Actions over a constructive semantics for $\mathcal{ALC}$

Loris Bozzato, Mauro Ferrari, and Paola Villa

Dipartimento di Informatica e Comunicazione
Università degli Studi dell'Insubria
Via Mazzini 5, 21100, Varese, Italy

Abstract. Following the approaches and motivations given in recent works about action languages over description logics, we propose an action formalism based on a constructive semantics for $\mathcal{ALC}$.

1 Introduction

In [3] we have introduced the *information terms semantics* for the basic description logic $\mathcal{ALC}$. This semantics is related to the BHK interpretation, the well known constructive explanation of logical connectives (see, e.g., [14]). Specifically, in our semantics the truth of a formula in a given interpretation is explained by a mathematical object that we call *information term*. For the time being, the latter can be visualized as a sort of *proof term* inhabiting a type/formula. One of the main features of information terms semantics is that it supports a natural notion of *state* which has already been used to define the semantics of CooML (Constructive Object Oriented Modeling Language) [7, 12], a modeling language based on a constructive first-order logic.

In recent papers [2, 5, 6] different approaches to the definition of action languages over description logics have been addressed and different solutions have been proposed. In this paper we describe an approach based on the notion of state given by information terms. In our context an action is an expression $\mathcal{P} \Rightarrow \mathcal{Q}$ where $\mathcal{P}$ and $\mathcal{Q}$ are sets of $\mathcal{ALC}$ formulas. An action can be informally understood as follows: if in a given state the formulas in $\mathcal{P}$ (*preconditions*) are true, then the action can be applied and in the resulting state the formulas in $\mathcal{Q}$ (*postconditions*) will be true. In this paper we address the problems to determine executability of an action, to build the state obtained by an action application and to check its consistency. As discussed in [7, 8], these problems are strictly connected with *snapshot generation* in CooML and with model generation in SAT [10] and in Answer Set Programming [11].

For expository reasons, in this paper we restrict our attention to the simple case where the formulas occurring in $\mathcal{P} \Rightarrow \mathcal{Q}$ are literals over the terminological alphabet. However, our approach can be extended to more complex actions.

2 $\mathcal{ALC}$ language and semantics

We begin introducing the language $\mathcal{L}$ for $\mathcal{ALC}$ [1, 13], based on the following denumerable sets: the set $\mathtt{NR}$ of *role names*, the set $\mathtt{NC}$ of *concept names* and the set $\mathtt{NI}$ of *individual names*. A *concept* H is an expression of the kind:

$$H \ ::= \ C \mid \neg H \mid H \sqcap H \mid H \sqcup H \mid \exists R.H \mid \forall R.H$$

where $C \in \mathtt{NC}$ and $R \in \mathtt{NR}$. Let $\mathtt{Var}$ be a denumerable set of *individual variables*; the *formulas* K of $\mathcal{L}$ are defined according to the following grammar:

$$K \ ::= \ \bot \mid (s,t) : R \mid (s,t) : \neg R \mid t : H \mid \forall H$$

where $s, t \in \mathtt{NI} \cup \mathtt{Var}$, $R \in \mathtt{NR}$ and H is a concept. An *atomic formula* of $\mathcal{L}$ is a formula of the kind $\bot$, $(s,t) : R$ or $t : C$ where $R \in \mathtt{NR}$ and $C \in \mathtt{NC}$; a *negated formula* is a formula of the kind $(s,t) : \neg R$ or $t : \neg H$. A formula is *closed* if it does not contain variables. We write $\neg((s,t) : R)$, $\neg((s,t) : \neg R)$, $\neg(t : H)$ as abbreviations for $(s,t) : \neg R$, $(s,t) : R$, $t : \neg H$ respectively; $A \sqsubseteq B$ stands for $\forall(\neg A \sqcup B)$.

Let $\mathcal{N}$ be a finite subset of $\mathtt{NI}$. By $\mathcal{L}_{\mathcal{N}}$ we denote the set of formulas K of $\mathcal{L}$ such that all the individual names occurring in K belong to $\mathcal{N}$.

A *substitution (over $\mathcal{N}$)* is a function $\sigma : \mathtt{Var} \to \mathcal{N}$; we extend σ to $\mathcal{L}_{\mathcal{N}}$ so that for every $c \in \mathcal{N}$, $\sigma(c) = c$. Given a set of formulas Γ of $\mathcal{L}_{\mathcal{N}}$ and a substitution σ over $\mathcal{N}$, $\sigma\Gamma$ denotes the set of closed formulas obtained replacing every variable x occurring in Γ with $\sigma(x)$.

A *model (interpretation)* $\mathcal{M}$ for $\mathcal{L}$ is a pair $(\mathcal{D}^{\mathcal{M}}, \cdot^{\mathcal{M}})$, where $\mathcal{D}^{\mathcal{M}}$ is a non-empty set (the *domain* of $\mathcal{M}$) and $\cdot^{\mathcal{M}}$ is a *valuation* map such that: for every $c \in \mathtt{NI}$, $c^{\mathcal{M}} \in \mathcal{D}^{\mathcal{M}}$; for every $C \in \mathtt{NC}$, $C^{\mathcal{M}} \subseteq \mathcal{D}^{\mathcal{M}}$; for every $R \in \mathtt{NR}$, $R^{\mathcal{M}} \subseteq \mathcal{D}^{\mathcal{M}} \times \mathcal{D}^{\mathcal{M}}$. A non atomic concept H is interpreted by a subset $H^{\mathcal{M}}$ of $\mathcal{D}^{\mathcal{M}}$ as usual:

- $(\neg A)^{\mathcal{M}} = \mathcal{D}^{\mathcal{M}} \setminus A^{\mathcal{M}}$, $(A \sqcap B)^{\mathcal{M}} = A^{\mathcal{M}} \cap B^{\mathcal{M}}$, $(A \sqcup B)^{\mathcal{M}} = A^{\mathcal{M}} \cup B^{\mathcal{M}}$
- $(\exists R.A)^{\mathcal{M}} = \{\, d \in \mathcal{D}^{\mathcal{M}} \mid \exists d' \in \mathcal{D}^{\mathcal{M}} \text{ s.t. } (d,d') \in R^{\mathcal{M}} \text{ and } d' \in A^{\mathcal{M}} \,\}$
- $(\forall R.A)^{\mathcal{M}} = \{\, d \in \mathcal{D}^{\mathcal{M}} \mid \forall d' \in \mathcal{D}^{\mathcal{M}}, (d,d') \in R^{\mathcal{M}} \text{ implies } d' \in A^{\mathcal{M}} \,\}$

We say that a model $\mathcal{M}$ of $\mathcal{L}_{\mathcal{N}}$ is *reachable* if, for every $d \in \mathcal{D}^{\mathcal{M}}$, there exists $c \in \mathtt{NI}$ such that $c^{\mathcal{M}} = d$.

An *assignment* on a model $\mathcal{M}$ is a map $\theta : \mathtt{Var} \to \mathcal{D}^{\mathcal{M}}$. If $t \in \mathtt{NI} \cup \mathtt{Var}$, $t^{\mathcal{M},\theta}$ is the element of $\mathcal{D}^{\mathcal{M}}$ denoting t in $\mathcal{M}$ w.r.t. θ, namely: $t^{\mathcal{M},\theta} = \theta(t)$ if $t \in \mathtt{Var}$; $t^{\mathcal{M},\theta} = t^{\mathcal{M}}$ if $t \in \mathtt{NI}$. A formula K is *valid* in $\mathcal{M}$ w.r.t. θ, and we write $\mathcal{M}, \theta \models K$, if $K \neq \bot$ and one of the following conditions holds:

- $\mathcal{M}, \theta \models t : H$ iff $t^{\mathcal{M},\theta} \in H^{\mathcal{M}}$
- $\mathcal{M}, \theta \models (s,t) : R$ iff $(s^{\mathcal{M},\theta}, t^{\mathcal{M},\theta}) \in R^{\mathcal{M}}$
- $\mathcal{M}, \theta \models (s,t) : \neg R$ iff $(s^{\mathcal{M},\theta}, t^{\mathcal{M},\theta}) \notin R^{\mathcal{M}}$
- $\mathcal{M}, \theta \models \forall H$ iff $H^{\mathcal{M}} = \mathcal{D}^{\mathcal{M}}$

We write $\mathcal{M} \models K$ iff $\mathcal{M}, \theta \models K$ for every assignment θ. Note that $\mathcal{M} \models \forall H$ iff $\mathcal{M} \models x : H$, with x any variable. If Γ is a set of formulas, $\mathcal{M} \models \Gamma$ means that $\mathcal{M} \models K$ for every $K \in \Gamma$. We say that K is a *logical consequence* of Γ, and we write $\Gamma \models K$, iff, for every $\mathcal{M}$ and every θ, $\mathcal{M}, \theta \models \Gamma$ implies $\mathcal{M}, \theta \models K$.

3 Information terms semantics

We introduce *information terms* that will be the base structure of our constructive semantics. Given a finite subset $\mathcal{N}$ of $\mathtt{NI}$ and a closed formula K of $\mathcal{L}_{\mathcal{N}}$, we define the set of information terms $\mathrm{IT}_{\mathcal{N}}(K)$ by induction on K as follows.

$$\begin{aligned}
\mathrm{IT}_{\mathcal{N}}(K) &= \{\mathtt{tt}\}, \text{ if } K \text{ is an atomic or negated formula}\\
\mathrm{IT}_{\mathcal{N}}(c : A_1 \sqcap A_2) &= \{\,(\alpha,\beta) \mid \alpha \in \mathrm{IT}_{\mathcal{N}}(c : A_1) \text{ and } \beta \in \mathrm{IT}_{\mathcal{N}}(c : A_2)\,\}\\
\mathrm{IT}_{\mathcal{N}}(c : A_1 \sqcup A_2) &= \{\,(k,\alpha) \mid k \in \{1,2\} \text{ and } \alpha \in \mathrm{IT}_{\mathcal{N}}(c : A_k)\,\}\\
\mathrm{IT}_{\mathcal{N}}(c : \exists R.A) &= \{\,(d,\alpha) \mid d \in \mathcal{N} \text{ and } \alpha \in \mathrm{IT}_{\mathcal{N}}(d : A)\,\}\\
\mathrm{IT}_{\mathcal{N}}(c : \forall R.A) &= \mathrm{IT}_{\mathcal{N}}(\forall A) = \{\,\phi : \mathcal{N} \to \textstyle\bigcup_{d\in\mathcal{N}} \mathrm{IT}_{\mathcal{N}}(d : A) \mid \phi(d) \in \mathrm{IT}_{\mathcal{N}}(d : A)\,\}
\end{aligned}$$

Information terms for a formula K provide possible justifications for the validity of K in a classical model. Formally, let $\mathcal{M}$ be a model for $\mathcal{L}$, K a closed formula of $\mathcal{L}_{\mathcal{N}}$ and $\eta \in \mathrm{IT}_{\mathcal{N}}(K)$. We define the *realizability relation* $\mathcal{M} \rhd \langle\eta\rangle\, K$ as follows:

$$\begin{aligned}
&\mathcal{M} \rhd \langle\mathtt{tt}\rangle\, K \text{ iff } \mathcal{M} \models K, \text{ where } K \text{ is an atomic or negated formula}\\
&\mathcal{M} \rhd \langle(\alpha,\beta)\rangle\, c : A_1 \sqcap A_2 \text{ iff } \mathcal{M} \rhd \langle\alpha\rangle\, c : A_1 \text{ and } \mathcal{M} \rhd \langle\beta\rangle\, c : A_2\\
&\mathcal{M} \rhd \langle(k,\alpha)\rangle\, c : A_1 \sqcup A_2 \text{ iff } \mathcal{M} \rhd \langle\alpha\rangle\, c : A_k\\
&\mathcal{M} \rhd \langle(d,\alpha)\rangle\, c : \exists R.A \text{ iff } \mathcal{M} \models (c,d) : R \text{ and } \mathcal{M} \rhd \langle\alpha\rangle\, d : A\\
&\mathcal{M} \rhd \langle\phi\rangle\, c : \forall R.A \text{ iff } \;\mathcal{M} \models c : \forall R.A \text{ and, for every } d \in \mathcal{N},\\
&\qquad\qquad\qquad\qquad\;\; \mathcal{M} \models (c,d) : R \text{ implies } \mathcal{M} \rhd \langle\phi(d)\rangle\, d : A\\
&\mathcal{M} \rhd \langle\phi\rangle\, \forall A \text{ iff } \mathcal{M} \models \forall A \text{ and, for every } d \in \mathcal{N}, \mathcal{M} \rhd \langle\phi(d)\rangle\, d : A
\end{aligned}$$

If $\Gamma = \{K_1, \ldots, K_n\}$ is a finite set of closed formulas of $\mathcal{L}_{\mathcal{N}}$, $\mathrm{IT}_{\mathcal{N}}(\Gamma)$ denotes the set of n-tuples $\overline{\eta} = (\eta_1, \ldots, \eta_n)$ such that, for every $1 \leq j \leq n$, $\eta_j \in \mathrm{IT}_{\mathcal{N}}(K_j)$; $\mathcal{M} \rhd \langle\overline{\eta}\rangle\, \Gamma$ iff, for every $1 \leq j \leq n$, $\mathcal{M} \rhd \langle\eta_j\rangle\, K_j$.

We remark that $\mathcal{M} \rhd \langle\eta\rangle\, K$ implies $\mathcal{M} \models K$, hence the constructive semantics is compatible with the usual classical one. The converse in general does not hold and stronger conditions are required:

Proposition 1. *Let K be a closed formula of $\mathcal{L}$ and let $\mathcal{M}$ be a finite model for $\mathcal{L}$. If $\mathcal{M} \models K$, there exists a finite subset $\mathcal{N}$ of* $\mathtt{NI}$ *and $\eta \in \mathrm{IT}_{\mathcal{N}}(K)$ such that $\mathcal{M} \rhd \langle\eta\rangle\, K$.* □

We point out that in our setting negation has a classical meaning, thus negated formulas are not constructively explained by an information term. However, information terms semantics can be extended to treat various kinds of constructive negation as those discussed in [9].

In the following we will indicate with the term *theory* a set $\mathbf{T}$ of closed formulas of $\mathcal{L}_{\mathcal{N}}$ consisting of a TBox and an ABox. The TBox is a set of formulas of the kind $K = \forall A$, representing the constraints of our system. The ABox is a set of concept and role assertions that represent our knowledge about the current state of the system. A *state* of the system is any $\overline{\gamma} \in \mathrm{IT}_{\mathcal{N}}(\mathbf{T})$.

Example 1 (The alert system). In this example we model a simple home alert system. The system has two kinds of sensors, namely to detect fire and flood events: whenever a sensor activates and signals the occurrence of one of these events, a corresponding alert goes off. When a sensor stops signaling an event, the alarm must stop.

The theory that models our system consists of the TBox $\textsc{Tas}$, representing the constraints of the model:

$$(Ax_1) : \forall(\neg\texttt{CurrentAlert} \sqcup (\exists\texttt{hasReason}.\texttt{CurrentSignal} \sqcap \texttt{Alert}))$$
$$(Ax_2) : \forall(\neg\texttt{CurrentSignal} \sqcup (\texttt{Active} \sqcap (\texttt{Fire} \sqcup \texttt{Flood})))$$

that can be restated as:

$$(Ax_1) : \texttt{CurrentAlert} \sqsubseteq \exists\texttt{hasReason}.\texttt{CurrentSignal} \sqcap \texttt{Alert}$$
$$(Ax_2) : \texttt{CurrentSignal} \sqsubseteq \texttt{Active} \sqcap (\texttt{Fire} \sqcup \texttt{Flood})$$

and an ABox $\textsc{Aas}_0$:

alert1:Alert	fire_s1:Fire	flood_s1:Flood
alert2:Alert	fire_s2:Fire	flood_s2:Flood

asserting that our system has two sensors for every kind. Moreover, in the initial state none of the signals is active.

Let $\mathbf{As}_0 = \textsc{Tas} \cup \textsc{Aas}_0$ and let W be the set of the individual names occurring in $\textsc{Aas}_0$. As an example, an element of $\textsc{it}_{\mathsf{W}}(Ax_1)$ is a function ϕ mapping each $c \in \mathsf{W}$ to an element

$$\delta \in \textsc{it}_{\mathsf{W}}(c : \neg\texttt{CurrentAlert} \sqcup (\exists\texttt{hasReason}.\texttt{CurrentSignal} \sqcap \texttt{Alert}))$$

where, δ is either $(1, \texttt{tt})$ (meaning that c is not a current alert) or $(2, ((d, \texttt{tt}), \texttt{tt}))$ (i.e., c is a current alert and his reason is the current signal d).

Now, we have to select an initial state of the system consistent with our knowledge contained in $\textsc{Aas}_0$. To this aim let $\gamma_1 \in \textsc{it}_{\mathsf{W}}(Ax_1)$ and $\gamma_2 \in \textsc{it}_{\mathsf{W}}(Ax_2)$ the functions mapping every $c \in \mathsf{W}$ in $(1, \texttt{tt})$. The initial state of our system is the information term $\overline{\gamma}_0 \in \textsc{it}_{\mathsf{W}}(\mathbf{As}_0)$ associating γ_1 to Ax_1, γ_2 to Ax_2 and $\texttt{tt}$ to every formula of the ABox $\textsc{Aas}_0$. It is easy to define a model $\mathcal{M}$ of $\mathbf{As}_0$ such that $\mathcal{M} \rhd \langle\overline{\gamma}_0\rangle\, \mathbf{As}_0$. We remark that $\overline{\gamma}_0$ is the only state that can justify our theory only assuming the information contained in $\textsc{Aas}_0$. We also notice that $\overline{\gamma}_0$ assumes a sort of closed world assumption about the current state, in the sense that what is not true in the current state is assumed as false. ◇

According to the above definitions an information term is a structured data whose correct reading is provided by the related formula. According to this interpretation we call *piece of information* over $\mathcal{L}_{\mathcal{N}}$ a pair $\langle\alpha\rangle F$, where F is a closed formula of $\mathcal{L}_{\mathcal{N}}$ and $\alpha \in \textsc{it}_{\mathcal{N}}(F)$. Given a theory $\mathbf{T}$ over $\mathcal{L}_{\mathcal{N}}$, we introduce two notions of consistency:

- A state (information term) $\overline{\gamma} \in \textsc{it}_{\mathcal{N}}(\mathbf{T})$ is *consistent* if there exists a model $\mathcal{M}$ such that $\mathcal{M} \rhd \langle\overline{\gamma}\rangle\, \mathbf{T}$.
- $\mathbf{T}$ is *state consistent* if there exists $\overline{\gamma} \in \textsc{it}_{\mathcal{N}}(\mathbf{T})$ such that $\overline{\gamma}$ is consistent.

Now, we will see how, introducing the notion of information content, we can reduce the problem to check consistency of a state to the problem to check classical consistency of a set of atomic and negated formulas.

Given a finite subset $\mathcal{N}$ of $\texttt{NI}$, let $\mathcal{R}_{\mathcal{N}} = \{(s,t) : R \mid R \in \texttt{NR} \text{ and } s,t \in \mathcal{N}\}$. This set intuitively represents the set of all the possible role assertions that we can express over $\mathcal{N}$. Given a finite subset $\mathcal{R}$ of $\mathcal{R}_{\mathcal{N}}$, we define the *information content* (w.r.t. $\mathcal{R}$) of a piece of information as follows:

- $\textsc{ic}_{\mathcal{R}}(\langle \texttt{tt} \rangle c : H) = \{c : H\}$, if $c : H$ is an atomic or negated formula
- $\textsc{ic}_{\mathcal{R}}(\langle (\alpha,\beta) \rangle c : A_1 \sqcap A_2) = \textsc{ic}_{\mathcal{R}}(\langle \alpha \rangle c : A_1) \cup \textsc{ic}_{\mathcal{R}}(\langle \beta \rangle c : A_2)$
- $\textsc{ic}_{\mathcal{R}}(\langle (k,\alpha) \rangle c : A_1 \sqcup A_2) = \textsc{ic}_{\mathcal{R}}(\langle \alpha \rangle c : A_k)$ ($k = 1$ or $k = 2$)
- $\textsc{ic}_{\mathcal{R}}(\langle (d,\alpha) \rangle c : \exists R.A) = \textsc{ic}_{\mathcal{R}}(\langle \alpha \rangle d : A) \cup \{(c,d) : R\}$
- $\textsc{ic}_{\mathcal{R}}(\langle \phi \rangle c : \forall R.A) = \bigcup_{(c,d):R \in \mathcal{R}} \textsc{ic}_{\mathcal{R}}(\langle \phi(d) \rangle d : A) \cup \{(c,d) : R \mid (c,d) : R \in \mathcal{R}\}$
- $\textsc{ic}_{\mathcal{R}}(\langle \phi \rangle \forall A) = \bigcup_{d \in \mathcal{N}} \textsc{ic}_{\mathcal{R}}(\langle \phi(d) \rangle d : A)$

If $\Gamma = \{K_1, \ldots, K_n\}$ is a set of closed formulas of $\mathcal{L}_{\mathcal{N}}$ and $\overline{\gamma} \in \textsc{it}_{\mathcal{N}}(\Gamma)$, $\textsc{ic}_{\mathcal{R}}(\langle \overline{\gamma} \rangle \Gamma) = \bigcup_{K_i \in \Gamma}(\langle \gamma_i \rangle K_i)$.

We remark that the information content of a piece of information is a set of atomic and negated formulas. The following relation holds between a piece of information and its information content:

Theorem 1. *Let $\mathcal{N}$ be a finite subset of* $\texttt{NI}$*, let K be a closed formula of $\mathcal{L}_{\mathcal{N}}$, let $\mathcal{R} \subseteq \mathcal{R}_{\mathcal{N}}$ and let $\mathcal{M}$ be a reachable model of $\mathcal{L}_{\mathcal{N}}$ such that $\mathcal{M} \models (c,d) : R$ iff $(c,d) : R \in \mathcal{R}$. Then $\mathcal{M} \rhd \langle \alpha \rangle K$ iff $\mathcal{M} \models \textsc{ic}_{\mathcal{R}}(\langle \alpha \rangle K)$.* □

According with this theorem, $\textsc{ic}_{\mathcal{R}}(\langle \alpha \rangle K)$ intuitively represents the minimum[1] amount of information needed to get evidence for K according to the information term α, assuming the roles in $\mathcal{R}$. We remark that, by Theorem 1, checking consistency of a state $\overline{\gamma}$ of $\mathbf{T}$ is equivalent to check consistency of $\textsc{ic}_{\mathcal{R}}(\langle \overline{\gamma} \rangle \mathbf{T})$ for the given $\mathcal{R}$.

Example 2 (Information content). Let us consider the theory $\mathbf{As}_0$ and the information terms γ_1 and γ_2 defined in Example 1. Let $\mathcal{R} = \emptyset$, corresponding to the initial state of our system where no role is specified. Then the information content of the initial state of our system is:

$$\textsc{ic}_{\mathcal{R}}(\langle \gamma_1 \rangle Ax_1) = \{a : \neg\texttt{CurrentAlert} \mid a \in \mathsf{W}\}$$
$$\textsc{ic}_{\mathcal{R}}(\langle \gamma_2 \rangle Ax_2) = \{a : \neg\texttt{CurrentSignal} \mid a \in \mathsf{W}\}$$

Moreover, since every formula in the $\textsc{Aas}_0$ is atomic, its information content is $\textsc{Aas}_0$ itself. So we have that:

$$\begin{aligned}\textsc{ic}_{\mathcal{R}}(\langle \overline{\gamma}_0 \rangle \mathbf{As}_0) = \{a : \neg\texttt{CurrentAlert} \mid a \in \mathsf{W}\} \cup \\ \{a : \neg\texttt{CurrentSignal} \mid a \in \mathsf{W}\} \cup \textsc{Aas}_0\end{aligned}$$

Now, let $\mathcal{M}_0$ be the model having the set of individual names W as domain and mapping every individual name in itself, every concept so to satisfy $\textsc{Aas}_0$ and every role in the empty set. $\mathcal{M}_0$ satisfies the hypothesis of Theorem 1 and hence the state $\overline{\gamma}_0$ is consistent. ◇

[1] Minimality of $\textsc{ic}_{\mathcal{R}}(\langle \alpha \rangle K)$ can be formalised in model-theoretic terms (see [7]).

4 Actions

We call *literal* a formula either of the kind $t : C$, $t : \neg C$, $(s,t) : R$ or $(s,t) : \neg R$ where $C \in \texttt{NC}$ and $R \in \texttt{NR}$. Given a set of literals L, we denote with $\overline{L} = \{\neg K \mid K \in L\}$ where we assume that $\neg\neg H = H$ if H is either a role or a concept name.

An *action* over $\mathcal{L}_{\mathcal{N}}$ is an expression of the kind $\mathcal{P} \Rightarrow \mathcal{Q}$ where $\mathcal{P}$ and $\mathcal{Q}$ are sets of literals over $\mathcal{L}_{\mathcal{N}}$ and every individual variable occurring in $\mathcal{Q}$ also occurs in $\mathcal{P}$. Informally an action can be understood as follows: if in a given state the formulas in $\mathcal{P}$ (*preconditions*) are true, then the action can be applied and in the resulting state the formulas in $\mathcal{Q}$ (*postconditions*) will be true. Given an action $\alpha \equiv \mathcal{P} \Rightarrow \mathcal{Q}$ we denote with $\mathsf{Pre}(\alpha)$ the preconditions of α and with $\mathsf{Post}(\alpha)$ the postconditions of α.

Now, let $\mathbf{T}$ be a theory with ABox $\mathcal{A}$ over $\mathcal{L}_{\mathcal{N}}$ and let $\overline{\gamma} \in \textsc{it}_{\mathcal{N}}(\mathbf{T})$. The action $\alpha \equiv \mathcal{P} \Rightarrow \mathcal{Q}$ is *active in the state* $\overline{\gamma}$ *w.r.t. a substitution* σ if $\sigma\mathcal{P} \subseteq \textsc{ic}_{\mathcal{R}}(\langle\overline{\gamma}\rangle\mathbf{T})$. An active action can be applied and its application in state $\overline{\gamma}$ has two effects:

- We get a new ABox $\mathcal{A}' = (\mathcal{A} \setminus \overline{\sigma\mathcal{Q}}) \cup \sigma\mathcal{Q}$ *(ABox update)*;
- We get the set $\mathsf{Out}(\alpha) = ((\textsc{ic}_{\mathcal{R}}(\langle\overline{\gamma}\rangle\mathbf{T}) \cup \mathcal{R}) \setminus \overline{\sigma\mathcal{Q}}) \cup \sigma\mathcal{Q}$ *(action output)*.

An action application changes both the ABox of the theory and the state of the system. Obviously, given a consistent state of our system, an action application could lead the system to an inconsistent state. According to Theorem 1 to guarantee that action application is consistent we must prove that the action output is consistent and that we can construct an information term for $\mathbf{T}$ from the formulas in the action output. Before discussing the consistency issues let us define the actions governing the behaviour of our alert system.

Example 3 (Actions of the alert system). For every fire and flood sensor, we define the following actions (here defined for `fire_s1`) that model the beginning and the end of a signal from the sensor:

$$SignalFireS1() : \emptyset \Rightarrow \{\texttt{fire\_s1} : \texttt{Active}\}$$

$$UnSignalFireS1() : \emptyset \Rightarrow \{\texttt{fire\_s1} : \neg\texttt{Active}\}$$

We remark that the above actions have an empty set of preconditions, thus they are active in every state and they simply update the knowledge base. The system reacts to these signals with the following actions:

$$\begin{aligned} StartFireAlert(x) : \; & \{x\texttt{:Fire},\ x\texttt{:Active}\} \Rightarrow \\ & \{\texttt{alert1:CurrentAlert},\ x\texttt{:CurrentSignal}, \\ & (\texttt{alert1},x)\texttt{:hasReason}\} \end{aligned}$$

$$\begin{aligned} StartFloodAlert(x) : \; & \{x\texttt{:Flood},\ x\texttt{:Active}\} \Rightarrow \\ & \{\texttt{alert2:CurrentAlert},\ x\texttt{:CurrentSignal}, \\ & (\texttt{alert2},x)\texttt{:hasReason}\} \end{aligned}$$

$$
\begin{aligned}
StopAlert(x,y) : \{&x : \neg\texttt{Active},\ (y,x) : \texttt{hasReason}\} \Rightarrow \\
&\{y : \neg\texttt{CurrentAlert},\ x : \neg\texttt{CurrentSignal}, \\
&(y,x) : \neg\texttt{hasReason}\}
\end{aligned}
$$

We write actions in a function-like form to emphasise the free variables and how they are instantiated. Note that the above three actions are not active in the initial state $\overline{\gamma}_0$.

Now, let us consider the situation where a fire is detected by `fire_s1`. This raises the action $SignalFireS1()$. Let $\mathbf{As}_1 = \textsc{Tas} \cup \textsc{Aas}_1$ where $\textsc{Aas}_1 = \textsc{Aas}_0 \cup \{\texttt{fire\_s1:Active}\}$ and let $\overline{\gamma}_1$ be the information term for $\mathbf{As}_1$ associating γ_1 and γ_2 of Example 1 to Ax_1 and Ax_2 respectively, and associating `tt` to every formula in $\textsc{Aas}_1$. It is easy to check that $\textsc{ic}_{\mathcal{R}}(\langle\overline{\gamma}_1\rangle\mathbf{As}_1)$ is exactly the result of applying this action to the state $\overline{\gamma}_0$.

In the state $\overline{\gamma}_1$ the action $StartFireAlert(\texttt{fire\_s1})$ can be activated, and the result of its application is the output

$$
\begin{aligned}
Out = \textsc{Aas}_0 &\cup \{a : \neg\texttt{CurrentAlert} \mid a \in (\mathsf{W} \setminus \{\texttt{alert1}\})\} \cup \\
&\{a : \neg\texttt{CurrentSignal} \mid a \in (\mathsf{W} \setminus \{\texttt{fire\_s1}\})\} \cup \\
&\{\texttt{fire\_s1:Active},\ \texttt{alert1:CurrentAlert},\ \texttt{fire\_s1:CurrentSignal}, \\
&\ \texttt{(alert1,fire\_s1):hasReason}\}
\end{aligned}
$$

Now, let $\textsc{Aas}_2 = \textsc{Aas}_1 \cup \mathsf{Post}(StartFireAlert(\texttt{fire\_s1}))$, and let $\mathbf{As}_2 = \textsc{Tas} \cup \textsc{Aas}_2$. Let $\mathcal{M}_2$ be a model of Out, it is easy to check that $\mathcal{M}_2$ is also a model of $\mathbf{As}_2$. $\diamond$

5 An algorithm to build up information terms

We remark that we have concluded the above example without giving the state corresponding to the action application. To build up the output state of an action we will use the algorithm $\textsc{GenIt}(X, F)$ of Figure 1. This algorithm takes as input a set X of closed atomic and negated formulas of $\mathcal{L}_{\mathcal{N}}$ and a closed formula F of $\mathcal{L}_{\mathcal{N}}$ and generates as output a (possibly empty) set of information terms in $\textsc{it}_{\mathcal{N}}(F)$. $\textsc{GenIt}$ invokes the function $\textsc{OpenIt}$ of Figure 2 to compute the information terms for compound and open formulas. $\textsc{OpenIt}$ taken as input the set X and a formula F and returns a (possibly empty) set of pairs (α, c) where $\alpha \in \textsc{it}_{\mathcal{N}}(F)$ and $c \in \mathcal{N}$ (intuitively, if the pair (α, c) is built, α justifies the formula F with respect to the individual name c).

It is easy to prove, by induction on the structure of the formula F, the following result:

Theorem 2. *Let X be a set of closed atomic and negated formulas of $\mathcal{L}_{\mathcal{N}}$, let $\mathcal{R} = \{(t,t') : R \mid (t,t') : R \in X\}$ and let F be a closed formula of $\mathcal{L}_{\mathcal{N}}$. Then, for every $\tau \in \textsc{GenIt}(X, F)$, $\textsc{ic}_{\mathcal{R}}(\langle\tau\rangle F) \subseteq X$.* $\square$

```
if ($F$ is either $(c,d):R$ or $(c,d):\neg R$) then
    if ($F \in X$) then return {tt};
    else return $\emptyset$;
else if ($F = \forall A$) then begin
    let $Z = \textsc{OpenIt}(X, x:A)$;
    let $\Gamma = \{c \mid (\alpha, c) \in Z\}$;
    if ($\Gamma \neq \mathcal{N}$) then return $\emptyset$;
    else return $\{\phi \mid \phi(c) = \alpha$ with $(\alpha, c) \in Z\}$;
end
else return $\{\alpha \mid (\alpha, c) \in \textsc{OpenIt}(X, F)\}$;
```

Fig. 1. The GenIt algorithm

Example 4 (State generation). The execution of $\textsc{GenIt}(Out, Ax_1)$ provides the following information term $\gamma_1' \in \textsc{it}_{\mathsf{W}}(Ax_1)$, where we enclose between square brackets the pairs $(c, \gamma_1'(c))$ belonging to the function:

```
[ (alert1,(2,((fire_s1, tt), tt))), (alert2,(1,tt)), (fire_s1,(1,tt)),
  (fire_s2,(1,tt)), (flood_s1,(1,tt)), (flood_s2,(1,tt)) ]
```

while the execution of $\textsc{GenIt}(Out, Ax_2)$ provides the information term $\gamma_2' \in \textsc{it}_{\mathsf{W}}(Ax_2)$:

```
[ (alert1,(1,tt)), (alert2,(1,tt)), (fire_s1,(2,(tt,(1,tt)))),
  (fire_s2,(1,tt)), (flood_s1,(1,tt)), (flood_s2,(1,tt)) ]
```

Consider $\textsc{Aas}_2$ as defined in the previous example. If $\overline{\gamma}_2$ is the information term associating γ_1' to Ax_1, γ_2' to Ax_2 and `tt` to every formula of $\textsc{Aas}_2$ and if $\mathcal{M}_2$ is a model of Out, then it follows that $\mathcal{M}_2 \rhd \langle\overline{\gamma}_2\rangle\, \mathbf{As}_2$. Hence the action application leads to a consistent state and thus $\mathbf{As}_2$ is state consistent.

If we consider $\mathcal{R} = \{\texttt{(alert1,fire\_s1):hasReason}\}$, which is the set of role formulas asserted by $\textsc{Aas}_2$, then the information content of $\mathbf{As}_2$ for the axioms in $\textsc{Tas}$ is defined as:

$$\begin{aligned}
\textsc{ic}_{\mathcal{R}}(\langle\gamma_1'\rangle Ax_1) = {} & \{\texttt{alert1:Alert}, \texttt{fire\_s1:CurrentSignal},\\
& \texttt{(alert1,fire\_s1):hasReason}\} \cup\\
& \{a : \neg\texttt{CurrentAlert} \mid a \in \mathsf{W} \setminus \{\texttt{alert1}\}\}\\
\textsc{ic}_{\mathcal{R}}(\langle\gamma_2'\rangle Ax_2) = {} & \{\texttt{fire\_s1:Active}, \texttt{fire\_s1:Fire}\} \cup\\
& \{a : \neg\texttt{CurrentSignal} \mid a \in \mathsf{W} \setminus \{\texttt{fire\_s1}\}\}
\end{aligned}$$

So we have that $\textsc{ic}_{\mathcal{R}}(\langle\overline{\gamma}_2\rangle\mathbf{As}_2) = \textsc{ic}_{\mathcal{R}}(\langle\gamma_1'\rangle Ax_1) \cup \textsc{ic}_{\mathcal{R}}(\langle\gamma_2'\rangle Ax_2) \cup \textsc{Aas}_2$. As in Example 2, if $\mathcal{M}_2$ satisfies the hypotheses, then Theorem 1 holds and $\mathcal{M}_2 \rhd \langle\overline{\gamma}_2\rangle\, \mathbf{As}_2$ iff $\mathcal{M}_2 \models \textsc{ic}_{\mathcal{R}}(\langle\overline{\gamma}_2\rangle\mathbf{As}_2)$.

Note that, if `fire_s1` stops being active and $UnSignalFireS1()$ is executed, in the new state it does not hold that `fire_s1:Active` and so the action $StopAlert(\texttt{fire\_s1}, \texttt{alert1})$ can be fired. If that is the case, it is easy to verify that the system returns in the initial state of our example.

```
if ($F = t : A$ with $A \in$ NC or $A = \neg H$) then
    if ($t \in \mathcal{N}$) then
        if ($t : A \in X$) then return $\{(\mathtt{tt}, t)\}$;
        else return $\emptyset$;
    else return $\{(\mathtt{tt}, c) \mid c : A \in X\}$;
if ($F = t : A \sqcap B$) then
    return $\{((\alpha, \beta), c) \mid (\alpha, c) \in \textsc{OpenIt}(X, t : A)$ and $(\beta, c) \in \textsc{OpenIt}(X, t : B)\}$;
if ($F = t : A \sqcup B$) then
    return $\{((1, \alpha), c)) \mid (\alpha, c) \in \textsc{OpenIt}(X, t : A)\} \cup \{((2, \beta), c) \mid (\beta, c) \in \textsc{OpenIt}(X, t : B)\}$;
if ($F = t : \exists R.A$) then
    if ($t \in \mathcal{N}$) then let $D = \{t\}$;
    else let $D = \{c \mid (c, d) : R \in X\}$
    return $\{((d, \alpha), c) \mid c \in D$ and $(c, d) : R \in X$ and $(\alpha, d) \in \textsc{OpenIt}(X, x : A)\}$;
if ($F = t : \forall R.A$) then begin
    if ($t \in \mathcal{N}$) then begin
        let $D = \{t\}$;
        let $Z = \{d \mid (t, d) : R \in X$ and $(\alpha, d) \in \textsc{OpenIt}(X, x : A)\}$;
    else begin
        let $D = \{c \mid (c, d) : R \in X\}$;
        let $Z = \{d \mid (\alpha, d) \in \textsc{OpenIt}(X, x : A)\}$;
    end let $\Phi = \emptyset$;
    for all ($c \in D$) do begin
        let $C = \{d \mid (c, d) : R \in X\}$;
        if ($C \subseteq Z$) then
            $\Phi = \Phi \cup \left\{ (\phi, c) \mid \phi(d) = \begin{cases} \alpha & \text{if } d \in C \text{ and } (a, d) \in Z \\ \text{any } \eta^+ \text{ of } \textsc{it}_{\mathcal{N}}(d : A) & \text{otherwise} \end{cases} \right\}$
    end;
    return $\Phi$;
end
```

Fig. 2. The OpenIt function

Moreover, it is possible to show that, for example, since $\textsc{GenIt}(Out, Ax_1) = \{\gamma'_1\}$ and $\textsc{ic}_{\mathcal{R}}(\langle\gamma'_1\rangle Ax_1) \subseteq Out$, then Theorem 2 holds. ◇

We remark that in general GenIt generates more than one state. In this case the action could lead the system to different states and a non deterministic choice has to be done. Moreover, given a state generated by GenIt, one has to show that this state is consistent. As for the latter point, the usual considerations about checking consistency hold (see, e.g., [7]). In relation with this problem we plan to study connections of our semantics with SAT [10] and ASP [11].

An important point of our approach is that we can use GenIt as a first consistency check for an action application. Indeed, if X is the output of an action application over $\textsc{ic}_{\mathcal{R}}(\langle\overline{\gamma}\rangle\mathbf{T})$ and $\textsc{GenIt}(X, F)$ is empty for some $F \in \mathbf{T}$ then $\mathbf{T}$ is not state consistent. This usually means that the action does not provide enough information to justify the system constraints as shown in the following example.

Example 5 (Action consistency check). Suppose that, in the development of our system, we write the following (wrong) version of $StartFireAlert(x)$:

$$\begin{aligned} StartFireAlert2(x) : \{& x{:}\texttt{Fire},\ x{:}\texttt{Active}\} \Rightarrow \\ & \{\texttt{alert1:CurrentAlert},\ \texttt{(alert1,}x\texttt{):hasReason}\}\end{aligned}$$

As can be noted, we forgot to set $x{:}\texttt{CurrentSignal}$ in $\mathsf{Post}(StartFireAlert2(x))$. This action is "wrong" in the sense that its execution leads to an inconsistent action output: if that is the case, GenIt will find this inconsistency as it is unable to generate a set of information terms for an axiom.

For example, consider the set Out' obtained applying the above action to the state $\overline{\gamma}_1$ of Example 3 (namely, $Out' = Out \setminus \{\texttt{fire\_s1} : \texttt{CurrentSignal}\}$). The inconsistency of Out' is verified with the execution of $\textsc{GenIt}(Out', Ax_1)$: this execution leads to the following recursive executions of OpenIt on the subformulas of Ax_1

$\textsc{OpenIt}(Out', x' : \texttt{CurrentSignal}) = \emptyset$
$\textsc{OpenIt}(Out', x : \exists\texttt{hasReason.CurrentSignal}) = \emptyset$
$\textsc{OpenIt}(Out', x : \exists\texttt{hasReason.CurrentSignal} \sqcap \texttt{Alert}) = \emptyset$
$\textsc{OpenIt}(Out', x : \neg\texttt{CurrentAlert} \sqcup (\exists\texttt{hasReason.CurrentSignal} \sqcap \texttt{Alert})) = H$

with $H = \{((1, \texttt{tt}), a) \mid a \in \mathsf{W} \setminus \{\texttt{alert1}\}\}$. Since no information term in H can be associated to $\texttt{alert1}$, $\textsc{GenIt}(Out', Ax_1) = \emptyset$. This means that the corresponding theory is not state consistent. We also remark that the execution of $\textsc{GenIt}(Out', x' : \texttt{CurrentSignal})$ traces the reason of state inconsistency. $\diamond$

We conclude this section noting that GenIt is exponential in the size of $\mathcal{N}$: this complexity is due to the case of $F = \forall A$, where the algorithm must generate all the possible functions ϕ such that $\phi : \mathcal{N} \to \bigcup_{d \in \mathcal{N}} \mathrm{IT}_{\mathcal{N}}(d : A)$. The number of such functions is obviously exponential in the number of elements of $\mathcal{N}$, hence the complexity of GenIt.

6 Conclusion and future works

In this paper we have presented an action formalism based on the information terms semantics. We have shown how our semantics supports a natural notion of state and how an action language can be defined on the top of this notion. The problem to determine the consistency of an action is reduced to the problem to study the information terms generated by the application of GenIt. We have shown, by means of an example, how GenIt can be used, in some cases, to debug inconsistent actions. For lack of space we have not treated in details the problem to study the consistency of an action when the output of GenIt is not empty. However, we remark that this problem is similar to the problem to check *snapshot* consistency in CooML [7]. As for the future works we plan to investigate this question also considering its relations with model generation in SAT [10] and ASP [11] and with the planning problem. Moreover, we plan to study the projection problem (see, e.g., [2]), that needs to be restated in our setting. We are also working on an implementation of our action language.

References

1. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.
2. F. Baader, M. Milicic, C. Lutz, U. Sattler, and F. Wolter. Integrating description logics and action formalisms: First results. In I. Horrocks, U. Sattler, and F. Wolter, editors, *Proceedings of the 2005 International Workshop on Description Logics (DL2005)*, volume 147 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2005.
3. L. Bozzato, M. Ferrari, C. Fiorentini, and G. Fiorino. A constructive semantics for $\mathcal{ALC}$. In Calvanese et al. [4], pages 219–226.
4. D. Calvanese, E. Franconi, V. Haarslev, D. Lembo, B. Motik, S. Tessaris, and A. Turhan, editors. *Proceedings of the 20th International Workshop on Description Logics (DL2007)*, volume 250 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.
5. D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati. Actions and programs over description logic ontologies. In Calvanese et al. [4], pages 29–40.
6. C. Drescher and M. Thielscher. Integrating action calculi and description logics. In J. Hertzberg, M. Beetz, and R. Englert, editors, *KI*, volume 4667 of *Lecture Notes in Computer Science*, pages 68–83. Springer-Verlag, 2007.
7. M. Ferrari, C. Fiorentini, A. Momigliano, and M. Ornaghi. Snapshot generation in a constructive object-oriented modeling language. In A. King, editor, *Logic Based Program Synthesis and Transformation, LOPSTR 2007, Selected Papers*, volume 4915 of *Lecture Notes in Computer Science*, pages 169–184. Springer-Verlag, 2008.
8. C. Fiorentini and M. Ornaghi. Answer set semantics vs. information term semantics. In *ASP2007: Answer Set Programming, Advances in Theory and Implementation.* `http://cooml.dsi.unimi.it/papers/asp.pdf`, 2007.
9. K. Kaneiwa. Negations in description logic - contraries, contradictories, and subcontraries. In *Proceedings of the 13th International Conference on Conceptual Structures (ICCS '05)*, pages 66–79. Kassel University Press, 2005.
10. F. Lin and Y. Zhao. ASSAT: computing answer sets of a logic program by SAT solvers. *Artif. Intell.*, 157(1-2):115–137, 2004.
11. I. Niemelä and P. Simons. Smodels - an implementation of the stable model and well-founded semantics for normal LP. In *LPNMR*, pages 421–430, 1997.
12. M. Ornaghi, M. Benini, M. Ferrari, C. Fiorentini, and A. Momigliano. A Constructive Modeling Language for Object Oriented Information Systems. In *Constructive Logic for Automated Software Engineering*, volume 153 of *Electronic Notes in Theoretical Computer Science*, pages 55–75, 2006.
13. M. Schmidt-Schauß and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.
14. A. S. Troelstra. From constructivism to computer science. *TCS*, 211(1-2):233–252, 1999.