{I}} \\
\mathcal{I} \models (O, H) : R & \text{iff} & ([\![O]\!]^{\mathcal{I}}, [\![[\![H]\!]^{\mathcal{I}}_{\mathcal{P}}]\!]^{\mathcal{I}}) \in [\![R]\!]^{\mathcal{I}} \\
\mathcal{I} \models O \preceq H & \text{iff} & O \in range([\![]\!]^{\mathcal{I}}_{\mathcal{P}})
\end{array}$$

An interpretation $\mathcal{I}$ is a model of a formula γ, iff $\mathcal{I} \models \gamma$. Note that the constraints on a single $o \preceq h$ is rather unspecific. In order to extend the definition from a single formula to sets of formulae, some additional restrictions on pluggings are necessary. Before be can define what a possible plugging is, we need some definitions:

Definition 1 (p-objects,holes) *Let* Γ *be a set of UDL formulae. The set of objects, that is available for plugging is called* $p\text{-}objects(\Gamma)$ *:*

$$p\text{-}objects(\Gamma) = \{x \mid \exists y : x \preceq y \in \Gamma\}$$

The set of holes, that have to be plugged are :

$$holes(\Gamma) = \{y \mid \exists x : x \preceq y \in \Gamma\}$$

Definition 2 (Subordination) *Let* Γ *be a set of UDL formulae. The* Γ*-induced relation of* subordination *is the smallest relation such that :*

1. $x \prec x$ *for all* $x \in p\text{-}object(\Gamma) \cup holes(\Gamma)$
2. $x \prec y$ *if* $x \preceq y \in \Gamma$
3. $x \prec y$ *if* $(y, x) : R \in \Gamma$ *for some R,* $x \in holes(\Gamma)$ $y \in p\text{-}objects(\Gamma)$ *and it is not the case that* $y \prec x$
4. $x \prec y$ *if there is a z such that* $x \prec z$ *and* $z \prec y$

The first clause expresses reflexivity, the second is the explicit way of defining subordination, the third one defines subordination on relation descriptions with a hole as one argument. The fourth expresses transitivity. $\prec$ is reflexive, transitive and antisymmetric and therefore a partial order. Note that $\prec$ is not defined for all named objects but only for the p-objects.

Definition 3 (Proper subordination) *The* Γ*-induced subordination relation* $\prec$ *is* proper *iff for all* $x, y \in holes(\Gamma) \cup p\text{-}objects(\Gamma)$ *there is a z such that* $x \prec z$ *and* $y \prec z$.

A proper subordination forms a join semi-lattice. Now we can describe what a possible plugging is: A plugging is possible, if it does not contradict the given lattice. More formal:

Definition 4 (possible plugging) *Let* Γ *be a set of UDL formulae. Let* $\mathcal{I}$ *be an interpretation. Let*

$$\phi_{\mathcal{I}}(x) = \begin{cases} [\![x]\!]^{\mathcal{I}}_{\mathcal{P}} & if \quad x \in H \\ x & if \quad x \in O \end{cases}$$

$[\![]\!]^{\mathcal{I}}_{\mathcal{P}}$ *is a possible plugging, iff for all* $x, y \in holes(\Gamma) \cup p\text{-}objects(\Gamma)$ *there is a z such that:* $\phi_{\mathcal{I}}(x) \prec z$ *and* $\phi_{\mathcal{I}}(y) \prec z$.

Now we can extend the definition of $\models$ to sets of formulae. An interpretation $\mathcal{I}$ is a model of a set of UDL formulae Γ iff $[\![]\!]^{\mathcal{I}}_{\mathcal{P}}$ is a possible plugging and $\mathcal{I}$ is a model of every formula in Γ. Γ implies a formula γ (written $\Gamma \models \gamma$) iff every interpretation that is a model of Γ is also a model of γ.

4 Example

The german sentence "Ich würde lieber nicht kommen" has two readings, which are in English "I would prefer not to come" (would(i,prefer(not(to_come)))) and "I would not prefer to come" (would(i,not(prefer(to_come)))). Both readings can be represented as a single unplugged ABox.

$$\begin{array}{ll}
\Gamma = \{ & (o_0, h_o) : would, \\
& (o_1, h_1) : not, \\
& (o_2, h_2) : prefer, \\
& o3 : to_come, \\
& (o_2, o_4) : arg, \\
& o_4 : i, \\
& o_1 \preceq h_0, \\
& o_2 \preceq h_0, \\
& o_3 \preceq h_1, \\
& o_3 \preceq h_2\}
\end{array}$$

'would' is the role with widest scope, in both readings 'prefer' and 'not' are in the scope of 'would', hence : $o_1 \preceq h_0$ and $o_2 \preceq h_0$. 'to_come' is in the scope of 'prefer' and 'not' : $o_3 \preceq h_1, o_3 \preceq h_2$.

There are two possible pluggings :

$$\{h_o \mapsto o_1, h_1 \mapsto o_2, h_2 \mapsto o_3\}$$

$$\{h_o \mapsto o_2, h_1 \mapsto o_3, h_2 \mapsto o_1\}$$

5 Terminology

A subsumption $A \sqsubseteq C$ where the left hand side is an atomic concept is called *concept specification*. Concept specifications define necessary conditions. An equivalence $A \doteq C$ where the left hand side is atomic is called *concept definition*. In this case necessary and sufficient conditions are given. A *terminology (TBox)* consists of a set of concept specifications and concept definitions with some additional restriction: On the left hand side only atomic concepts are allowed. Every atomic concept may appear at most once as a left side. Furthermore we do not allow cycles in the TBox. A cycle-free TBox can be transformed into an "equivalent" normalized one. For details on what is meant with "equivalent" see [3] A TBox $\mathcal{T}$ is normalized iff

1. $\mathcal{T}$ contains only concept definitions $A \doteq C$ and
2. on the right hand side there are only concept names which do not appear on the left hand side of other definitions and
3. every concept description is simple (see below)

The process of normalization of a TBox consists of the following three steps:

1. Elimination of specifications
   Substitute every $A \sqsubseteq C$ with $A \doteq C \sqcap A^*$, where A^* is a new symbol.
2. Expansion
   Substitute every concept name on the right hand side which appears also on the left hand side with its right hand side.
3. Simplification
   A concept description is *simple*, if it contains only complements of the form $\neg A$, where A is atomic. The following rewrite rules transforms a concept description into an equivalent simple concept description.

$$
\begin{array}{rcl}
\neg\bot & \rightarrow & \top \\
\neg\top & \rightarrow & \bot \\
\neg\neg C & \rightarrow & C \\
\neg\forall R.C & \rightarrow & \exists R.\neg C \\
\neg\exists R.C & \rightarrow & \forall R.\neg C \\
\neg(C \sqcap D) & \rightarrow & \neg C \sqcup \neg D \\
\neg(C \sqcup D) & \rightarrow & \neg C \sqcap \neg D
\end{array}
$$

6 Inferences

There are several inference services, that can be offered from a DL-system. The most common are:

Subsumption Problem Does a concept description C subsume a concept description D?

Consistency Problem Has set of formulae a model?

Instance Problem Does a set of formulae imply o:C?

In [3] the consistency problem is identified as the basic inference problem. The subsumption problem can be reduced to the instance problem, which can be reduced to the consistency problem:

Proposition 1 (Reduction of subsumption to instance checking) *Let Γ be set of formulae, C and D are concept descriptions.*

$$\Gamma \models C \sqsubseteq D \textit{ iff } \Gamma \cup \{A \doteq C, B \doteq D, a : B\} \models a : A$$

where A,B and a are new symbols.

Proposition 2 (Reduction of instance checking to consistency) *Let Γ be a set of formulae, $a : A$ an object description, and $\bar{A}$ a new concept. Then :*

$$\Gamma \models a : A \;\; \textit{iff} \;\; \Gamma \cup \{\bar{A} \doteq \neg A, a : \bar{A}\} \textit{ has no model}$$

Hence for all interesting services it is sufficient to have a solution for the consistency problem. The next section presents such an algorithm for UDL.

7 The Consistency Problem

A constraint system consists of a nonempty finite set of constraints and an alphabet of symbols which is a superset of the objects and holes. In the following we use x, y, z for symbols. A constraint is a syntactic object of one of the forms

$$x : C, x \mapsto y, x \prec y, x \preceq y, xRy$$

Definition 5 (induced constraint system) *Let Γ be a set of normalized UDL-formulae. The Γ-induced constraint system is defined as follows:*

$$
\begin{array}{rcl}
S & = & \{x : C \mid x : A \in \Gamma, A \doteq C \in \Gamma\} \cup \\
& & \{xRy \mid (x, y) : R \in \Gamma\} \cup \\
& & \{x \preceq y, x \prec y \mid x \preceq y \in \Gamma\}
\end{array}
$$

An induced constraint system is consistent if there is an interpretation $\mathcal{I}$ and $\mathcal{I}$-assignment such that all constraints are satisfied. For a detailed description of constraint-systems for DL see [6] or [3].

Proposition 3 *Let S be a constraint system induced by Γ. Γ has no model iff S is inconsistent.*

Definition 6 (plug) *The application of a single plug leads to a modified constraint system*

$$S^{x \mapsto y} = \{aRy \mid \exists a, R : aRx \in S\} \cup \{X \mid X \in S, X \neq aRx\}$$

For consistency checking a set of propagation rules are applied to the induced system. The propagation rules are:

$S \quad \rightarrow_{\sqcap} \quad \{x : C_1, x : C_2\} \cup S$
if $x : C_1 \sqcap C_2$ is in S and
$x : C_1$ and $x : C_2$ are not both in S

$S \quad \rightarrow_{\sqcup} \quad \{x : C\} \cup S$
if $x : C_1 \sqcup C_2$ is in S and neither $x : C_1$
nor $x : C_2$ are in S and $C = C_1$ or $C = C_2$

$S \quad \rightarrow_{\mapsto} \quad \{x \prec y\} \cup S$
if $z \mapsto y$ is in S and for some x with $x \neq y$
there is $x \preceq z$ in S and $x \prec y$ is not in S or
if $z \mapsto x$ is in S and $z \prec y$ is in S and
$x \neq y$ and $x \prec y$ is not in S

$S \quad \rightarrow_{\exists} \quad \{xRy, y : C\} \cup S$
if $x : \exists R.C$ is in S and there is no z such that
xRz and $z : C$are in S

$S \quad \rightarrow_{\preceq} \quad \{x \mapsto y\} \cup S^{x \mapsto y}$
if $x \in hole(S)$(i.e. $x \preceq z$ is in S) and
$y \in p\text{-}object(S)$ and $x \mapsto z$ is not in S for
any z

$S \quad \rightarrow_{\prec} \quad \{x \prec y\} \cup S$
if $x \prec z$ and $z \prec y$ is in S and $x \prec y$ is not in S

$S \quad \rightarrow_{R} \quad \{x \prec y\} \cup S$
if yRx is in S and $y \in hole(S)$ and $x \prec y$ is
not in S

$S \quad \rightarrow_{\forall} \quad \{y : C\} \cup S$
if $x : \forall R.C$ and xRy are in S and
$y : C$ is not in S

$S \quad \rightarrow_{\perp} \quad \{\perp\}$
if $x : A$ and $x : \neg A$ are in S, or
if $x \prec y$ and $y \prec x$ are in S or
if $x \mapsto y$ and $x \prec y$ are in S

The rules $\rightarrow_{\sqcap}, \rightarrow_{\sqcup}, \rightarrow_{\exists.}, \rightarrow_{\forall.}$ are the standard ones ([3]). For the gives rule set, we assume that the Γ-induced subordination relation is proper. In this case we call the constraint system itself proper. Only under this assumption the propagation rules construct the graph of a possible plugging with the constraints of the form $x \mapsto y$. The test for properness can be done in a preprocessing step. Note that the rules $\rightarrow_{\sqcup}, \rightarrow_{\preceq}$ are nondeterministic. A constraint system is complete, if no propagation rule applies. A clash is a constraint system of the form $\{x : \perp\}$.

Proposition 4 (consistency) *A constraint system is* consistent *iff it is proper and there exists a complete system, which is no clash. Otherwise it is* inconsistent.

8 Conclusion

We have presented an underspecified description logic, which can be used to represent scope ambiguities of natural language expressions. Instead of explicitly telling the fillers, a set of scope constraints is added. The next interesting step would be the integration of a preference relation on models for disambiguation. In the NLP context usually there is a 'default' plugging, based on syntactic information. If no other rules are specified this default model would be the prefered one.

References

[1] J. Bos *Predicate Logic Unplugged* Tenth Amsterdam Colloquium, 1995

[2] M. Dorna *The ADT PACKAGE for the Vermobil Interface Term* VERBMOBIL Report 104, 1996

[3] B. Holunder *Hybrid Inferences in KL-ONE-based Knowledge Representation Systems* DFKI Report RR-90-06 1990

[4] U. Küssner *Applying DL in Automatic Dialogue Interpreting* International Workshop on Description Logics Gif sur Yvette (Paris) 1997

[5] J.J. Quantz, G. Dunker, F. Bergmann, I. Kellner, *The FLEX System*, KIT Report 124 Technische Universität Berlin, 1995

[6] M. Schmidt-Schauß, G. Smolka *Attribute Concept descriptions with complements* SEKI Report SR-88-21 Universität Kaiserslautern 1988 (and in Artificial Intelligence 48(1991) 1-26)

[7] W. Wahlster, Verbmobil : *Übersetzung von Verhandlungsdialogen* VERBMOBIL Report 1,DFKI Saabrücken, 1993