Dynamic Description Logic based on DL-Lite

Na Zhang, Liang Chang, Zhoubo Xu, Tianlong Gu

Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China

zhnamengluo@163.com,{changl,xzbli_11,cctlgu}@guet.edu.cn

Abstract. Description logics offer considerable expressive power for describing knowledge about static application domains while reasoning is still decidable. The dynamic description logic DDL is a family of dynamic extensions of description logics for representing and reasoning about knowledge of dynamic application domains. In order to provide effective reasoning mechanisms, systems of DDL investigated in the literatures assume that there is no general concept inclusion(GCI) contained in the knowledge base. In this paper, we build a system of dynamic description logic based on the tractable description logic $\text{DL-Lite}_{\text{R}}^{\text{pr}}$, in such a way that all the knowledge described by $\text{DL-Lite}_{\text{R}}^{\text{pr}}$ is supported by our system. A decision algorithm is provided for our system $\text{DDL-Lite}_{\text{R}}^{\text{pr}}$.Termination and correctness of the algorithm are proved.

Keywords: description logic, dynamic description logic, action theory, satisfiability, tableau algorithm

1 Introduction

With the rapid development of the Semantic Web, description logics [1] are playing an important role in it, which are recommended by W3C as the basis of Web Ontology Language OWL [2]. About static application domains, they provide considerable expressive power and decidable reasoning mechanisms [3]. But they can't directly deal with knowledge of dynamic application domains which are characterized by actions.

For this limitation, Shi et al. [4] put forward a dynamic description logic DDL based on a combination of description logic ALC, dynamic logic and action theory. Based on DDL, a family of dynamic description logics named DDL($X^{@}$)[5] was proposed for representing and reasoning about actions, where the minimal change semantics[6] were used to define the semantics of atomic action definitions. But all the current decision algorithms for dynamic description logics are all restricted to requiring that TBoxes of description logics don't include GCIs any more. The DL-Lite family [7] is a family of DLs tailored to capture conceptual modeling constructs while keeping reasoning.

Based on [5], in this paper, we first of all propose a dynamic description logic $\text{DDL-Lite}_{\text{R}}^{\text{pr}}$ and give its syntax and semantics. And we adopt model-based semantics $\mathcal{L}_{\subseteq}^{a}$ [8] of ABox update to define the semantics of atomic action definitions. Then a

tableau decision algorithm which supports GCIs for DDL-Lite$_R^{pr}$ is given. Finally, the termination and correctness of the algorithm are proved.

2 Dynamic Description Logic DDL-Lite$_R^{pr}$

DDL-Lite$_R^{pr}$ primitive symbols include a set N_C of concept names, a set N_R of role names, a set N_I of individual names and a set N_A of action names. With the help of a set of constructors, starting from these symbols, roles, concepts, formulas and actions can be inductively constructed respectively.

Definition 1. Roles of DDL-Lite$_R^{pr}$ are formed according to the following syntax rule:

$$R ::= P \mid P^{-} \text{ where } P \in N_R.$$

Definition 2. Concepts of DDL-Lite$_R^{pr}$ are formed according to the following syntax rule:

$$C, C' ::= A_i \mid \neg C \mid \exists R \text{ where } A_i \in N_C, R \text{ is a role.}$$

The form of $C_1 \sqsubseteq C_2$ is called a GCI short for a general concept inclusion assertion, where C_1 and C_2 are any concept. The form of $R_1 \sqsubseteq R_2$ is called a role inclusion assertion where R_1 and R_2 are any role.Each finite set $\mathcal{T}$ of GCIs and role inclusion assertions is called a TBox of DDL-Lite$_R^{pr}$, where disjointness that involves roles is forbidden.

Definition 3. Formulas of DDL-Lite$_R^{pr}$ are formed according to the following syntax rule:

$\varphi, \varphi' ::= C(p) \mid R(p, q) \mid <\pi> \varphi \mid \neg\varphi$ where $p, q \in N_I$, $R \in N_R$, C is a concept, π is an action.

Formulas of the form $C(p)$, $R(p, q)$, $<\pi>\varphi$ and $\neg\varphi$ are respectively called concept assertion, role assertion, diamond assertion and negation formula. Concept assertions, role assertions, negations of concept assertions, and negations of role assertions are all called ABox assertions. A finite set of ABox assertions is called an ABox of DDL-Lite$_R^{pr}$. For any ABox $\mathcal{A}$, we use $\mathcal{A}^{\neg}$ to denote the set $\{\neg\varphi \mid \varphi \in \mathcal{A}\}$.

Definition 4. With respect to a TBox $\mathcal{T}$, an atomic action definition of DDL-Lite$_R^{pr}$ is of the form $\alpha \equiv (P, E)$, where,

(1) $\alpha \in N_A$ is an atomic action name;
(2) P is a finite set of ABox assertions for describing the pre-conditions of the action;
(3) E is a finite set of ABox assertions for describing the post-conditions.

For each finite set $\mathcal{A}c$ of atomic action definitions, if no action name occurs on the left-hand sides for more than once, then we call $\mathcal{A}c$ an ActBox of DDL-Lite$_R^{pr}$.

Definition 5. With respect to a TBox $\mathcal{T}$, an ActBox $\mathcal{A}c$, actions of DDL-Lite$_R^{pr}$ are formed according to the following syntax rule:

$\pi, \pi' ::= \alpha \mid \varphi? \mid \pi \cup \pi' \mid \pi; \pi' \mid \pi^*$ where α is an atomic action, φ is a formula.

Actions of the form α, $\varphi?$, $\pi \cup \pi'$, $\pi; \pi'$ and π^* are respectively called atomic action, test action, choice action, sequential action and iterated action.

Definition 6. A DDL-Lite$_R^{pr}$ -model is of the form $M = (W, T, \Delta, I)$, where,

1) W is a non-empty finite set composed of states;
2) T is a function that maps every action name $\alpha \in N_A$ to a binary relation $T(\alpha) \subseteq W \times W$;
3) Δ is a non-empty set made up of individuals;
4) I is a function which associates every state $w \in W$ a DL-interpretation $I(w) = (\Delta, \cdot^{I(w)})$, where the function $\cdot^{I(w)}$

– maps each concept name $C_i \in N_C$ to a set $C_i^{I(w)} \subseteq \Delta$,
– 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 $p_i \in N_I$ to an individual $p_i^{I(w)} \in \Delta$, with the constraints that $p_i^{I(w)} = p_i^{I(w')}$ for any state $w' \in W$.

Definition 7. Let $\mathcal{T}$, $\mathcal{A}c$ be a TBox, an ActBox respectively. $M = (W, T, \Delta, I)$ is a model of DDL-Lite$_R^{pr}$. The semantics of roles, concepts, formulas and actions of DDL-Lite$_R^{pr}$ are defined inductively as follows.

Firstly, for any state $w \in W$, each role R is interpreted as a binary relation $R^{I(w)} \subseteq \Delta \times \Delta$ and each concept C is interpreted as a set $C^{I(w)} \subseteq \Delta$. The semantics of roles and concepts of DDL-Lite$_R^{pr}$ are defined inductively as follows:

1. $(R^-)^{I(w)} = \{(y, x) | x \in \Delta, y \in \Delta \text{ and } (x, y) \in R^{I(w)}\}$;
2. $(\neg C)^{I(w)} = \Delta \setminus C^{I(w)}$, where "\" is the set difference operator;
3. $(\exists R)^{I(w)} = \{ x \in \Delta |$ there is some $y \in \Delta$ such that $(x, y) \in R^{I(w)}\}$.

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

4. $(M, w) \models C(p)$ iff $p^I \in C^{I(w)}$;
5. $(M, w) \models R(p, q)$ iff $(p^I, q^I) \in R^{I(w)}$;
6. $(M, w) \models \neg\varphi$ iff it is not the case that $(M, w) \models \varphi$;
7. $(M, w) \models <\pi>\varphi$ iff some state $w' \in W$ exists with $(w, w') \in T(\pi)$ and $(M, w') \models \varphi$;
8. $(M, w) \models [\pi]\ \varphi$ iff for every state $w' \in W$: if $(w, w') \in T(\pi)$ then $(M, w') \models \varphi$.

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

9. $T(\varphi?) = \{(w, w) | (M, w) \models \varphi\}$;
10. $T(\pi \cup \pi') = T(\pi) \cup T(\pi')$;
11. $T(\pi;\ \pi') = \{(w, w')$ |there is some state $w'' \in W$ such that $(w, w'') \in T(\pi)$ and $(w'', w') \in T(\pi')\}$;
12. $T(\pi^*)$ = reflexive transitive closure of $T(\pi)$.

A model M satisfies a TBox $\mathcal{T}$, denoted by $M \models \mathcal{T}$, if and only if for every state $w \in W$, $C_1^{I(w)} \subseteq C_2^{I(w)}$ for every concept inclusion assertion $C_1 \sqsubseteq C_2 \in \mathcal{T}$ and $R_1^{I(w)} \subseteq R_2^{I(w)}$ for every role inclusion assertion $R_1 \sqsubseteq R_2 \in \mathcal{T}$. A state w of a model M satisfies an ABox $\mathcal{A}$, denoted by $(M, w) \models \mathcal{A}$, if and only if $(M, w) \models \varphi$ for every ABox assertion $\varphi \in \mathcal{A}$.

According to $\mathcal{A}c$, any atomic action α is specified by some atomic action definition $\alpha \equiv (P, E)$. Then w.r.t. a TBox $\mathcal{T}$, a model M satisfies an atomic action definition $\alpha \equiv (P, E)$, in symbols $M \models_{\mathcal{T}} \alpha \equiv (P, E)$, if and only if $M \models \mathcal{T}$ and $T(\alpha) = \{(w, w') | ①(M, w) \models P$, $②(M, w') \models E$, ③for every state $w'' \in W$, if $(M, w'') \models E$, then $\text{dist}_{\subseteq}^{a}(I(w), I(w')) \subseteq \text{dist}_{\subseteq}^{a}(I(w), I(w''))\}$.

The distance function between interpretations is specified in details as follows.

Firstly, given two interpretations $I(w) = (\Delta, \cdot^{I(w)})$ and $I(w') = (\Delta, \cdot^{I(w')})$, the distance between them is denoted by $\text{dist}_{\subseteq}^{a}(I(w), I(w'))$. Then $\text{dist}_{\subseteq}^{a}(I(w), I(w')) = I(w) \ominus I(w')$

$$= \bigcup_{C \in N_C} ((C^{I(w)} - C^{I(w')}) \cup (C^{I(w')} - C^{I(w)})) \cup \bigcup_{R \in N_R} ((R^{I(w)} - R^{I(w')}) \cup (R^{I(w')} - R^{I(w)}))$$

where $\ominus$ is the set symmetric difference operator. Distances under $\text{dist}_{\subseteq}^{a}$ are compared by set inclusion.

Definition 8. A model M satisfies an ActBox $\mathcal{A}c$ w.r.t. a TBox $\mathcal{T}$, in symbols $M \models_{\mathcal{T}} \mathcal{A}c$, if and only if $M \models_{\mathcal{T}} \alpha \equiv (P, E)$ for every atomic action definition $\alpha \equiv (P, E) \in \mathcal{A}c$.

Definition 9. A formula φ is satisfiable w.r.t. a TBox $\mathcal{T}$ and an ActBox $\mathcal{A}c$ if and only if there is a model $M = (W, T, \Delta, I)$ and a state $w \in W$ such that $M \models \mathcal{T}$, $M \models_{\mathcal{T}} \mathcal{A}c$ and $(M, w) \models \varphi$.

3 Tableau Decision Algorithm for DDL-Lite$_{\mathbf{R}}^{\mathbf{pr}}$

Let $\mathcal{T}$, $\mathcal{A}c$ be a TBox and an ActBox respectively. Let φ be a DDL-Lite$_{\mathrm{R}}^{\mathrm{pr}}$ -formula which is defined w.r.t. $\mathcal{A}c$. For the convenience of presentation, we firstly transform the formula φ into a normal form nf(φ) according to the following steps.

(1) Replace every occurrence of atomic actions with their atomic action definitions.Let φ' be the resulted formula.

(2) Transform φ' into an equivalent one in negation normal form by pushing negations inwards according to the following equivalences:

$$\neg(\langle\pi\rangle\psi) = [\pi]\neg\psi \qquad \neg([\pi]\psi) = \langle\pi\rangle\neg\psi \qquad \neg\neg\psi = \psi$$

A full closure of an ABox $\mathcal{A}$ w.r.t. TBox $\mathcal{T}$, denoted by $fcl_{\mathcal{T}}(\mathcal{A})$, is the set of all ABox assertions f such that $\mathcal{A} \models_{\mathcal{T}} f$.

Next, we introduce some definitions involved in the algorithm as follows.

Given a TBox $\mathcal{T}$, a prefix σ.μ with a sequential action σ and a set μ of ABox assertions is constructed according to the following syntax rule:

$$\sigma.\mu ::= (\varnothing, \varnothing).\varnothing \mid \sigma;(P, E).(\mu\backslash(fcl_{\mathcal{T}}(E))^{\neg})\cup E$$

where $(\varnothing, \varnothing)$ and (P, E) are atomic actions, $\sigma;(P, E)$ is a sequential action.

We also use $\sigma_0.\mu_0$ to denote the prefix $(\varnothing, \varnothing).\varnothing$ and call it the initial prefix. A prefixed formula is a pair $\sigma.\mu:\varphi$, where σ.μ is a prefix and φ is a formula.

A branch $\mathcal{B}$ is a union of a set $\mathcal{B}_{PF}$ of prefixed formulas and a set $\mathcal{B}_E$ of eventuality records which is of the form $X \equiv \langle\pi^*\rangle\varphi$.

A branch $\mathcal{B}$ is completed if and only if it can't be expanded by any tableau expansion rule.

An eventuality record $X \equiv \langle\pi^*\rangle\varphi$ is fulfilled in a branch $\mathcal{B}$ if and only if there is a prefix σ.μ such that both $\sigma.\mu: X \in \mathcal{B}$, and $\sigma.\mu:\varphi \in \mathcal{B}$.

A branch $\mathcal{B}$ is ignorable if and only if it is completed but contains some eventuality record $X \equiv \langle\pi^*\rangle\varphi$ which is not fulfilled.

A branch $\mathcal{B}$ is contradictory if and only if there is some prefix σ.μ and some formula φ such that both $\sigma.\mu:\varphi \in \mathcal{B}$ and $\sigma.\mu:\neg\varphi \in \mathcal{B}$.

For any branch $\mathcal{B}$, $\{\psi \mid \sigma_0.\mu_0:\psi \in \mathcal{B}$ and ψ is an ABox assertion $\}$ is denoted by $IV_{\mathcal{B}}$.

Tableau expansion rules on inverse roles, non-atomic actions are the same with [5].

$\neg$atom$_{\langle\rangle}$-rule	If $\sigma.\mu:\neg\langle P, E\rangle\varphi \in \mathcal{B}$, $\{\sigma.\mu:\psi^{\neg} \mid \psi \in P\} \cap \mathcal{B} = \varnothing$, and there is a prefix $\sigma'.\mu'$ with both $\mu' = (\mu\backslash(fcl_{\mathcal{T}}(E))^{\neg})\cup E$ and $\sigma'.\mu':\neg\varphi \notin \mathcal{B}$, then set either $\mathcal{B} := \mathcal{B} \cup \{\sigma'.\mu':\neg\varphi\}$ or $\mathcal{B} := \mathcal{B} \cup \{\sigma.\mu:\neg\psi\}$ for some $\psi \in P$.
atom$_{\langle\rangle}$-rule	If $\sigma.\mu:\langle P, E\rangle\varphi \in \mathcal{B}$, and if $\{\sigma.\mu: \psi \mid \psi \in P\} \subsetneq \mathcal{B}$, or no prefix $\sigma'.\mu'$ exists with both $\mu' = (\mu\backslash(fcl_{\mathcal{T}}(E))^{\neg})\cup E$ and $\sigma'.\mu':\varphi \in \mathcal{B}$, then: if there is no prefix $\sigma'.\mu'$ with $\mu' = (\mu\backslash(fcl_{\mathcal{T}}(E))^{\neg})\cup E$, then introduce a prefix $\sigma''.\mu'' := \sigma;(P, E).(\mu\backslash(fcl_{\mathcal{T}}(E))^{\neg})\cup E$ and set $\mathcal{B} := \mathcal{B} \cup \{\sigma.\mu: \psi \mid \psi \in P\} \cup \{\sigma''.\mu'':\varphi\} \cup \{\sigma''.\mu'': \phi \mid \phi \in \mu''\}$, else find a prefix $\sigma'.\mu'$ with $\mu' = (\mu\backslash(fcl_{\mathcal{T}}(E))^{\neg})\cup E$, and set $\mathcal{B} := \mathcal{B} \cup \{\sigma.\mu: \psi \mid \psi \in P\} \cup \{\sigma'.\mu':\varphi\} \cup \{\sigma'.\mu': \phi \mid \phi \in \mu'\}$.

Fig.1. Tableau expansion rules on atomic actions

B-rule	If $\sigma.\mu:\varphi \in \mathcal{B}$, and φ is an ABox assertion, then : 1. $\mathcal{A}=\{\varphi\}$; 2. if $\{\sigma_0.\mu_0: \phi \mid \phi \in \mathcal{A}^{\mathrm{Regress}(\sigma.\mu)}\} \subsetneq \mathcal{B}$, then any $\psi \in \mathcal{A}^{\mathrm{Regress}(\sigma.\mu)}$ and $\sigma_0.\mu_0:\psi \notin \mathcal{B}$ set $\mathcal{B} := \mathcal{B} \cup \{\sigma_0.\mu_0: \psi\}$.

Fig.2. Tableau expansion rules on ABox assertions

For B-rule, ABox $\mathcal{A}^{\mathrm{Regress}(\sigma.\mu)}$ is constructed according to the following Algorithm 1.

Algorithm 1 Given a regress setting $\mathcal{K}=(\mathcal{T}, \mathcal{A})$ and $\mu^{\neg}$, given that $\mathcal{A}$ and $\mu^{\neg}$ w.r.t. $\mathcal{T}$ are satisfiable respectively, construct an ABox $\mathcal{A}'$ by the following steps.

Step 1. Let $\mathcal{A}'$ is an empty set and S is equivalent to $fcl_{\mathcal{T}}(\mathcal{A})$.

Step 2. If S is not an empty set, do the following operations:

1. Choose some ψ of S and at the same time remove ψ from S.

2. If a union of the two sets $\{\psi\}, fcl_{\mathcal{T}}(\mu^{\neg})$ is satisfiable, then add ψ to $\mathcal{A}'$.

3. Repeat Step 2 until the condition is unallowed.

The construction process ensures the following Lemma 3 and $\mathcal{A}^{\mathrm{Regress}(\sigma.\mu)}$ is a union of $\mathcal{A}'$, $\mu^{\neg}$.

Finally, the tableau decision algorithm is described as follows.

Algorithm 2 The satisfiability of a formula φ w.r.t. a TBox $\mathcal{T}$ and an ActBox $\mathcal{A}c$ is decided according to the following steps:

Step1. Construct a branch $\mathcal{B} := \{\sigma_0.\mu_0: \mathrm{nf}(\varphi)\}$.

Step2. If all the tableau expansion rules are applied to $\mathcal{B}$ in random order so that they yield a completed branch $\mathcal{B}'$, and $\mathcal{B}'$ is neither contradictory nor ignorable, and $IV_{\mathcal{B}'}$ of $\mathcal{B}'$ is consistent w.r.t. $\mathcal{T}$, then the algorithm returns "*TRUE*", else returns "*FALSE*".

Theorem 1. Algorithm 2 terminates.

Firstly, the number of branches which will be investigated by Algorithm 2 is finite. Then, for each branch $\mathcal{B}$ investigated by Algorithm 2, the number of ABox assertions contained in the initial view $IV_{\mathcal{B}}$ is also finite. Finally, the consistency of $IV_{\mathcal{B}}$ w.r.t. $\mathcal{T}$ can be decided with terminable procedures.

To demonstrate the correctness of Algorithm 2, some notations are introduced as follows.

Firstly, with reference to Algorithm AlignAlg($(\mathcal{T}, \mathcal{A})$, $\mathcal{N}$)[8], Algorithm 1 has the following properties.

Lemma 1. Let $\mathcal{T}$, $\mathcal{A}$, $\mathcal{N}$ be a TBox, an ABox and an ABox, respectively. $\mathcal{A}$, $\mathcal{N}$ w.r.t. $\mathcal{T}$ are consistent, if $\mathcal{A}^{\mathcal{N}}$ is an ABox which is constructed according to Algorithm 1 under $\mathcal{A}$ and the update information $\mathcal{N}$, then we have $\mathcal{A}^{\mathcal{N}} = fcl_{\mathcal{T}}(\mathcal{A})\text{-}(fcl_{\mathcal{T}}(\mathcal{N}))^{\neg} \cup \mathcal{N}$.

Lemma 2. Let $\mathcal{T}$ be a TBox, $\sigma.\mu$ is a prefix w.r.t. $\mathcal{T}$, where $\sigma = (\varnothing, \varnothing); (P_1, E_1); (P_2, E_2); \ldots; (P_n, E_n)$. For any ABox $\mathcal{A}$, if $\mathcal{A}$ w.r.t. $\mathcal{T}$ is consistent, then there must be $((\mathcal{A}^{E_1})^{E_2}\cdots)^{E_n} = \mathcal{A}^{\mu}$.

Secondly, we introduce branch-model mappings to act as bridges between branches and models.

Let $\mathcal{T}$, $\mathcal{A}c$, $\mathcal{B}$ and $M=(W, T, \Delta, I)$ be a TBox, an ActBox, a branch and a model respectively. A branch-model mapping δ w.r.t. $\mathcal{T}$, $\mathcal{B}$ and M is a function from $\sum$ of prefixes occurring in $\mathcal{B}$ to states of M, satisfying that for each pair of prefixes $\sigma.\mu$ and $\sigma'.\mu'$ occurring in $\mathcal{B}$: if there is an atomic action (P, E), $\sigma'=\sigma;(P, E)$ and $\mu' = (\mu \backslash (fcl_{\mathcal{T}}(E))^{\neg}) \cup E$, then:

(1) $M \models \mathcal{T}$; (2) $(M, \delta(\sigma.\mu)) \models P$; (3) $(M, \delta(\sigma'.\mu')) \models E$; (4) for any $\sigma''.\mu'' \in \sum$,

if $(M, \delta(\sigma''.\mu'')) \models E$, then $\mathrm{dist}^{a}_{\subseteq}(I(\delta(\sigma.\mu)), I(\delta(\sigma'.\mu'))) \subseteq \mathrm{dist}^{a}_{\subseteq}(I(\delta(\sigma.\mu)), I(\delta(\sigma''.\mu'')))$.

By means of a branch-model mapping, we can get the following property.

Lemma 3. Let $\mathcal{A}^{\mathrm{Regress}(\sigma.\mu)}$ be a ABox constructed by the regression operator in the B-rule. Let $M = (W, T, \Delta, I)$ be a model with $M \models \mathcal{T}$ .Then, for any branch-model mapping δ w.r.t. $\mathcal{T}$, $\mathcal{B}$ and M, we have $(M, \delta(\sigma.\mu)) \vDash \mathcal{A}$, iff $(M, \delta(\sigma_0.\mu_0)) \vDash \mathcal{A}^{\mathrm{Regress}(\sigma.\mu)}$.

Finally, it is easy to prove the correctness of Algorithm 2.

Theorem 2. Algorithm 2 returns "φ is satisfiable w.r.t. $\mathcal{T}$and $\mathcal{A}c$ " if and only if φ w.r.t. $\mathcal{T}$and $\mathcal{A}c$ is satisfiable.

4 Conclusion

Dynamic description logic $\mathrm{DDL\text{-}Lite}_{\mathrm{R}}^{\mathrm{pr}}$ not only inherits the feature of representing and reasoning about knowledge of dynamic application domains as the same with the existing dynamic description logics, but also at the same time its TBoxes are composed of GCIs. We broaden knowledge offered by TBoxes of dynamic description logics and provide a reliable algorithm foundation for the practical application of a dynamic description logic with GCIs. Further work is to optimize the algorithm and develop a corresponding dynamic description logic reasoning machine.

Acknowledgements. This work is supported by the National Natural Science Foundation of China (Nos. 61363030, 61100025, 61262030), the Natural Science Foundation of Guangxi Province (No.2012GXNSFBA053169) and the Science Foundation of Guangxi Key Laboratory of Trusted Software.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F.:The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, Cambridge (2003)
2. Horrocks, I., Patel-Schneider, P.F., Harmelen, F.V.: From SHIQ and RDF to OWL: the making of a web ontology language. J. Web Semant. 1(1), 7–26 (2003)
3. Baader, F., Sattler, U.: An Overview of Tableau Algorithms for Description Logics. Studia Logica 69, 5–40(2001)
4. Shi, Z.Z., Dong, M.K., Jiang, Y.C., Zhang, H.J.: A logical foundation for the semantic Web. Science in China, Ser. F: Information Sciences 48(2), 161–178 (2005)
5. Chang, L., Shi, Z.Z., Gu, T.L. , Zhao, L.Z. : A family of dynamic description logics for representing and reasoning about actions. Journal of Automated Reasoning 49(1), 19–70 (2012)
6. Baader, F., Lutz, C., Mili či ć , M., Sattler, U., Wolter., F.: Integrating description logics and action formalisms: first results. In: Proc. of the 12th Nat. Conf. on Artificial Intelligence (AAAI'05), pp. 572 – 577. AAAI Press / MIT Press (2005)
7. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The DL-Lite family and relations. J. of Artificial Intelligence Research 36, 1–69 (2009)
8. Kharlamov, E., Zheleznyakov, D.: Capturing instance level ontology evolution for DL-Lite. In: Proc. of the 10th Int. Conf. on Semantic Web(ISWC'11), pp. 321–337. Springer(2011)