Model Checking of Restricted CTL* formulas using $\mathcal{ALCK}_{R^+}$

Taufiq Rochaeli and Claudia Eckert

Department of Computer Science
Technische Universitt Darmstadt
{rochaeli,eckert}@sec.informatik.tu-darmstadt.de

Introduction The purpose of model checking technique is to verify whether the implemented computer program (the model) satisfies the specified requirements (the formulas). Now let K be the Kripke model representing the behavior of the system and R be the set of formulas. The model checking process verifies whether every formula in R is satisfied by the model, which has I as the set of initial states. Formally, it checks whether $\forall f \in R, \forall s \in I : K, s \vDash f$ is true or not. We propose an approach to perform model checking of the restricted CTL* formula based on the top of description logics reasoning services. This approach allows us to build the ontology of the atomic propositions used both in the model and in the formula.

We realize our approach by performing the following steps: (i) representing the model in the assertion box (ABox) of the description-logic based knowledge base, (ii) defining the ontology of atomic propositions in the terminology box (TBox) of the knowledge base, (iii) defining a part of the formal semantics of CTL* logic on top of the DL semantics, (iv) defining the translation rules for the formula.

Representation of the Kripke Model and the Ontology of Atomic Propositions The Kripke model is represented as a set of axioms $\mathcal{A}_M$. There are three types of axiom in $\mathcal{A}_M$. The first type of the axiom, $\mathsf{State}(x)$, defines the state s of Kripke model K, where x is a unique individual name representing state s. The second type of the axiom, $\mathit{next}(x_i, x_j)$,defines the state transition between s_i and s_j. The last kind of axiom, $V_i(x)$, is created for each $p_i \in L(s)$. The concept V_i represents the atomic proposition p_i. For each state s of the Kripke model K, an assertion is created in $\mathcal{A}_M$, where x is a unique individual name representing the state s. The state transitions (s_i, s_j) are defined by the assertions in form of $\mathit{next}(x_i, x_j)$. The set of atomic propositions that label a state are represented as an assertion about instance membership. That is, for each $p_i \in L(s)$ an assertion $V_i(x)$ is created in $\mathcal{A}_M$. The concept V_i represents the atomic proposition p_i. The ontology of atomic propositions is defined in $\mathcal{T}_{AP}$, which has the purpose as the common vocabulary.

Formal Semantics of CTL* Logic in Description Logics The basic idea in *emulating* the formal semantics of CTL* temporal logic on top of the description logic semantics is to construct the Kripke model in the ABox. Furthermore, we require the epistemic operator **K**, which is presented in [1], to fix the interpretation of the Kripke model represented in the ABox. Therefore, the epistemic operator **K** immediately appears in the concepts representing atomic propositions and the roles representing state transitions. (see equations 1, 7 - 10.)

$$
\begin{aligned}
s \vDash p_i &\Leftrightarrow \mathcal{KB} \vDash \mathbf{K}V_i(x) && (1)\\
s \vDash \mathbf{E}\,(g_1) &\Leftrightarrow \mathcal{KB} \vDash D_1(x) && (2)\\
\pi \vDash g_1 &\Leftrightarrow \mathcal{KB} \vDash D_1(\sigma_1) && (3)\\
\pi \vDash \neg g_1 &\Leftrightarrow \mathcal{KB} \vDash \neg D_1(\sigma_1) && (4)\\
\pi \vDash g_1 \vee g_2 &\Leftrightarrow \mathcal{KB} \vDash (D_1 \sqcup D_2)(\sigma_1) && (5)\\
\pi \vDash g_1 \wedge g_2 &\Leftrightarrow \mathcal{KB} \vDash (D_1 \sqcap D_2)(\sigma_1) && (6)\\
\pi \vDash \mathbf{X}\, g_1 &\Leftrightarrow \mathcal{KB} \vDash (\exists \mathbf{K}\mathit{next}.D_1)(\sigma_1) && (7)\\
\pi \vDash \mathbf{G}\, g_1 &\Leftrightarrow \langle \mathcal{T}_M \cup \{D_{aux} \equiv D_1 \sqcap \exists \mathbf{K}\mathit{next}.D_{aux}\}, \mathcal{A}_M\rangle \vDash D_{aux}\,(\sigma_1) && (8)\\
\pi \vDash \mathbf{F}\, g_1 &\Leftrightarrow \mathcal{KB} \vDash (D_1 \sqcup \exists \mathbf{K}\mathit{next}^{+}.D_1)\,(\sigma_1) && (9)\\
\pi \vDash g_1\, \mathcal{U}\, g_2 &\Leftrightarrow \langle \mathcal{T}_M \cup \{D_{aux} \equiv D_2 \sqcup (D_1 \sqcap \exists \mathbf{K}\mathit{next}.D_{aux})\}, \mathcal{A}_M\rangle && (10)\\
&\qquad \vDash D_{aux}\,(\sigma_1)
\end{aligned}
$$

Fig. 1. Formal semantics of a subset of CTL* logic in $\mathcal{ALCK}$.

Translating CTL* Formulas into TBox Concepts The restricted CTL* formulas has the form of $\mathbf{E}\,[\phi]$, where ϕ is the LTL formula enclosed in a $\mathbf{E}$ path quantifier. The first step is to decompose ϕ into a parse tree. Now we can perform the translation of the LTL formula ϕ, by visiting the nodes in the parse tree starting from the leaf nodes and toward the root node. Suppose that we have a counter idx, which has the initial value 0. For each visit in the node, we create a new axiom in the TBox $\mathcal{T}_f$ according to the rules in Fig. 2 and increase the counter idx. The concepts D_i and D_j refer to the concepts created after translating the previous sub formulas g_i and g_j, respectively.

LTL formula	concepts added to $\mathcal{T}_f$
p_i	$D_{idx} \equiv \mathbf{K}V_i$
$\neg g_i$	$D_{idx} \equiv \neg D_i$
$g_i \vee g_j$	$D_{idx} \equiv D_i \sqcup D_j$
$g_i \wedge g_j$	$D_{idx} \equiv D_i \sqcap D_j$
$\mathbf{X}g_i$	$D_{idx} \equiv \exists \mathbf{K}\mathit{next}.D_i$
$\mathbf{G}g_i$	$D_{idx} \equiv D_i \sqcap \exists \mathbf{K}\mathit{next}.D_{idx}$
$\mathbf{F}g_i$	$D_{idx} \equiv D_i \sqcup \exists \mathbf{K}\mathit{next}^{+}.D_i$
$g_i\, \mathcal{U}\, g_j$	$D_{idx} \equiv D_j \sqcup (D_i \sqcap \exists \mathbf{K}\mathit{next}.D_{idx})$

Fig. 2. Translation table of a subset of CTL* sub formulas into $\mathcal{ALCK}$ concepts.

Performing the Model Checking Let K and s be the Kripke model representing the dynamic behavior of a system and the initial state of the model. The formula f represents the desired property. The model checking $K, s \vDash f$ can be performed as an instance checking $\mathcal{KB}_M \vDash C(x)$, whereas $\mathcal{KB}_M : \langle \mathcal{T}_{AP} \cup \mathcal{T}_f, \mathcal{A}_M\rangle$.

References

1. Donini, F.M., Nardi, D., Rosati, R.: Description logics of minimal knowledge and negation as failure. ACM Trans. Comput. Logic **3** (2002) 177–225