Description Logic Programs: A Practical Choice For the Modelling of Ontologies*

Pascal Hitzler[1], Rudi Studer[1,2,3], York Sure[1]

[1] AIFB, University of Karlsruhe
[2] FZI, Karlsruhe
[3] Ontoprise GmbH, Karlsruhe

Abstract

Knowledge representation using ontologies constitutes the heart of semantic technologies. Despite successful standardization efforts by the W3C, however, there are still numerous different ontology representation languages being used, and interoperability between them is in general not given. The problem is aggrevated by the fact that current standards lay foundations only and are well-known to be insufficient for the modelling of finer details. Thus, a plethora of extensions of the basic languages is being proposed, rendering the picture of ontology representation languages to be chaotic, to say the least. While semantic technologies start to become applicable and are being applied in adjacent areas of research and in research projects with industrial participation, and can soon be expected to become an integral part of industrial applications, the practitioner is faced with the difficult task of choosing his basic ontology representation paradigm. We will argue that the OWL subset known as Description Logic Programs constitutes a very reasonable choice.

1 Semantic Technologies are everywhere

Accelerated by the vision of the semantic web, semantic technologies have recently made significant advances towards applications. The underlying methods and paradigms are already being transferred to adjacent areas of research in artificial intelligence, knowledge management, and elsewhere. Textbooks explaining the foundations, e.g. [12], have appeared. Large national and international projects on the topic are under way, like the EU funded SEKT[1] project, the KnowledgeWeb[2] and REWERSE[3] Networks of Excellence, or the SmartWeb[4] project financed by the German Federal Ministry of Education and Research (BMBF), to mention only a few.

*The authors acknowledge support by the European Union under the SEKT project and the KnowledgeWeb Network of Excellence, and by the German Federal Ministry for Education and Research (BMBF) under the SmartWeb project.

[1] http://www.sekt-project.com
[2] http://knowledgeweb.semanticweb.org
[3] http://rewerse.net
[4] http://www.smartweb-project.org

Dagstuhl Seminar Proceedings 05371
Principles and Practices of Semantic Web Reasoning
http://drops.dagstuhl.de/opus/volltexte/2006/478

Partners from the industry play an important role in the just mentioned projects and networks, as they help focussing the research on practical needs and therefore accelerate the transfer of basic research to real applications. Besides the involvement of major companies of wide international impact, small and medium-sized spin-off companies like Ontoprise and others are successfully leading semantic technologies to real applications.

As an example, we mention the HALO project[5] by Vulcan Inc.[6] whose ultimate goal is the creation of a "digital Aristotle", an expert tutor in a wide variety of subjects. In a preliminary six-month phase the state-of-the-art in question-answering, with an emphasis on deep reasoning, was assessed. The effort was structured around the challenge of responding to variants of AP Chemistry questions that focused on a portion of the "Advanced Placement test: Chemistry", used in the US as a qualification test before entering unversity as a student. The system developed by Ontoprise, OntoNova, answers questions from this AP test. OntoNova justifies its answers in detail-giving, natural-language explanations. The results of the intial assessment showed that semantic technologies are very well suited for such complex modelling and reasoning tasks, indeed system performance was in general much better than that of real students taking the exam.

2 The central role of ontologies

Semantic technologies rest on ontologies as the central paradigm for modelling knowledge. They act as reference points for the meaning of data, and thus enable the sharing of conceptualizations. Logical aspects of ontologies furthermore allow for complex reasoning tasks over the shared knowledge, and for a formally — i.e. *semantically* — sound treatment of data.

The use of ontologies for the representation of meaning indeed is one of the distinguishing aspects of semantic technologies. The development of suitable ontology representation languages consequently is and has been one of the central research tasks for establishing and driving the field. A number of different paradigms have been proposed and are still being developed and applied. Some basic languages, like OWL [10, 2], have been established as standards, but there is general agreement that further extensions and refinements will be needed, taking some of the basic intuitions of competing paradigms into account. In consequence, the quest for suitable ontology representation standards is still open and being pursued with frenzy.

3 The Babel of ontology representation languages

Apparently, some of the reasons why there are competing ontology representation languages lie in the differing requirements imposed by the respective application scenarios to which semantic technologies are being applied. Simpler languages are easier and more efficient to deal with but lack the complex mod-

[5] http://www.projecthalo.com
[6] http://www.vulcan.com

elling facilities of the richer paradigms. Some scenarios may require particular modelling facilities or even tools which are only available for certain paradigms.

Of the major ontology representation languages, we mention a few. The Resource Description Framework RDF [11] — and its Vocabulary Description Language RDF Schema (RDFS) — is a W3C standard for the basic modelling of resources. While it is often being used directly as an ontology representation language, it lacks some of the basic logical features and semantic expressivity needed for taking real advantage of the added-value which semantic technologies offer.

The Web Ontology Language OWL [10, 2], in contrast, is a very expressive ontology representation language which can be described as a fragment of first-order predicate logic. Despite its being a W3C standard, it comes in three different versions, namely as OWL Lite, OWL DL and OWL Full, where the first two are decidable but not fully RDFS-compatible, while the latter is undecidable and does not harmonize well with the first two. Furthermore, there currently do not exist any implemented reasoners which support all of OWL DL or OWL Full, and even for the implemented fragments reasoning tasks scale badly due to the underlying high complexities.

Another major paradigm used for ontology modelling is F-Logic [8, 1]. While OWL is based on the Description Logic paradigm [3], F-Logic is strongly related to logic and object-oriented programming, and allows for the modelling of knowledge by implication rules, a feature missing in OWL. F-Logic, for example, underlies the OntoBroker system which was used in the abovementioned HALO project.

4 DLP in a nutshell — and why it is good for you

For applying semantic technologies, the practicioner is faced with the question, which of the above — or other — ontology modelling paradigms should be adopted. For performance reasons, simpler paradigms are often preferable, but equally important is interoperability between different systems and reusability in the future, i.e. compatibility with to-be established standards.

We argue that Description Logic Programs (DLP)[7] as described in [4, 13] provide a basic ontology modelling paradigm which meets most of the requirements above while being a flexible choice for future developments. In a nutshell, DLP is a fragment of OWL corresponding to Horn clauses[8], i.e. to the logic programming fragment of OWL. As such, it is a proper fragment of both OWL and F-Logic. While it is intuitively clear what DLP is, formal characterizations are not entirely straightforward, mainly due to the question what "intersection" between two languages is supposed to mean exactly, e.g. whether it should be understood in a semantic or a syntactic way — we will discuss this in more detail in Section 5.

But in order to facilitate the usage of OWL DLP, we can describe an easy to use sublanguage of DLP by listing all the constructors which can be used

[7] See also http://logic.aifb.uni-karlsruhe.de.

[8] A Horn clause is a formula in first order predicate logic which is in conjunctive normal (i.e. clausal) form and contains at most one positive (non-negated) literal. See e.g. [9].

freely in an OWL ontology without running the risk of leaving DLP — provided the usual OWL DL constraints are also being adhered to. This does not mean that other constructors are entirely forbidden in OWL DLP. It is just that their usage underlies further constraints which we will discuss in Section 5.

Allowed OWL constructors: *Class, Thing, subClassOf, Property, subPropertyOf, domain, range, Individual, equivalentClass, equivalentProperty, sameAs, differentFrom, AllDifferent, ObjectProperty, DatatypeProperty, inverseOf, TransitiveProperty, SymmetricProperty, FunctionalProperty, InverseFunctionalProperty, intersectionOf.*

In order to use DLP, there is no need for special software as standard OWL editors can be used to create valid DLP ontologies. As for expressivity, it was shown in [13] that existing available ontologies often use very few constructs outside the DLP language fragment. It thus supports most of the requirements currently made in practice. The use of DLP in major projects such as SEKT, SmartWeb, WSMO[9] or SweetRules[10] further justifies our argument. As for performance, DLP enjoys polynomial data complexity and exptime combined complexity, which renders it to be far better than the more expressive languages we mentioned.

DLP also provides a flexible choice for the future. As it is a common fragment of major paradigms, it is compatible in principle with whatever paradigm will turn out to be more popular. Extensions made for either more general language can be adopted for the fragment in a straightforward manner. Modelling and reasoning tools available for OWL or F-Logic can naturally deal with DLP, and interoperability is guaranteed to the largest extent possible.

5 What DLP is

DLP is the Horn fragment of OWL DL. This apparently neat statement, examined closely, turns out to be of limited use in practice. This is caused by the fact that "*Horn fragment*" refers to a *syntactic* fragment of first-order predicate logic (FOL), while OWL DL is commonly perceived as a *semantic* fragment of FOL. Consequently, the initial statement of this section reads that *DLP is the syntactic Horn fragment (in the sense of FOL syntax) of something (namely OWL DL) which is not in FOL syntax, but can semantically be mapped to a syntactic fragement of FOL.* Obviously, this needs to be explained, which is what we will do in the following. Along the way, we will also come up with clear explanations what DLP is. Let us start with two perspectives on the DLP definition problem, one semantic, the other syntactic. If the reader is not interested in these rather messy details, then he may safely skip to Section 5.1.

The semantic perspective says that *an OWL-DL statement is in DLP if and only if it can be written — semantically equivalently — as a set of Horn clauses in FOL.* Sadly enough, this perspective is not readily algorithmized as it amounts to checking whether for a given FOL formula there exists a set of

[9] http://www.wsmo.org
[10] http://sweetrules.projects.semwebcentral.org

Horn formulae with exactly the same models. In this generality, the checking task is actually undecidable!

In order to exemplify the difficulties, consider the following example — we choose description logic syntax for this and all subsequent examples, as it is the most easily human-readable syntax for written documents.

Example 1 *The (admittedly strange) OWL-DL statement*

$$\mathit{Father} \sqsubseteq \mathit{Human} \sqcup \mathit{Human}$$

translates to the FOL formula

$$(\forall x)(\mathit{Father}(x) \rightarrow \mathit{Human}(x) \vee \mathit{Human}(x)),$$

which is not Horn. However, it is obvious that the following statement is semantically equivalent, and is indeed a Horn statement:

$$(\forall x)(\mathit{Father}(x) \rightarrow \mathit{Human}(x)).$$

The final translation, however, was not syntactic, but semantic.

A more sophisticated (and more generic) example is the following. The statement

$$\forall.R.\langle X\rangle \sqsubseteq D \sqcup E,$$

for R a role, D, E concept identifiers, and $\langle X\rangle$ some composite concept, translates naively to the FOL formula

$$(\forall x)((\forall y)(R(x,y) \wedge \langle X\rangle(y)) \rightarrow (D(x) \vee E(x))),$$

which is in general not syntactically transformable to a Horn clause. However, if $(\forall y)(R(x,y) \wedge \langle X\rangle(y))$ is unsatisfiable, then the formula is a tautology and can thus be expressed by means of any Horn tautology. Checking satisfiability of $(\forall y)(R(x,y) \wedge \langle X\rangle(y))$ for arbitrarily sophisticated $\langle X\rangle$, however, amounts to reasoning over OWL-DL, which is too complex a task to be included in a language definition!

A more practical approach along the same intuition is to state that *an OWL DL statement is in DLP if and only if some given transformation algorithm can rewrite it as a semantically equivalent Horn clause in FOL.* Indeed, it is easy to come up with reasonable transformation algorithms, and corresponding implementations have this way shown that most of the currently available ontologies are in DLP (see [13]). While this approach is not entirely satisfactory from a principled-based perspective, we believe that it is a practical one, as the finer details will rarely matter in practice, and reference to a concrete (implementation of a) transformation algorithm will suffice to clarify which "dialect" of DLP one refers to.

The syntactic perspective is more helpful than the semantic perspective for the ontology engineer who wants to create DLP ontologies from scratch, as a constant checking for Hornness is unpractical.[11] The syntactic approach

[11] It may actually become practical if ontology editors be equipped with real-time Horn checkers, but such a thing does currently not exist.

thus strives for a concrete syntax definition which tells the ontology engineer how to manually create DLP ontologies. The discussion provided in Example 1 reveals the difficulties inherent in this approach: Any concretely given syntax will disallow constructs which the ontology engineer may be inclined to use, but which are semantically redundant and should thus not inflict on the DLPness of a statement.

We therefore believe that a bottom-up syntactic definition of DLP will not be entirely satisfactory, and defining DLP in relation to a concrete transformation algorithm is the more flexible approach. Nevertheless, the provision of a syntax (or different versions of increasing complexity) which guarantees that the written statements are DLP statements will be of much practical help. A first step towards this was provided by the list of *allowed OWL constructors* in Section 4. A more sophisticated syntax definition will be provided below.

5.1 Two working definitions

Before we move on to defining what we mean by *DLP*, we first need to clarify four additional issues, concerning concrete domains, equality for instances, number restrictions, and integrity constraints

Concrete domains. Adhering to the naive intuition as DLP being a Horn fragment of OWL, we disallow concrete domains. We actually understand that concret domains are of utmost importance in practice, but we nevertheless think that they should not be part of the pure core language. It appears, however, that the addition of concrete domains to DLP, for example in the form of built-ins in a logic programming framework, will not pose any particular difficulties if practical issues require it.

Equality for instances can be expressed if a logic with equality theory is used. We perceive this as an optional feature of DLP.

Number restrictions will in general be disallowed for the same reason as concrete domains. However, some number restrictions can be rewritten using equality for instances or the existential quantifier, and may therefore be included.

Integrity constraints are not Horn clauses, but are commonly used in logic programming besides Horn clauses.[12] As for equality, we perceive this as an optional feature of DLP.

The semantic approach

Adhering to the semantic perspective discussed earlier, we tie our definition of DLP to a concrete implementation of a transformation algorithm. The reference implementation is KAON2[13], respectively the KAON2-based conversion tool dlpconvert[14], which accepts OWL DL ontologies as input, checks them for

[12] Integrity constraints are formulae in conjunctive normal form which contain negated literals only.

[13] http://kaon2.semanticweb.org

[14] http://logic.aifb.uni-karlsruhe.de/dlpconvert

Hornness, and returns the corresponding DLP ontology in Prolog syntax. The exact Horn checking algorithm is detailed in [7], which we do not repeat here. It suffices to say that it boils down to a standard syntactic translation into conjunctive normal form, with some straightforward improvements.

The syntactic approach

We now proceed with defining a concrete syntactic fragment of DLP, which is expressive enough to be used by the ontology engineer in practice. Allowed are the following, where a, b, a_i stand for individuals, C stands for a concept name and $R, Q, R_i, Q_{i,j}$ stand for role names.

- ABox:

$C(a)$	(indiv. assertion)
$R(a, b)$	(property assertion)
$a = b$	(indiv. equivalence)

- Property Characteristics:

$R \equiv Q$		(equivalence)
$R \sqsubseteq Q$		(subproperty)
$\top \sqsubseteq \forall R.C$	$(C \neq \bot)$	(domain)
$\top \sqsubseteq \forall R^-.C$	$(C \neq \bot)$	(range)
$R \equiv Q^-$		(inverse)
$R \equiv R^-$		(symmetry)
$\top \sqsubseteq \leq 1 R$		(functionality)
$\top \sqsubseteq \leq 1 R^-$		(inverseFunctionality)

- TBox: We allow all (and only) expressions of the form

$$\exists Q_{1,1}^{(-)} \ldots \exists Q_{1,m_1}^{(-)}.\texttt{Left}_1 \sqcap \cdots \sqcap \exists Q_{k,1}^{(-)} \ldots \exists Q_{k,m_k}^{(-)}.\texttt{Left}_k \sqsubseteq \forall R_1^{(-)} \ldots \forall R_n^{(-)}.\texttt{Right}$$

  where we allow $\texttt{Left}_j$ to be of the forms C, $\{o_1, \ldots, o_n\}$, $\bot$ or $\top$, and $\texttt{Right}$ to be of the forms C or $\top$. The superscript $^{(-)}$ shall indicate, that an inverse symbol may occur in these places. Note that (by a common abuse of notation) we allow any of k, m_i, n to be zero. For $k = 0$ the left hand side becomes $\top$.

  If integrity constraints are allowed, we furthermore allow $\texttt{Right}$ to be of the form $\bot$. If equality of individuals is allowed, we furthermore allow $\texttt{Right}$ to be of the form $\{o\}$. We finally remark that $\texttt{Right}$ may also be of the form $\exists R^{(-)}.\{a\}$; in general, however, this causes the DLP ontology to lie outside of the OWL Lite fragment.

We will proceed with examples in the next section, in order to explain the definition just given.

It can be shown formally, that the syntactic definition above yields DLP ontologies, and in fact includes all expressive features which were also allowed in the original publications on DLP, i.e. in [4, 13]. Formal proof of this can be found in [6].

6 Examples

A rule of thumb for the creation of DLP ontologies is: *Avoid concrete domains and number restrictions, and be careful with quantifiers, disjunction, and nominals.* We give a small example ontology which includes the safe usage of the latter constructs. It shall display the modelling expressivity of DLP.

Example 2 *For the TBox, we model the following sentences.*

(1) Every man or woman is an adult.
(2) An adult is a human who is not a child.
(3) A woman who has somebody as a child, is a mother.
(4) A man who is a parent of somebody, is a father.
(5) An orphan is the child of dead humans.
(6) A lonely child has no siblings.

They can be written in DLP as follows.

$$Man \sqcup Woman \sqsubseteq Adult \tag{1}$$
$$Adult \sqsubseteq Human \sqcap \neg Child \tag{2}$$
$$Woman \sqcap \exists childOf^{-}.\top \sqsubseteq Mother \tag{3}$$
$$Man \sqcap \exists parentOf.\top \sqsubseteq Father \tag{4}$$
$$Orphan \sqsubseteq \forall childOf.(Dead \sqcap Human) \tag{5}$$
$$LonelyChild \sqsubseteq \leq 0\, childOf.parentOf.\top \tag{6}$$

For the RBox, we use the following.

$parentOf \equiv childOf^{-}$	*parentOf and childOf are inverse roles.*
$parentOf \sqsubseteq ancestorOf$	*parentOf is a subrole of ancestorOf.*
$fatherOf \sqsubseteq parentOf$	*fatherOf is a subrole of parentOf.*
$\top \sqsubseteq \forall ancestorOf.Human$	*Human is the domain of ancestorOf.*
$\top \sqsubseteq \leq 1 fatherOf^{-}$	*fatherOf is inverse functional.*

We can populate the classes and roles by means of an ABox in the following way.

$$\{Ian, Frank, Rudi, Horrocks\} \sqsubseteq Man$$
$$\{Carole, Asun, Roberta\} \sqsubseteq Woman$$
$$fatherOf(Rudi, \dots)$$
$$Ian = Horrocks$$
$$\dots$$

Note that an ABox statement such as

$$\{\text{Carole}, \text{Asun}, \text{Roberta}\} \sqsubseteq \text{Woman}$$

is simply syntactic sugar for the three statements

$$\text{Woman(Carole)} \qquad \text{Woman(Asun)} \qquad \text{Woman(Roberta)}.$$

We therefore consider it to be part of the ABox. To be precise, the original statement is (syntactically) not in OWL Lite, but the equivalent set of three ABox statements is.

Note also that class inclusions cannot in general be replaced by equivalences. For example, the statement

$$\text{Adult} \sqsubseteq \text{Man} \sqcup \text{Woman}$$

is not in DLP. The inverse inclusion

$$\text{Man} \sqcup \text{Woman} \sqsubseteq \text{Adult},$$

however, is in DLP, as it can be written semantically equivalently as the two statements

$$\begin{aligned} \text{Man} &\sqsubseteq \text{Adult} \\ \text{Woman} &\sqsubseteq \text{Adult}. \end{aligned}$$

7 Pointers to DLP

The initial publications introducing DLP language are [4, 13]. A website with pointers to further resources is being maintained at http://logic.aifb.uni-karlsruhe.de. Special efforts will be made to make the forthcoming KAON2 OWL DL reasoner[15] fully compatible with DLP requirements.

We also provide the KAON2-based tool dlpconvert that converts DLP ontologies written in OWL/XML or RDF/XML syntax to logic programming syntax, retaining the semantics[16], thus allowing the use of standard out-of-the-box logic programming systems for DLP reasoning. We further plan to integrate DLP into our OWL evolution framework evOWLution [5] in order to provide the means to develop OWL ontologies with the guarantee to remain within DLP. It is also planned to adapt existing ontology editors in order to help the user to account for DLP automatically, thus enabling the standard user to fully leverage the power of DLP. Since DLP can be understood as a subset of OWL DL, there is no need for recreating the whole range of ontology management tools. Small adjustments — if any — of existing software will suffice for achieveing full benefit of the existing systems and methodologies.

References

[1] Jürgen Angele and Georg Lausen. Ontologies in F-logic. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, pages 29–50. Springer, 2004.

[2] Grigoris Antoniou and Frank van Harmelen. Web Ontology Language: OWL. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, pages 67–92. Springer, 2004.

[15] http://kaon2.semanticweb.org
[16] http://logic.aifb.uni-karlsruhe.de/dlpconvert

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

[4] Benjamin Grosof, Ian Horrocks, Raphael Volz, and Stefan Decker. Description logic programs: Combining logic programs with description logics. In *Proc. of WWW 2003, Budapest, Hungary, May 2003*, pages 48–57. ACM, 2003.

[5] Peter Haase, York Sure, and Denny Vrandecic. Ontology management and evolution – survey, methods and prototype. SEKT formal deliverable D3.1.1, Institute AIFB, University of Karlsruhe, December 2004.

[6] Pascal Hitzler and Andreas Eberhart. Description logic programs: Normal forms. Technical report, AIFB, University of Karlsruhe, September 2004. Available from the first author's homepage.

[7] Ulrich Hustadt, Boris Motik, and Ulrike Sattler. Reasoning for description logics around SHIQ in a resolution framework. Technical report, FZI Karlsruhe, 2004. Available from the second author's homepage.

[8] Michael Kifer, Georg Lausen, and James Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM*, 42:741–843, 1995.

[9] John W. Lloyd. *Foundations of Logic Programming*. Springer, Berlin, 1988.

[10] Web ontology language (OWL). http://www.w3.org/2004/OWL/, 2004.

[11] Resource description framework (RDF). http://www.w3.org/2004/RDF/, 2004.

[12] Steffen Staab and Rudi Studer, editors. *Handbook on Ontologies*. International Handbooks on Information Systems. Springer, 2004.

[13] Raphael Volz. *Web Ontology Reasoning with Logic Databases*. PhD thesis, AIFB, University of Karlsruhe, 2004.