A Tractable Rule Language in the Modal and Description Logics that Combine CPDL with Regular Grammar Logic

Linh Anh Nguyen

Institute of Informatics, University of Warsaw
Banacha 2, 02-097 Warsaw, Poland
nguyen@mimuw.edu.pl

Abstract. Combining CPDL (Propositional Dynamic Logic with Converse) and regular grammar logic results in an expressive modal logic denoted by CPDL$_{reg}$. This logic covers TeamLog, a logical formalism used to express properties of agents' cooperation in terms of beliefs, goals and intentions. It can also be used as a description logic for expressing terminological knowledge, in which both regular role inclusion axioms and CPDL-like role constructors are allowed. In this paper, we develop an expressive rule language called Horn-CPDL$_{reg}$ that has PTime data complexity. As a special property, this rule language allows the concept constructor "universal restriction" to appear at the left hand side of general concept inclusion axioms. We use a special semantics for Horn-CPDL$_{reg}$ that is based on pseudo-interpretations. It is called the constructive semantics and coincides with the traditional semantics when the concept constructor "universal restriction" is disallowed at the left hand side of concept inclusion axioms or when the language is used as an epistemic formalism and the accessibility relations are serial. We provide an algorithm with PTime data complexity for checking whether a knowledge base in Horn-CPDL$_{reg}$ has a pseudo-model. This shows that the instance checking problem in Horn-CPDL$_{reg}$ with respect to the constructive semantics has PTime data complexity.

1 Introduction

Combining CPDL (Propositional Dynamic Logic with Converse) [16] and regular grammar logic [6, 7, 31] results in an expressive modal logic denoted by CPDL$_{reg}$ [10, 26]. This logic covers TeamLog [12, 13], a logical formalism used to express properties of agents' cooperation in terms of beliefs, goals and intentions. It can also be used as a description logic, in which both regular role inclusion axioms and CPDL-like role constructors are allowed.

Description logics (DLs) are variants of modal logics suitable for expressing terminological knowledge. They represent the domain of interest in terms of individuals (objects), concepts and roles. A concept stands for a set of individuals, a role stands for a binary relation between individuals. In comparison with modal logic, concepts correspond to formulas, role names correspond to modal indices,

roles correspond to programs in dynamic logic, and the constructors $\forall R.C$ and $\exists R.C$ correspond to the modalities $[R]C$ and $\langle R\rangle C$, respectively.

In this work, CPDL$_{reg}$ is considered as a DL and the objective is to develop an expressive rule language in CPDL$_{reg}$ that has PTIME data complexity.

1.1 Related Work and Motivation

The data complexity of the general Horn fragment in the basic DL $\mathcal{ALC}$ is NP-hard [25]. The hardness is caused by that basic roles are not required to be serial (i.e., to satisfy the condition $\forall x \exists y\, R(x, y)$). A naive approach for overcoming the NP-hardness is to disallow the concept constructor $\forall R.C$ at the LHS (left hand side) of $\sqsubseteq$ in TBox axioms [15, 1, 2, 17, 18, 20, 34, 33, 4].

$\mathcal{EL}$ [1, 2], DL-Lite [5, 4], DLP [15], Horn-$\mathcal{SHIQ}$ [17] and Horn-$\mathcal{SROIQ}$ [33] are well-known rule languages in DLs with PTIME data complexity. The combined complexity of Horn fragments of DLs were considered, amongst others, in [19]. Some tractable Horn fragments of DLs without ABoxes have also been isolated in [1, 3]. To guarantee PTIME data or combined complexity, all of the rule languages in the mentioned works disallow the concept constructor $\forall R.C$ at the LHS of $\sqsubseteq$ in TBox axioms.

More sophisticated approaches for dealing with the mentioned NP-hardness are as follows:

- allowing $\forall R.C$ to appear at the LHS of $\sqsubseteq$ in TBox axioms when R is serial and using the traditional semantics for it,
- allowing a special kind of $\forall R.C$ like $\forall \exists R.C$ (defined as $\forall R.C \sqcap \exists R.C$) to appear at the LHS of $\sqsubseteq$ in TBox axioms and using the traditional semantics for it,
- allowing $\forall R.C$ to appear at the LHS of $\sqsubseteq$ in TBox axioms and using a special semantics for it.

As discussed in the long version [27] of the current paper, our previous works [21–24, 8, 32, 30, 11, 29, 28] on rule languages in propositional modal and description logics follow the first two of the above approaches.

The objective of this paper is to formulate an as rich as possible Horn fragment in CPDL$_{reg}$ together with an appropriate semantics for it. As discussed in [25, 28], for $R \in \mathbf{R}$, the concept constructor $\forall \exists R.C$ is more constructive than $\forall R.C$ at the LHS of $\sqsubseteq$ in TBox axioms. For the case when R is not a basic role, a constructor similar to $[\pi]_\diamond \varphi$ of [23] seems to be too strong and complicated for practical applications. A natural question is: *Can the concept constructor $\forall R.C$ be directly used at the LHS of $\sqsubseteq$ in TBox axioms?* Our answer is: *Yes, why not? To obtain the* PTIME *data complexity, just formulate and use an appropriate semantics for that constructor.*

1.2 Our Contributions and the Structure of This Paper

We introduce a rule language called Horn-CPDL$_{reg}$ that is a fragment of CPDL$_{reg}$ with PTIME data complexity. As a special property, it allows the

concept constructors $\forall\exists R.C$ and $\forall R.C$ to appear at the LHS of $\sqsubseteq$ in TBox axioms. We use a special semantics for Horn-CPDL$_{reg}$ that is based on pseudo-interpretations. It is called the *constructive semantics* and coincides with the traditional semantics when the concept constructor $\forall R.C$ is disallowed at the LHS of TBox axioms or when the language is used as an epistemic formalism and the accessibility relations are serial. We provide an algorithm with PTIME data complexity for checking whether a knowledge base in Horn-CPDL$_{reg}$ has a pseudo-model. This shows that the instance checking problem in Horn-CPDL$_{reg}$ with respect to the constructive semantics has PTIME data complexity.

The rest of this paper is structured as follows. Section 2 recalls the notation and semantics of CPDL$_{reg}$. Section 3 defines the rule language Horn-CPDL$_{reg}$. Section 4 presents the constructive semantics of Horn-CPDL$_{reg}$ and its properties. Section 5 provides our algorithm for checking whether a given knowledge base in Horn-CPDL$_{reg}$ has a pseudo-model. Section 6 contains concluding remarks. Due to the lack of space, proofs of our results are presented in [27].

2 Preliminaries

Our language uses a countable set $\mathbf{C}$ of *concept names*, a countable set $\mathbf{R}_+$ of *role names*, and a finite set $\mathbf{I}$ of *individual names*. We use letters like a, b to denote individual names, letters like A, B to denote concept names, and letters like r, s to denote role names. We use $\overline{r}$ to denote the *inverse* of r. For $R = \overline{r}$, let $\overline{R}$ stand for r. Let $\mathbf{R}_- = \{\overline{r} \mid r \in \mathbf{R}_+\}$ and $\mathbf{R} = \mathbf{R}_+ \cup \mathbf{R}_-$. We call the roles from $\mathbf{R}$ *basic roles*.

A *context-free semi-Thue system* $\mathcal{S}$ over $\mathbf{R}$ is a finite set of context-free production rules $R \to S_1 \dots S_k$ over alphabet $\mathbf{R}$ (i.e., R, S_1, $\dots$, $S_k \in \mathbf{R}$). It is *symmetric* if, for every rule $R \to S_1 \dots S_k$ of $\mathcal{S}$, the rule $\overline{R} \to \overline{S}_k \dots \overline{S}_1$ is also in $\mathcal{S}$.[1] It is *regular* if, for every $R \in \mathbf{R}$, the set of words derivable from R using the system is a regular language over $\mathbf{R}$.

A context-free semi-Thue system is like a context-free grammar, but it has no designated start symbol and there is no distinction between terminal and non-terminal symbols. We assume that, for $R \in \mathbf{R}$, the word R is derivable from R using such a system.

A *role inclusion axiom* (RIA for short) is an expression of the form $S_1 \circ \dots \circ S_k \sqsubseteq R$, where $k \geq 0$ and $S_1, \dots, S_k, R \in \mathbf{R}$. In the case $k = 0$, the LHS of the inclusion axiom stands for the empty word ε.

A *regular RBox* $\mathcal{R}$ is a finite set of RIAs such that

$$\{R \to S_1 \dots S_k \mid (S_1 \circ \dots \circ S_k \sqsubseteq R) \in \mathcal{R}\}$$

is a symmetric regular semi-Thue system $\mathcal{S}$ over $\mathbf{R}$. We assume that $\mathcal{R}$ is given together with a mapping $\mathbf{A}$ that associates every $R \in \mathbf{R}$ with a finite automaton $\mathbf{A}_R$ recognizing the words derivable from R using $\mathcal{S}$. We call $\mathbf{A}$ the *RIA-automaton-specification* of $\mathcal{R}$.

[1] In the case $k = 0$, the right hand sides of the rules stand for ε.

$\top^{\mathcal{I}} = \Delta^{\mathcal{I}} \qquad \bot^{\mathcal{I}} = \emptyset$	$\varepsilon^{\mathcal{I}} = \{\langle x, x\rangle \mid x \in \Delta^{\mathcal{I}}\}$
$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$	$\overline{R}^{\mathcal{I}} = (R^{\mathcal{I}})^{-1}$
$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$	$(R \circ S)^{\mathcal{I}} = R^{\mathcal{I}} \circ S^{\mathcal{I}}$
$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$	$(R \sqcup S)^{\mathcal{I}} = R^{\mathcal{I}} \cup S^{\mathcal{I}}$
$(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y\,(\langle x, y\rangle \in R^{\mathcal{I}} \Rightarrow y \in C^{\mathcal{I}})\}$	$(R^*)^{\mathcal{I}} = (R^{\mathcal{I}})^*$
$(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y\,(\langle x, y\rangle \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}})\}$	

Fig. 1. Interpretation of complex concepts and complex roles.

Recall that a *finite automaton* A over alphabet $\mathbf{R}$ is a tuple $\langle \mathbf{R}, Q, q_0, \delta, F\rangle$, where Q is a finite set of states, $q_0 \in Q$ is the initial state, $\delta \subseteq Q \times \mathbf{R} \times Q$ is the transition relation, and $F \subseteq Q$ is the set of accepting states. A *run* of A on a word $R_1 \dots R_k$ over alphabet $\mathbf{R}$ is a finite sequence of states $q_0, q_1, \dots, q_k$ such that $\delta(q_{i-1}, R_i, q_i)$ holds for every $1 \leq i \leq k$. It is an *accepting run* if $q_k \in F$. We say that A *accepts* a word w if there exists an accepting run of A on w. The set of all words accepted by A is denoted by $\mathcal{L}(\mathsf{A})$.

Concepts and *roles* are defined, respectively, by the following BNF grammar rules, where $A \in \mathbf{C}$ and $r \in \mathbf{R}_+$:

$$C ::= \top \mid \bot \mid A \mid \neg C \mid C \sqcap C \mid C \sqcup C \mid \forall R.C \mid \exists R.C$$
$$R ::= \varepsilon \mid r \mid \overline{R} \mid R \circ R \mid R \sqcup R \mid R^* \mid C?$$

We use letters like C, D to denote concepts, and letters like R, S to denote roles.

A *terminological axiom*, also called a *TBox axiom*, is an expression of the form $C \sqsubseteq D$. A *TBox* is a finite set of *TBox axioms*. An *ABox* is a finite set of assertions of the form $C(a)$ or $r(a, b)$. A *knowledge base* in CPDL$_{reg}$ is a tuple $\langle \mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ consisting of a regular RBox $\mathcal{R}$, a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$.

An *interpretation* is a pair $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\rangle$, where $\Delta^{\mathcal{I}}$ is a non-empty set called the *domain* of $\mathcal{I}$ and $\cdot^{\mathcal{I}}$ is a mapping called the *interpretation function* of $\mathcal{I}$ that associates each individual name $a \in \mathbf{I}$ with an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, each concept name $A \in \mathbf{C}$ with a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and each role name $r \in \mathbf{R}_+$ with a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The interpretation function $\cdot^{\mathcal{I}}$ is extended to complex concepts and complex roles as shown in Figure 1.

Given an interpretation $\mathcal{I}$ and an axiom/assertion φ, the satisfaction relation $\mathcal{I} \models \varphi$ is defined as follows, where $\circ$ at the right hand side of "if" stands for the composition of binary relations:

$$\begin{array}{lll}
\mathcal{I} \models S_1 \circ \dots \circ S_k \sqsubseteq R & \text{if} & S_1^{\mathcal{I}} \circ \dots \circ S_k^{\mathcal{I}} \subseteq R^{\mathcal{I}} \\
\mathcal{I} \models \varepsilon \sqsubseteq R & \text{if} & \varepsilon^{\mathcal{I}} \sqsubseteq R^{\mathcal{I}} \\
\mathcal{I} \models C \sqsubseteq D & \text{if} & C^{\mathcal{I}} \subseteq D^{\mathcal{I}} \\
\mathcal{I} \models C(a) & \text{if} & a^{\mathcal{I}} \in C^{\mathcal{I}} \\
\mathcal{I} \models r(a, b) & \text{if} & \langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in r^{\mathcal{I}}.
\end{array}$$

If $\mathcal{I} \models \varphi$ then we say that $\mathcal{I}$ *validates* φ.

An interpretation $\mathcal{I}$ is a *model* of an RBox $\mathcal{R}$, a TBox $\mathcal{T}$ or an ABox $\mathcal{A}$ if it validates all the axioms/assertions of that "box". It is a *model* of a knowledge base $KB = \langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$, denoted by $\mathcal{I} \models KB$, if it is a model of $\mathcal{R}$, $\mathcal{T}$ and $\mathcal{A}$.

A knowledge base is *satisfiable* if it has a model. For a knowledge base KB, we write $KB \models \varphi$ to mean that every model of KB validates φ. If $KB \models C(a)$ then we say that a is an *instance* of C w.r.t. KB.

The *length* of a concept, an assertion or an axiom φ is the number of symbols occurring in φ. The *size* of an ABox is the sum of the lengths of its assertions. The *size* of a TBox is the sum of the lengths of its axioms.

A *reduced ABox* is a finite set of assertions of the form $A(a)$, $\neg A(a)$ or $r(a, b)$. The *data complexity* of the instance checking problem $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle \models C(a)$ is defined when $\mathcal{A}$ is a reduced ABox and is measured w.r.t. the size of $\mathcal{A}$, while assuming that $\mathbf{R}_+$, $\mathcal{R}$, $\mathcal{T}$ and $C(a)$ are fixed.

3 The Horn-CPDL$_{reg}$ Fragment

A *Horn-CPDL$_{reg}$ TBox axiom* is an expression of the form $C_l \sqsubseteq C_r$, where l stands for "left", r stands for "right", C_l and C_r are concepts defined by the following BNF grammar:

$$C_l ::= \top \mid A \mid C_l \sqcap C_l \mid C_l \sqcup C_l \mid \exists R_{l\exists}.C_l \mid \forall R_{l\forall}.C_l \mid \forall\exists r.C_l \mid \forall\exists \overline{r}.C_l \quad (1)$$

$$R_{l\exists} ::= r \mid \overline{R} \mid R_{l\exists} \circ R_{l\exists} \mid R_{l\exists} \sqcup R_{l\exists} \mid R^*_{l\exists} \mid C_l? \quad (2)$$

$$R_{l\forall} ::= r \mid \overline{R} \mid R_{l\forall} \circ R_{l\forall} \mid R_{l\forall} \sqcup R_{l\forall} \mid R^*_{l\forall} \mid (\neg C_l)? \quad (3)$$

$$C_r ::= \top \mid \bot \mid A \mid \neg C_l \mid C_r \sqcap C_r \mid \neg C_l \sqcup C_r \mid \exists R_{r\exists}.C_r \mid \forall R_{l\exists}.C_r \quad (4)$$

$$R_{r\exists} ::= r \mid \overline{R} \mid R_{r\exists} \circ R_{r\exists} \mid C_r? \quad (5)$$

A *Horn-CPDL$_{reg}$ TBox* is a finite set of *Horn-CPDL$_{reg}$ TBox axioms*.

A *Horn-CPDL$_{reg}$ clause* is a TBox axiom of the form $C_1 \sqcap \ldots \sqcap C_k \sqsubseteq D$, $\top \sqsubseteq D$, $\top \sqsubseteq \exists r.\top$ or $\top \sqsubseteq \exists \overline{r}.\top$, where:[2]

- each C_i is of the form A, $\exists R_{l\exists}.A$, $\forall R_{l\forall}.A$, $\forall\exists r.A$ or $\forall\exists \overline{r}.A$,
- D is of the form $\bot$, A, $\exists r.A$, $\exists \overline{r}.A$ or $\forall R_{l\exists}.A$,
- $R_{l\exists}$ and $R_{l\forall}$ are now restricted by the following BNF grammar:

$$R_{l\exists} ::= r \mid \overline{r} \mid R_{l\exists} \circ R_{l\exists} \mid R_{l\exists} \sqcup R_{l\exists} \mid R^*_{l\exists} \mid A? \quad (6)$$

$$R_{l\forall} ::= r \mid \overline{r} \mid R_{l\forall} \circ R_{l\forall} \mid R_{l\forall} \sqcup R_{l\forall} \mid R^*_{l\forall} \mid (\neg A)? \quad (7)$$

A *clausal Horn-CPDL$_{reg}$ TBox* consists of Horn-CPDL$_{reg}$ clauses.

A *Horn-CPDL$_{reg}$ ABox* is a finite set of assertions of the form $C_r(a)$ or $r(a, b)$, where C_r is a concept of the form specified by (4).

A *Horn-CPDL$_{reg}$ knowledge base* is a tuple $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ consisting of a regular RBox $\mathcal{R}$, a Horn-CPDL$_{reg}$ TBox $\mathcal{T}$ and a Horn-CPDL$_{reg}$ ABox $\mathcal{A}$. When $\mathcal{T}$ is a

[2] The clauses $\top \sqsubseteq \exists r.\top$ and $\top \sqsubseteq \exists \overline{r}.\top$ can be replaced by $\top \sqsubseteq \exists r.A_\top$ or $\top \sqsubseteq \exists \overline{r}.A_\top$, respectively, where $A_\top$ is a fresh concept name. We include them just for convenience.

$\bot^{\mathcal{I}} = \emptyset$	$\varepsilon^{\mathcal{I}_\exists} = \{\langle x, x\rangle \mid x \in \Delta^{\mathcal{I}}\}$	$\varepsilon^{\mathcal{I}_\forall} = \varepsilon^{\mathcal{I}_\exists}$
$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$	$\overline{R}^{\mathcal{I}_\exists} = (R^{\mathcal{I}_\exists})^{-1}$	$\overline{R}^{\mathcal{I}_\forall} = (R^{\mathcal{I}_\forall})^{-1}$
$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$	$(R \circ S)^{\mathcal{I}_\exists} = R^{\mathcal{I}_\exists} \circ S^{\mathcal{I}_\exists}$	$(R \circ S)^{\mathcal{I}_\forall} = R^{\mathcal{I}_\forall} \circ S^{\mathcal{I}_\forall}$
$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$	$(R \sqcup S)^{\mathcal{I}_\exists} = R^{\mathcal{I}_\exists} \cup S^{\mathcal{I}_\exists}$	$(R \sqcup S)^{\mathcal{I}_\forall} = R^{\mathcal{I}_\forall} \cup S^{\mathcal{I}_\forall}$
$(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$	$(R^*)^{\mathcal{I}_\exists} = (R^{\mathcal{I}_\exists})^*$	$(R^*)^{\mathcal{I}_\forall} = (R^{\mathcal{I}_\forall})^*$
$(\forall\exists R.C)^{\mathcal{I}} = (\forall R.C \sqcap \exists R.C)^{\mathcal{I}}$	$(C?)^{\mathcal{I}_\exists} = \{\langle x, x\rangle \mid C^{\mathcal{I}}(x)\}$	$(C?)^{\mathcal{I}_\forall} = (C?)^{\mathcal{I}_\exists}$
$(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y\,(\langle x, y\rangle \in R^{\mathcal{I}_\forall} \Rightarrow y \in C^{\mathcal{I}})\}$		
$(\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y\,(\langle x, y\rangle \in R^{\mathcal{I}_\exists} \wedge y \in C^{\mathcal{I}})\}$		

Fig. 2. The meaning of complex concepts and complex roles in a pseudo-interpretation.

clausal Horn-CPDL$_{reg}$ TBox and $\mathcal{A}$ is a reduced ABox, we call such a knowledge base a *clausal Horn-CPDL$_{reg}$ knowledge base*.

A *Horn-CPDL$_{reg}$ query* for the instance checking problem is an expression of the form $C(a)$, where $a \in \mathbf{I}$ and C is a concept of the family C_l specified by (1).

4 The Constructive Semantics of Horn-CPDL$_{reg}$

Pseudo-interpretations were introduced by us in [22, 23, 25]. Here, we extend that notion for CPDL$_{reg}$ to deal with inverse roles, using a slightly different notation that is closer to the traditional notation of DLs.

Definition 4.1. A *pseudo-interpretation* is a pair $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\rangle$, where $\Delta^{\mathcal{I}}$ is a non-empty set called the *domain* of $\mathcal{I}$ and $\cdot^{\mathcal{I}}$ is a mapping called the *interpretation function* of $\mathcal{I}$ that associates each individual name $a \in \mathbf{I}$ with an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, each concept name $A \in \mathbf{C}$ with a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and each role name $r \in \mathbf{R}_+$ with a pair $\langle r^{\mathcal{I}_\exists}, r^{\mathcal{I}_\forall}\rangle$ of binary relations such that:

- $r^{\mathcal{I}_\exists} \subseteq r^{\mathcal{I}_\forall} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$,
- for every $x \in \Delta^{\mathcal{I}}$, if $Y = \{y \mid \langle x, y\rangle \in r^{\mathcal{I}_\exists}\} \neq \emptyset$ then $\{y \mid \langle x, y\rangle \in r^{\mathcal{I}_\forall}\} = Y$.

The interpretation function $\cdot^{\mathcal{I}}$ is extended to complex concepts and complex roles as shown in Figure 2. ◁

Observe that, given a pseudo-interpretation $\mathcal{I}$ and a role R, we have that $R^{\mathcal{I}_\exists} \subseteq R^{\mathcal{I}_\forall}$, and $(\forall R.C)^{\mathcal{I}}$ may differ from $(\neg\exists R.\neg C)^{\mathcal{I}}$. If $\langle x, y\rangle \in R^{\mathcal{I}_\exists}$ then we call $\langle x, y\rangle$ a *firm* R-edge. If $\langle x, y\rangle \in R^{\mathcal{I}_\forall} \setminus R^{\mathcal{I}_\exists}$ then call $\langle x, y\rangle$ a *pseudo* R-edge.

Definition 4.2. Given a pseudo-interpretation $\mathcal{I}$ and an axiom/assertion φ, the satisfaction relation $\mathcal{I} \models\!\!\!\sim \varphi$ is defined as follows:

$\mathcal{I} \models\!\!\!\sim S_1 \circ \cdots \circ S_k \sqsubseteq R$	if $S_1^{\mathcal{I}_\exists} \circ \cdots \circ S_k^{\mathcal{I}_\exists} \subseteq R^{\mathcal{I}_\exists}$ and $S_1^{\mathcal{I}_\forall} \circ \cdots \circ S_k^{\mathcal{I}_\forall} \subseteq R^{\mathcal{I}_\forall}$
$\mathcal{I} \models\!\!\!\sim \varepsilon \sqsubseteq R$	if $\varepsilon^{\mathcal{I}_\exists} \sqsubseteq R^{\mathcal{I}_\exists}$
$\mathcal{I} \models\!\!\!\sim C \sqsubseteq D$	if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
$\mathcal{I} \models\!\!\!\sim C(a)$	if $a^{\mathcal{I}} \in C^{\mathcal{I}}$
$\mathcal{I} \models\!\!\!\sim r(a, b)$	if $\langle a^{\mathcal{I}}, b^{\mathcal{I}}\rangle \in r^{\mathcal{I}_\exists}$.

If $\mathcal{I} \models\!\!\!\sim \varphi$ then we say that $\mathcal{I}$ *validates* φ. A pseudo-interpretation $\mathcal{I}$ is a *pseudo-model* of an RBox $\mathcal{R}$, a TBox $\mathcal{T}$ or an ABox $\mathcal{A}$ if it validates all the axioms/assertions of that "box". It is a *pseudo-model* of a knowledge base $KB = \langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$, denoted by $\mathcal{I} \models\!\!\!\sim KB$, if it is a pseudo-model of $\mathcal{R}$, $\mathcal{T}$ and $\mathcal{A}$. A knowledge base is *satisfiable w.r.t. the constructive semantics* if it has a pseudo-model. We define that $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle \models\!\!\!\sim C(a)$ if, for every pseudo-model $\mathcal{I}$ of $\langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$, it holds that $\mathcal{I} \models\!\!\!\sim C(a)$. $\triangleleft$

Remark 4.3. An interpretation $\mathcal{I}$ can be treated as a pseudo-interpretation with $r^{\mathcal{I}_\exists} = r^{\mathcal{I}_\forall} = r^{\mathcal{I}}$ for all $r \in \mathbf{R}_+$. Thus, given an interpretation $\mathcal{I}$, $\mathcal{I} \models KB$ iff $\mathcal{I} \models\!\!\!\sim KB$, and $\mathcal{I} \models C(a)$ iff $\mathcal{I} \models\!\!\!\sim C(a)$. Conversely, a pseudo-interpretation $\mathcal{I}$ satisfying $r^{\mathcal{I}_\exists} = r^{\mathcal{I}_\forall} = r^{\mathcal{I}}$ for all $r \in \mathbf{R}_+$ can be treated as an interpretation. In particular, if $\mathcal{I} \models (\top \sqsubseteq \exists R.\top)$ for all $R \in \mathbf{R}$, then $\mathcal{I}$ can be treated as an interpretation.

Proposition 4.4. *Let $KB = \langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ be a Horn-CPDL$_{reg}$ knowledge base.*

1. *If $C(a)$ is a Horn-CPDL$_{reg}$ query then, for the Horn-CPDL$_{reg}$ knowledge base $KB' = \langle \mathcal{R}, \mathcal{T} \cup \{C \sqsubseteq A\}, \mathcal{A} \cup \{\neg A(a)\} \rangle$, where A is a fresh concept name, we have that:*
   (a) $KB \models\!\!\!\sim C(a)$ iff KB' does not have any pseudo-model,
   (b) $KB \models C(a)$ iff KB' does not have any model.
2. *KB can be converted in polynomial time in the sizes of $\mathcal{T}$ and $\mathcal{A}$ to a Horn-CPDL$_{reg}$ knowledge base $KB' = \langle \mathcal{R}, \mathcal{T}', \mathcal{A}' \rangle$ with $\mathcal{A}'$ being a reduced ABox such that KB has a pseudo-model (resp. model) iff KB' has a pseudo-model (resp. model).*
3. *KB can be converted in polynomial time in the size of $\mathcal{T}$ to a Horn-CPDL$_{reg}$ knowledge base $KB' = \langle \mathcal{R}, \mathcal{T}', \mathcal{A} \rangle$ with $\mathcal{T}'$ being a clausal Horn-CPDL$_{reg}$ TBox such that:*
   - *KB has a pseudo-model (resp. model) iff KB' has a pseudo-model (resp. model),*
   - *if $\mathcal{T}$ does not use the constructor $\forall R.C$ at the LHS of $\sqsubseteq$ then $\mathcal{T}'$ does neither.*

Corollary 4.5. *Every Horn-CPDL$_{reg}$ knowledge base KB can be converted in polynomial time in the sizes of $\mathcal{T}$ and $\mathcal{A}$ to a clausal Horn-CPDL$_{reg}$ knowledge base $KB' = \langle \mathcal{R}, \mathcal{T}', \mathcal{A}' \rangle$ such that KB has a pseudo-model (resp. model) iff KB' has a pseudo-model (resp. model).*

We present basic properties of the constructive semantics of Horn-CPDL$_{reg}$.

Theorem 4.6. *Let KB be a clausal Horn-CPDL$_{reg}$ knowledge base and $C(a)$ be a Horn-CPDL$_{reg}$ query. Then:*

1. *If $KB \models\!\!\!\sim C(a)$ then $KB \models C(a)$.*
2. *If $\{\top \sqsubseteq \exists R.\top \mid R \in \mathbf{R}\} \subseteq \mathcal{T}$ then:*
   (a) if KB has a pseudo-model then it also has a model,
   (b) $KB \models\!\!\!\sim C(a)$ iff $KB \models C(a)$.

3. *If KB is specified without using the constructor* $\forall R_{l\forall}.C_l$ *in the grammar rule (1) and has a pseudo-model then it also has a model.*
4. *If KB and C are specified without using the constructor* $\forall R_{l\forall}.C_l$ *in the grammar rule (1), then* $KB \models\!\!\sim C(a)$ *iff* $KB \models C(a)$.

5 Checking Constructive Satisfiability in Horn-CPDL$_{reg}$

In this section we present an algorithm that, given a clausal Horn-CPDL$_{reg}$ knowledge base $KB = \langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ together with the RIA-automaton-specification $\mathbf{A}$ of $\mathcal{R}$, checks whether the knowledge base has a pseudo-model.

5.1 Automaton-Modal Operators

We say that a role is in the inverse-and-test normal form (ITNF) if in its construction the inverse operation is applied only to role names and the test operator $C?$ is applied only to concepts C of the form A or $\neg A$. Such a role can be treated as a regular expression over the alphabet $\Sigma = \mathbf{R} \cup \{A?, (\neg A)? \mid A \in \mathbf{C}\}$ (where $\circ$ corresponds to ; and $\sqcup$ corresponds to $\cup$). The regular language characterized by such a role R is denoted by $\mathcal{L}(R)$. A word $R_1 R_2 \dots R_k$ over Σ is also treated as the role $R_1 \circ R_2 \circ \dots \circ R_k$.

For each role R in ITNF, let $\mathbb{A}_R$ be a finite automaton recognizing the regular language $\mathcal{L}(R)$. For each role R in ITNFsuch that $R \notin \mathbf{R}$, let $\mathbf{A}_R$ be a finite automaton recognizing the language $\mathcal{L}(R')$, where R' is obtained from R by simultaneously substituting each $S \in \mathbf{R}$ by a regular expression representing $\mathcal{L}(\mathbf{A}_S)$.

The automaton $\mathbb{A}_R$ can be constructed from R in polynomial time, and $\mathbf{A}_R$ can be constructed in polynomial time in the length of R and the sizes of the automata $(\mathbf{A}_S)_{S \in \mathbf{R}}$. Roughly speaking, $\mathbf{A}_R$ can be obtained from $\mathbb{A}_R$ by simultaneously substituting each transition $\langle q_1, S, q_2 \rangle$ by the automaton $\mathbf{A}_S$.

Given a role R in ITNF, by $\mathbf{A}_{\overline{R}}$ we denote $\mathbf{A}_S$ with S being $\overline{R}$ in ITNF.

Given an interpretation (resp. pseudo-interpretation) $\mathcal{I}$ and a finite automaton A over alphabet Σ, we define $\mathsf{A}^{\mathcal{I}}$ (resp. $\mathsf{A}^{\mathcal{I}_\forall}$, $\mathsf{A}^{\mathcal{I}_\exists}$) to be $\{\langle x, y \rangle \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid$ there exist a word $R_1 \dots R_k$ accepted by A and elements $x_0 = x, x_1, \dots, x_k = y$ of $\Delta^{\mathcal{I}}$ such that $\langle x_{i-1}, x_i \rangle \in R_i^{\mathcal{I}}$ (resp. $\langle x_{i-1}, x_i \rangle \in R_i^{\mathcal{I}_\forall}$, $\langle x_{i-1}, x_i \rangle \in R_i^{\mathcal{I}_\exists}$) for all $1 \leq i \leq k\}$.

We will use auxiliary concept constructors $[\mathsf{A}]C$, $[\mathsf{A}]_\exists C$ and $\langle \mathsf{A} \rangle C$, where A is a finite automaton over alphabet Σ and C is a concept. Such constructors (called formulas with automaton-modal operators) were used earlier, among others, in [16, 14, 24, 9, 25, 10]. The semantics of concepts $[\mathsf{A}]C$, $[\mathsf{A}]_\exists C$, $\langle \mathsf{A} \rangle C$ are specified below:

– given an interpretation $\mathcal{I}$,

$$([\mathsf{A}]C)^{\mathcal{I}} = \big\{x \in \Delta^{\mathcal{I}} \mid \forall y \big(\langle x, y \rangle \in \mathsf{A}^{\mathcal{I}} \text{ implies } y \in C^{\mathcal{I}}\big)\big\},$$
$$(\langle \mathsf{A} \rangle C)^{\mathcal{I}} = \big\{x \in \Delta^{\mathcal{I}} \mid \exists y \big(\langle x, y \rangle \in \mathsf{A}^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\big)\big\};$$

– given a pseudo-interpretation $\mathcal{I}$,

$$([\mathsf{A}]C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y(\langle x, y\rangle \in \mathsf{A}^{\mathcal{I}_\forall} \text{ implies } y \in C^{\mathcal{I}})\},$$
$$([\mathsf{A}]_\exists C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y(\langle x, y\rangle \in \mathsf{A}^{\mathcal{I}_\exists} \text{ implies } y \in C^{\mathcal{I}})\},$$
$$(\langle\mathsf{A}\rangle C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y(\langle x, y\rangle \in \mathsf{A}^{\mathcal{I}_\exists} \text{ and } y \in C^{\mathcal{I}})\}.$$

For a finite automaton A over Σ, let the components of A be denoted as in

$$\mathsf{A} = \langle \Sigma, Q_{\mathsf{A}}, q_{\mathsf{A}}, \delta_{\mathsf{A}}, F_{\mathsf{A}}\rangle.$$

If q is a state of a finite automaton A then by A_q we denote the finite automaton obtained from A by replacing the initial state by q.

Lemma 5.1. *Let $\mathcal{I}$ be a pseudo-model of a regular RBox $\mathcal{R}$, $\mathbf{A}$ the RIA-automaton-specification of $\mathcal{R}$, and C a concept. Then:*

– $(\forall R.C)^{\mathcal{I}} = ([\mathbf{A}_R]C)^{\mathcal{I}}$ *and* $(\exists R.C)^{\mathcal{I}} = (\langle\mathbf{A}_R\rangle C)^{\mathcal{I}}$,
– $C^{\mathcal{I}} \subseteq ([\mathbf{A}_{\overline{R}}]_\exists \langle\mathbf{A}_R\rangle C)^{\mathcal{I}}$ *and* $C^{\mathcal{I}} \subseteq ([\mathbf{A}_{\overline{R}}]_\exists \exists R.C)^{\mathcal{I}}$.

The proof of this lemma is straightforward.

5.2 Our Algorithm

We will treat each TBox axiom $C \sqsubseteq D$ from $\mathcal{T}$ as a concept standing for a global assumption. That is, $C \sqsubseteq D$ is logically equivalent to $\neg C \sqcup D$, and it is a global assumption for an interpretation $\mathcal{I}$ if $(\neg C \sqcup D)^{\mathcal{I}} = \Delta^{\mathcal{I}}$.

Let X be a set of concepts. The *saturation* of X (w.r.t. $\mathbf{A}$ and $\mathcal{T}$), denoted by $\mathsf{Satr}(X)$, is defined to be the least extension of X such that:

1. for every $R \in \mathbf{R}$, $[\mathbf{A}_{\overline{R}}]_\exists \exists R.\top \in \mathsf{Satr}(X)$,
2. if $\forall R.C \in \mathsf{Satr}(X)$ then $[\mathbf{A}_R]C \in \mathsf{Satr}(X)$,
3. if $[\mathsf{A}]C \in \mathsf{Satr}(X)$, $\langle q_{\mathsf{A}}, B?, q\rangle \in \delta_{\mathsf{A}}$ and $B \in \mathsf{Satr}(X)$ then $[\mathsf{A}_q]C \in \mathsf{Satr}(X)$,
4. if $[\mathsf{A}]_\exists C \in \mathsf{Satr}(X)$, $\langle q_{\mathsf{A}}, B?, q\rangle \in \delta_{\mathsf{A}}$ and $B \in \mathsf{Satr}(X)$ then $[\mathsf{A}_q]_\exists C \in \mathsf{Satr}(X)$,
5. if $([\mathsf{A}]C \in \mathsf{Satr}(X)$ or $[\mathsf{A}]_\exists C \in \mathsf{Satr}(X))$ and $q_{\mathsf{A}} \in F_{\mathsf{A}}$ then $C \in \mathsf{Satr}(X)$,
6. if $B \in \mathsf{Satr}(X)$ and $\exists R.B$ occurs at the LHS of $\sqsubseteq$ in some clause of $\mathcal{T}$ then $[\mathbf{A}_{\overline{R}}]_\exists \langle\mathbf{A}_R\rangle B \in \mathsf{Satr}(X)$.

For $R \in \mathbf{R}$, there are two kinds of *transfer* of X through R:

$$\mathsf{Trans}(X, R) = \{[\mathsf{A}_q]C \mid [\mathsf{A}]C \in X \text{ and } \langle q_{\mathsf{A}}, R, q\rangle \in \delta_{\mathsf{A}}\}$$
$$\mathsf{Trans}_\exists(X, R) = \mathsf{Trans}(X, R) \cup \{[\mathsf{A}_q]_\exists C \mid [\mathsf{A}]_\exists C \in X \text{ and } \langle q_{\mathsf{A}}, R, q\rangle \in \delta_{\mathsf{A}}\}.$$

Our algorithm for checking whether $KB = \langle\mathcal{R}, \mathcal{T}, \mathcal{A}\rangle$ has a pseudo-model uses the data structure $G = \langle\Delta_0, \Delta, \mathit{Label}, \mathit{Next}, \mathit{LeastSucc}, \mathit{Status}\rangle$, which is called a *Horn-CPDL$_{reg}$ graph*, where:

– Δ_0 : the set of all individual names occurring in $\mathcal{A}$,
– Δ : a set of objects including Δ_0,

– *Label* : a function mapping each $x \in \Delta$ to a set of concepts,
– *Next* : $\Delta \times \{\exists R.\top, \exists R.A \mid R \in \mathbf{R}, A \in \mathbf{C}\} \to \Delta$ is a partial mapping,
– *LeastSucc* : $\Delta \times \mathbf{R} \to \Delta$ is a partial mapping,
– *Status* $\in \{unknown, unsat, sat\}$.

Define $Edges = \{\langle x, R, y\rangle \mid R(x, y) \in \mathcal{A}$ or $Next(x, \exists R.C) = y$ for some C or $LeastSucc(x, R) = y\}$. A tuple $\langle x, R, y\rangle \in Edges$ represents an edge $\langle x, y\rangle$ with label R of the graph. If $R(x, y) \in \mathcal{A}$ or $Next(x, \exists R.C) = y$ then we call $\langle x, R, y\rangle$ a *firm edge*, else if $LeastSucc(x, R) = y$ then we call $\langle x, R, y\rangle$ a *pseudo edge*. The notions of *predecessor* and *successor* are defined as usual. We say that $x \in \Delta$ is *reachable* from Δ_0 if there exist $x_0, \ldots, x_k \in \Delta$ and elements $R_1, \ldots, R_k$ of $\mathbf{R}$ such that $k \geq 0$, $x_0 \in \Delta_0$, $x_k = x$ and $\langle x_{i-1}, R_i, x_i\rangle \in Edges$ for all $1 \leq i \leq k$.

For $x \in \Delta$, $Label(x)$ is called the *label* of x. A fact $Next(x, \exists R.C) = y$ means that $\exists R.C \in Label(x)$, $C \in Label(y)$, and $\exists R.C$ is "realized" at x by going to y. When defined, $Next(x, \exists R.\top)$ denotes the "logically smallest firm R-successor of x", and $LeastSucc(x, R)$ denotes the "logically smallest R-successor of x". A fact $Status = unsat$ means the knowledge base does not have any pseudo-model. A fact $Status = unsat$ means the knowledge base has a pseudo-model.

Definition 5.2. Let $G, x \not\models_c [\mathsf{A}]B$ stand for "it is not certain that G satisfies $[\mathsf{A}]B$ at x", where $x \in \Delta$, A is a finite automaton over Σ and $B \in \mathbf{C}$. We define $\not\models_c$ to be the smallest relation such that $G, x \not\models_c [\mathsf{A}]B$ holds if one of the following holds (for some B' or R when it is related):

– $q_{\mathsf{A}} \in F_{\mathsf{A}}$ and $B \notin Label(x)$;
– $\langle q_{\mathsf{A}}, (\neg B')?, q\rangle \in \delta_{\mathsf{A}}$, $B' \notin Label(x)$ and $G, x \not\models_c [\mathsf{A}_q]B$;
– $\langle q_{\mathsf{A}}, R, q\rangle \in \delta_{\mathsf{A}}$, $\exists R.\top \notin Label(x)$ and $LeastSucc(x, R)$ is not defined;
– $\langle q_{\mathsf{A}}, R, q\rangle \in \delta_{\mathsf{A}}$, $\exists R.\top \notin Label(x)$, $LeastSucc(x, R) = y$ and $G, y \not\models_c [\mathsf{A}_q]B$;
– $\langle q_{\mathsf{A}}, R, q\rangle \in \delta_{\mathsf{A}}$, $\exists R.\top \in Label(x)$ and $Next(x, \exists R.\top)$ is not defined;
– $\langle q_{\mathsf{A}}, R, q\rangle \in \delta_{\mathsf{A}}$, $Next(x, \exists R.\top) = y$ and $G, y \not\models_c [\mathsf{A}_q]B$.

We define that $G, x \not\models_c \forall R.A$ if $G, x \not\models_c [\mathbf{A}_R]A$. ◁

Algorithm 1 attempts to construct a pseudo-model of KB by initializing a Horn-CPDL$_{reg}$ graph and then expanding it by the rules in Table 1. The intended pseudo-model extends $\mathcal{A}$ with disjoint trees rooted at the named individuals occurring in $\mathcal{A}$. The trees may be infinite. However, we represent such a semi-forest as a graph with global caching: if two nodes that are not named individuals occur in a tree or in different trees and have the same label, then they should be merged.

Theorem 5.3. *Algorithm 1 runs in polynomial time in the size of the ABox $\mathcal{A}$ and correctly checks whether the clausal Horn-CPDL$_{reg}$ knowledge base KB has a pseudo-model.*

Corollary 5.4. *The Horn-CPDL$_{reg}$ rule language has* PTIME *data complexity (when used with the constructive semantics).*

See [27] for an explanation of Algorithm 1 and an illustrative example.

Function Find(X)

if *there exists* $z \in \Delta \setminus \Delta_0$ *with* $\mathit{Label}(z) = X$ **then**
  return z
else
  add a new element z to Δ with $\mathit{Label}(z) := X$;
  return z

Procedure ExtendLabel(z, X)

if $X \subseteq \mathit{Label}(z)$ **then return**;
if $z \in \Delta_0$ **then**
  $\mathit{Label}(z) := \mathit{Label}(z) \cup \mathsf{Satr}(X)$
else
  $z_* := \mathtt{Find}(\mathit{Label}(z) \cup \mathsf{Satr}(X))$;
  foreach y, R, C *such that* $\mathit{Next}(y, \exists R.C) = z$ **do** $\mathit{Next}(y, \exists R.C) := z_*$;
  foreach y *and* R *such that* $\mathit{LeastSucc}(y, R) = z$ **do** $\mathit{LeastSucc}(y, R) := z_*$;

Function CheckPremise(x, C)

if $C = \top$ **then return** *true*
else let $C = C_1 \sqcap \ldots \sqcap C_k$;
foreach $1 \le i \le k$ **do**
  if $C_i = A$ *and* $A \notin \mathit{Label}(x)$ **then return** *false*
  else if $C_i = \forall\exists R.A$ *and* ($\exists R.\top \notin \mathit{Label}(x)$ *or* $\mathit{Next}(x, \exists R.\top)$ *is not defined or* $A \notin \mathit{Label}(\mathit{Next}(x, \exists R.\top))$) **then**
    return *false*
  else if $C_i = \exists R.A$ *and* $\langle \mathbf{A}_R \rangle A \notin \mathit{Label}(x)$ **then return** *false*
  else if $C_i = \forall R.A$ *and* $G, x \not\models_c \forall R.A$ **then return** *false*
return *true*

Algorithm 1: checking constructive satisfiability in Horn-CPDL$_{reg}$

Input: a clausal Horn-CPDL$_{reg}$ knowledge base $KB = \langle \mathcal{R}, \mathcal{T}, \mathcal{A} \rangle$ and the RIA-automaton-specification $\mathbf{A}$ of $\mathcal{R}$.
Output: *true* if KB has a pseudo-model, or *false* otherwise.
Global data: a Horn-CPDL$_{reg}$ graph G and a TBox $\mathcal{T}'$.

let Δ_0 be the set of all individuals occurring in $\mathcal{A}$;
if $\Delta_0 = \emptyset$ **then** $\Delta_0 := \{\tau\}$;
$\Delta := \Delta_0$, $\mathcal{T}' := \mathsf{Satr}(\mathcal{T})$, set *Next* and *LeastSucc* to the empty mappings;
foreach $a \in \Delta_0$ **do**
  $\mathit{Label}(a) := \mathsf{Satr}(\{A \mid A(a) \in \mathcal{A}\}) \cup \mathcal{T}'$
while *some rule in Table 1 can make changes* **do**
  choose such a rule and execute it; `// any strategy can be used`
  if *Status* = *unsat* **then return** *false*
return *true*

$(\forall_1)$	**if** $r(a,b) \in \mathcal{A}$ **then** $\texttt{ExtendLabel}(b, \mathsf{Trans}_\exists(\mathit{Label}(a), r))$, $\texttt{ExtendLabel}(a, \mathsf{Trans}_\exists(\mathit{Label}(b), \overline{r}))$;
$(\forall_2)$	**if** x is reachable from Δ_0 and $\mathit{Next}(x, \exists R.C) = y$ **then** $\mathit{Next}(x, \exists R.C) := \texttt{Find}(\mathit{Label}(y) \cup \mathsf{Satr}(\mathsf{Trans}_\exists(\mathit{Label}(x), R)))$;
$(\forall_3)$	**if** x is reachable from Δ_0 and $\mathit{Next}(x, \exists R.C) = y$ **then** $\texttt{ExtendLabel}(x, \mathsf{Trans}_\exists(\mathit{Label}(y), \overline{R}))$;
$(\forall_4)$	**if** x is reachable from Δ_0 and $\mathit{LeastSucc}(x, R) = y$ **then** $\mathit{LeastSucc}(x, R) := \texttt{Find}(\mathit{Label}(y) \cup \mathsf{Satr}(\mathsf{Trans}(\mathit{Label}(x), R)))$;
$(\forall_5)$	**if** x is reachable from Δ_0 and $\mathit{LeastSucc}(x, R) = y$ **then** $\texttt{ExtendLabel}(x, \mathsf{Trans}(\mathit{Label}(y), \overline{R}))$;
$(\exists)$	**if** x is reachable from Δ_0, $\exists R.C \in \mathit{Label}(x)$, $R \in \mathbf{R}$ and $\mathit{Next}(x, \exists R.C)$ is not defined **then** $\mathit{Next}(x, \exists R.C) := \texttt{Find}(\mathsf{Satr}(\{C\} \cup \mathsf{Trans}_\exists(\mathit{Label}(x), R)) \cup \mathcal{T}')$;
(LS)	**if** x is reachable from Δ_0, $R \in \mathbf{R}$ and $\mathit{LeastSucc}(x, R)$ is not defined **then** $\mathit{LeastSucc}(x, R) := \texttt{Find}(\mathsf{Satr}(\mathsf{Trans}(\mathit{Label}(x), R)) \cup \mathcal{T}')$;
$(\sqsubseteq)$	**if** x is reachable from Δ_0, $(C \sqsubseteq D) \in \mathit{Label}(x)$ and $\texttt{CheckPremise}(x, C)$ **then** $\texttt{ExtendLabel}(x, \{D\})$;
$(\bot)$	**if** $\bot \in \mathit{Label}(x)$ or there exists $\{A, \neg A\} \subseteq \mathit{Label}(x)$ **then** $\mathit{Status} := \mathit{unsat}$;

Table 1. Expansion rules for Horn-CPDL$_{reg}$ graphs.

6 Concluding Remarks

We have developed the rule language Horn-CPDL$_{reg}$ and proved that it has PTime data complexity by providing an algorithm for checking whether a given knowledge base in Horn-CPDL$_{reg}$ has a pseudo-model.

Horn-CPDL$_{reg}$ is more general than the Horn fragments introduced and studied in our (joint) works [21, 22, 24, 8, 32, 30, 11]. As it has PTime data complexity and is more general than Horn-TeamLog [11], it is a useful rule language for formalizing agents' cooperation.

In contrast to all the well-known Horn fragments $\mathcal{EL}$ [1, 2], DL-Lite [5], DLP [15], Horn-$\mathcal{SHIQ}$ [17], Horn-$\mathcal{SROIQ}$ [33] of DLs, Horn-CPDL$_{reg}$ allows the concept constructors $\forall\exists R.C$ (for $R \in \mathbf{R}$) and $\forall R.C$ (for any role R) to appear at the LHS of TBox axioms.

In comparison with Horn-DL [29, 28], apart from the concept constructor $\forall\exists R.C$ (for $R \in \mathbf{R}$), Horn-CPDL$_{reg}$ also allows the concept constructor $\forall R.C$ (for any role R) to appear at the LHS of TBox axioms. However, Horn-CPDL$_{reg}$ is not more general than Horn-DL because the latter additionally allows nominals, quantified number restrictions, the $\exists r.\mathsf{Self}$ constructor, the universal role as well as assertions of the form $\mathsf{disjoint}(s, s')$, $\mathsf{irreflexive}(s)$, $\neg s(a, b)$, $a \neq b$. As future work, we will extend Horn-CPDL$_{reg}$ with these features to obtain a rule

language Horn-DL2 that is more general than Horn-DL, and hence also more general than Horn-$\mathcal{SHIQ}$ and Horn-$\mathcal{SROIQ}$.

Our approach and method for Horn-CPDL$_{reg}$ make important steps in developing richer and richer tractable rule languages in modal and description logics.

Acknowledgments. This work was supported by the Polish National Science Centre (NCN) under Grant No. 2011/01/B/ST6/02769.

References

1. F. Baader, S. Brandt, and C. Lutz. Pushing the EL envelope. In *Proceedings of IJCAI'2005*, pages 364–369. Morgan-Kaufmann Publishers, 2005.
2. F. Baader, S. Brandt, and C. Lutz. Pushing the EL envelope further. In *Proceedings of the OWLED 2008 DC Workshop on OWL: Experiences and Directions*, 2008.
3. S. Brandt. Polynomial time reasoning in a description logic with existential restrictions, GCI axioms, and - what else? In *Proceedings of ECAI'2004*, pages 298–302. IOS Press, 2004.
4. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Data complexity of query answering in description logics. *Artif. Intell.*, 195:335–360, 2013.
5. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *J. Autom. Reasoning*, 39(3):385–429, 2007.
6. S. Demri. The complexity of regularity in grammar logics and related modal logics. *Journal of Logic and Computation*, 11(6):933–960, 2001.
7. S. Demri and H. de Nivelle. Deciding regular grammar logics with converse through first-order logic. *Journal of Logic, Language and Information*, 14(3):289–329, 2005.
8. B. Dunin-Kęplicz, L.A. Nguyen, and A. Szałas. Tractable approximate knowledge fusion using the Horn fragment of serial propositional dynamic logic. *Int. J. Approx. Reasoning*, 51(3), 2010.
9. B. Dunin-Kęplicz, L.A. Nguyen, and A. Szałas. Tractable approximate knowledge fusion using the Horn fragment of serial propositional dynamic logic. *Int. J. Approx. Reasoning*, 51(3):346–362, 2010.
10. B. Dunin-Kęplicz, L.A. Nguyen, and A. Szałas. Converse-PDL with regular inclusion axioms: A framework for MAS logics. *J. Applied Non-Classical Logics*, 21(1):61–91, 2011.
11. B. Dunin-Kęplicz, L.A. Nguyen, and A. Szałas. Horn-TeamLog: A Horn fragment of TeamLog with PTime data complexity. In *Proceedings of ICCCI'2013*, volume 8083 of *LNCS*, pages 143–153. Springer, 2013.
12. B. Dunin-Kęplicz and R. Verbrugge. Collective intentions. *Fundam. Inform.*, 51(3):271–295, 2002.
13. M. Dziubiński, R. Verbrugge, and B. Dunin-Kęplicz. Complexity issues in multiagent logics. *Fundam. Inform.*, 75(1-4):239–262, 2007.
14. R. Goré and L.A. Nguyen. A tableau system with automaton-labelled formulae for regular grammar logics. In B. Beckert, editor, *Proceedings of TABLEAUX 2005, LNAI 3702*, pages 138–152. Springer-Verlag, 2005.
15. B.N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description logic programs: combining logic programs with description logic. In *Proceedings of WWW'2003*, pages 48–57, 2003.

16. D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic*. MIT Press, 2000.
17. U. Hustadt, B. Motik, and U. Sattler. Reasoning in description logics by a reduction to disjunctive Datalog. *J. Autom. Reasoning*, 39(3):351–384, 2007.
18. A. Krisnadhi and C. Lutz. Data complexity in the EL family of description logics. In *Proceedings of LPAR'2007*, volume 4790 of *LNCS*, pages 333–347. Springer, 2007.
19. M. Krötzsch, S. Rudolph, and P. Hitzler. Complexity boundaries for Horn description logics. In *Proceedings of AAAI'2007*, pages 452–457. AAAI Press, 2007.
20. M. Krötzsch, S. Rudolph, and P. Hitzler. Conjunctive queries for a tractable fragment of OWL 1.1. In *Proceedings of ISWC'2007 + ASWC'2007, LNCS 4825*, pages 310–323. Springer, 2007.
21. L.A. Nguyen. Constructing the least models for positive modal logic programs. *Fundamenta Informaticae*, 42(1):29–60, 2000.
22. L.A. Nguyen. A bottom-up method for the deterministic Horn fragment of the description logic $\mathcal{ALC}$. In *Proceedings of JELIA'2006*, volume 4160 of *LNAI*, pages 346–358. Springer-Verlag, 2006.
23. L.A. Nguyen. On the deterministic Horn fragment of test-free PDL. In I. Hodkinson and Y. Venema, editors, *Advances in Modal Logic - Volume 6*, pages 373–392. King's College Publications, 2006.
24. L.A. Nguyen. Constructing finite least Kripke models for positive logic programs in serial regular grammar logics. *Logic Journal of the IGPL*, 16(2):175–193, 2008.
25. L.A. Nguyen. Horn knowledge bases in regular description logics with PTime data complexity. *Fundamenta Informaticae*, 104(4):349–384, 2010.
26. L.A. Nguyen. Cut-free ExpTime tableaux for Converse-PDL extended with regular inclusion axioms. In *Proceedings of KES-AMSTA'2013*, volume 252 of *Frontiers in Artificial Intelligence and Applications*, pages 235–244. IOS Press, 2013.
27. L.A. Nguyen. A long version of the current paper. Available at `http://www.mimuw.edu.pl/~nguyen/HornCPDLreg-long.pdf`, June 2014.
28. L.A. Nguyen, T.-B.-L. Nguyen, and A. Szałas. A long version of the paper [29]. Available at `http://www.mimuw.edu.pl/~nguyen/horn_dl_long.pdf`.
29. L.A. Nguyen, T.-B.-L. Nguyen, and A. Szałas. Horn-DL: An expressive Horn description logic with PTime data complexity. In *Proceedings of RR'2013*, volume 7994 of *LNCS*, pages 259–264. Springer, 2013.
30. L.A. Nguyen, T.-B.-L. Nguyen, and A. Szałas. On Horn knowledge bases in regular description logic with inverse. In *Proceedings of KSE'2013*, volume 244 of *Advances in Intelligent Systems and Computing*, pages 37–49. Springer, 2013.
31. L.A. Nguyen and A. Szałas. ExpTime tableau decision procedures for regular grammar logics with converse. *Studia Logica*, 98(3):387–428, 2011.
32. L.A. Nguyen and A. Szałas. On the Horn fragments of serial regular grammar logics with converse. In *Proceedings of KES-AMSTA'2013*, volume 252 of *Frontiers in Artificial Intelligence and Applications*, pages 225–234. IOS Press, 2013.
33. M. Ortiz, S. Rudolph, and M. Simkus. Query answering in the Horn fragments of the description logics SHOIQ and SROIQ. In *Proceedings of IJCAI 2011*, pages 1039–1044, 2011.
34. R. Rosati. On conjunctive query answering in $\mathcal{EL}$. In *Proceedings of DL'2007*, pages 451–458.