Extend Atomic Action Definitions of DDL to Support Occlusions and Conditional Post-conditions

Liang Chang, Zhongzhi Shi, Tianlong Gu

Extend Atomic Action Definitions of DDL to Support Occlusions and Conditional Post-conditions

Liang Chang[1], Zhongzhi Shi[2], and Tianlong Gu[1]

[1] School of Computer and Control, Guilin University of Electronic Technology, Guilin, 541004, China
[2] The Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100080, China
{changl,cctlgu}@guet.edu.cn,shizz@ics.ict.ac.cn

Abstract. The dynamic description logic *DDL* provides a kind of action theories based on description logics (DLs). Compared with another important DL-based action formalism constructed by Baader et.al., a shortcoming of *DDL* is the absence of occlusions and conditional post-conditions in the description of atomic actions. In this paper, we extend atomic action definitions of *DDL* to overcome this limitation. Firstly, we introduce extended atomic action definitions in which the occlusions and conditional post-conditions are incorporated. Secondly, for each atomic action specified by an extended atomic action definition, a function named *Expand* is introduced to transform it into a choice action which is composed of atomic actions defined by ordinary atomic action definitions. Finally, based on the *Expand* function, the satisfiability-checking algorithm of *DDL* is extended to support occlusions and conditional post-conditions.

1 Introduction

Description logics (DLs) are well-known for representing and reasoning about knowledge of static application domains. The main strength of description logics is that they offer considerable expressive power going far beyond propositional logic, while reasoning is still decidable.

The study of integrating description logics with action formalisms is driven by two factors. One is the demand to represent and reason about semantic web services [7]. Another factor is the fact that there is an expressive gap between existing action formalisms: they are either based on first- or higher-order logics and do not admit decidable reasoning, like the Situation Calculus [9] and the Fluent Calculus [11], or are decidable but only propositional, like those based on propositional dynamic logics [5] or propositional temporal logics [2].

One approach to integrate description logics with action formalisms was proposed by Baader et.al.[1]. That approach is characterized by constructing action formalisms over description logics of the *ALCQIO* family. In that formalism,

acyclic TBoxes and ABox assertions of description logics are used to specify the domain constraints and the states of the world respectively. Each atomic action is described as a triple $(pre, occ, post)$, where the set pre is composed of ABox assertions for specifying the pre-conditions under which the action is applicable; the set $post$ is composed of conditional post-conditions of the form φ/ψ with φ an ABox assertion and ψ a primitive literal, each conditional postcondition φ/ψ says that if φ is true before executing the action, then ψ should be true after the executions; the set occ is composed of occlusions for indicating these primitive literals that can change arbitrarily as while as the action is executed, where each occlusion is of the form $A_i(p)$ or $r(p, q)$, with A_i a primitive concept name and r a role name. The semantics of atomic actions is defined according to the minimal-change semantics; each atomic action is defined as a transition relation on DL-interpretations. Taking each finite sequence of atomic actions as a composite action, Baader et.al. investigated the executability problem and the projection problem of actions, and demonstrated that both of them can be reduced to standard inference problems of description logics and therefore were remained decidable.

A limitation of Baader et.al.'s formalism is that atomic actions can only be organized as finite sequences. Many complex control structures required by Web services [8], such as the "Choice", "Any-Order", "Iterate", "If-Then-Else", "Repeat-While" and "Repeat-Until" structures specified in the OWL-based Web service ontology OWL-S [6], are not supported by it.

Another typical approach to integrate description logics with action formalisms was proposed by Shi et.al.[10]. That approach is characterized by constructing a kind of dynamic description logics named DDL, which is in fact a combination of description logics, propositional dynamic logics and action formalisms. In that approach, domain knowledge of each action theory is captured by acyclic TBoxes of description logics; based on these knowledge, both the states of the world and the pre- and post-conditions of each atomic action are described by ABox assertions. Starting from atomic actions and ABox assertions, complex actions are constructed with the help of regular program constructors of propositional dynamic logics, so that not only the sequence structure, but also the "Choice", "Any-Order", "Iterate", "If-Then-Else", "Repeat-While" and "Repeat-Until" structures required by Web services are all supported by the formalism. Finally, both atomic actions and complex actions are used as modal operators to construct formulas, so that properties on actions can be stated explicitly by formulas. Chang et.al. [4] provided a terminable, sound and complete algorithm for checking the satisfiability of DDL-formulas; based on that algorithm, reasoning tasks on the realizability of actions, the executability of actions and the consequence of executing actions can all be effectively carried out [3].

Compared with Baader et.al.'s formalism, a merits of DDL is the capability of representing complex actions. However, a shortcoming of DDL is that occlusions and conditional post-conditions are not supported in the description of atomic actions. In this paper, we extend atomic action definitions of DDL to include occlusions and conditional post-conditions.

The rest of this paper is organized as follows. A brief introduction of DDL is presented in Section 2. The atomic action definitions of DDL is extended to include occlusions and conditional post-conditions in Section 3. Section 4 provides a satisfiability-checking algorithm for DDL-formulas in the case that occlusions and conditional post-conditions are embraced in the description of atomic actions. Section 5 concludes the paper.

2 The Dynamic Description Logic *DDL*

As a kind of dynamic description logics, DDL is constructed by embracing an action theory into description logics. Be corresponding to the family of description logics, DDL is embodied as different logic systems. In this section, we take the description logic $ALCQIO$ as an example and introduce the dynamic description logic constructed over it.

Primitive symbols of the logic $DDL(ALCQIO)$ are a set N_I of individual names, a set N_R of role names, a set N_C of concept names, and a set N_A of action names. Basic citizens of $DDL(ALCQIO)$ are roles, concepts, actions and formulas; all of them are defined inductively by constructors starting from primitive symbols.

Roles of $DDL(ALCQIO)$ are formed according to the following syntax rule:

$$R \ ::= \ R_i \ \mid \ R^{-}$$

where $R_i \in N_R$.

Concepts of $DDL(ALCQIO)$ are constructed according to the following syntax rule:

$$\begin{aligned} C, C' \ ::= \ & A_i \mid \neg C \mid C \sqcup C' \mid C \sqcap C' \\ & \mid \forall R.C \mid \exists R.C \mid \leq nR.C \mid \geq nR.C \mid \{p\} \end{aligned}$$

where $A_i \in N_C$, $p \in N_I$, and R is role.

A *concept definition* is of the form $A \equiv C$, where A is a concept name and C is a concept. A *TBox* of $DDL(ALCQIO)$ is a finite set of concept definitions with unique left-hand sides. A TBox is said to be *acyclic* if there are no cyclic dependencies between the definitions.

With respect to an acyclic TBox $\mathcal{T}$, a concept name $A_i \in N_C$ is called *defined* if and only if it occurs on the left-hand side of some concept definition contained in $\mathcal{T}$, and is called *primitive* otherwise.

Formulas of $DDL(ALCQIO)$ are formed according to the following syntax rule:

$$\varphi, \varphi' \ ::= \ C(p) \mid R(p,q) \mid < \pi > \varphi \mid [\pi]\varphi \mid \neg\varphi \mid \varphi \vee \varphi' \mid \varphi \wedge \varphi'$$

where $p, q \in N_I$, R is a role, C is a concept and π is an action.

An *ABox assertion* is of the form $C(p)$, $R(p,q)$ or $\neg R(p,q)$, where $p, q \in N_I$, C is a concept, and R is a role. A finite set of ABox assertions is called an *ABox* of $DDL(ALCQIO)$.

With respect to an acyclic TBox $\mathcal{T}$, an ABox assertion ψ is called a *primitive literal* if and only if it is of the form $A_i(p)$, $(\neg A_i)(p)$, $R(p,q)$ or $\neg R(p,q)$, with A_i a primitive concept name, R a role and $p, q \in N_I$.

Actions of $DDL(ALCQIO)$ are formed according to the following syntax rule:

$$\pi, \pi' \ ::= \ \alpha \mid \varphi? \mid \pi \cup \pi' \mid \pi; \pi' \mid \pi^*$$

where $\alpha \in N_A$, and φ is an ABox assertion.

With respect to an acyclic TBox $\mathcal{T}$, an *atomic action definition* of $DDL(ALCQIO)$ is of the form $\alpha \equiv (P, E)$, where

- $\alpha \in N_A$,
- P is a finite set of ABox assertions for describing the pre-conditions, and
- E is a finite set of primitive literals for describing the post-conditions.

An *ActBox* of $DDL(ALCQIO)$ is a finite set of atomic action definitions with unique left-hand sides.

An atomic action α is said to *be defined in an ActBox* $\mathcal{A}_\mathcal{C}$ if and only if α occurs on the left-hand side of some atomic action definition contained in $\mathcal{A}_\mathcal{C}$. A formula φ is said to *be defined w.r.t. an ActBox* $\mathcal{A}_\mathcal{C}$ if and only if all the atomic actions occurring in φ are defined in $\mathcal{A}_\mathcal{C}$.

A *knowledge base* of $DDL(ALCQIO)$ is of the form $K = (\mathcal{T}, \mathcal{A}_\mathcal{C}, \mathcal{A})$, where $\mathcal{T}$, $\mathcal{A}_\mathcal{C}$ and $\mathcal{A}$ are respectively a TBox, an ActBox and an ABox.

The semantic model of $DDL(ALCQIO)$ is of the form $M = (W, T, \Delta, I)$, where,

- W is a non-empty set of states;
- $T : N_A \to 2^{W \times W}$ is a function which maps action names into binary relations on W;
- Δ is a non-empty set of individuals; and
- I is a function which associates with each state $w \in W$ a DL-interpretation $I(w) =< \Delta, \cdot^{I(w)} >$, where the function $\cdot^{I(w)}$
  - maps each concept name $A_i \in N_C$ to a set $A_i^{I(w)} \subseteq \Delta$,
  - maps each role name $R_i \in N_R$ to a binary relation $R_i^{I(w)} \subseteq \Delta \times \Delta$, and
  - maps each individual name $p \in N_I$ to an element $p^{I(w)} \in \Delta$, with the constraints that $p^{I(w)} = p^{I(w')}$ for any state $w' \in W$, and $p^{I(w)} \neq q^{I(w)}$ for any individual name q which is different from p. Since interpretations of p are the same in every state, the interpretation $p^{I(w)}$ is also represented as p^I.

Given a model $M = (W, T, \Delta, I)$, the semantics of concepts, formulas and actions of $DDL(ALCQIO)$ are defined inductively as follows.

Firstly, with respect to any state $w \in W$, each role R will be interpreted as a binary relation $R^{I(w)} \subseteq \Delta \times \Delta$, and each concept C will be interpreted as a set $C^{I(w)} \subseteq \Delta$; the definition is as follows:

1. $(R^-)^{I(w)} = \{(y, x) \mid (x, y) \in R^{I(w)}\}$;

2. $(\neg C)^{I(w)} = \Delta \backslash C^{I(w)}$;
3. $(C \sqcup D)^{I(w)} = C^{I(w)} \cup D^{I(w)}$;
4. $(C \sqcap D)^{I(w)} = C^{I(w)} \cap D^{I(w)}$;
5. $(\forall R.C)^{I(w)} = \{\ x \in \Delta \mid \text{for all } y \in \Delta\text{: if } (x,y) \in R^{I(w)}, \text{ then } y \in C^{I(w)}\}$;
6. $(\exists R.C)^{I(w)} = \{\ x \in \Delta \mid \text{there is a } y \in \Delta \text{ with } (x,y) \in R^{I(w)} \text{ and } y \in C^{I(w)}\}$;
7. $(\leq nS.C)^{I(w)} = \{x \in \Delta \mid \sharp\{y \in \Delta \mid (x,y) \in S^{I(w)} \text{ and } y \in C^{I(w)}\} \leq n\ \}$;
8. $(\geq nS.C)^{I(w)} = \{x \in \Delta \mid \sharp\{y \in \Delta \mid (x,y) \in S^{I(w)} \text{ and } y \in C^{I(w)}\} \geq n\ \}$;

Secondly, for any formula φ and any state $w \in W$, the truth-relation $(M,w) \models \varphi$ is defined inductively as follows:

9. $(M,w) \models C(p)$ iff $p^I \in C^{I(w)}$;
10. $(M,w) \models R(p,q)$ iff $(p^I, q^I) \in R^{I(w)}$;
11. $(M,w) \models < \pi > \varphi$ iff there is a state $w' \in W$ with $(w,w') \in T(\pi)$ and $(M,w') \models \varphi$;
12. $(M,w) \models [\pi]\varphi$ iff for every $w' \in W$: if $(w,w') \in T(\pi)$ then $(M,w') \models \varphi$;
13. $(M,w) \models \neg\varphi$ iff it is not the case that $(M,w) \models \varphi$;
14. $(M,w) \models \varphi \vee \psi$ iff $(M,w) \models \varphi$ or $(M,w) \models \psi$;
15. $(M,w) \models \varphi \wedge \psi$ iff $(M,w) \models \varphi$ and $(M,w) \models \psi$;

Finally, each action π is interpreted as a binary relation $T(\pi) \subseteq W \times W$ according to the following definitions:

16. $T(\varphi?) = \{\ (w,w) \mid w \in W \text{ and } (M,w) \models \varphi\ \}$;
17. $T(\pi \cup \pi') = T(\pi) \cup T(\pi')$;
18. $T(\pi;\pi') = \{\ (w,w') \mid \text{there is a state } u \in W \text{ with } (w,u) \in T(\pi) \text{ and } (u,w') \in T(\pi')\ \}$;
19. $T(\pi^*) =$ reflexive and transitive closure of $T(\pi)$.

Let $K = (\mathcal{T}, \mathcal{A}_\mathcal{C}, \mathcal{A})$ be a knowledge base and let $M = (W, T, \Delta, I)$ be a semantic model, then:

- a state w of the model M *satisfies the ABox* $\mathcal{A}$, denoted by $(M,w) \models \mathcal{A}$, if and only if $(M,w) \models \varphi_i$ for every ABox assertion $\varphi_i \in \mathcal{A}$;
- M is a *model of the TBox* $\mathcal{T}$, denoted by $M \models \mathcal{T}$, if and only if $A^{I(w)} = C^{I(w)}$ for every state $w \in W$ and every concept definition $A \equiv C \in \mathcal{T}$; and
- M is *a model of the ActBox* $\mathcal{A}_\mathcal{C}$, denoted by $M \models \mathcal{A}_\mathcal{C}$, if and only if the following equation holds for every atomic action definition $\alpha \equiv (P,E) \in \mathcal{A}_\mathcal{C}$:

$$T(\alpha) = \{\ (w,w') \mid w \in W, w' \in W, (M,w) \models P, \\ C^{I(w')} = (C^{I(w)} \cup \{p^I \mid C(p) \in E\}) \setminus \{p^I \mid (\neg A)(p) \in E\}\ for \\ each\ concept\ name\ A\ which\ is\ primitive\ w.r.t.\ \mathcal{T},\ and \\ R^{I(w')} = (R^{I(w)} \cup \{(p^I,q^I) \mid R(p,q) \in E\}) \setminus \{(p^I,q^I) \mid \neg R(p,q) \in E\} \\ for\ each\ role\ name\ R.\}.$$

A primary inference problem for $DDL(ALCQIO)$ is to decide the satisfiability of formulas. A formula φ is called *satisfiable* w.r.t. a TBox $\mathcal{T}$ and an ActBox $\mathcal{A}_\mathcal{C}$ if and only if there exists a model $M = (W, T, \Delta, I)$ and a state $w \in W$ such that $M \models \mathcal{T}$, $M \models \mathcal{A}_\mathcal{C}$ and $(M,w) \models \varphi$.

In the literature, a terminable, sound and complete algorithm for deciding the satisfiability of $DDL(ALCQIO)$-formulas is presented [4].

3 Extended atomic action definitions

In this section, we extend atomic action definitions of DDL to include occlusions and conditional post-conditions. To be distinguished from original atomic action definitions discussed in DDL, we refer to these extended definitions as extended atomic action definitions.

With respect to an acyclic TBox $\mathcal{T}$, an *extended atomic action definition* is of the form $\alpha \equiv (P, O, E)$, where

- $\alpha \in N_A$;
- P is a finite set of ABox assertions for describing the pre-conditions;
- O is a finite set of occlusions of the form $A(p)$ or $r(p, q)$, with A primitive concept name, r role name, and $p, q \in N_I$; and
- E is a finite set of conditional post-conditions of the form φ/ψ, where φ is an ABox assertion and ψ is a primitive literal.

Intuition of the above definition is as follows. The pre-conditions specify under which conditions the action is applicable. Each conditional postcondition φ/ψ says that, if φ is true before executing the action, then ψ should be true after the execution. The occlusions indicate those primitive literals that can change arbitrarily as while as the action is executed.

The semantics of extended atomic action definitions is defined as follows: a semantic model $M = (W, T, \Delta, I)$ *satisfies an extended atomic action definition* $\alpha \equiv (P, O, E)$, in symbols $M \models \alpha \equiv (P, O, E)$, if and only if

$$T(\alpha) = \{\, (w, w') \in W \times W \mid (M, w) \models P, \textit{ both } A_w^+ \cap A_w^- = \emptyset \textit{ and } A^{I(w')} \cap I_A^w = ((A^{I(w)} \cup A_w^+) \setminus A_w^-) \cap I_A^w \textit{ for each concept name } A \textit{ which is primitive w.r.t. } \mathcal{T}, \textit{ and both } R_w^+ \cap R_w^- = \emptyset \textit{ and } R^{I(w')} \cap I_R^w = ((R^{I(w)} \cup R_w^+) \setminus R_w^-) \cap I_R^w \textit{ for each role name } R. \,\},$$

where A_w^+, A_w^-, I_A^w, R_w^+, R_w^- and I_R^w denote some sets constructed as follows:

- $A_w^+ := \{\, p^I \mid \varphi/A(p) \in E \text{ and } (M, w) \models \varphi \,\}$,
- $A_w^- := \{\, p^I \mid \varphi/(\neg A)(p) \in E \text{ and } (M, w) \models \varphi \,\}$,
- $I_A^w := (\, \Delta \setminus \{\, p^I \mid A(p) \in O \,\}\,) \cup A_w^+ \cup A_w^-$,
- $R_w^+ := \{\, (p^I, q^I) \mid \text{there is some role } S \text{ with } S \sqsubseteq_{\mathcal{R}}^* R,\ \varphi/S(p, q) \in E \text{ and } (M, w) \models \varphi \,\}$,
- $R_w^- := \{\, (p^I, q^I) \mid \text{there is some role } S \text{ with } R \sqsubseteq_{\mathcal{R}}^* S,\ \varphi/\neg S(p, q) \in E \text{ and } (M, w) \models \varphi \,\}$,
- $I_R^w := (\, (\Delta \times \Delta) \setminus \{\, (p^I, q^I) \mid R(p, q) \in O \,\}\,) \cup R_w^+ \cup R_w^-$.

Be similar with the semantics of atomic action definitions, this definition is also based on the minimal-change semantics [12]. For any pair $(w, w') \in T(\alpha)$, any primitive concept name A and any role name R, it must be $A_w^+ \subseteq A^{I(w')}$, $A_w^- \cap A^{I(w')} = \emptyset$, and nothing else changes from $A^{I(w)}$ to $A^{I(w')}$ with the possible exception of the occluded literals. Similarly, the interpretations $R^{I(w)}$ and $R^{I(w')}$

should satisfy that $R_w^+ \subseteq R^{I(w')}$, $R_w^- \cap R^{I(w')} = \emptyset$, and nothing else changes from $R^{I(w)}$ to $R^{I(w')}$ with the possible exception of the occluded literals. Those parts that might change arbitrarily by the presence of occluded literals are captured by the set I_A^w and the set I_R^w.

For example, consider a Web service system in which customers are able to buy books online, a Web service for the customer *Tom* to buy the book *KingLear* can be described according to the following extended atomic action definition:

$$
\begin{aligned}
& BuyBook_{Tom,Kin} \\
\equiv\ & (\ \{\ customer(Tom), book(KingLear)\ \},\quad \{\ \}, \\
& \quad \{\ instore(KingLear)/bought(Tom, KingLear), \\
& \quad\ instore(KingLear)/\neg instore(KingLear), \\
& \quad\ instore(KingLear)/notify(Tom, NotifySucceed), \\
& \quad\ \neg instore(KingLear)/notify(Tom, NotifyBookOutOfStock)\ \}\ )
\end{aligned}
$$

According to this definition, if the book *KingLear* is in store, then the formula $bought(Tom, KingLear)$, $\neg instore(KingLear)$ and $notify(Tom, NotifySucceed)$ will become true after the execution of the service, otherwise *Tom* will be notified by the notification *NotifyBookOutOfStock*.

4 Reasoning mechanisms for extended atomic action definitions

In order to provide reasoning services for extended atomic action definitions, for any atomic action α defined by some extended atomic action definition $\alpha \equiv (P, O, E)$, we introduce a procedure $Expand(\alpha)$ to transform it into some action of the form $\alpha_1 \cup ... \cup \alpha_n$, where each α_i $(1 \leq i \leq n)$ is an atomic action defined by an atomic action definition.

More precisely, for the definition $\alpha \equiv (P, O, E)$, let $O = \{\ \phi_1, ..., \phi_m\ \}$ and let $E = \{\ \varphi_1/\psi_1, ..., \varphi_k/\psi_k\ \}$, then the procedure $Expand(\alpha)$ operates according to the following steps:

1. Construct an empty set $\mathcal{A}_\mathcal{C}$ on atomic action definitions.
2. According to the set P and these k conditional post-conditions contained in E, construct 2^k atomic action definitions as follows:

$$
\begin{aligned}
\alpha_0 &\equiv (P \cup \{\neg\varphi_k, ..., \neg\varphi_3, \neg\varphi_2, \neg\varphi_1\}, \{\ \}) \\
\alpha_1 &\equiv (P \cup \{\neg\varphi_k, ..., \neg\varphi_3, \neg\varphi_2, \varphi_1\}, \{\psi_1\}) \\
\alpha_2 &\equiv (P \cup \{\neg\varphi_k, ..., \neg\varphi_3, \varphi_2, \neg\varphi_1\}, \{\psi_2\}) \\
\alpha_3 &\equiv (P \cup \{\neg\varphi_k, ..., \neg\varphi_3, \varphi_2, \varphi_1\}, \{\psi_2, \psi_1\}) \\
\alpha_4 &\equiv (P \cup \{\neg\varphi_k, ..., \varphi_3, \neg\varphi_2, \neg\varphi_1\}, \{\psi_3\}) \\
&... \\
\alpha_{2^k-1} &\equiv (P \cup \{\varphi_k, ..., \varphi_3, \varphi_2, \varphi_1\}, \{\psi_k, ..., \psi_3, \psi_2, \psi_1\})
\end{aligned}
$$

I.e., start from the set P of pre-conditions and an empty set of post-conditions, be corresponding to each conditional post-condition φ_i/ψ_i $(1 \leq i \leq k)$, add either φ_i or $\varphi_i^{\neg}$ into the pre-condition set, and add ψ_i into the postcondition set as while as φ_i is added into the pre-condition set.

3. For each atomic action definition $\alpha_i \equiv (P_i, E_i)$ $(0 \leq i \leq 2^k - 1)$ constructed above, if it is consistent w.r.t. $\mathcal{R}$ and $\mathcal{T}$, then do the following operations sequentially:
   (a) Construct an empty set O_i on primitive literals.
   (b) For each occlusion ϕ_j $(1 \leq j \leq m)$, if both $\phi_j \notin E_i{}^*_{\mathcal{R}}$ and $\phi_j^{\neg} \notin E_i{}^*_{\mathcal{R}}$, then add ϕ_j into the set O_i.
   (c) Let $\phi_{i,1}$,...,ϕ_{i,m_i} be all the primitive literals contained in the set O_i, construct 2^{m_i} atomic action definitions as follows:

$$\begin{aligned}
\alpha_{i,0} &\equiv (P_i, E_i \cup \{\phi_{i,m_i}^{\neg}, ..., \phi_{i,3}^{\neg}, \phi_{i,2}^{\neg}, \phi_{i,1}^{\neg}\}) \\
\alpha_{i,1} &\equiv (P_i, E_i \cup \{\phi_{i,m_i}^{\neg}, ..., \phi_{i,3}^{\neg}, \phi_{i,2}^{\neg}, \phi_{i,1}\}) \\
\alpha_{i,2} &\equiv (P_i, E_i \cup \{\phi_{i,m_i}^{\neg}, ..., \phi_{i,3}^{\neg}, \phi_{i,2}, \phi_{i,1}^{\neg}\}) \\
\alpha_{i,3} &\equiv (P_i, E_i \cup \{\phi_{i,m_i}^{\neg}, ..., \phi_{i,3}^{\neg}, \phi_{i,2}, \phi_{i,1}\}) \\
\alpha_{i,4} &\equiv (P_i, E_i \cup \{\phi_{i,m_i}^{\neg}, ..., \phi_{i,3}, \phi_{i,2}^{\neg}, \phi_{i,1}^{\neg}\}) \\
&... \\
\alpha_{i,2^{m_i}-1} &\equiv (P_i, E_i \cup \{\phi_{i,m_i}, ..., \phi_{i,3}, \phi_{i,2}, \phi_{i,1}\})
\end{aligned}$$

   I.e., start from the set P_i of pre-conditions and the set E_i of post-conditions, be corresponding to each primitive literal $\phi_{i,j}$ $(1 \leq j \leq m_i)$, add either $\phi_{i,j}$ or $\phi_{i,j}^{\neg}$ into the post-condition set.
   (d) For each atomic action definition constructed above, if it is consistent w.r.t. $\mathcal{R}$ and $\mathcal{T}$, then add it into the set $\mathcal{A}_{\mathcal{C}}$.
4. If the set $\mathcal{A}_{\mathcal{C}}$ is empty, then construct an atomic action definition $\beta_0 \equiv (\{false\}, \emptyset)$ and return the action β_0, else let $\beta_1 \equiv (P_{\beta_1}, E_{\beta_1})$, ..., $\beta_n \equiv (P_{\beta_1}, E_{\beta_1})$ be all the atomic action definitions contained in $\mathcal{A}_{\mathcal{C}}$, construct a choice action $\beta_1 \cup ... \cup \beta_n$ and return it.

As an example, taken the atomic action $BuyBook_{Tom,Kin}$ defined in previous example as an input, the procedure $Expand(BuyBook_{Tom,Kin})$ will return a choice action as follows:

$$BuyBookNotified_1 \cup BuyBookNotified_2$$

where $BuyBookNotified_1$ and $BuyBookNotified_2$ are defined by the following atomic action definitions respectively:

$$\begin{aligned}
&BuyBookNotified_1 \\
\equiv\ &(\ \{\ customer(Tom), book(KingLear), instore(KingLear)\ \}, \\
&\quad \{\ bought(Tom, KingLear), \neg instore(KingLear), \\
&\qquad notify(Tom, NotifyOrderSucceed)\ \}\ ) \\
&BuyBookNotified_2 \\
\equiv\ &(\ \{\ customer(Tom), book(KingLear), \neg instore(KingLear)\ \}, \\
&\quad \{\ notify(Tom, NotifyBookOutOfStock)\ \}\ )
\end{aligned}$$

The procedure *Expand*() is technically designed to guarantee the following property.

Theorem 1. *Let α be an atomic action defined by some extended atomic action definition $\alpha \equiv (P, O, E)$ w.r.t. an acyclic TBox $\mathcal{T}$, let $\alpha_1 \cup ... \cup \alpha_n$ be the action returned by the procedure Expand(α), and let $\mathcal{A}_\mathcal{C}$ be an ActBox composed of atomic action definitions of every α_i ($1 \leq i \leq n$). Then, for any model $M = (W, T, \Delta, I)$ with $M \models \mathcal{T}$, $M \models \alpha \equiv (P, O, E)$ and $M \models \mathcal{A}_\mathcal{C}$, it must be $T(\alpha_1 \cup ... \cup \alpha_n) = T(\alpha)$.*

Now we are ready to present a satisfiability-checking algorithm for formulas that might contain atomic actions defined by extended atomic action definitions.

Algorithm 1 *Let $\mathcal{A}_\mathcal{C}$ be an ActBox which might contains some extended atomic action definitions, let φ be a formula defined w.r.t. $\mathcal{A}_\mathcal{C}$. Then, the satisfiability of φ w.r.t. a TBox $\mathcal{T}$ and ActBox $\mathcal{A}_\mathcal{C}$ is decided according to the following steps.*

1. *Construct a formula φ' and an ActBox $\mathcal{A}_\mathcal{C}{}'$ according to the following steps:*
   (a) *set $\mathcal{A}_\mathcal{C}{}' := \emptyset$ and $\varphi' := \varphi$;*
   (b) *for each atomic action occurring in φ', if it is defined in $\mathcal{A}_\mathcal{C}$ by some atomic action definition $\alpha \equiv (P, E)$, then add $\alpha \equiv (P, E)$ into the set $\mathcal{A}_\mathcal{C}{}'$;*
   (c) *for each atomic action occurring in φ', if it is defined in $\mathcal{A}_\mathcal{C}$ by some extended atomic action definition $\alpha \equiv (P, O, E)$, then do the following operations sequentially:*
      i. *call the procedure Expand(α) and let $\alpha_1 \cup ... \cup \alpha_n$ be the action returned by it;*
      ii. *add all the atomic action definitions of α_1, ..., α_n into the set $\mathcal{A}_\mathcal{C}{}'$; and*
      iii. *for each occurrence of the action α in the formula φ', replace it with the action $\alpha_1 \cup ... \cup \alpha_n$.*
2. *Since every atomic action occurring in φ' is defined by original atomic action definitions discussed in DDL, we can call the procedure provided by DDL to decide whether the formula φ' is satisfiable w.r.t. $\mathcal{T}$ and $\mathcal{A}_\mathcal{C}{}'$; if φ' is satisfiable w.r.t. $\mathcal{T}$ and $\mathcal{A}_\mathcal{C}{}'$, then return "TRUE", otherwise return "FALSE".*

For the formula φ' constructed in this algorithm, according to Theorem 1, it is straightforward that φ' is satisfiable w.r.t. $\mathcal{T}$ and $\mathcal{A}_\mathcal{C}{}'$ if and only if φ is satisfiable w.r.t. $\mathcal{T}$ and $\mathcal{A}_\mathcal{C}$. Therefore, this deciding algorithm is correct; i.e.,

Theorem 2. *Algorithm 1 returns "TRUE" if and only if the formula φ is satisfiable w.r.t. $\mathcal{T}$ and $\mathcal{A}_\mathcal{C}$.*

5 Conclusion

In this paper, the dynamic description logic DDL is extended to support occlusions and conditional post-conditions in the description of atomic actions. As a result, the action theory supported by DDL is compatible with the action formalism constructed by Baader et.al. [1].

DDL provides an approach to bring the power and character of description logics into the description and reasoning of dynamic application domains. One of our future work is to optimize the reasoning mechanisms of DDL. Another work is to apply DDL to model and reason about semantic Web services.

Acknowledgments. This work was partially supported by the National Natural Science Foundation of China under Grant Nos. 60903079, 60775035 and 60963010.

References

1. Baader, F., Lutz, C., Milicic, M., Sattler, U., Wolter, F.: Integrating Description Logics and Action Formalisms: First Results. In: Veloso, M., Kambhampati, S.(eds.) Proceedings of the 12th Nat. Conf. on Artif. Intell., pp. 572-577. AAAI Press (2005)
2. Calvanese, D., De Giacomo, G., Vardi, M.: Reasoning about Actions and Planning in LTL Action Theories. In: Fensel, D., Giunchiglia, F., McGuinness, D., Williams, M.(eds.) 8th Int. Conf. on Principles and Knowledge Representation and Reasoning, pp. 593-602. Morgan Kaufmann (2002)
3. Chang, L., Lin, F., Shi, Z.: A Dynamic Description Logic for Representation and Reasoning about Actions. In: 2nd International Conference on Knowledge Science, Engineering and Management, LNCS vol.4798, pp. 115-127. Springer (2007)
4. Chang, L., Shi, Z., Qiu L., Lin, F.: A Tableau Decision Algorithm for Dynamic Description Logic. Chinese Journal of Computers, 31(6), 896-909 (2008)
5. De Giacomo, G., Lenzerini, M.: PDL-based Framework for Reasoning about Actions. In: Gori, M., Soda, G.(eds.) Topics in Artif. Intell., LNCS, vol.992, pp. 103-114. Springer (1995)
6. Martin, D., Burstein, M., McDermott, D., McIlraith, S., Paolucci, M., Sycara, K., McGuinness, D., Sirin, E., Srinivasan, N.: Bringing semantics to web services with OWL-S. *World Wide Web Journal*, 10(3), 243-277 (2007)
7. McIlraith, S., Son, T., Zeng, H.: Semantic Web Services. IEEE Intelligent Systems. 16(2), 46-53 (2001)
8. Narayanan, S., McIlraith, S.: Simulation, verification and automated composition of web services. In: *Proc. of the 11th Int. World Wide Web Conference (WWW'02)*, pp. 77-88 (2002)
9. Reiter, R.: Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems. MIT Press (2001)
10. Shi, Z., Dong, M., Jiang, Y., Zhang, H.: A Logic Foundation for the Semantic Web. Science in China, Series F. 48(2), 161-178 (2005)
11. Thielscher, M.: Introduction to the Fluent Calculus. Electron. Trans. Artif. Intell. 2(3-4), 179-192 (1998)
12. Winslett, M.: Reasoning about Action Using a Possible Models Approach. In: 7th Nat. Conf. on Artif. Intell. pp. 89-93. AAAI Press (1988)