Advanced Languages of Terms for Ontologies

Philippe Balbiani[1], Martin Diéguez[2] and Çiğdem Gencer[1,3]

[1]Institut de recherche en informatique de Toulouse. CNRS-INPT-UT3, Université de Toulouse, Toulouse, France
[2]Laboratoire d'étude et de recherche en informatique d'Angers. University of Angers, Angers, France
[3]Faculty of Arts and Sciences. Istanbul Aydın University, Istanbul, Turkey

Abstract

This paper is about the integration in a unique formalism of knowledge representation languages such as those provided by description logic languages and rule-based reasoning paradigms such as those provided by logic programming languages. We aim at creating an hybrid formalism where description logics constructs are used for defining concepts that are given as arguments to the predicates of the logic programs.

1. A Short Introduction

A crucial issue in the development of the semantic web is the possibility to combine rule-based systems and ontologies. There exists already several types of such combination [23, 24, 30, 32, 35]. These approaches either build rules on top of ontologies allowing rule-based systems to use the vocabulary specified in ontologies, or build ontologies on top of rules supplementing ontological definitions by rules. None of them completely answer to the question of the combination of logic programming with description logics that we are seeking for: an hybrid formalism where description logics constructs are used for defining concepts that are given as arguments to the predicates of the logic programs. In this paper, we develop such an hybrid formalism.

The section-by-section breakdown of this paper is as follows. A case study motivating the combination of logic programming with description logics that we are seeking for is presented in Section 2. In Sections 3 and 4, we introduce the syntax and the semantics of our hybrid formalism. Decision problems are presented in Sections 5, 6 and 7. A research program is presented in Section 8.

2. A Case Study

Examining role-based access control and organization-based access control, we present a case study motivating the combination of logic programming with description logics that we are seeking for.

DL 2022: 35th International Workshop on Description Logics, August 7–10, 2022, Haifa, Israel
Philippe.Balbiani@irit.fr (P. Balbiani); martin.dieguezlodeiro@univ-angers.fr (M. Diéguez); Cigdem.Gencer@irit.fr (Ç. Gencer)

Access of subjects to objects in a computer system are permitted in accordance with a security policy embodied in an access control database. Many computer systems use the access control matrix model to represent security policies [31]. Formally, an access control matrix is a structure consisting of a set of subjects (users, processes, etc), a set of objects (files, tables, etc) and binary relations $(p_i)_{i \in I}$ between objects and subjects giving to subjects permissions to access objects. In this setting, asserting that subject a possesses permission p_i on object b comes down to asserting that p_i holds for b and a.

Access control with a lot of subjects is space-consuming. To reduce the cost of security, within the context of role-based access control (RBAC), it has been proposed that access control administrators treat sets of subjects as instances of a concept called role[1] [38]. Formally, an RBAC-structure consists of a set of subjects, a set of objects, a set of roles, a binary relation r between subjects and roles defining the roles of subjects and binary relations $(p_i)_{i \in I}$ between objects and roles giving to roles permissions to access objects. In this setting, asserting that subject a has role A comes down to asserting that r holds for a and A, whereas asserting that role A possesses permission p_i on object b comes down to asserting that p_i holds for b and A. It is possible to refine the RBAC model by including the concept of role hierarchy which allows permissions to be inherited through it. This hierarchy is specified by means of assertions of the form $A' \sqsubseteq A''$ where A' and A'' are roles. To put it simply, the idea behind RBAC is the following: in a computer system, subject a possesses a permission p on object b if and only if there are roles $A_0, \ldots, A_m$ such that r holds for a and A_0, for all positive integers $i \leq m$, $A_{i-1} \sqsubseteq A_i$ has been asserted and p holds for b and A_m.

RBAC with a lot of objects is space-consuming. To reduce the cost of security, within the context of organization-based access control (OrBAC), it has been proposed that RBAC administrators treat sets of objects as instances of a concept called view [1]. Formally, an OrBAC-structure consists of a set of subjects, a set of objects, a set of roles, a set of views, a binary relation r between subjects and roles defining the roles of subjects, a binary relation v between objects and views defining the views of objects and binary relations $(p_i)_{i \in I}$ between views and roles giving to roles permissions to access views. In this setting, asserting that object b has view B comes down to asserting that v holds for b and B, whereas asserting that role A possesses permission p_i on view B comes down to asserting that p_i holds for B and A. It is possible to refine the OrBAC model by including the concept of view hierarchy which allows permissions to be inherited through it. This hierarchy is specified by means of assertions of the form $B' \sqsubseteq B''$ where B' and B'' are views. To put it simply, the idea behind OrBAC is the following: in a computer system, subject a possesses a permission p on object b if and only if there are roles $A_0, \ldots, A_m$ and there are views $B_0, \ldots, B_n$ such that r holds for a and A_0 and v holds for b and B_0, for all positive integers $i \leq m$, $A_{i-1} \sqsubseteq A_i$ has been asserted and for all positive integers $j \leq n$, $B_{j-1} \sqsubseteq B_j$ has been asserted and p holds for B_n and A_m.

[1] The roles in RBAC should not be mistaken for the roles in description logics. In RBAC security policies, roles correspond to sets of subjects, whereas in description logic frames, roles correspond to binary relations.

It is a great pity that neither RBAC, nor OrBAC allow atomic assertions of the form $p_i(D, C)$ where C and D are, respectively, Boolean combinations of roles and Boolean combinations of views. By using assertions of that form, one may more succinctly define more precise access control policies. For instance, to say that subjects having the role A but not having the role A' possess a permission p_i on objects having the view B but not having the view B', one can simply assert that p_i holds for $B \wedge \neg B'$ and $A \wedge \neg A'$ instead of asserting that p_i holds for B'' and A'' where A'' is a new role such that for all subjects a, $r(a, A'')$ if and only if $r(a, A)$ and not $r(a, A')$ and B'' is a new view such that for all objects b, $v(b, B'')$ if and only if $v(b, B)$ and not $v(b, B')$.

Finally, it is also a great pity that neither RBAC, nor OrBAC allow conditional assertions of the form $p_i(D, C) \leftarrow p_j(D', C')$. By using conditional assertions of that form, one may more succinctly define more precise access control policies. For instance, to say that subjects having the role C possess a permission p_i on objects having the view D if subjects having the role C' possess a permission p_j on objects having the view D', one can simply say that $p_i(D, C) \leftarrow p_j(D', C')$. This is particularly interesting when p_j does not denote a permission, but an obligation corresponding to the permission denoted by p_i[2]. In that case, a conditional assertion like $p_i(D, C) \leftarrow p_j(D, C)$ expresses the deontic rule saying that subjects having the role C possess the permission p_i on objects having the view D if subjects having the role C possess the corresponding obligation p_j on objects having the view D.

Knowledge representation languages such as those provided by description logic languages [5] (allowing expressions of the form $C \sqsubseteq D$ where C and D are complex concepts) and rule-based reasoning paradigms such as those provided by logic programming languages [26, 34] (allowing expressions of the form $\alpha \leftarrow \beta_1, \ldots, \beta_n$ where α, β_1, . . ., β_n are atoms) are well-known and widely used in Computer Science and Artificial Intelligence. Their integration in a unique formalism would be a natural solution for many application problems requiring the following features: allowing rule-based systems to use the vocabulary specified in ontologies and supplementing ontological definitions by rules. Hybrid knowledge bases are the main approaches proposed so far. They integrate some aspects of description logic and some aspects of logic programming [23, 24, 32, 35]. Nevertheless, they hardly address all aspects of our aim: the development of an hybrid formalism where description logics constructs are used for defining concepts that are given as arguments to the predicates of the logic programs.

3. Syntax

We introduce the syntax of our hybrid formalism.

3.1. Complex Concepts

Let $\mathbf{VAR}$ be a countable set of *variable concepts* (with typical members denoted X, Y, etc). Let $\mathbf{CON}$ be a countable set of *constant concepts* (with typical members denoted A, B, etc) and

[2] We are assuming the deontic principle saying that permissions are implied by their corresponding obligations [37].

$\mathbf{ROL}$ be a countable set of *constant roles* (with typical members denoted R, S, etc). The set of *complex concepts* (with typical members denoted C, D, etc) is defined by the rule[3]

- $C ::= X \mid A \mid \top \mid (C \sqcap D) \mid \exists R.C$,

where X ranges over $\mathbf{VAR}$, A ranges over $\mathbf{CON}$ and R ranges over $\mathbf{ROL}$. We adopt standard rules for omission of the parentheses. A complex concept C is $\mathbf{VAR}$-*free* if C contains no occurrence of a variable concept. A complex concept C is $\mathbf{ROL}$-*free* if C contains no occurrence of a constant role. For all $k \in \mathbb{N}$, the concept construct $(\exists R.)^k$ is inductively defined as follows for each $R \in \mathbf{ROL}$:

- if $k=0$ then $(\exists R.)^k C ::= C$,
- otherwise, $(\exists R.)^k C ::= \exists R.(\exists R.)^{k-1} C$.

3.2. Substitutions

A *substitution* is a function from $\mathbf{VAR}$ to the set of all complex concepts equal to the identity function on a cofinite subset of $\mathbf{VAR}$ [9]. To apply a substitution σ to a complex concept C amounts to replace each occurrence in C of a variable concept $X \in \mathbf{VAR}$ by the corresponding complex concept $\sigma(X)$.

3.3. Inclusions and Equations

Concept inclusions are expressions of the form $C \sqsubseteq D$ (read "C is contained in D") for all complex concepts C, D. *Concept equations* are expressions of the form $C = D$ (read "C is equal to D") for all complex concepts C, D.

3.4. Clauses

Let $\mathbf{PRE}$ be a countable set of *predicate symbols* (with typical members denoted p, q, etc). For all $p \in \mathbf{PRE}$, let $\mathtt{ar}(p)$ be the *arity* of p. An *atom* is an expression of the form $p(C_1, \ldots, C_{\mathtt{ar}(p)})$ (read "p holds for $C_1, \ldots, C_{\mathtt{ar}(p)}$") where p is a predicate symbol and $C_1, \ldots, C_{\mathtt{ar}(p)}$ are complex concepts. *Clauses* are expressions of the form $\alpha_1, \ldots, \alpha_m \leftarrow \beta_1, \ldots, \beta_n$ (read "if $\beta_1, \ldots, \beta_n$ then either $\alpha_1, \ldots$, or α_m") where $\alpha_1, \ldots, \alpha_m, \beta_1, \ldots, \beta_n$ are atoms. *Definite clauses* are clauses of the form $\alpha \leftarrow \beta_1, \ldots, \beta_n$, *unit clauses* are clauses of the form $\alpha \leftarrow$ and *definite goals* are clauses of the form $\leftarrow \beta_1, \ldots, \beta_n$.

3.5. Assertions

Let $\mathbf{IND}$ be a countable set of *individual constants* (with typical members denoted a, b, etc). A *concept assertion* is an expression of the form $C{:}a$ (read "a belongs to C") where C is a $\mathbf{VAR}$-free complex concept and a is an individual constant. A *role assertion* is an expression of the form $R{:}(a, b)$ (read "a is R-related to b") where $R \in \mathbf{ROL}$ and a and b are individual constants.

[3]The set of complex concepts we define here is the one of description logic $\mathcal{EL}$ [8]. Most of our definitions can be easily adapted to cases where other description logics are considered instead of description logic $\mathcal{EL}$ [6, 8, 15].

3.6. Deductive Ontologies

A *T-box* is a finite set of concept inclusions and concept equations. A *program* is a finite set of clauses. An *A-box* is a finite set of concept assertions and role assertions. A *deductive ontology* is a triple $(\mathcal{T}, \Pi, \mathcal{A})$ consisting of a T-box $\mathcal{T}$, a program Π and an A-box $\mathcal{A}$.

4. Semantics

We introduce the semantics of our hybrid formalism[4].

4.1. Frames and Var-interpretations

The semantics is defined in terms of *frames,* i.e. structures (W, K, Rel) where W is a nonempty set, K: $\mathbf{CON} \longrightarrow \mathcal{P}(W)$ and Rel: $\mathbf{ROL} \longrightarrow \mathcal{P}(W \times W)$. In a frame (W, K, Rel), for all $R \in \mathbf{ROL}$,

- the R*-image* of a subset S of W is the set of all $t \in W$ such that there exists $s \in S$ such that $Rel(R)(s, t)$,
- the R*-pre-image* of a subset T of W is the set of all $s \in W$ such that there exists $t \in T$ such that $Rel(R)(s, t)$,
- the *domain* of R is the set of all $s \in W$ such that there exists $t \in W$ such that $Rel(R)(s, t)$,
- the *range* of R is the set of all $t \in W$ such that there exists $s \in W$ such that $Rel(R)(s, t)$.

Obviously, in a frame (W, K, Rel), for all $R \in \mathbf{ROL}$, the domain of R is the R-pre-image of W and the range of R is the R-image of W. A *var-interpretation* on a frame (W, K, Rel) is a function V: $\mathbf{VAR} \longrightarrow \mathcal{P}(W)$. For all frames (W, K, Rel), the value of the complex concept C with respect to a var-interpretation V on (W, K, Rel) is the subset $\|C\|_V$ of W defined by

- $\|X\|_V = V(X)$,
- $\|A\|_V = K(A)$,
- $\|\top\|_V = W$,
- $\|C \sqcap D\|_V = \|C\|_V \cap \|D\|_V$,
- $\|\exists R.C\|_V = \{s \in W :$ there exists $t \in W$ such that $Rel(R)(s, t)$ and $t \in \|C\|_V\}$.

Obviously, $\|C\|_V$ does not depend on V when C is $\mathbf{VAR}$-free. In that case, $\|C\|_V$ will be denoted $\|C\|$.

4.2. Pre-interpretations

A *pre-interpretation* on a frame (W, K, Rel) is a function I: $\mathbf{PRE} \longrightarrow \mathcal{P}(\mathcal{P}(W)^\star)$ such that for all $p \in \mathbf{PRE}$, $I(p) \subseteq \mathcal{P}(W)^{\mathtt{ar}(p)}$. For all frames (W, K, Rel) and for all pre-interpretations I on (W, K, Rel), the *value* of an atom $p(C_1, \ldots, C_{\mathtt{ar}(p)})$ with respect to a var-interpretation V on (W, K, Rel) is the element $|p(C_1, \ldots, C_{\mathtt{ar}(p)})|_V^I$ in $\{0, 1\}$ such that

- if $I(p)$ contains $(\|C_1\|_V, \ldots, \|C_{\mathtt{ar}(p)}\|_V)$ then $|p(C_1, \ldots, C_{\mathtt{ar}(p)})|_V^I = 1$,
- otherwise, $|p(C_1, \ldots, C_{\mathtt{ar}(p)})|_V^I = 0$.

[4] In this paper, for all sets E, $\mathcal{P}(E)$ denotes the set of all subsets of E, $E^\star$ denotes the set of all tuples of elements of E and for all $k \in \mathbb{N}$, E^k denotes the set of all k-tuples of elements of E.

4.3. Ind-interpretations

An *ind-interpretation* on a frame (W, K, Rel) is a function g: $\mathbf{IND} \longrightarrow W$. For all frames (W, K, Rel) and for all ind-interpretations g on (W, K, Rel), the *value* of a concept assertion $C{:}a$ is the element $|C{:}a|^g$ in $\{0, 1\}$ such that

- if $\|C\|$ contains $g(a)$ then $|C{:}a|^g{=}1$,
- otherwise, $|C{:}a|^g{=}0$,

and the *value* of a role assertion $R{:}(a, b)$ is the element $|R{:}(a, b)|^g$ in $\{0, 1\}$ such that

- if $Rel(R)$ contains $(g(a), g(b))$ then $|R{:}(a, b)|^g{=}1$,
- otherwise, $|R{:}(a, b)|^g{=}0$.

4.4. Models

For all T-boxes $\mathcal{T}$, a $\mathcal{T}$*-model* (or a *model* of $\mathcal{T}$) is a frame (W, K, Rel) such that for all var-interpretations V on (W, K, Rel),

- for all concept inclusions $C{\sqsubseteq}D$ in $\mathcal{T}$, $\|C\|_V{\subseteq}\|D\|_V$,
- for all concept equations $C{=}D$ in $\mathcal{T}$, $\|C\|_V{=}\|D\|_V$.

For all deductive ontologies $(\mathcal{T}, \Pi, \mathcal{A})$, a $(\mathcal{T}, \Pi, \mathcal{A})$*-model* (or a *model* of $(\mathcal{T}, \Pi, \mathcal{A})$) is a structure (W, K, Rel, I, g) consisting of a $\mathcal{T}$-model (W, K, Rel), a pre-interpretation I on (W, K, Rel) and an ind-interpretation g on (W, K, Rel) such that for all var-interpretations V on (W, K, Rel),

- for all clauses $\alpha_1, \ldots, \alpha_m{\leftarrow}\beta_1, \ldots, \beta_n$ in Π, if $|\beta_1|^I_V{=}1$, …, $|\beta_n|^I_V{=}1$ then either $|\alpha_1|^I_V{=}1$, …, or $|\alpha_m|^I_V{=}1$,
- for all concept assertions $C{:}a$ in $\mathcal{A}$, $|C{:}a|^g{=}1$,
- for all role assertions $R{:}(a, b)$ in $\mathcal{A}$, $|R{:}(a, b)|^g{=}1$.

Notice that in a model (W, K, Rel, I, g) of a deductive ontology $(\mathcal{T}, \Pi, \mathcal{A})$, for all var-interpretations V on (W, K, Rel),

- for all definite clauses $\alpha{\leftarrow}\beta_1, \ldots, \beta_n$ in Π, if $|\beta_1|^I_V{=}1$, …, $|\beta_n|^I_V{=}1$ then $|\alpha|^I_V{=}1$,
- for all unit clauses $\alpha{\leftarrow}$ in Π, $|\alpha|^I_V{=}1$,
- for all definite goals ${\leftarrow}\beta_1, \ldots, \beta_n$ in Π, either $|\beta_1|^I_V{=}0$, …, or $|\beta_n|^I_V{=}0$.

5. Correspondence Theory

We briefly present the correspondence theory of our hybrid formalism.

Although of limited expressive power, concept constructs can be used for characterizing classes of frames. As observed by [6, 39], description logic languages are modal languages in disguise. Therefore, the following relationships that can be easily established for all frames (W, K, Rel) will not come as a surprise:

(**1**) (W, K, Rel) is a model of $X \sqsubseteq \exists R.\top$ if and only if $Rel(R)$ is serial[5],
(**2**) (W, K, Rel) is a model of $\exists R.\top \sqsubseteq X$ if and only if $Rel(R)$ is empty,
(**3**) (W, K, Rel) is a model of $X \sqsubseteq \exists R.X$ if and only if $Rel(R)$ is reflexive[6],
(**4**) (W, K, Rel) is a model of $\exists R.X \sqsubseteq X$ if and only if $Rel(R)$ is included in the identity relation on W,
(**5**) (W, K, Rel) is a model of $\exists R.X \sqsubseteq \exists R.\exists R.X$ if and only if $Rel(R)$ is dense[7],
(**6**) (W, K, Rel) is a model of $\exists R.\exists R.X \sqsubseteq \exists R.X$ if and only if $Rel(R)$ is transitive[8],
(**7**) (W, K, Rel) is a model of $\exists R.\exists R.\top \sqsubseteq X$ if and only if the R-pre-image of the R-pre-image of W is empty,
(**8**) (W, K, Rel) is a model of $\exists R.X \equiv \exists S.X$ if and only if $Rel(R)$ is equal to $Rel(S)$,
(**9**) (W, K, Rel) is a model of $A \sqcap B \sqsubseteq X$ if and only if $K(A)$ and $K(B)$ do not intersect,
(**10**) (W, K, Rel) is a model of $A \sqsubseteq B$ if and only if $K(A)$ is included in $K(B)$,
(**11**) (W, K, Rel) is a model of $\exists R.X \sqsubseteq A$ if and only if the domain of R is included in $K(A)$,
(**12**) (W, K, Rel) is a model of $\exists R.X \sqsubseteq \exists R.(X \sqcap A)$ if and only if the range of R is included in $K(A)$.

Within our setting, elementary conditions — like "$Rel(R)$ is serial", "$Rel(R)$ is empty", etc — are first-order conditions that can be expressed as sentences in a function-free first-order language with equality based on a set of unary predicate symbols in one-to-one correspondence with **CON** and a set of binary predicate symbols in one-to-one correspondence with **ROL**. As a result, the following decision problems are of interest:

— **deciding elementary definability (DED):** given a T-box $\mathcal{T}$, determine whether there exists an elementary condition F such that for all frames (W, K, Rel), F holds in (W, K, Rel) if and only if (W, K, Rel) is a model of $\mathcal{T}$,

— **deciding concept definability (DCD):** given an elementary condition F, determine whether there exists a T-box $\mathcal{T}$ such that for all frames (W, K, Rel), (W, K, Rel) is a model of $\mathcal{T}$ if and only if F holds in (W, K, Rel),

— **deciding elementary equivalence (DEE):** given a T-box $\mathcal{T}$ and an elementary condition F, determine whether for all frames (W, K, Rel), (W, K, Rel) is a model of $\mathcal{T}$ if and only if F holds in (W, K, Rel).

DED, **DCD** and **DEE** stem from the corresponding definability problems in modal logics [16]. It is not known whether **DED**, **DCD** and **DEE** are decidable[9].

[5]That is to say, for all $s \in W$, there exists $t \in W$ such that $Rel(R)(s, t)$.
[6]That is to say, for all $s \in W$, $Rel(R)(s, s)$.
[7]That is to say, for all $s, t \in W$, if $Rel(R)(s, t)$ then there exists $u \in W$ such that $Rel(R)(s, u)$ and $Rel(R)(u, t)$.
[8]That is to say, for all $s, t \in W$, if there exists $u \in W$ such that $Rel(R)(s, u)$ and $Rel(R)(u, t)$ then $Rel(R)(s, t)$.
[9]Description logic languages being modal languages in disguise [6, 39], the undecidability of **DED**, **DCD** and **DEE** are immediate consequences of Chagrova's Theorems [16] when description logic $\mathcal{ALC}$ is considered instead of description logic $\mathcal{EL}$.

6. Deciding Inclusions and Equations

We present decision problems about concept inclusions and concept equations.

Let $\mathcal{T}$ be a T-box.

A concept inclusion $C \sqsubseteq D$ is a *logical consequence* of $\mathcal{T}$ (denoted $\mathcal{T} \models C \sqsubseteq D$) if for all $\mathcal{T}$-models (W, K, Rel) and for all var-interpretations V on (W, K, Rel), $\|C\|_V \subseteq \|D\|_V$. A concept equation $C = D$ is a *logical consequence* of $\mathcal{T}$ (denoted $\mathcal{T} \models C = D$) if for all $\mathcal{T}$-models (W, K, Rel) and for all var-interpretations V on (W, K, Rel), $\|C\|_V = \|D\|_V$. As a result, the following decision problems are of interest:

- **deciding concept inclusions (DCI):** given a concept inclusion $C \sqsubseteq D$, determine whether $\mathcal{T} \models C \sqsubseteq D$,

- **deciding concept equations (DCE):** given a concept equation $C = D$, determine whether $\mathcal{T} \models C = D$.

If $\mathcal{T}$ is $\mathbf{VAR}$-free then $\mathbf{DCI}$ and $\mathbf{DCE}$ are in $\mathbf{P}$ [3, 4][10]. Otherwise, it is not known whether $\mathbf{DCI}$ and $\mathbf{DCE}$ are decidable.

7. Deciding Consequences and Answers

We present decision problems about logical consequences and correct answers.

Let $(\mathcal{T}, \Pi, \mathcal{A})$ be a deductive ontology.

A clause $\alpha_1, \ldots, \alpha_m \leftarrow \beta_1, \ldots, \beta_n$ is a *logical consequence* of $(\mathcal{T}, \Pi, \mathcal{A})$ (denoted $(\mathcal{T}, \Pi, \mathcal{A}) \models \alpha_1, \ldots, \alpha_m \leftarrow \beta_1, \ldots, \beta_n$) if for all $(\mathcal{T}, \Pi, \mathcal{A})$-models (W, K, Rel, I, g) and for all var-interpretations V on (W, K, Rel), if $|\beta_1|_V^I = 1$, ..., $|\beta_n|_V^I = 1$ then either $|\alpha_1|_V^I = 1$, ..., or $|\alpha_m|_V^I = 1$. Notice that a definite clause $\alpha \leftarrow \beta_1, \ldots, \beta_n$ is a logical consequence of $(\mathcal{T}, \Pi, \mathcal{A})$ if and only if for all $(\mathcal{T}, \Pi, \mathcal{A})$-models (W, K, Rel, I, g) and for all var-interpretations V on (W, K, Rel), if $|\beta_1|_V^I = 1$, ..., $|\beta_n|_V^I = 1$ then $|\alpha|_V^I = 1$, a unit clause $\alpha \leftarrow$ is a logical consequence of $(\mathcal{T}, \Pi, \mathcal{A})$ if and only if for all $(\mathcal{T}, \Pi, \mathcal{A})$-models (W, K, Rel, I, g) and for all var-interpretations V on (W, K, Rel), $|\alpha|_V^I = 1$ and a definite goal $\leftarrow \beta_1, \ldots, \beta_n$ is a logical consequence of $(\mathcal{T}, \Pi, \mathcal{A})$ if and only if for all $(\mathcal{T}, \Pi, \mathcal{A})$-models (W, K, Rel, I, g) and for all var-interpretations V on (W, K, Rel), either $|\beta_1|_V^I = 0$, ..., or $|\beta_n|_V^I = 0$. As a result, the following decision problems are of interest:

- **deciding definite clauses (DDC):** given a definite clause $\alpha \leftarrow \beta_1, \ldots, \beta_n$, determine whether $(\mathcal{T}, \Pi, \mathcal{A}) \models \alpha \leftarrow \beta_1, \ldots, \beta_n$,

- **deciding unit clauses (DUC):** given a unit clause $\alpha \leftarrow$, determine whether $(\mathcal{T}, \Pi, \mathcal{A}) \models \alpha \leftarrow$,

[10] See [22] when other description logics are considered instead of description logic $\mathcal{EL}$.

– **deciding definite goals (DDG):** given a definite goal $\leftarrow\beta_1, \ldots, \beta_n$, determine whether $(\mathcal{T}, \Pi, \mathcal{A}) \models \leftarrow\beta_1, \ldots, \beta_n$.

A substitution σ is a *correct answer* for the definite goal $\leftarrow\beta_1, \ldots, \beta_n$ with respect to $(\mathcal{T}, \Pi, \mathcal{A})$ if for all $(\mathcal{T}, \Pi, \mathcal{A})$-models (W, K, Rel, I, g) and for all var-interpretations V on (W, K, Rel), $|\sigma(\beta_1)|^I_V{=}1$, …, $|\sigma(\beta_n)|^I_V{=}1$. As a result, the following decision problem is of interest:

– **deciding correct answers (DCA):** given a definite goal $\leftarrow\beta_1, \ldots, \beta_n$, determine whether there exists a correct answer for $\leftarrow\beta_1, \ldots, \beta_n$ with respect to $(\mathcal{T}, \Pi, \mathcal{A})$.

$\mathbf{DDC}$, $\mathbf{DUC}$, $\mathbf{DDG}$ and $\mathbf{DCA}$ stem from the corresponding derivability problems in logic programming [26, 34]. It is not known whether $\mathbf{DDC}$, $\mathbf{DUC}$, $\mathbf{DDG}$ and $\mathbf{DCA}$ are decidable[11].

8. A Research Program

We present a research program. As can be seen from its presentation, this research program covers different aspects of description logics and logic programming: recursion theory with $(\mathbf{RP_1})$, computational complexity with $(\mathbf{RP_2})$, model theory and fixpoint theory with $(\mathbf{RP_3})$, automated deduction with $(\mathbf{RP_4})$ and non-monotonic reasoning with $(\mathbf{RP_5})$. Needless to say, to carry out it, one must neither work in isolation, nor lose sight of the possible applications of the hybrid formalism developed in this paper. In other respect, with respect to expressivity, one must also compare this formalism to the main approaches proposed so far. These approaches include the above-mentioned hybrid knowledge bases [23, 24, 30, 32, 35]. They also include approaches such as the existential rule framework [13, 36].

8.1. Turing-completeness

Our hybrid formalism can be seen as a programming language. It is not known whether it is Turing-complete. When description logic $\mathcal{ALC}$ is considered instead of description logic $\mathcal{EL}$, the Turing-completeness of our hybrid formalism can be easily proved by means of a reduction from the Turing-completeness of Minsky machines. Hence, the following item in our research program: $(\mathbf{RP_1})$ separate the description logics that do give rise to a Turing-complete hybrid formalism from the description logics that do not. In particular, find simple and natural conditions on concept inclusions, concept equations and clauses such that deductive ontologies satisfying them give rise to a Turing-complete hybrid formalism.

8.2. Tractability

The success of the logic programming languages comes from the fact that it is relatively easy to define Turing-incomplete restrictions of clauses that can be used as a domain-specific language taking advantage of efficient algorithms developed for them [19, 29]. Thus, the following item

[11]When description logic $\mathcal{ALC}$ is considered instead of description logic $\mathcal{EL}$, the undecidability of $\mathbf{DDC}$, $\mathbf{DUC}$, $\mathbf{DDG}$ and $\mathbf{DCA}$ can be easily proved by means of reductions from the undecidability of the reachability problem in Minsky machines.

in our research program: $(\mathbf{RP_2})$ for the description logics that do not give rise to a Turing-complete hybrid formalism, separate those that do give rise to a hybrid formalism tractable in polynomial time from those that do not. In particular, find simple and natural conditions on concept inclusions, concept equations and clauses such that deductive ontologies satisfying them give rise to a hybrid formalism tractable in polynomial time.

8.3. Declarative and Fixpoint Semantics

In logic programming, the declarative semantics of programs is given by the usual semantics of first-order logic. It is defined in terms of Herbrand interpretations [26, 34]. In this setting, given a program, the main result is the standard characterization of its Herbrand models as the pre-fixpoints of some continuous mapping associated to it. Consequently, the following item in our research program: $(\mathbf{RP_3})$ develop the declarative and fixpoint semantics of our hybrid formalism. In particular, given a deductive ontology, characterize its Herbrand models as the pre-fixpoints of some continuous mapping associated to it.

8.4. Procedural Semantics

In logic programming, the refutation procedure of interest is called SLD-resolution where an inference step is based on the unifiability between the selected atom in a given definite goal and the left side of a variant of a definite clause in a given program. Hence, the following item in our research program: $(\mathbf{RP_4})$ develop the procedural semantics of our hybrid formalism. In particular, considering the unification problem in description logics with empty T-boxes [7], adapt the related unification algorithms to the context of our hybrid formalism[12]. In this respect, the tools and techniques developed in [2, 10, 11, 27, 28, 40] might be useful.

8.5. Negation

By using conditional assertions of the form $\alpha_1, \ldots, \alpha_m \leftarrow \beta_1, \ldots, \beta_n, \mathtt{not}(\gamma_1), \ldots, \mathtt{not}(\gamma_o)$ where α_1, …, α_m, β_1, …, β_n, γ_1, …, γ_o are atoms, one may write more expressive deductive ontologies. For instance, in our example about security policies, the deontic principle saying that every non-prohibited access is permitted and the deontic principle saying that every non-compulsory access is optional can be expressed by the following conditional assertions:

- $\mathsf{perm}(X, Y) \leftarrow \mathtt{not}(\mathsf{proh}(X, Y))$,
- $\mathsf{opti}(X, Y) \leftarrow \mathtt{not}(\mathsf{comp}(X, Y))$.

In logic programming, the declarative semantics of a program containing, possibly, negation in the right side of clauses is given by the so-called answer set semantics. It is defined in terms of stable models [21, 25, 33]. In this setting, the question of the existence of stable models for a given program is of the utmost interest. Thus, the following item in our research program: $(\mathbf{RP_5})$ develop the answer set semantics of our hybrid formalism when programs contain, possibly, negation in the right side of their clauses.

[12]The computability of the unification problem with arbitrary T-boxes is not known. In other respect, when description logic $\mathcal{ALC}$ is considered instead of description logic $\mathcal{EL}$, the computability of the unification problem either with empty T-boxes, or with arbitrary T-boxes is not known too.

9. Last Words

Our idea of an hybrid formalism where description logics constructs are used for defining concepts that are given as arguments to the predicates of the logic programs has only one ancestor: the formalism developed in [14]. In this formalism, Boolean constructs are used for defining expressions that are given as arguments to the predicates of the logic programs, allowing clauses of the form

- $\mathsf{adder}(X, Y, Z, T, U \vee V) \leftarrow \mathsf{halfAdder}(X, Y, W, U), \mathsf{halfAdder}(W, Z, T, V)$,
- $\mathsf{halfAdder}(X, Y, X \oplus Y, X \wedge Y) \leftarrow$.

where X, Y, Z, T, U, V and W denote propositional variables, $\vee$, $\oplus$ and $\wedge$ denote the Boolean constructs of, respectively, disjunction, exclusive disjunction and conjunction and adder and halfAdder are predicate symbols of, respectively, arity 5 and arity 4. Obviously, the Boolean expressions $U \vee V$, $X \oplus Y$ and $X \wedge Y$ used in these clauses can be seen as $\mathbf{ROL}$-free complex concepts when description logic $\mathcal{ALC}$ is considered instead of description logic $\mathcal{EL}$.

Knowledge representation languages such as those provided by description logic languages and rule-based reasoning paradigms such as those provided by logic programming languages are well-known and widely used in Computer Science and Artificial Intelligence. Therefore, it is quite amazing that their integration in a unique formalism similar to the formalism proposed by [14] has not been put forward during the last 30 years. A narrow-minded explanation would consist of saying that this lack of interest is the result of the lack of importance of hybrid formalisms such as the one introduced in this paper. The case study presented in Section 2 indicates that this lack of interest might just be the result of a lack of imagination. Indeed, we believe that it is time to give space to advanced languages of terms for ontologies as introduced in Sections 3 and 4, to consider the decision problems presented in Sections 5, 6 and 7 and to address the research program presented in Section 8.

Acknowledgments

Special acknowledgement is heartily granted to Stéphane Demri, Esra Erdem, Luis Fariñas del Cerro, Andreas Herzig, Rosalie Iemhoff, George Metcalfe, Mojtaba Mojtahedi, Linh Anh Nguyen, Christophe Ringeissen, Maryam Rostamigiv, Renate Schmidt and Tinko Tinchev for their feedback. We also make a point of strongly thanking the referees for their useful suggestions.

References

[1] Abou El Kalam, A., El Baida, R., Balbiani, P., Benferhat, S., Cuppens, F., Deswarte, Y., Miège, A., Saurel, C., Trouessin, G. 'Organization based access control'. In: *Proceedings POLICY 2003.* IEEE (2003) 120–131.

[2] Alizadeh, M., Ardeshir, M., Balbiani, P., Mojtahedi, M. 'Unification types in Euclidean modal logics'. *Logic Journal of the IGPL* doi.org/10.1093/jigpal/jzab036.

[3] Baader, F. ‘Terminological cycles in a description logic with existential restrictions’. In: *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence.* Morgan Kaufmann (2003) 325–330.

[4] Baader, F., Brandt, S., Lutz, C. ‘Pushing the $\mathcal{EL}$ envelope’. In: *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence.* Morgan Kaufmann (2005) 364–369.

[5] Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. *The Description Logic Handbook. Theory, Implementation and Applications.* Cambridge University Press (2003).

[6] Baader, F., Lutz, C. ‘Description logic’. In: *Handbook of Modal Logic.* Elsevier (2007) 757–819.

[7] Baader, F., Morawska, B. ‘Unification in the description logic $\mathcal{EL}$’. In: *Rewriting Techniques and Applications.* Springer (2009) 350–364.

[8] Baader, F., Nutt, W. ‘Basic description logics’. In: *The Description Logic Handbook. Theory, Implementation and Applications.* Cambridge University Press (2003) 47–100.

[9] Baader, F., Snyder, W. ‘Unification theory’, In: *Handbook of Automated Reasoning,* Elsevier (2001) 439–526.

[10] Balbiani, P., Gencer, Ç. ‘Unification in epistemic logics’. *Journal of Applied Non-Classical Logics* **27** (2017) 91–105.

[11] Balbiani, P., Gencer, Ç., Rostamigiv, M., Tinchev, T. ‘About the unification type of $\mathbf{K}+\Box\Box\bot$’. *Annals of Mathematics and Artificial Intelligence* doi.org/10.1007/s10472-021-09768-w.

[12] Ben Abbès, S., Scheuermann, A., Meilender, T., d’Aquin, M. ‘Characterizing modular ontologies’. In: *Workshop on Modular Ontologies (WoMO) 2012.* CEUR-WS.org (2012) 16–27.

[13] Bienvenu, M., Leclère, M., Mugnier, M.-L., Rousset, M.-C. ‘Reasoning with ontologies’. In: *A Guided Tour of Artificial Intelligence Research.* Springer (2020) 185–215.

[14] Büttner, W., Simonis, H. ‘Embedding Boolean expressions into logic programming’. *Journal of Symbolic Computation* **4** (1987) 191–205.

[15] Calvanese, D., De Giacomo, G. ‘Expressive description logics’. In: *The Description Logic Handbook. Theory, Implementation and Applications.* Cambridge University Press (2003) 178–218

[16] Chagrov, A., Chagrova, L. ‘The truth about algorithmic problems in correspondence theory’. In: *A*dvances in Modal Logic. Vol. 6. College Publications (2006) 121–138.

[17] Chen, J., Alghamdi, G., Schmidt, R., Walther, D., Gao, Y. ‘Ontology extraction for large ontologies via modularity and forgetting’. In: *Proceedings of the* 10*th International Conference on Knowledge Capture.* ACM (2019) 45–52.

[18] Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U. ‘A logical framework for modularity of ontologies’. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence. Morgan Kaufmann (2007) 298–303.

[19] Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A. ‘Complexity and expressive power of logic programming’. *ACM Computing Surveys* **33** (2001) 374–425.

[20] Del-Pinto, W., Schmidt, R. ‘ABox abduction via forgetting in ALC’. In: Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence. AAAI (2019) 2768–2775.

[21] Diéguez, M. *Temporal Answer Set Programming.* Doctoral thesis of the University of A Coruña (2015).

[22] Donini, F. ‘Complexity of reasoning’. In: *The Description Logic Handbook. Theory, Implementation and Applications.* Cambridge University Press (2003) 101–141.

[23] Eiter, T., Ianni, G., Lukasiewicz, T., Schindlauer, R., Tompits, H. ‘Combining answer set

programming with description logics for the semantic web'. *Artificial Intelligence* **172** (2011) 1495–1539.

[24] Eiter, T., Ianni, G., Lukasiewicz, T., Schindlauer, R. 'Well-founded semantics for description logic programs in the semantic web'. *ACM Transactions on Computational Logic* **12** (2011) article 11.

[25] Erdem, E. *Theory and Applications of Answer Set Programming.* Doctoral thesis of the University of Texas at Austin (2002).

[26] Gabbay, D., Hogger, C., Robinson, J. *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 5: Logic Programming.* Oxford University Press (1998).

[27] Gencer, Ç. 'Description of modal logics inheriting admissible rules for $\mathbf{K}4$'. *Logic Journal of the IGPL* **10** (2002) 401–411.

[28] Gencer, Ç., de Jongh, D. 'Unification in extensions of $\mathbf{K}4$'. *Logic Journal of the IGPL* **17** (2009) 159–172.

[29] Gottlob, G., Papadimitriou, C. 'On the complexity of single-rules datalog queries'. *Information and Computation* **183** (2003) 104–122.

[30] Grosof, B., Horrocks, I., Volz, R., Decker, S. 'Description logic programs: combining logic programs with description logic'. In: *Proceedings of the 12th International Conference on World Wide Web.* ACM (2003) 48–57.

[31] Lampson, B. 'Protection'. *Operating Systems Review* **8** (1974) 18–24.

[32] Levy, A., Rousset, M.-C. 'Combining Horn rules and description logics in CARIN'. *Artificial Intelligence* **104** (1998) 165–209.

[33] Lifschitz, V. 'What is answer set programming'. In: *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence.* Association for the Advancement of Artificial Intelligence (2008) 1594–1597.

[34] Lloyd, J. *Foundations of Logic Programming.* Springer (1987).

[35] Motik, B., Rosati, R. 'Reconciling description logics and rules'. *Journal of the ACM* **57** (2010) article 30.

[36] Mugnier, M.-L., Thomazo, M. 'An introduction to ontology-based query answering with existential rules'. In: *Reasoning Web.* Springer (2014) 245–278.

[37] Parent, X., van der Torre, L. *Introduction to Deontic Logic and Normative Systems.* College Publications (2018).

[38] Sandhu, R., Coyne, E., Feinstein, H., Youman, C. 'Role-based access control models'. *IEEE Computer* **29** (1996) 38–47.

[39] Schild, K. 'A correspondence theory for terminological logics: preliminary report'. In: *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence.* Morgan Kaufmann (1991) 466–471.

[40] Sofronie-Stokkermans, V. 'On unification for bounded distributive lattices'. *ACM Transactions on Computational Logic* **8** (2007) article 12.