Forgetting in CTL to Compute Necessary and Sufficient Conditions

Written by AAAI Press Staff[1]*
AAAI Style Contributions by Pater Patel Schneider,
Sunil Issar, J. Scott Penberthy, George Ferguson, Hans Guesgen
[1]Association for the Advancement of Artificial Intelligence
2275 East Bayshore Road, Suite 160
Palo Alto, California 94303
publications20@aaai.org

Abstract

Computation Tree Logic (CTL) is one of the central formalisms in formal verification. As a specification language, it is used to express a property that the system at hand is expected to satisfy. From both the verification and the system design points of view, some information content of such property might become irrelevant for the system due to various reasons e.g., it might become obsolete by time, or perhaps infeasible due to practical difficulties. Then, the problem arises on how to subtract such piece of information without altering the relevant system behaviour or violating the existing specifications. Moreover, in such a scenario, two crucial notions are informative: the strongest necessary condition (SNC) and the weakest sufficient condition (WSC) of a given property.

To address such a scenario in a principled way, we introduce a forgetting-based approach in CTL and show that it can be used to compute SNC and WSC of a property under a given model. We study its theoretical properties and also show that our notion of forgetting satisfies existing essential postulates. Furthermore, we analyse the computational complexity of basic tasks, including various results for the relevant fragment CTL_{AF}.

Introduction

Consider a car-manufacturing company which produces two types of automobiles: a sedan car (basically a four-doored classical passenger car) and a sports car. No matter a sedan or a sports car, both production lines are subject to a single standard criterion which is indispensable: safety restrictions. This shared feature is also complemented several major differences aligned with these types in general. That is, a sedan car is produced with a small engine, while a sports car is produced with a large one. Moreover, due to its large amount of production, the sedan car is subject to some very restrictive low-carbon emission regulations, while a sports car is not. On the verge of shifting to an upcoming new engine technology, the company aims to adapt the sedan production to electrical engines. Such major shift in production also comes with one in regulations; electric sedans are not subject to low-carbon emission restrictions any more. In fact, due to the difference in its underlying technology, a sedan car drastically emits much less carbon, hence such standard is obsolete, and can be dropped. Yet dropping some restrictions in a large and complex production system in automative industry, without affecting the working system components or violating dependent specifications is a non-trivial task.

Similar scenarios may arise in many different domains such as business-process modelling, software development, concurrent systems and more (Baier and Katoen 2008). In general, some information content of such property might become irrelevant for the system due to various reasons e.g., it might become obsolete by time like in the above example, or perhaps becomes infeasible due to practical difficulties. Then, the problem arises on how to subtract such piece of information without altering the behaviour of the relevant system components or violating the existing specifications. Moreover, in such a scenario, two logical notions introduced by E. Dijkstra in (Dijkstra 1978) are very informative: the *strongest necessary condition* (SNC) and the *weakest sufficient condition* (WSC) of a given specification. These correspond to *most general consequence* and the *most specific abduction* of such specification, respectively.

To address such a scenario in a principled way, we employ a method based on formal verification.[1] In particular, we introduce a *forgetting*-based approach in Computation Tree Logic (CTL) (Clarke and Emerson 1981) a central formalism in formal verification, and show that it can be used to compute SNC and WSC, in the same spirit of (Lin 2001).

The scenario we mentioned concerning car-engine manufacturing can be easily represented as a small example by the following Kripke structure $\mathcal{M} = (S, R, L, s_0)$ in Figure 1 on $V = \{sl, sr, se, le, lc\}$ whose elements correspond to *select*, *safety restrictions*, *small engine*, *large engine* and *low-carbon emission requirements*, respectively. Moreover, s_0 is the initial state where we select either choose producing an engine for *sedan* which corresponds to the state s_1 or a *sports car* which corresponds to the state s_2. Since only a single-type of production is possible at a time, after each production state (s_1 or s_2), we turn back to the initial state

*Primarily Mike Hamilton of the Live Oak Press, LLC, with help from the AAAI Publications Committee

[1]This is especially useful for abstracting away the domain-dependent problems, and focusing on conceptual ones.

(s_0) to start over.

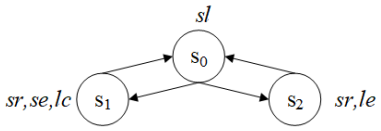

Figure 1: Car Engine Manufacturing Scenario

The notions of SNC and WSC were considered in the scope of formal verification among others, in generating counterexamples (Dailler et al. 2018) and refinement of system (Woodcock and Morgan 1990). On the *forgetting* side, it was first formally defined in propositional and first order logics by Lin and Reiter (Lin and Reiter 1994). Over the last decades, researchers have developed forgetting notions and theories not only in classical logic but also in non-classical logic systems (Eiter and Kern-Isberner 2019), such as forgetting in logic programs under answer-set semantics (Zhang and Foo 2006; Eiter and Wang 2008; Wong 2009; Wang et al. 2012; Wang, Wang, and Zhang 2013), description logics (Wang et al. 2010; Lutz and Wolter 2011; Zhao and Schmidt 2017) and knowledge forgetting in modal logic (Zhang and Zhou 2009; Su et al. 2009; Liu and Wen 2011; Fang, Liu, and Van Ditmarsch 2019). It also has been considered in planning (Lin 2003) and conflict solving (Lang and Marquis 2010; Zhang, Foo, and Wang 2005), creating restricted views of ontologies (Zhao and Schmidt 2017), strongest and weakest definitions (Lang and Marquis 2008), SNC (WSC) (Lin 2001), among others.

Although forgetting has been extensively investigated from various aspects of different logical systems, the existing forgetting techniques are not directly applicable in CTL. For instance, in propositional forgetting theory, forgetting atom q from φ is equivalent to a formula $\varphi[q/\top] \vee \varphi[q/\ \bot]$, where $\varphi[q/X]$ is a formula obtained from φ by replacing each q with X ($X \in \{\top, \bot\}$). This method cannot be extended to a CTL formula. Consider a CTL formula $\psi = \text{AG}p \wedge \neg\text{AG}q \wedge \neg\text{AG}\neg q$. If we want to forget atom q from ψ by using the above method, we would have $\psi[q/\top] \vee \psi[q/\ \bot] \equiv \bot$. This is obviously not correct since after forgetting q this specification should not become inconsistent. Similar to (Zhang and Zhou 2009), we research forgetting in CTL from the semantic forgetting point of view. And it is shown that our definition of forgetting satisfies those four postulates of forgetting presented in (Zhang and Zhou 2009).

The rest of the paper is organised as follows. Section 2 introduces the notation and technical preliminaries. As key contributions, Section 3, introduces the notion of forgetting in CTL, via developing the notion of V-bisimulation. Such bisimulation is constructed through a set-based bisimulation and more general than the classical bisimulation. Moreover, it provides a CTL characterization for model structures (with the initial state), and studies the semantic properties of forgetting. In addition, a complexity analysis, including a relevant fragment CTL_{AF}, is carried out. Section 4 explores the relation between forgetting and SNC (WSC). Section 5 gives a model-based algorithm for computing forgetting in CTL and outline its complexity. Conclusion closes the paper.

Due to space restrictions and to avoid hindering the flow of content, all the proofs are put to the supplementary material.

Preliminaries

We start with some technical and notational preliminaries. Throughout this paper, we fix a finite set $\mathcal{A}$ of propositional variables (or atoms), and use V, V' for subsets of $\mathcal{A}$.

Model structure in CTL

In general, a transition system can be described by a *model structure* (or *Kripke structure*) (see (Baier and Katoen 2008) for details). A model structure is a triple $\mathcal{M} = (S, R, L)$, where

- S is a finite nonempty set of states [2],
- $R \subseteq S \times S$ and, for each $s \in S$, there is $s' \in S$ such that $(s, s') \in R$,
- L is a labeling function $S \to 2^{\mathcal{A}}$.

Given a model structure $\mathcal{M} = (S, R, L)$, a *path* π_{s_i} starting from s_i of $\mathcal{M}$ is an infinite sequence of states $\pi_{s_i} = (s_i, s_{i+1}s_{i+2}, \dots)$, where for each j ($0 \leq i \leq j$), $(s_j, s_{j+1}) \in R$. By $s' \in \pi_{s_i}$ we mean that s' is a state in the path π_{s_i}. A state $s \in S$ is *initial* if for any state $s' \in S$, there is a path π_s s.t $s' \in \pi_s$. If s_0 is an initial state of $\mathcal{M}$, then we denote this model structure $\mathcal{M}$ as (S, R, L, s_0).

For a given model structure $\mathcal{M} = (S, R, L, s_0)$ and $s \in S$, the *computation tree* $\text{Tr}_n^{\mathcal{M}}(s)$ of $\mathcal{M}$ (or simply $\text{Tr}_n(s)$), that has depth n and is rooted at s, is recursively defined as (Browne, Clarke, and Grumberg 1988), for $n \geq 0$,

- $\text{Tr}_0(s)$ consists of a single node s with label s.
- $\text{Tr}_{n+1}(s)$ has as its root a node m with label s, and if $(s, s') \in R$ then the node m has a subtree $\text{Tr}_n(s')$.

A K-*structure* (or K-*interpretation*) is a model structure $\mathcal{M} = (S, R, L, s_0)$ associating with a state $s \in S$, which is written as $(\mathcal{M}, s)$ for convenience in the following. In the case $s = s_0$ is an initial state of $\mathcal{M}$, the K-structure is *initial*.

Syntax and semantics of CTL

In the following we briefly review the basic syntax and semantics of the CTL (Clarke, Emerson, and Sistla 1986). The *signature* of the language $\mathcal{L}$ of CTL includes:

- a finite set of Boolean variables, called *atoms* of $\mathcal{L}$: $\mathcal{A}$;
- constant symbols: $\bot$ and $\top$;
- the classical connectives: $\vee$ and $\neg$;
- the path quantifiers: A and E;
- the temporal operators: X, F, G U and W, that means 'neXt state', 'some Future state', 'all future states (Globally)', 'Until' and 'Unless', respectively;
- parentheses: (and).

[2] Indeed, every state is identified by a configuration of atoms i.e., which holds in that state.

The *(existential normal form or ENF in short) formulas* of $\mathcal{L}$ are inductively defined via a Backus Naur form:

$$\phi ::= \bot \mid \top \mid p \mid \neg\phi \mid \phi \vee \phi \mid \text{EX}\phi \mid \text{EG}\phi \mid \text{E}[\phi \text{ U } \phi] \quad (1)$$

where $p \in \mathcal{A}$. The formulas $\phi \wedge \psi$ and $\phi \rightarrow \psi$ are defined in a standard manner of propositional logic. The other form formulas of $\mathcal{L}$ are abbreviated using the forms of (1).

We are now in the position to define the semantics of $\mathcal{L}$. Let $\mathcal{M} = (S, R, L, s_0)$ be a model structure, $s \in S$ and ϕ a formula of $\mathcal{L}$. The *satisfiability* relationship between $(\mathcal{M}, s)$ and ϕ, written $(\mathcal{M}, s) \models \phi$, is inductively defined on the structure of ϕ as follows:

- $(\mathcal{M}, s) \not\models \bot$ and $(\mathcal{M}, s) \models \top$;
- $(\mathcal{M}, s) \models p$ iff $p \in L(s)$;
- $(\mathcal{M}, s) \models \phi_1 \vee \phi_2$ iff $(\mathcal{M}, s) \models \phi_1$ or $(\mathcal{M}, s) \models \phi_2$;
- $(\mathcal{M}, s) \models \neg\phi$ iff $(\mathcal{M}, s) \not\models \phi$;
- $(\mathcal{M}, s) \models \text{EX}\phi$ iff $(\mathcal{M}, s_1) \models \phi$ for some $s_1 \in S$ and $(s, s_1) \in R$;
- $(\mathcal{M}, s) \models \text{EG}\phi$ iff $\mathcal{M}$ has a path $(s_1 = s, s_2, \ldots)$ such that $(\mathcal{M}, s_i) \models \phi$ for each $i \geq 1$;
- $(\mathcal{M}, s) \models \text{E}[\phi_1 \text{U} \phi_2]$ iff $\mathcal{M}$ has a path $(s_1 = s, s_2, \ldots)$ such that, for some $i \geq 1$, $(\mathcal{M}, s_i) \models \phi_2$ and $(\mathcal{M}, s_j) \models \phi_1$ for each $1 \leq j < i$.

Similar to the work in (Browne, Clarke, and Grumberg 1988; Bolotov 1999), only initial K-structures are considered to be candidate models in the following, unless otherwise noted. Formally, an initial K-structure $\mathcal{K}$ is a *model* of a formula ϕ whenever $\mathcal{K} \models \phi$. We denote *Mod*(ϕ) the set of models of ϕ. The formula ϕ is *satisfiable* if *Mod*(ϕ) $\neq \emptyset$. Given two formulas ϕ_1 and ϕ_2, $\phi_1 \models \phi_2$ we mean *Mod*(ϕ_1) $\subseteq$ *Mod*(ϕ_2), and by $\phi_1 \equiv \phi_2$, we mean $\phi_1 \models \phi_2$ and $\phi_2 \models \phi_1$. In this case ϕ_1 is *equivalent* to ϕ_2. The set of atoms occurring in ϕ_1, is denoted by *Var*(ϕ_1). ϕ_1 is V*-irrelevant*, written IR(ϕ_1, V), if there is a formula ψ with *Var*(ψ) $\cap V = \emptyset$ such that $\phi_1 \equiv \psi$.

The normal form of CTL

It has proved that any CTL formula φ can be transformed into a set T_φ of $\text{SNF}^g_{\text{CTL}}$ (Separated Normal Form with Global Clauses for CTL) clauses in polynomial time such that φ is satisfiable iff T_φ is satisfiable (Zhang, Hustadt, and Dixon 2008). An important difference between CTL formulae and $\text{SNF}^g_{\text{CTL}}$ is that $\text{SNF}^g_{\text{CTL}}$ is an extension of the syntax of CTL to use indices. These indices can be used to preserve a particular path context. The language of $\text{SNF}^g_{\text{CTL}}$ clauses is defined over an extension of CTL. That is the language is based on: (1) the language of CTL; (2) a propositional constant **start**; (3) a countably infinite index set Ind; and (4) temporal operators: $\text{E}_{\langle ind\rangle}\text{X}$, $\text{E}_{\langle ind\rangle}\text{F}$, $\text{E}_{\langle ind\rangle}\text{G}$, $\text{E}_{\langle ind\rangle}\text{U}$ and $\text{E}_{\langle ind\rangle}\text{W}$.

Before talked about the sematic of this language, we introduce the $\text{SNF}^g_{\text{CTL}}$ clauses at first. A $\text{SNF}^g_{\text{CTL}}$ clauses is a formula with one of following forms:

$$\text{AG}(\textbf{start} \supset \bigvee_{j=1}^{k} m_j) \qquad (initial\ clause)$$

$$\text{AG}(true \supset \bigvee_{j=1}^{k} m_j) \qquad (global\ clause)$$

$$\text{AG}(\bigwedge_{i=1}^{n} l_i \supset \text{AX} \bigvee_{j=1}^{k} m_j) \qquad (\text{A} - step\ clause)$$

$$\text{AG}(\bigwedge_{i=1}^{n} l_i \supset \text{E}_{\langle ind\rangle}\text{X} \bigvee_{j=1}^{k} m_j) \qquad (\text{E} - step\ clause)$$

$$\text{AG}(\bigwedge_{i=1}^{n} l_i \supset \text{AF}l) \qquad (\text{A} - sometime\ clause)$$

$$\text{AG}(\bigwedge_{i=1}^{n} l_i \supset \text{E}_{\langle ind\rangle}\text{F}l) \qquad (\text{E} - sometime\ clause).$$

where $k \geq 0$, $n > 0$, **start** is a propositional constant, l_i ($1 \leq i \leq n$), m_j ($1 \leq j \leq k$) and l are literals, that is atomic propositions or their negation and ind is an element of Ind (Ind is a countably infinite index set). By clause we mean the classical clause or the $\text{SNF}^g_{\text{CTL}}$ clause unless explicitly stated.

Formulae of $\text{SNF}^g_{\text{CTL}}$ over $\mathcal{A}$ are interpreted in Ind-model structure $\mathcal{M} = (S, R, L, [\_], s_0)$, where S, R, L and s_0 is the same as our model structure talked above and $[\_] : \text{Ind} \rightarrow 2^{(S*S)}$ maps every index $ind \in \text{Ind}$ to a successor function $[ind]$ which is a functional relation on S and a subset of the binary accessibility relation R, such that for every $s \in S$ there exists exactly a state $s' \in S$ such that $(s, s') \in [ind]$ and $(s, s') \in R$. An infinite path $\pi^{\langle ind\rangle}_{s_i}$ is an infinite sequence of states $s_i, s_{i+1}, s_{i+2}, \ldots$ such that for every $j \geq i$, $(s_j, s_{j+1}) \in [ind]$.

Similarly, an *Ind-structure* is an Ind-model structure $\mathcal{M} = (S, R, L, [\_], s_0)$ associating with a state $s \in S$, which is written as $(\mathcal{M}, s)$ for convenience in the following. In the case s is an initial state of $\mathcal{M}$, the Ind-structure is *initial*.

The semantics of $\text{SNF}^g_{\text{CTL}}$ is then defined as shown next as an extension of the semantics of CTL. Let φ and ψ be two $\text{SNF}^g_{\text{CTL}}$ formulae and $\mathcal{M} = (S, R, L, [\_], s_0)$ be an Ind-model structure, the relation "$\models$" between $\text{SNF}^g_{\text{CTL}}$ formulae and $\mathcal{M}$ is defined recursively as follows:

- $(\mathcal{M}, s_i) \models$ **start** iff $s_i = s_0$;
- $(\mathcal{M}, s_i) \models \text{E}_{\langle ind\rangle}\text{X}\psi$ iff for the path $\pi^{\langle ind\rangle}_{s_i}$, $(\mathcal{M}, s_{i+1}) \models \psi$;
- $(\mathcal{M}, s_i) \models \text{E}_{\langle ind\rangle}\text{G}\psi$ iff for every $s_j \in \pi^{\langle ind\rangle}_{s_i}$, $(\mathcal{M}, s_j) \models \psi$;
- $(\mathcal{M}, s_i) \models \text{E}_{\langle ind\rangle}[\varphi\text{U}\psi]$ iff there exists $s_j \in \pi^{\langle ind\rangle}_{s_i}$ such that $(\mathcal{M}, s_j) \models \psi$ and for every $s_k \in \pi^{\langle ind\rangle}_{s_i}$, if $i \leq k < j$, then $(\mathcal{M}, s_k) \models \varphi$;
- $(\mathcal{M}, s_i) \models \text{E}_{\langle ind\rangle}\text{F}\psi$ iff $(\mathcal{M}, s_i) \models \text{E}_{\langle ind\rangle}[\top\text{U}\psi]$;

- $(\mathcal{M}, s_i) \models \mathrm{E}_{\langle ind\rangle}[\varphi \mathrm{W} \psi]$ iff $(\mathcal{M}, s_i) \models \mathrm{E}_{\langle ind\rangle}\mathrm{G}\varphi$ or $(\mathcal{M}, s_i) \models \mathrm{E}_{\langle ind\rangle}[\varphi \mathrm{U} \psi]$.

The semantics of the remaining operators is analogous to that given previously but in the extended Ind-model structure $\mathcal{M} = (S, R, L, [\_], s_0)$.

A $\mathrm{SNF}^g_{\mathrm{CTL}}$ formula φ is satisfiable, iff for some Ind-model structure $\mathcal{M}$ there is $(\mathcal{M}, s_0) \models \varphi$, and unsatisfiable otherwise. And if $(\mathcal{M}, s_0) \models \varphi$ then $(\mathcal{M}, s_0)$ is called an Ind-model of φ, and we say that $(\mathcal{M}, s_0)$ satisfies φ. By $T \wedge \varphi$ we mean $\bigwedge_{\psi \in T} \psi \wedge \varphi$, where T is a set of formulae. Other terminologies are similar with those in subsection of CTL.

Forgetting in CTL

In this section, we will define the forgetting in CTL by V-bisimulation constructed via a set-based bisimulation. Besides, some properties of forgetting are also explored. For convenience, let $\mathcal{M} = (S, R, L, s_0)$, $\mathcal{M}' = (S', R', L', s'_0)$ and $\mathcal{K}_i = (\mathcal{M}_i, s_i)$ with $\mathcal{M}_i = (S_i, R_i, L_i, s^i_0)$, $s_i \in S_i$ (is any state in S_i) and $i \in \mathbb{N}$.

Set-based bisimulation

To present a formal definition of forgetting, we need the concept of V-bisimulation. Inspired by the notion of bisimulation in (Browne, Clarke, and Grumberg 1988), we define the relations $\mathcal{B}_0, \mathcal{B}_1, \ldots$ between K-structures on V as follows: let $\mathcal{K}_i = (\mathcal{M}_i, s_i)$ with $i \in \{1, 2\}$,

- $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_0$ if $L_1(s_1) - V = L_2(s_2) - V$;
- for $n \geq 0$, $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_{n+1}$ if:
  - $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_0$,
  - for every $(s_1, s'_1) \in R_1$, there is a $(s_2, s'_2) \in R_2$ such that $(\mathcal{K}'_1, \mathcal{K}'_2) \in \mathcal{B}_n$, and
  - for every $(s_2, s'_2) \in R_2$, there is a $(s_1, s'_1) \in R_1$ such that $(\mathcal{K}'_1, \mathcal{K}'_2) \in \mathcal{B}_n$,

  where $\mathcal{K}'_i = (\mathcal{M}_i, s'_i)$ with $i \in \{1, 2\}$.

Now, we define the notion of V-bisimulation between K-structures:

Definition 1 (V-bisimulation) *Let $V \subseteq \mathcal{A}$. Given two K-structures $\mathcal{K}_1$ and $\mathcal{K}_2$ are V-*bisimilar*, denoted $\mathcal{K}_1 \leftrightarrow_V \mathcal{K}_2$ if and only if $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_i$ for all $i \geq 0$. Moreover, two paths $\pi_i = (s_{i,1}, s_{i,2}, \ldots)$ of $\mathcal{M}_i$ with $i \in \{1, 2\}$ are V-*bisimilar *if $\mathcal{K}_{1,j} \leftrightarrow_V \mathcal{K}_{2,j}$ for every $j \geq 1$ where $\mathcal{K}_{i,j} = (\mathcal{M}_i, s_{i,j})$.*

It is apparent that $\leftrightarrow_V$ is a binary relation. In the sequel, we abbreviate $\mathcal{K}_1 \leftrightarrow_V \mathcal{K}_2$ by $s_1 \leftrightarrow_V s_2$ whenever the underlying model structures of states s_1 and s_2 are clear from the context.

Lemma 1 *The relation $\leftrightarrow_V$ is an equivalence relation.*

Besides, we have the following properties:

Proposition 1 *Let $i \in \{1, 2\}$, $V_1, V_2 \subseteq \mathcal{A}$, s'_is be two states and π'_is be two paths, and $\mathcal{K}_i = (\mathcal{M}_i, s_i)$ $(i = 1, 2, 3)$ be K-structures such that $\mathcal{K}_1 \leftrightarrow_{V_1} \mathcal{K}_2$ and $\mathcal{K}_2 \leftrightarrow_{V_2} \mathcal{K}_3$. Then:*

(i) $s'_1 \leftrightarrow_{V_i} s'_2$ $(i = 1, 2)$ implies $s'_1 \leftrightarrow_{V_1 \cup V_2} s'_2$;

(ii) $\pi'_1 \leftrightarrow_{V_i} \pi'_2$ $(i = 1, 2)$ implies $\pi'_1 \leftrightarrow_{V_1 \cup V_2} \pi'_2$;

(iii) for each path π_{s_1} of $\mathcal{M}_1$ there is a path π_{s_2} of $\mathcal{M}_2$ such that $\pi_{s_1} \leftrightarrow_{V_1} \pi_{s_2}$, and vice versa;

(iv) $\mathcal{K}_1 \leftrightarrow_{V_1 \cup V_2} \mathcal{K}_3$;

(v) If $V_1 \subseteq V_2$ then $\mathcal{K}_1 \leftrightarrow_{V_2} \mathcal{K}_2$.

Intuitively, if two K-structures are V-bisimilar, then they satisfy the same formula φ that dose not contain any atoms in V, *i.e.* $\mathrm{IR}(\varphi, V)$.

Theorem 1 *Let $V \subseteq \mathcal{A}$, $\mathcal{K}_i$ $(i = 1, 2)$ be two K-structures such that $\mathcal{K}_1 \leftrightarrow_V \mathcal{K}_2$ and ϕ a formula with IR(ϕ, V). Then $\mathcal{K}_1 \models \phi$ if and only if $\mathcal{K}_2 \models \phi$.*

Let $V \subseteq \mathcal{A}$, $\mathcal{M}_i$ $(i = 1, 2)$ be model structures. A computation tree $\mathrm{Tr}_n(s_1)$ of $\mathcal{M}_1$ is V-*bisimilar* to a computation tree $\mathrm{Tr}_n(s_2)$ of $\mathcal{M}_2$, written $(\mathcal{M}_1, \mathrm{Tr}_n(s_1)) \leftrightarrow_V (\mathcal{M}_2, \mathrm{Tr}_n(s_2))$ (or simply $\mathrm{Tr}_n(s_1) \leftrightarrow_V \mathrm{Tr}_n(s_2)$), if

- $L_1(s_1) - V = L_2(s_2) - V$,
- for every subtree $\mathrm{Tr}_{n-1}(s'_1)$ of $\mathrm{Tr}_n(s_1)$, $\mathrm{Tr}_n(s_2)$ has a subtree $\mathrm{Tr}_{n-1}(s'_2)$ such that $\mathrm{Tr}_{n-1}(s'_1) \leftrightarrow_V \mathrm{Tr}_{n-1}(s'_2)$, and vice versa.

Note that the last condition in the above definition hold trivially for $n = 0$.

Proposition 2 *Let $V \subseteq \mathcal{A}$ and $(\mathcal{M}_i, s_i)$ $(i = 1, 2)$ be two K-structures. Then*

$(s_1, s_2) \in \mathcal{B}_n$ *iff $Tr_j(s_1) \leftrightarrow_V Tr_j(s_2)$ for every $0 \leq j \leq n$.*

This means that $\mathrm{Tr}_j(s_1) \leftrightarrow_V \mathrm{Tr}_j(s_2)$ for all $j \geq 0$ if $s_1 \leftrightarrow_V s_2$, otherwise there is some k such that $\mathrm{Tr}_k(s_1)$ and $\mathrm{Tr}_k(s_2)$ are not V-bisimilar.

Proposition 3 *Let $V \subseteq \mathcal{A}$, $\mathcal{M}$ be a model structure and $s, s' \in S$ such that $s \not\leftrightarrow_V s'$. There exists a least k such that $Tr_k(s)$ and $Tr_k(s')$ are not V-bisimilar.*

In this case the model structure $\mathcal{M}$ is called V-*distinguishable* (by states s and s' at the least depth k), which is denoted by $\mathrm{dis}_V(\mathcal{M}, s, s', k)$. The V-*characterization number* of $\mathcal{M}$, written $ch(\mathcal{M}, V)$, is defined as

$$ch(\mathcal{M}, V) = \begin{cases} \max\{k \mid s, s' \in S \text{ and } \mathrm{dis}_V(\mathcal{M}, s, s', k)\}, \\ \qquad\qquad \mathcal{M} \text{ is } V\text{-distinguishable}; \\ \min\{k \mid \mathcal{B}_k = \mathcal{B}_{k+1}\}, \qquad \text{otherwise.} \end{cases}$$

Similar with the V-bisimulation between K-structures, we define the $\langle V, I\rangle$-bisimulation between Ind-structures as follows:

Definition 2 ($\langle V, I\rangle$-*bisimulation*) *Let $\mathcal{M}_i = (S_i, R_i, L_i, [\_]_i, s^i_0)$ with $i \in \{1, 2\}$ be two Ind-structures, V be a set of atoms and $I \subseteq Ind$. The $\langle V, I\rangle$-bisimulation between initial Ind-structures is a set $\beta_{\langle V,I\rangle}$ that satisfy $((\mathcal{M}_1, s^1_0), (\mathcal{M}_2, s^2_0)) \in \beta_{\langle V,I\rangle}$ if and only if $(\mathcal{M}_1, s^1_0) \leftrightarrow_V (\mathcal{M}_2, s^2_0)$ and $\forall j \notin I$ there is*

(i) $\forall (s, s_1) \in [j]_1$ there is $(s', s'_1) \in [j]_2$ such that $s \leftrightarrow_V s'$ and $s_1 \leftrightarrow_V s'_1$, and

(ii) $\forall (s', s'_1) \in [j]_2$ there is $(s, s_1) \in [j]_1$ such that $s \leftrightarrow_V s'$ and $s_1 \leftrightarrow_V s'_1$.

Apparently, this definition is similar with our concept V-bisimulation. Similar with the Proposition 1, we have:

Proposition 4 *Let $i \in \{1,2\}$, $V_1, V_2 \subseteq \mathcal{A}$, $I_1, I_2 \subseteq Ind$ and $\mathcal{K}_i = (\mathcal{M}_i, s_0^i)$ $(i = 1,2,3)$ be Ind-structures such that $\mathcal{K}_1 \leftrightarrow_{\langle V_1, I_1\rangle} \mathcal{K}_2$ and $\mathcal{K}_2 \leftrightarrow_{\langle V_2, I_2\rangle} \mathcal{K}_3$. Then:*

(i) $\mathcal{K}_1 \leftrightarrow_{\langle V_1 \cup V_2, I_1 \cup I_2\rangle} \mathcal{K}_3$*;*

(ii) If $V_1 \subseteq V_2$ *and* $I_1 \subseteq I_2$ *then* $\mathcal{K}_1 \leftrightarrow_{\langle V_2, I_2\rangle} \mathcal{K}_2$.

Characterization of Initial K-structure

In order to introduce our notion of forgetting, and to compute strongest necessary and weakest sufficient conditions, we need a formula that captures the initial K-structure on V syntactically. We call such formula as characterizing formula. In the following, we present such a characterization.

Given a set $V \subseteq \mathcal{A}$, we define a formula φ of V (that is $Var(\varphi) \subseteq V$) in CTL that describes a computation tree.

Definition 3 *Let $V \subseteq \mathcal{A}$, $\mathcal{M} = (S, R, L, s_0)$ be a model structure and $s \in S$. The* characterizing formula *of the computation tree $Tr_n(s)$ on V, written $\mathcal{F}_V(Tr_n(s))$, is defined recursively as:*

$$\mathcal{F}_V(Tr_0(s)) = \bigwedge_{p \in V \cap L(s)} p \wedge \bigwedge_{q \in V - L(s)} \neg q,$$

$$\mathcal{F}_V(Tr_{k+1}(s)) = \bigwedge_{(s,s') \in R} \mathrm{EX}\mathcal{F}_V(Tr_k(s')) \wedge \mathrm{AX}\left(\bigvee_{(s,s') \in R} \mathcal{F}_V(Tr_k(s'))\right) \wedge \mathcal{F}_V(Tr_0(s))$$

for $k \geq 0$.

The characterizing formula of a computation tree formally exhibit the content of each node on V (i.e., atoms that are true at this node if they are in V, false otherwise) and the temporal relation between states recursively. The following result shows that the V-bisimulation between two computation trees imply the semantic equivalence of the corresponding characterizing formulas.

Lemma 2 *Let $V \subseteq \mathcal{A}$, $\mathcal{M}$ and $\mathcal{M}'$ be two model structures, $s \in S$, $s' \in S'$ and $n \geq 0$. If $Tr_n(s) \leftrightarrow_{\overline{V}} Tr_n(s')$, then $\mathcal{F}_V(Tr_n(s)) \equiv \mathcal{F}_V(Tr_n(s'))$.*

Let $s' = s$, it shows that for any formula φ of V, if φ is a characterizing formula of $\mathrm{Tr}_n(s)$ then $\varphi \equiv \mathcal{F}_V(\mathrm{Tr}_n(s))$.

Let $V \subseteq \mathcal{A}$, $\mathcal{K} = (\mathcal{M}, s_0)$ be an initial K-structure and $T(s') = \mathcal{F}_V(\mathrm{Tr}_c(s'))$. The *characterizing formula* of $\mathcal{K}$ on V, written $\mathcal{F}_V(\mathcal{M}, s_0)$ (or $\mathcal{F}_V(\mathcal{K})$ in short), is defined as the following formula:

$$\mathcal{F}_V(\mathrm{Tr}_c(s_0)) \wedge \bigwedge_{s \in S} \mathrm{AG}\left(\mathcal{F}_V(\mathrm{Tr}_c(s)) \rightarrow \bigwedge_{(s,s') \in R} \mathrm{EX}T(s') \wedge \mathrm{AX} \bigvee_{(s,s') \in R} T(s')\right)$$

where $c = ch(\mathcal{M}, V)$. It is apparent that $\mathrm{IR}(\mathcal{F}_V(\mathcal{M}, s_0), \overline{V})$.

The following example show how to compute characterizing formula:

Example 1 Let $V = \{sr\}$ and $\overline{V} = \{sl, se, lc, le\}$ ($\mathcal{A} = V \cup \{sr\}$), and $\mathcal{M}$ is as illustrated in Figure 1. We have $\mathrm{Tr}_0(s_0) \not\leftrightarrow_{\overline{V}} \mathrm{Tr}_0(s_1)$ and $\mathrm{Tr}_0(s_0) \not\leftrightarrow_{\overline{V}} \mathrm{Tr}_0(s_2)$, then $\mathrm{dis}_{\overline{V}}(\mathcal{M}, s_0, s_1, 0)$ and $\mathrm{dis}_{\overline{V}}(\mathcal{M}, s_0, s_2, 0)$. Besides, it is easy checking that $s_1 \leftrightarrow_{\overline{V}} s_2$ since they have the same direct successor s_0. Hence, $ch(\mathcal{M}, \overline{V}) = 0$. Therefore, we have:

$\mathcal{F}_V(\mathrm{Tr}_0(s_0)) = \neg sr$

$\mathcal{F}_V(\mathrm{Tr}_0(s_1)) = \mathcal{F}_V(\mathrm{Tr}_0(s_2)) = sr$, and then

$\mathcal{F}_V(\mathcal{M}, s_0) = \neg sr \wedge \mathrm{AG}(\neg sr \rightarrow \mathrm{AX}sr) \wedge \mathrm{AG}(sr \rightarrow \mathrm{AX}\neg sr)$.

By the following theorem, we also have that given $V \subseteq \mathcal{A}$, the characterizing formula of an initial K-structure is unique (up to semantic equivalence) which describes this initial K-structure on V.

Theorem 2 *Given $V \subseteq \mathcal{A}$, let $\mathcal{M} = (S, R, L, s_0)$ and $\mathcal{M}' = (S', R', L', s_0')$ be two model structures. Then,*

(i) $(\mathcal{M}', s_0') \models \mathcal{F}_V(\mathcal{M}, s_0)$ *iff* $(\mathcal{M}, s_0) \leftrightarrow_{\overline{V}} (\mathcal{M}', s_0')$*;*

(ii) $s_0 \leftrightarrow_{\overline{V}} s_0'$ *implies* $\mathcal{F}_V(\mathcal{M}, s_0) \equiv \mathcal{F}_V(\mathcal{M}', s_0')$.

Moreover, we know that any initial K-structure can be described as a CTL formula from the definition of characterizing formula. Then,

Lemma 3 *Let φ be a formula. We have*

$$\varphi \equiv \bigvee_{(\mathcal{M}, s_0) \in Mod(\varphi)} \mathcal{F}_{\mathcal{A}}(\mathcal{M}, s_0). \tag{2}$$

It follows that any CTL formula can rewritten in the form of a characterizing formula (in the form of a disjunction of characterizing formulas; each corresponds to a single model.

Semantic Properties of Forgetting in CTL

In this subsection we will give the definition of forgetting in CTL and study it's semantic properties. We will first show via a representation theorem that our definition of forgetting correspond to the readily existing notion of forgetting which is characterised by several desirable properties (also called postulates) suggested in (Zhang and Zhou 2009). Next, we discuss various additional semantic properties of forgetting.

Now, we give the formal definition of forgetting in CTL from the semantic point view.

Definition 4 (Forgetting) *Let $V \subseteq \mathcal{A}$ and ϕ a formula. A formula ψ with $Var(\psi) \cap V = \emptyset$ is a* result of forgetting V from ϕ*, if*

$$Mod(\psi) = \{\mathcal{K} \text{ is initial} \mid \exists \mathcal{K}' \in Mod(\phi) \ \&\ \mathcal{K}' \leftrightarrow_V \mathcal{K}\}. \tag{3}$$

Note that if both ψ and ψ' are results of forgetting V from ϕ, then $Mod(\psi) = Mod(\psi')$, i.e., ψ and ψ' have the same models. In the sense of equivalence, the forgetting result is unique (up to equivalence). By Lemma 3, such a formula always exists, which is equivalent to

$$\bigvee_{\mathcal{K} \in \{\mathcal{K}' \mid \exists \mathcal{K}'' \in Mod(\phi) \text{ and } \mathcal{K}'' \leftrightarrow_V \mathcal{K}'\}} \mathcal{F}_{\overline{V}}(\mathcal{K}).$$

For this reason, the forgetting result is denoted by $\mathrm{F}_{\mathrm{CTL}}(\phi, V)$.

Assume you are given a formula φ, and φ' is the formula after forgetting V, then we have the following desired properties, also called *postulates* of forgetting (Zhang and Zhou 2009).

- Weakening (**W**): $\varphi \models \varphi'$;
- Positive Persistence (**PP**): Given $\eta \in$ CTL if IR(η, V) and $\varphi \models \eta$, then $\varphi' \models \eta$;
- Negative Persistence (**NP**): Given $\eta \in$ CTL if IR(η, V) and $\varphi \not\models \eta$, then $\varphi' \not\models \eta$;
- Irrelevance (**IR**): IR(φ', V).

Intuitive enough, the postulate (**W**) says, forgetting weakens the original formula. (**PP**) and (**NP**) correspond to the fact that so long as forgotten atoms V are irrelevant to the remaining positive and the negative information, respectively, they do not affect such information. (**IR**) states that forgotten atoms V are not relevant for the final formula (i.e., φ' is V-irrelevant).

Theorem 3 (Representation Theorem) *Let φ, φ' and ϕ be CTL formulas and $V \subseteq \mathcal{A}$. Then the following statements are equivalent:*

(i) $\varphi' \equiv \mathrm{F}_{\mathrm{CTL}}(\varphi, V)$,
(ii) $\varphi' \equiv \{\phi | \varphi \models \phi$ *and IR*$(\phi, V)\}$,
*(iii) Postulates (**W**), (**PP**), (**NP**) and (**IR**) hold.*

The above theorem says that, the notion of forgetting we defined, satisfies (and entailed by) the four postulates of forgetting. Moreover, CTL is closed under our definition of forgetting, i.e., for any CTL formula the result of forgetting is also a CTL formula.

Lemma 4 *Let φ and α be two CTL formulae and $q \in$ Var$(\varphi) \cup$ Var(α). Then* $\mathrm{F}_{CTL}(\varphi \wedge (q \leftrightarrow \alpha), q) \equiv \varphi$.

Proposition 5 *Given a formula $\varphi \in$ CTL, V a set of atoms and p an atom such that $p \notin V$. Then,*

$$\mathrm{F}_{\mathrm{CTL}}(\varphi, \{p\} \cup V) \equiv \mathrm{F}_{\mathrm{CTL}}(\mathrm{F}_{\mathrm{CTL}}(\varphi, p), V).$$

This means that the result of forgetting V from φ can be obtained by forgetting atoms in V one by one. Moreover, the order of atoms does not matter (commutativity), which follows from Proposition 5.

Corollary 4 (Commutativity) *Let φ be a formula and $V_i \subseteq \mathcal{A}$ $(i = 1, 2)$. Then:*

$$\mathrm{F}_{\mathrm{CTL}}(\varphi, V_1 \cup V_2) \equiv \mathrm{F}_{\mathrm{CTL}}(\mathrm{F}_{\mathrm{CTL}}(\varphi, V_1), V_2).$$

The following results, which are satisfied in both classical propositional logic and modal logic **S5** (Zhang and Zhou 2009), further illustrate other essential semantic properties of forgetting.

Proposition 6 *Let φ, φ_i, ψ_i $(i = 1, 2)$ be formulas and $V \subseteq \mathcal{A}$. We have*

(i) $\mathrm{F}_{\mathrm{CTL}}(\varphi, V)$ *is satisfiable iff φ is;*
(ii) If $\varphi_1 \equiv \varphi_2$, *then* $\mathrm{F}_{\mathrm{CTL}}(\varphi_1, V) \equiv \mathrm{F}_{\mathrm{CTL}}(\varphi_2, V)$;
(iii) If $\varphi_1 \models \varphi_2$, *then* $\mathrm{F}_{\mathrm{CTL}}(\varphi_1, V) \models \mathrm{F}_{\mathrm{CTL}}(\varphi_2, V)$;
(iv) $\mathrm{F}_{\mathrm{CTL}}(\psi_1 \vee \psi_2, V) \equiv \mathrm{F}_{\mathrm{CTL}}(\psi_1, V) \vee \mathrm{F}_{\mathrm{CTL}}(\psi_2, V)$;
(v) $\mathrm{F}_{\mathrm{CTL}}(\psi_1 \wedge \psi_2, V) \models \mathrm{F}_{\mathrm{CTL}}(\psi_1, V) \wedge \mathrm{F}_{\mathrm{CTL}}(\psi_2, V)$;

Another interesting result is that the forgetting of $QT\varphi$ ($Q \in \{\mathrm{E}, \mathrm{A}\}$, $T \in \{\mathrm{F}, \mathrm{X}\}$) on $V \subseteq \mathcal{A}$ can be computed by $QT\mathrm{F}_{\mathrm{CTL}}(\varphi, V)$. This gives us a convenient method to compute forgetting, since we can push the forgetting operator to a subformula without affecting the semantics, and rather compute the forgetting on this subformula.

Proposition 7 (Homogeneity) *Let $V \subseteq \mathcal{A}$ and $\phi \in$ CTL,*

(i) $\mathrm{F}_{\mathrm{CTL}}(\mathrm{AX}\phi, V) \equiv \mathrm{AX}\mathrm{F}_{\mathrm{CTL}}(\phi, V)$.
(ii) $\mathrm{F}_{\mathrm{CTL}}(\mathrm{EX}\phi, V) \equiv \mathrm{EX}\mathrm{F}_{\mathrm{CTL}}(\phi, V)$.
(iii) $\mathrm{F}_{\mathrm{CTL}}(\mathrm{AF}\phi, V) \equiv \mathrm{AF}\mathrm{F}_{\mathrm{CTL}}(\phi, V)$.
(iv) $\mathrm{F}_{\mathrm{CTL}}(\mathrm{EF}\phi, V) \equiv \mathrm{EF}\mathrm{F}_{\mathrm{CTL}}(\phi, V)$.

Complexity Results

In the following, we outline the computational complexity of the various tasks regarding the forgetting in CTL and its popular fragment $\mathrm{CTL}_{\mathrm{AF}}$. It turns out that the model-checking on forgetting without any restriction is NP-complete.

Proposition 8 (Model Checking on Forgetting) *Let $(\mathcal{M}, s_0)$ be an initial* K*-structure, φ be a CTL formula and V a set of atoms. Deciding whether $(\mathcal{M}, s_0)$ is a model of* $\mathrm{F}_{CTL}(\varphi, V)$ *is NP-complete.*

The fragment of CTL, in which each formula contains only AF temporal connective correspond to specifications that are desired to hold in all branches eventually. Such properties are of special interest in concurrent systems e.g., mutual exclusion and waiting events (Baier and Katoen 2008). In the following, we report various complexity results concerning forgetting and the logical entailment in this fragment.

Theorem 5 (Entailment on Forgetting) *Let φ and ψ be two $\mathrm{CTL}_{\mathrm{AF}}$ formulas and V a set of atoms. Then, results:*

(i) deciding $\mathrm{F}_{CTL}(\varphi, V) \models^{?} \psi$ *is co-NP-complete,*
(ii) deciding $\psi \models^{?} \mathrm{F}_{CTL}(\varphi, V)$ *is Π_2^P-complete,*
(iii) deciding $\mathrm{F}_{CTL}(\varphi, V) \models^{?} \mathrm{F}_{CTL}(\psi, V)$ *is Π_2^P-complete.*

The following results follow from Theorem 5 and extends them to semantic equivalence.

Corollary 6 *Let φ and ψ be two $\mathrm{CTL}_{\mathrm{AF}}$ formulas and V a set of atoms. Then*

(i) deciding $\psi \equiv^{?} \mathrm{F}_{CTL}(\varphi, V)$ *is Π_2^P-complete,*
(ii) deciding $\mathrm{F}_{CTL}(\varphi, V) \equiv^{?} \varphi$ *is co-NP-complete,*
(iii) deciding $\mathrm{F}_{CTL}(\varphi, V) \equiv^{?} \mathrm{F}_{CTL}(\psi, V)$ *is Π_2^P-complete.*

Strongest Necessary and Weakest Sufficient Conditions

In this section, we will give the definitions of strongest necessary and weakest sufficient conditions (SNC and WSC, respectively) of a specification in CTL and show that they can be obtained by forgetting under a given initial K-structure and set V of atoms. We first start with defining these notions of an atomic proposition.

Definition 5 (sufficient and necessary condition) *Let ϕ be a formula (or an initial* K*-structure), ψ be a formula, $V \subseteq$ Var(ϕ), $q \in$ Var$(\phi) - V$ and Var$(\psi) \subseteq V$.*

- ψ *is a* necessary condition *(NC in short) of* q *on* V *under* ϕ *if* $\phi \models q \rightarrow \psi$.
- ψ *is a* sufficient condition *(SC in short) of* q *on* V *under* ϕ *if* $\phi \models \psi \rightarrow q$.
- ψ *is a* strongest necessary condition *(SNC in short) of* q *on* V *under* ϕ *if it is a NC of* q *on* V *under* ϕ *and* $\phi \models \psi \rightarrow \psi'$ *for any NC* ψ' *of* q *on* V *under* ϕ.
- ψ *is a* weakest sufficient condition *(SNC in short) of* q *on* V *under* ϕ *if it is a SC of* q *on* V *under* ϕ *and* $\phi \models \psi' \rightarrow \psi$ *for any SC* ψ' *of* q *on* V *under* ϕ.

Note that if both ψ and ψ' are SNC (WSC) of q on V under ϕ, then $Mod(\psi) = Mod(\psi')$, i.e., ψ and ψ' have the same models. In the sense of equivalence the SNC (WSC) is unique (up to equivalence).

Proposition 9 ***(dual)*** *Let* V, q, φ *and* ψ *are like in Definition 5. The* ψ *is a SNC (WSC) of* q *on* V *under* φ *iff* $\neg\psi$ *is a WSC (SNC) of* $\neg q$ *on* V *under* φ.

This shows that the SNC and WSC are in fact dual conditions. Under this dual property, we can bother ourselves with only one of them e.g., SNC, and WSC can be obtained easily.

In order to generalise Definition 5 to arbitrary formulas, one can replace q (in the definition) by any formula α, and redefine V as a subset of $Var(\alpha) \cup Var(\phi)$. It turns out that the previous notion of SNC and WSC for an atomic proposition can be lifted to any formula, or conversely the SNC and WSC of any formula can be reduced to that of a proposition.

Proposition 10 *Let* Γ *and* α *be two formulas,* $V \subseteq Var(\alpha) \cup Var(\phi)$ *and* q *is a new proposition not in* Γ *and* α. *Then, a formula* φ *of* V *is the SNC (WSC) of* α *on* V *under* Γ *iff it is the SNC (WSC) of* q *on* V *under* $\Gamma' = \Gamma \cup \{q \leftrightarrow \alpha\}$.

The following result establishes the bridge between these two notions which are central to the paper; basically SNC (WSC) and the forgetting in which the former is obtained through the latter.

Theorem 7 *Let* φ *be a formula,* $V \subseteq Var(\varphi)$ *and* $q \in Var(\varphi) - V$.

(i) $\mathrm{F}_{\mathrm{CTL}}(\varphi \wedge q, (Var(\varphi) \cup \{q\}) - V)$ *is a SNC of* q *on* V *under* φ.

(ii) $\neg\mathrm{F}_{\mathrm{CTL}}(\varphi \wedge \neg q, (Var(\varphi) \cup \{q\}) - V)$ *is a WSC of* q *on* V *under* φ.

As aforementioned, since any initial K-structure can be characterized by a CTL formula, we can obtain the SNC (and its dual WSC) of a target property (a formula) under an initial K-structure by forgetting.

Theorem 8 *Let* $\mathcal{K} = (\mathcal{M}, s)$ *be an initial* K*-structure with* $\mathcal{M} = (S, R, L, s_0)$ *on the set* $\mathcal{A}$ *of atoms,* $V \subseteq \mathcal{A}$ *and* $q \in V' = \mathcal{A} - V$. *Then:*

(i) the SNC of q *on* V *under* $\mathcal{K}$ *is* $\mathrm{F}_{\mathrm{CTL}}(\mathcal{F}_{\mathcal{A}}(\mathcal{K}) \wedge q, V')$.

(ii) the WSC of q *on* V *under* $\mathcal{K}$ *is* $\neg\mathrm{F}_{\mathrm{CTL}}(\mathcal{F}_{\mathcal{A}}(\mathcal{K}) \wedge \neg q, V')$.

An Algorithm to Compute Forgetting

Resolution in CTL is a method to decide the satisfiability of a CTL formula. In this part, we will explore a resolution-based method to compute forgetting in CTL. We use the transformation rules Trans(1) to Trans(12) and resolution rules (SRES1), ..., (SRES8), RW1, RW2, (ERES1), (ERES2) in (Zhang, Hustadt, and Dixon 2009).

The key problems of this method include (1) How to fill the gap between CTL and $\mathrm{SNF}^g_{\mathrm{CTL}}$ since there is index for existential quantifier in $\mathrm{SNF}^g_{\mathrm{CTL}}$; and (2) How to eliminate the irrelevant atoms, which we want to forget, in the formula. We will resolve these two problems by $\langle V, I\rangle$-bisimulation and *eliminate* operator respectively. For convenient, we use $V \subseteq \mathcal{A}$ denote the set we want to forget, $V' \subseteq \mathcal{A}$ with $V \cap V' = \emptyset$ the set of atoms introduced in the transformation process below, φ the CTL formula, T_φ be the set of $\mathrm{SNF}^g_{\mathrm{CTL}}$ clause obtained from φ by using transformation rules and $\mathcal{M} = (S, R, L, [\_], s_0)$ unless explicitly stated. Let T, T' be two sets of formulae, I a set of indexes and $V'' \subseteq \mathcal{A}$, by $T \equiv_{\langle V'', I\rangle} T'$ we mean that $\forall(\mathcal{M}, s_0) \in Mod(T)$ there is a $(\mathcal{M}', s'_0)$ such that $(\mathcal{M}, s_0) \leftrightarrow_{\langle V'', I\rangle} (\mathcal{M}', s'_0)$ and $(\mathcal{M}', s'_0) \models T'$ and vice versa.

The algorithm of computing the forgetting in CTL is as Algorithm 1. The main idea of this algorithm is to change the CTL formula into a set of $\mathrm{SNF}^g_{\mathrm{CTL}}$ clauses at first (the Transform process), and then compute all the possible resolutions on the specified set of atoms (the Resolution process). Third, eliminating all the irrelevant atoms which dose not be eliminated by the resolution. We will describe this process, which include *Instantiate*, *Connect* and *Removing_atoms* sub-processes, in detail below. Changing the result obtained before into a CTL formula at last, this will include three sub-processes: *Removing_index* (removing the index in the formula), *Replacing_atoms* (replacing the atoms in V' with an formula) and T_{CTL} (removing the **start** in the formula). To describe our algorithm clearly, we illustrate it with the following example.

Example 2 Let $\varphi = \mathrm{A}((p \wedge q)\mathrm{U}(f \vee m)) \wedge r$ and $V = \{p\}$.

In the following context we will show how to compute the $\mathrm{F}_{\mathrm{CTL}}(\varphi, V)$ step by step using our algorithm.

Input: A CTL formula φ and a set V of atoms
Output: $ERes(\varphi, V)$
1 $T_\varphi = \emptyset$ // the initial set of $\mathrm{SNF}^g_{\mathrm{CTL}}$ clauses of φ ;
2 $V' = \emptyset$ // the set of atoms introduced in the process of transforming φ into $\mathrm{SNF}^g_{\mathrm{CTL}}$ clauses;
3 $T_\varphi, V' \leftarrow$ *Transform*(φ) ;
4 $Res \leftarrow$ *Resolution*(T_φ, V') ;
5 $\mathrm{Inst}_{V'} \leftarrow$ *Instantiate*(Res, V') ;
6 $\mathrm{Com}_{\mathrm{EF}} \leftarrow$ *Connect*$(\mathrm{Inst}_{V'})$;
7 *RemA* $\leftarrow$ *Removing_atoms*$(\mathrm{Com}_{\mathrm{EF}}, \mathrm{Inst}_{V'})$;
8 NI $\leftarrow$ *Removing_index*(*RemA*);
9 Rp $\leftarrow$ *Replacing_atoms*(NI) ;
10 **return** $\bigwedge_{\psi \in \mathrm{Rp}_{\mathrm{CTL}}} \psi$.

Algorithm 1: Computing forgetting based on Resolution

The Transform process

The *Transform* process, denoted as *Transform*(φ), is to transform the CTL formula into a set of $\text{SNF}^g_{\text{CTL}}$ clauses by using the rules Trans(1) to Trans(12) in (Zhang, Hustadt, and Dixon 2009)). The transformation of an arbitrary CTL formula into the set T_φ is a sequence $T_0, T_1, \ldots, T_n = T_\varphi$ of sets of formulae with $T_0 = \{\text{AG}(\textbf{start} \supset p), \text{AG}(p \supset \textbf{simp}(\textbf{nnf}(\varphi)))\}$ such that for every i ($0 \leq i < n$), $T_{i+1} = (T_i \setminus \{\psi\}) \cup R_i$ (Zhang, Hustadt, and Dixon 2009)), where p is a new atom not appearing in φ, ψ is a formula in T_i not in $\text{SNF}^g_{\text{CTL}}$ clause and R_i is the result set of applying a matching transformation rule to ψ. Note that throughout the transformation formulae are kept in negation normal form.

Proposition 11 $\varphi \equiv_{\langle V', I\rangle} T_\varphi$.

This means that φ has the same models with T_φ excepting those atoms in V' and those relations $[i]$ with $i \in I$.

Example 3 By the *Transform* process, the result T_φ of the Example 2 can be listed as follows:

1.**start** $\supset z$	2.$\top \supset \neg z \vee r$	3.$\top \supset \neg x \vee f \vee m$
4.$\top \supset \neg z \vee x \vee y$	5.$\top \supset \neg y \vee p$	6.$\top \supset \neg y \vee q$
7.$z \supset \text{AF}x$	8.$y \supset \text{AX}(x \vee y)$.	

Besides, the set of new atoms introduced in this process is $V' = \{x, y, x\}$.

The Resolution process

The *Resolution* process is to compute all the possible resolutions of T_φ on $V \cup V'$, denoted as *Resolution*($T_\varphi, V \cup V'$). A *derivation* on a set $V \cup V'$ of atoms and T_φ is a sequence $T_0, T_1, T_2, \ldots, T_n = Res$ of sets of $\text{SNF}^g_{\text{CTL}}$ clauses such that $T_0 = T_\varphi$ and $T_{i+1} = T_i \cup R_i$ where R_i is a set of clauses obtained as the conclusion of the application of a resolution rule to premises in T_i. Note that all the T_i ($0 \leq i \leq n$) are set of $\text{SNF}^g_{\text{CTL}}$ clauses. Besides, if there is a T_i containing **start** $\supset \bot$ or $\top \supset \bot$, then we have $\text{F}_{\text{CTL}}(\varphi, V) = \bot$.

Proposition 12 $T_\varphi \equiv_{\langle V \cup V', \emptyset\rangle} Res$.

Proposition 11 and Proposition 12 mean that $\varphi \equiv_{\langle V \cup V', I\rangle} Res$, this resolve part of the problem (1).

Example 4 The resolution of T_φ obtained from Example 3 on $V \cup V'$ is as follows:

start $\supset r$	**start** $\supset x \vee y$
$\top \supset \neg z \vee y \vee f \vee m$	$y \supset \text{AX}(f \vee m \vee y)$
$\top \supset \neg z \vee x \vee p$	$\top \supset \neg z \vee x \vee q$
$y \supset \text{AX}(x \vee p)$	$y \supset \text{AX}(x \vee q)$
start $\supset f \vee m \vee y$	**start** $\supset x \vee p$
start $\supset x \vee q$	$\top \supset p \vee \neg z \vee f \vee m$
$\top \supset q \vee \neg z \vee f \vee m$	$y \supset \text{AX}(p \vee f \vee m)$
$y \supset \text{AX}(q \vee f \vee m)$	**start** $\supset f \vee m \vee p$
start $\supset f \vee m \vee q$	

The Elimination process

For resolving problem (2), we should pay attention to the following properties that obtained from the transformation and resolution rules at first:

- **(GNA)** for all atom p in *Var*(φ), p do not positively appear in the left hand of the $\text{SNF}^g_{\text{CTL}}$ clause;
- **(PI)** for each atom $p \in V'$, if p appearing in the left hand of a $\text{SNF}^g_{\text{CTL}}$ clause, then p appear positively.

This *Elimination* process include three sub-processes: *Instantiate*, *Connect* and *Removing_atoms*. We will described those sub-processes carefully now.

The Instantiation process An *instantiate formula* ψ of set V'' of atoms is a formula such that *Var*(ψ) $\cap V'' = \emptyset$. Given a formula of the form $p \supset \psi$ with p is an atom not in $V'' \cup$ *Var*(ψ), if ψ is an instantiate formula of set V'' then we call p is instantiated by ψ. A key point to compute forgetting is eliminate those irrelevant atoms, for this purpose we define the follow instantiation process.

Definition 6 *[instantiation] Let $V'' = V'$ and $\Gamma = Res$, then the process of instantiation is as follows:*

(i) for each global clause $C = \top \supset D \vee \neg p \in \Gamma$, if there is one and on one atom $p \in V'' \cap$ Var(C) and Var(D) $\cap (V \cup V'') = \emptyset$ then let $C = p \supset D$ and $V'' := V'' \setminus \{p\}$;

(ii) find out all the possible instantiate formulae $\varphi_1, ..., \varphi_m$ of $V \cup V''$ in the $p \supset \varphi_i \in \Gamma$ ($1 \leq i \leq m$);

(iii) if there is $p \supset \varphi_i$ for some $i \in \{1, \ldots, m\}$, then let $V'' := V'' \setminus \{p\}$, which means p is a instantiate formula;

(iv) for $\bigwedge_{j=1}^m p_j \supset \varphi_i \in \Gamma$ ($i \in \{1, \ldots, m\}$), if there is $\alpha \supset p_1, \ldots, \alpha \supset p_m \in \Gamma$ then let $\Gamma_1 := \Gamma \cup \{\alpha \supset \varphi\}$. if $\Gamma_1 \neq \Gamma$ then let $\Gamma := \Gamma_1$ go to step (i), else return $V \cup V''$.

*Where p, p_i ($1 \leq i \leq m$) are atoms and α is a conjunction of literals or **start**.*

We denote this process as *Instantiate*(Γ, V'). After this process we obtain a set of atoms that do not has been instantiated by any instantiate formula of $V \cup V''$ in this process.

Example 5 By using the instantiation process on result of Example 4, we obtain that x is instantiated by $f \vee m$ at first since there is $\top \supset \neg x \vee f \vee m \in T_\varphi$ with $x \in V'$ and *Var*($f \vee m$) $\cap (V \cup V') = \emptyset$, then $V'' = \{y, z\}$.

Similarly, due to $\top \supset \neg y \vee q \in T_\varphi$ and $y \supset \text{AX}(q \vee f \vee m) \in T_\varphi$, then y can be instantiated by $q \wedge \text{AX}(q \vee f \vee m)$. And z can be instantiated by r. Therefore $V'' = \emptyset$ That is *Instantiate*($T_\varphi^{r, V \cup V'}, V'$) $= V$, which means all the introduced atoms are instantiated.

By instantiation operator, we guarantee those atoms in $V \cup V''$ are really irrelevant, *i.e.* should be forgot.

The Connect process Let P be a conjunction of literals, l, l_1 be literals, in which *Var*(C_1) $\cap V \cup V' = \emptyset$, and C_i ($i \in \{2, 3, 4\}$) be classical clauses. Let $A = \{l \supset \neg l_1 \vee C_2, l \supset C_3 \vee C_2\}$, $\alpha = P \supset ((\neg C_3 \wedge \neg C_2) \supset (\text{E}_{\langle ind\rangle}\text{X}(C_3 \wedge$

$\neg(C_2 \vee C_4) \supset \text{AXAF}(C_3 \vee C_2))))$ and $\beta = P \supset ((\neg C_3 \wedge \neg C_2) \supset (\text{AX}(C_3 \wedge \neg(C_2 \vee C_4) \supset \text{AXAF}(C_3 \vee C_2))))$,we add following new rules, we call it **EF** imply.

(EF1)$\{P \supset \text{AF}l, P \supset \text{E}_{\langle ind\rangle}\text{X}(l_1 \vee C_4)\} \cup A \to \alpha$

(EF2)$\{P \supset \text{AF}l, P \supset \text{AX}(l_1 \vee C_4)\} \cup A \to \beta$

(EF3)$\{P \supset \text{E}_{\langle ind\rangle}\text{F}l, P \supset \text{E}_{\langle ind\rangle}\text{X}(l_1 \vee C_4)\} \cup A \to \alpha$

(EF4)$\{P \supset \text{E}_{\langle ind\rangle}\text{F}l, P \supset \text{AX}(l_1 \vee C_4)\} \cup A \to \alpha$

By *Connect*(*Instantiate*(Res, V')) we mean using (EF1) to (EF4) on Res and replacing $P \supset \text{AX}(\neg l \vee C_2 \vee C_4)$ with $P \supset \text{AX}(\neg l \vee C_2 \vee C_4) \vee \beta$ for rule (EF2) and replacing $P \supset \text{E}_{\langle ind\rangle}\text{X}(\neg l \vee C_2 \vee C_4)$ with $P \supset \text{E}_{\langle ind\rangle}\text{X}(\neg l \vee C_2 \vee C_4) \vee \alpha$ for other rules when l, C_2, C_3 and C_4 are instantiate formulae of Sub(Res, V') and *Var*$(l_1) \in V \cup V'$.

Proposition 13 *Let* $\Gamma = Res$*, we have* $\Gamma \equiv_{\langle V', \emptyset\rangle}$ Connect(Instantiate(Γ, V')).

The Removing_atoms process For eliminate those irrelevant atoms, we can do the following elimination operator.

Definition 7 (Removing_atoms) *Let T be a set of formulae, $C \in T$ and V a set of atoms, then the elimination operator is defined as:*

$$\text{Removing\_atoms}(C, V) = \begin{cases} \top, & if\ Var(C) \cap V \neq \emptyset \\ C, & else. \end{cases}$$

For convenience, for any set T of formula we have *Removing_atoms*$(T, V) = \{$*Removing_atoms*$(r, V) | r \in T\}$.

Proposition 14 *Let* $V'' = V \cup V'$, $\Gamma =$ Instantiate(Res, V') *and* $\Gamma_1 =$ Removing_atoms (Connect$(\Gamma), \Gamma)$*, then* $\Gamma_1 \equiv_{\langle V \cup V', \emptyset\rangle} Res$ *and* $\Gamma_1 \equiv_{\langle V \cup V', I\rangle} \varphi$.

Example 6 After removing the clauses that include atoms in $V = \{p\}$, the following clauses have been left:

start $\supset z$	$\top \supset \neg z \vee r$	$\top \supset \neg x \vee f \vee m$
$\top \supset \neg z \vee x \vee y$	**start** $\supset f \vee m \vee q$	$\top \supset \neg y \vee q$
$z \supset \text{AF}x$	$y \supset \text{AX}(x \vee y)$	**start** $\supset r$
start $\supset x \vee y$	$\top \supset \neg z \vee y \vee f \vee m$	$y \supset \text{AX}(f \vee m \vee y)$
$\top \supset \neg z \vee x \vee q$	$y \supset \text{AX}(x \vee q)$	**start** $\supset f \vee m \vee y$
start $\supset x \vee q$	$\top \supset q \vee \neg z \vee f \vee m$	$y \supset \text{AX}(q \vee f \vee m)$

Remove the Index and start

The *Removing_index*$(RemA)$ process is to change the set *RemA* obtained above into a set of formulas without the index by using the equations in Proposition 15.

Proposition 15 *Let P, P_i and φ_i be CTL formulas, then*

(i) $\bigwedge_{i=1}^{n} P \supset \text{E}_{\langle ind\rangle}\text{X}\varphi_i \equiv_{\langle \emptyset, \{ind\}\rangle} P \supset \text{EX}\bigwedge_{i=1}^{n}\varphi_i$,

(ii) $\bigwedge_{i=1}^{n} P_i \supset \text{E}_{\langle ind\rangle}\text{X}\varphi_i \equiv_{\langle \emptyset, \{ind\}\rangle} \bigwedge_{e \in 2^{\{0,\ldots,n\}} \setminus \{\emptyset\}} (\bigwedge_{i \in e} P_i \supset \text{EX}(\bigwedge_{i \in e}\varphi_i))$,

(iii) $\bigwedge_{i=1}^{n} P \supset \text{E}_{\langle ind\rangle}\text{F}\varphi_i \equiv_{\langle \emptyset, \{ind\}\rangle} P \supset \bigvee \text{EF}(\varphi_{j_1} \wedge \text{EF}(\varphi_{j_2} \wedge \text{EF}(\cdots \wedge \text{EF}\varphi_{j_n})))$*, where* $(j_1, \ldots, j_n)$ *are sequences of all elements in* $\{0, \ldots, n\}$,

(iv) $P \supset (C \vee \text{E}_{\langle ind\rangle}\text{X}\varphi_1) \wedge P \supset \text{E}_{\langle ind\rangle}\text{X}\varphi_2 \equiv_{\langle \emptyset, \{ind\}\rangle} P \supset ((C \wedge \text{EX}\varphi_2) \vee \text{EX}(\varphi_1 \wedge \varphi_2))$,

(v) $P \supset (C \vee \text{E}_{\langle ind\rangle}\text{X}\varphi_1) \vee P \supset \text{E}_{\langle ind\rangle}\text{X}\varphi_2 \equiv_{\langle \emptyset, \{ind\}\rangle} P \supset (C \vee \text{EX}(\varphi_1 \vee \varphi_2))$.

Lemma 5 ***(NI-BRemain)*** *Let I is the set of indexes in* RemA*, we have* RemA $\equiv_{\langle \emptyset, I\rangle}$ Removing_index(RemA).

In our Example 6 we do not need this process due to there is no index in the set of formulae. Let T be a set of $\text{SNF}^g_{\text{CTL}}$ clauses, then we define the following operator:

$T_{\text{CTL}} = \{C | C' \in T$ and $C = D$ if $C'$is the form $\text{AG}(\textbf{start} \supset D)$, else $C = C'\}$.

Then $T \equiv T_{\text{CTL}}$ by $\varphi \equiv \text{AG}(\textbf{start} \supset \varphi)$ (Bolotov 2000).

The last step of our algorithm is to eliminate all the atoms in V' which has been introduced in the *Transform* process. Let $V'' = V \cup V'$, $\Gamma =$ *Instantiate*(Res, V') and $\Gamma_1 =$ *Removing_atoms*(*Connect*(Γ)), then *Replacing_atoms*(*Removing_index*(Γ_1)) is obtained from *Removing_index*(Γ_1) by doing the following two steps for each $p \in (V' \setminus \Gamma)$:

- replacing each $p \supset \varphi_1 \vee \cdots \vee p \supset \varphi_n$ with $p \supset \bigvee_{i=1}^{n}\varphi_i$;
- replacing $p \supset \varphi_1 \wedge \cdots \wedge p \supset \varphi_m$ with φ_j are instantiate formulae of Γ $(j \in \{1, \ldots, m\})$ with $p \leftrightarrow \psi$, where $\psi = \bigwedge_{j=1}^{m}\varphi_j$ and p do not appear in φ_j, .
- For other formula $C \in \Gamma_1$, replacing every p in C with ψ.

Apparently, this process is just a process of replacing each atom with an equivalent formula. Then we have:

Proposition 16 *Let* $\Gamma_1 =$ Instantiate(Res, V'), $\Gamma_2 =$ Removing_atoms (Connect$(\Gamma_1), \Gamma_1)$ *and* $\Gamma_3 =$ Replacing_atoms(Removing_index(Γ_2))*, then* $\Gamma_2 \equiv_{\langle V' \setminus \Gamma_1, I\rangle} \Gamma_3$ *and* $\varphi \equiv_{\langle V \cup V', \emptyset\rangle} (\Gamma_3)_{CTL}$.

Example 7 By using the *Replacing_atoms* process on result of Example 6 directly since there is not index in those clauses, we obtain that x is replaced by $f \vee m$ at first, then y is replaced by $q \wedge \text{AX}(q \vee f \vee m)$ and z is replaced by $r \wedge (f \vee m \vee q) \wedge (f \vee m \vee (q \wedge \text{AX}(f \vee m \vee q))) \wedge \text{AF}(f \vee m)$.

The Correction and Complexity of the Algorithm

In the case that formula dose not include index, we use model structure $\mathcal{M} = (S, R, L, s_0)$ to interpret formula instead of Ind-model structure.

Theorem 9 (Resolution-based CTL-forgetting) *Let* $V'' = V \cup V'$ *and* $\Gamma_1 =$ ERes(φ, V)*, then*

(i) $\text{F}_{\text{CTL}}(\varphi, V'') \equiv \Gamma_1$;

(ii) $\text{F}_{\text{CTL}}(\varphi, V) \equiv \bigwedge_{\psi \in \Gamma_1}\psi$.

Then we can obtain the result of forgetting of Example 5:

$\text{F}_{\text{CTL}}(\varphi, \{p\}) \equiv r \wedge (f \vee m \vee q) \wedge \text{AF}(f \vee m) \wedge (f \vee m \vee (q \wedge \text{AX}(f \vee m \vee q))) \wedge \text{AG}((q \wedge \text{AX}(f \vee m \vee q) \supset \text{AX}(f \vee m \vee (q \wedge \text{AX}(f \vee m \vee q)))))$.

Example 8 Consider the model given in Figure 1. Assume that we are given a property $\alpha = \text{AGEF}(lc \wedge sr)$ and $\{lc\}$. Then, it is easy to check that $\text{F}_{\text{CTL}}(\alpha, \{lc\}) \equiv \text{AGEF}sr$.

Proposition 17 *Let φ be a CTL formula and $V \subseteq \mathcal{A}$. The time and space complexity of Algorithm 1 are $O((m+1)2^{4(n+n')}$. Where $|Var(\varphi)| = n$, $|V'| = n'$ (V' is set of atoms introduced in transformation) and m is the number of the set Ind of indices introduced during transformation.*

Concluding Remarks

Summary In this article, we generalized the notion of bisimulation from model structures of Computation Tree Logic (CTL) to model structures with initial states over a given signature V, named *V-bisimulation*. Based on this new bisimulation, we presented the notion of forgetting for CTL, which enables computing weakest sufficient and strongest necessary conditions of specifications. Further properties and complexity issues in this context were also explored. In particular, we have shown that our notion of forgetting satisfies the existing postulates of forgetting, which means it faithfully extends the notion of forgetting from classical propositional logic and modal logic S5 to CTL. On the complexity theory side, we investigated the model checking of forgetting, which turned out to be NP-complete, and the relevant fragment of $\mathrm{CTL}_{\mathrm{AF}}$ which ranged from co-NP to Π_2^p-completeness. And finally, we proposed a model-based algorithm which computes the forgetting of a given formula and a set of variables, and outlined its complexity.

Future work Note that, when a transition system $\mathcal{M}$ does not satisfy a specification ϕ, one can evaluate the weakest sufficient condition ψ over a signature V under which $\mathcal{M}$ satisfies ϕ, viz., $\mathcal{M} \models \psi \supset \phi$ and ψ mentions only atoms from V. It is worthwhile to explore how the condition ψ can guide the design of a new transition system $\mathcal{M}'$ satisfying ψ.

Moreover, a further study regarding the the computational complexity for other general fragments is required and part of the future research agenda. Such investigation can be coupled with fine-grained parameterized analysis.

Supplementary Material: Proof Appendix

Lemma 6 *Let $\mathcal{B}_0, \mathcal{B}_1, \ldots$ be the ones in the definition of section 3.1. Then, for each $i \geq 0$,*

(i) $\mathcal{B}_{i+1} \subseteq \mathcal{B}_i$;

(ii) there is a (smallest) $k \geq 0$ such that $\mathcal{B}_{k+1} = \mathcal{B}_k$;

(iii) $\mathcal{B}_i$ is reflexive, symmetric and transitive.

Proof: (i) Base: it is clear for $i = 0$ by the above definition.

Step: suppose it holds for $i = n$, i.e., $\mathcal{B}_{n+1} \subseteq \mathcal{B}_n$.
$(s, s') \in \mathcal{B}_{n+2}$
$\Rightarrow$ (a) $(s, s') \in \mathcal{B}_0$, (b) for every $(s, s_1) \in R$, there is $(s', s'_1) \in R'$ such that $(s_1, s'_1) \in \mathcal{B}_{n+1}$, and (c) for every $(s', s'_1) \in R'$, there is $(s, s_1) \in R$ such that $(s_1, s'_1) \in \mathcal{B}_{n+1}$
$\Rightarrow$ (a) $(s, s') \in \mathcal{B}_0$, (b) for every $(s, s_1) \in R$, there is $(s', s'_1) \in R'$ such that $(s_1, s'_1) \in \mathcal{B}_n$ by inductive assumption, and (c) for every $(s', s'_1) \in R'$, there is $(s, s_1) \in R$ such that $(s_1, s'_1) \in \mathcal{B}_n$ by inductive assumption
$\Rightarrow (s, s') \in \mathcal{B}_{n+1}$.

(ii) and (iii) are evident from (i) and the definition of $\mathcal{B}_i$. ■

Lemma 1 The relation $\leftrightarrow_V$ is an equivalence relation.

Proof: It is clear from Lemma 6 (ii) such that there is a $k \geq 0$ where $\mathcal{B}_k = \mathcal{B}_{k+1}$ which is $\leftrightarrow_V$, and it is reflexive, symmetric and transitive by (iii). ■

Proposition 1 Let $i \in \{1, 2\}$, $V_1, V_2 \subseteq \mathcal{A}$, s'_is be two states and π'_is be two paths, and $\mathcal{K}_i = (\mathcal{M}_i, s_i)$ $(i = 1, 2, 3)$ be K-structures such that $\mathcal{K}_1 \leftrightarrow_{V_1} \mathcal{K}_2$ and $\mathcal{K}_2 \leftrightarrow_{V_2} \mathcal{K}_3$. Then:

(i) $s'_1 \leftrightarrow_{V_i} s'_2$ $(i = 1, 2)$ implies $s'_1 \leftrightarrow_{V_1 \cup V_2} s'_2$;

(ii) $\pi'_1 \leftrightarrow_{V_i} \pi'_2$ $(i = 1, 2)$ implies $\pi'_1 \leftrightarrow_{V_1 \cup V_2} \pi'_2$;

(iii) for each path π_{s_1} of $\mathcal{M}_1$ there is a path π_{s_2} of $\mathcal{M}_2$ such that $\pi_{s_1} \leftrightarrow_{V_1} \pi_{s_2}$, and vice versa;

(iv) $\mathcal{K}_1 \leftrightarrow_{V_1 \cup V_2} \mathcal{K}_3$;

(v) If $V_1 \subseteq V_2$ then $\mathcal{K}_1 \leftrightarrow_{V_2} \mathcal{K}_2$.

Proof: In order to distinguish the relations $\mathcal{B}_0, \mathcal{B}_1, \ldots$ for different set $V \subseteq \mathcal{A}$, by $\mathcal{B}_i^V$ we mean the relation $\mathcal{B}_1, \mathcal{B}_2, \ldots$ for $V \subseteq \mathcal{A}$. Denote as $\mathcal{B}_0, \mathcal{B}_1, \ldots$ when the underlying set V is clear from the context. Moreover, for the ease of notation, we will refer to $\leftrightarrow_V$ by $\mathcal{B}$ (i.e., without subindex).

(i) Base: it is clear for $n = 0$.

Step: For $n > 0$, supposing if $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_i^{V_1}$ and $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_i^{V_2}$ then $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_i^{V_1 \cup V_2}$ for all $0 \leq i \leq n$. We will show that if $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_{n+1}^{V_1}$ and $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_{n+1}^{V_2}$ then $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_{n+1}^{V_1 \cup V_2}$.
(a) It is evident that $L_1(s_1) - (V_1 \cup V_2) = L_2(s_2) - (V_1 \cup V_2)$.
(b) We will show that for each $(s_1, s_1^1) \in R_1$ there is a $(s_2, s_2^1) \in R_2$ such that $(s_1^1, s_2^1) \in \mathcal{B}_n^{V_1 \cup V_2}$. There is $(\mathcal{K}_1^1, \mathcal{K}_2^1) \in \mathcal{B}_{n-1}^{V_1 \cup V_2}$ due to $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_n^{V_1 \cup V_2}$ by inductive assumption. Then we only need to prove for each $(s_1^1, s_1^2) \in R_1$ there is a $(s_2^1, s_2^2) \in R_2$ such that $(\mathcal{K}_1^2, \mathcal{K}_2^2) \in \mathcal{B}_{n-2}^{V_1 \cup V_2}$ and for each $(s_2^1, s_2^2) \in R_2$ there is a $(s_1^1, s_1^2) \in R_1$ such that $(\mathcal{K}_1^2, \mathcal{K}_2^2) \in \mathcal{B}_{n-2}^{V_1 \cup V_2}$. Therefore, we only need to prove that for each $(s_1^n, s_1^{n+1}) \in R_1$ there is a $(s_2^n, s_2^{n+1}) \in R_2$ such that $(\mathcal{K}_1^{n+1}, \mathcal{K}_2^{n+1}) \in \mathcal{B}_0^{V_1 \cup V_2}$ and for each $(s_2^n, s_2^{n+1}) \in R_2$ there is a $(s_1^n, s_1^{n+1}) \in R_1$ such that $(\mathcal{K}_1^{n+1}, \mathcal{K}_2^{n+1}) \in \mathcal{B}_0^{V_1 \cup V_2}$. It is apparent that $L_1(s_1^{n+1}) - (V_1 \cup V_2) = L_1(s_2^{n+1}) - (V_1 \cup V_2)$ due to $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_{n+1}^{V_1}$ and $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_{n+1}^{V_2}$. Where $\mathcal{K}_i^j = (\mathcal{M}_i, s_i^j)$ with $i \in \{1, 2\}$ and $0 < j \leq n + 1$.
(c) It is similar with (b).

(ii) It is clear from (i).

(iii) The following property show our result directly. Let $V \subseteq \mathcal{A}$ and $\mathcal{K}_i = (\mathcal{M}_i, s_i)$ $(i = 1, 2)$ be K-structures. Then $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}$ if and only if

(a) $L_1(s_1) - V = L_2(s_2) - V$,

(b) for every $(s_1, s'_1) \in R_1$, there is $(s_2, s'_2) \in R_2$ such that $(\mathcal{K}'_1, \mathcal{K}'_2) \in \mathcal{B}$, and

(c) for every $(s_2, s'_2) \in R_2$, there is $(s_1, s'_1) \in R_1$ such that $(\mathcal{K}'_1, \mathcal{K}'_2) \in \mathcal{B}$,

where $\mathcal{K}'_i = (\mathcal{M}_i, s'_i)$ with $i \in \{1, 2\}$.

We prove it from the following two aspects:

$(\Rightarrow)$ (a) It is apparent that $L_1(s_1) - V = L_2(s_2) - V$; (b) $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}$ iff $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_i$ for all $i \geq 0$, then for each $(s_1, s'_1) \in R_1$, there is a $(s_2, s'_2) \in R_2$ such that $(\mathcal{K}'_1, \mathcal{K}'_2) \in \mathcal{B}_{i-1}$ for all $i > 0$ and then $L_1(s'_1) - V = L_2(s'_2) - V$. Therefore, $(\mathcal{K}'_1, \mathcal{K}'_2) \in \mathcal{B}$. (c) This is similar with (b).

$(\Leftarrow)$ (a) $L_1(s_1) - V = L_2(s_2) - V$ implies that $(s_1, s_2) \in \mathcal{B}_0$; (b) Condition (ii) implies that for every $(s_1, s'_1) \in R_1$, there is $(s_2, s'_2) \in R_2$ such that $(\mathcal{K}'_1, \mathcal{K}'_2) \in \mathcal{B}_i$ for all $i \geq 0$; (c) Condition (iii) implies that for every $(s_2, s'_2) \in R_2$, there is $(s_1, s'_1) \in R_1$ such that $(\mathcal{K}'_1, \mathcal{K}'_2) \in \mathcal{B}_i$ for all $i \geq 0$
$\Rightarrow (\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_i$ for all $i \geq 0$
$\Rightarrow (\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}$.

(iv) Let $\mathcal{M}_i = (S_i, R_i, L_i, s_i)$ $(i = 1, 2, 3)$, $s_1 \leftrightarrow_{V_1} s_2$ via a binary relation $\mathcal{B}$, and $s_2 \leftrightarrow_{V_2} s_3$ via a binary relation $\mathcal{B}''$. Let $\mathcal{B}' = \{(w_1, w_3) | (w_1, w_2) \in \mathcal{B}$ and $(w_2, w_3) \in \mathcal{B}_2\}$. It's apparent that $(s_1, s_3) \in \mathcal{B}'$. We prove $\mathcal{B}'$ is a $V_1 \cup V_2$-bisimulation containing (s_1, s_3) from the (a), (b) and (c) of the previous step (iii) of X-bisimulation (where X is a set of atoms). For all $(w_1, w_3) \in \mathcal{B}'$:

(a) there is $w_2 \in S_2$ such that $(w_1, w_2) \in \mathcal{B}$ and $(w_2, w_3) \in \mathcal{B}''$, and $\forall q \notin V_1$, $q \in L_1(w_1)$ iff $q \in L_2(w_2)$ by $w_1 \leftrightarrow_{V_1} w_2$ and $\forall q' \notin V_2$, $q' \in L_2(w_2)$ iff $q' \in L_3(w_3)$ by $w_2 \leftrightarrow_{V_2} w_3$. Then we have $\forall r \notin V_1 \cup V_2, r \in L_1(w_1)$ iff $r \in L_3(w_3)$.

(b) if $(w_1, u_1) \in \mathcal{R}_1$, then $\exists u_2 \in S_2$ such that $(w_2, u_2) \in \mathcal{R}_2$ and $(u_1, u_2) \in \mathcal{B}$ (due to $(w_1, w_2) \in \mathcal{B}$ and $(w_2, w_3) \in \mathcal{B}''$ by the definition of $\mathcal{B}'$); and then $\exists u_3 \in S_3$ such that $(w_3, u_3) \in \mathcal{R}_3$ and $(u_2, u_3) \in \mathcal{B}''$, hence $(u_1, u_3) \in \mathcal{B}'$ by the definition of $\mathcal{B}'$.

(c) if $(w_3, u_3) \in \mathcal{R}_3$, then $\exists u_2 \in S_2$ such that $(w_2, u_2) \in \mathcal{R}_2$ and $(u_2, u_3) \in \mathcal{B}_2$; and then $\exists u_1 \in S_1$ such that $(w_1, u_1) \in \mathcal{R}_1$ and $(u_1, u_2) \in \mathcal{B}$, hence $(u_1, u_3) \in \mathcal{B}'$ by the definition of $\mathcal{B}'$.

(v) Let $\mathcal{K}_{i,j} = (\mathcal{M}_i, s_{i,j})$ and $(s_{i,k}, s_{i,k+1}) \in R_i$ mean that $s_{i,k+1}$ is the $(k + 2)$-th node in the path $(s_i, s_{i,1}, s_{i,2}, \ldots, s_{i,k+1}, \ldots)$ $(i = 1, 2)$. We will show that $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_n^{V_2}$ for all $n \geq 0$ inductively.

Base: $L_1(s_1) - V_1 = L_2(s_2) - V_1$
$\Rightarrow \forall q \in \mathcal{A} - V_1$ there is $q \in L_1(s_1)$ iff $q \in L_2(s_2)$

$\Rightarrow \forall q \in \mathcal{A} - V_2$ there is $q \in L_1(s_1)$ iff $q \in L_2(s_2)$ due to $V_1 \subseteq V_2$
$\Rightarrow L_1(s_1) - V_2 = L_2(s_2) - V_2$, i.e., $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_0^{V_2}$.

Step: Supposing that $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_i^{V_2}$ for all $0 \leq i \leq k$ $(k > 0)$, we will show $(\mathcal{K}_1, \mathcal{K}_2) \in \mathcal{B}_{k+1}^{V_2}$.

(a) It is apparent that $L_1(s_1) - V_2 = L_2(s_2) - V_2$ by base.

(b) $\forall (s_1, s_{1,1}) \in R_1$, we will show that there is a $(s_2, s_{2,1}) \in R_2$ *s.t.* $(\mathcal{K}_{1,1}, \mathcal{K}_{2,1}) \in \mathcal{B}_k^{V_2}$. $(\mathcal{K}_{1,1}, \mathcal{K}_{2,1}) \in \mathcal{B}_{k-1}^{V_2}$ by inductive assumption, we need only to prove the following points:
(a) $\forall (s_{1,k}, s_{1,k+1}) \in R_1$ there is a $(s_{2,k}, s_{2,k+1}) \in R_2$ *s.t.* $(\mathcal{K}_{1,k+1}, \mathcal{K}_{2,k+1}) \in \mathcal{B}_0^{V_2}$ due to $(\mathcal{K}_{1,1}, \mathcal{K}_{2,1}) \in \mathcal{B}_k^{V_1}$. It is easy to see that $L_1(s_{1,k+1}) - V_1 = L_1(s_{2,k+1}) - V_1$, then there is $L_1(s_{1,k+1}) - V_2 = L_1(s_{2,k+1}) - V_2$. Therefore, $(\mathcal{K}_{1,k+1}, \mathcal{K}_{2,k+1}) \in \mathcal{B}_0^{V_2}$.
(b) $\forall (s_{2,k}, s_{2,k+1}) \in R_1$ there is a $(s_{1,k}, s_{1,k+1}) \in R_1$ *s.t.* $(\mathcal{K}_{1,k+1}, \mathcal{K}_{2,k+1}) \in \mathcal{B}_0^{V_2}$ due to $(\mathcal{K}_{1,1}, \mathcal{K}_{2,1}) \in \mathcal{B}_k^{V_1}$. This can be proved as (a).

(c) $\forall (s_2, s_{2,1}) \in R_1$, we will show that there is a $(s_1, s_{1,1}) \in R_2$ *s.t.* $(\mathcal{K}_{1,1}, \mathcal{K}_{2,1}) \in \mathcal{B}_k^{V_2}$. This can be proved as (ii).

■

Theorem1 Let $V \subseteq \mathcal{A}$, $\mathcal{K}_i$ $(i = 1, 2)$ be two K-structures such that $\mathcal{K}_1 \leftrightarrow_V \mathcal{K}_2$ and ϕ a formula with IR(ϕ, V). Then $\mathcal{K}_1 \models \phi$ if and only if $\mathcal{K}_2 \models \phi$.

Proof: This theorem can be proved by inducting on the formula ϕ and supposing $Var(\phi) \cap V = \emptyset$.

Here we only prove the only-if direction. The other direction can be similarly proved.

Case $\phi = p$ where $p \in \mathcal{A} - V$:
$(\mathcal{M}, s) \models \phi$ iff $p \in L(s)$ (by the definition of satisfiability)
$\Leftrightarrow p \in L'(s')$ $(s \leftrightarrow_V s')$
$\Leftrightarrow (\mathcal{M}', s') \models \phi$

Case $\phi = \neg\psi$:
$(\mathcal{M}, s) \models \phi$ iff $(\mathcal{M}, s) \not\models \psi$
$\Leftrightarrow (\mathcal{M}', s') \not\models \psi$ (induction hypothesis)
$\Leftrightarrow (\mathcal{M}', s') \models \phi$

Case $\phi = \psi_1 \vee \psi_2$:
$(\mathcal{M}, s) \models \phi$
$\Leftrightarrow (\mathcal{M}, s) \models \psi_1$ or $(\mathcal{M}, s) \models \psi_2$
$\Leftrightarrow (\mathcal{M}', s') \models \psi_1$ or $(\mathcal{M}', s') \models \psi_2$ (induction hypothesis)
$\Leftrightarrow (\mathcal{M}', s') \models \phi$

Case $\phi = \text{EX}\psi$:
$\mathcal{M}, s \models \phi$
$\Leftrightarrow$ There is a path $\pi = (s, s_1, ...)$ such that $\mathcal{M}, s_1 \models \psi$
$\Leftrightarrow$ There is a path $\pi' = (s', s_1', ...)$ such that $\pi \leftrightarrow_V \pi'$ $(s \leftrightarrow_V s'$, Proposition 1)
$\Leftrightarrow s_1 \leftrightarrow_V s_1'$ $(\pi \leftrightarrow_V \pi')$
$\Leftrightarrow (\mathcal{M}', s_1') \models \psi$ (induction hypothesis)
$\Leftrightarrow (\mathcal{M}', s') \models \phi$

Case $\phi = \text{EG}\psi$:
$\mathcal{M}, s \models \phi$
$\Leftrightarrow$ There is a path $\pi = (s = s_0, s_1, ...)$ such that for each $i \geq 0$ there is $(\mathcal{M}, s_i) \models \psi$
$\Leftrightarrow$ There is a path $\pi' = (s' = s_0', s_1', ...)$ such that $\pi \leftrightarrow_V \pi'$ $(s \leftrightarrow_V s'$, Proposition 1)
$\Leftrightarrow s_i \leftrightarrow_V s_i'$ for each $i \geq 0$ $(\pi \leftrightarrow_V \pi')$
$\Leftrightarrow (\mathcal{M}', s_i') \models \psi$ for each $i \geq 0$ (induction hypothesis)
$\Leftrightarrow (\mathcal{M}', s') \models \phi$

Case $\phi = \text{E}[\psi_1 \text{U} \psi_2]$:
$\mathcal{M}, s \models \phi$
$\Leftrightarrow$ There is a path $\pi = (s = s_0, s_1, ...)$ such that there is $i \geq 0$ such that $(\mathcal{M}, s_i) \models \psi_2$, and for all $0 \leq j < i$, $(\mathcal{M}, s_j) \models \psi_1$
$\Leftrightarrow$ There is a path $\pi' = (s = s_0', s_1', ...)$ such that $\pi \leftrightarrow_V \pi'$ $(s \leftrightarrow_V s'$, Proposition 1)
$\Leftrightarrow (\mathcal{M}', s_i') \models \psi_2$, and for all $0 \leq j < i$ $(\mathcal{M}', s_j') \models \psi_1$ (induction hypothesis)
$\Leftrightarrow (\mathcal{M}', s') \models \phi$ ■

Proposition 2 Let $V \subseteq \mathcal{A}$ and $(\mathcal{M}_i, s_i)$ $(i = 1, 2)$ be two K-structures. Then

$(s_1, s_2) \in \mathcal{B}_n$ iff $\text{Tr}_j(s_1) \leftrightarrow_V \text{Tr}_j(s_2)$ for every $0 \leq j \leq n$.

Proof: We will prove this from two aspects:

$(\Rightarrow)$ If $(s_1, s_2) \in \mathcal{B}_n$, then $Tr_j(s_1) \leftrightarrow_V Tr_j(s_2)$ for all $0 \leq j \leq n$. $(s, s') \in \mathcal{B}_n$ implies both roots of $Tr_n(s_1)$ and $Tr_n(s_2)$ have the same atoms except those atoms in V. Besides, for any $s_{1,1}$ with $(s_1, s_{1,1}) \in R_1$, there is a $s_{2,1}$ with $(s_2, s_{2,1}) \in R_2$ s.t. $(s_{1,1}, s_{2,1}) \in \mathcal{B}_{n-1}$ and vice versa. Then we have $Tr_1(s_1) \leftrightarrow_V Tr_1(s_2)$. Therefore, $Tr_n(s_1) \leftrightarrow_V Tr_n(s_2)$ by use such method recursively, and then $Tr_j(s_1) \leftrightarrow_V Tr_j(s_2)$ for all $0 \leq j \leq n$.

$(\Leftarrow)$ If $Tr_j(s_1) \leftrightarrow_V Tr_j(s_2)$ for all $0 \leq j \leq n$, then $(s_1, s_2) \in \mathcal{B}_n$. $Tr_0(s_1) \leftrightarrow_V Tr_0(s_2)$ implies $L(s_1) - V = L'(s_2) - V$ and then $(s, s') \in \mathcal{B}_0$. $Tr_1(s_1) \leftrightarrow_V Tr_1(s_2)$ implies $L(s_1) - V = L'(s_2) - V$ and for every successors s of the root of one, it is possible to find a successor of the root of the other s' such that $(s, s') \in \mathcal{B}_0$. Therefore $(s_1, s_2) \in \mathcal{B}_1$, and then we will have $(s_1, s_2) \in \mathcal{B}_n$ by use such method recursively. ■

Proposition 3 Let $V \subseteq \mathcal{A}$, $\mathcal{M}$ be a model structure and $s, s' \in S$ such that $s \not\leftrightarrow_V s'$. There exists a least k such that $\text{Tr}_k(s)$ and $\text{Tr}_k(s')$ are not V-bisimilar.

Proof: If $s \not\leftrightarrow_V s'$, then there exists a least constant k such that $(s_i, s_j) \notin \mathcal{B}_k$, and then there is a least constant m $(m \leq k)$ such that $\text{Tr}_m(s_i)$ and $\text{Tr}_m(s_j)$ are not V-corresponding by Proposition 2. Let $c = m$, the lemma is proved. ■

Lemma2 Let $V \subseteq \mathcal{A}$, $\mathcal{M}$ and $\mathcal{M}'$ be two model structures, $s \in S$, $s' \in S'$ and $n \geq 0$. If $\text{Tr}_n(s) \leftrightarrow_{\overline{V}} \text{Tr}_n(s')$, then $\mathcal{F}_V(\text{Tr}_n(s)) \equiv \mathcal{F}_V(\text{Tr}_n(s'))$.

Proof: This result can be proved by inducting on n.

Base. It is apparent that for any $s_n \in S$ and $s_n' \in S'$, if $\text{Tr}_0(s_n) \leftrightarrow_{\overline{V}} \text{Tr}_0(s_n')$ then $\mathcal{F}_V(\text{Tr}_0(s_n)) \equiv \mathcal{F}_V(\text{Tr}_0(s_n'))$ due to $L(s_n) - \overline{V} = L'(s_n') - \overline{V}$ by known.

Step. Supposing that for $k = m$ $(0 < m \leq n)$ there is if $\text{Tr}_{n-k}(s_k) \leftrightarrow_{\overline{V}} \text{Tr}_{n-k}(s_k')$ then $\mathcal{F}_V(\text{Tr}_{n-k}(s_k)) \equiv \mathcal{F}_V(\text{Tr}_{n-k}(s_k'))$, then we will show if $\text{Tr}_{n-k+1}(s_{k-1}) \leftrightarrow_{\overline{V}} \text{Tr}_{n-k+1}(s_{k-1}')$ then $\mathcal{F}_V(\text{Tr}_{n-k+1}(s_{k-1})) \equiv \mathcal{F}_V(\text{Tr}_{n-k+1}(s_{k-1}'))$. Apparent that:

$\mathcal{F}_V(\mathrm{Tr}_{n-k+1}(s_{k-1})) = \left(\bigwedge_{(s_{k-1},s_k)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_{n-k}(s_k))\right)\wedge$ $\mathrm{AX}\left(\bigvee_{(s_{k-1},s_k)\in R} \mathcal{F}_V(\mathrm{Tr}_{n-k}(s_k))\right) \wedge \mathcal{F}_V(\mathrm{Tr}_0(s_{k-1}))$
$\mathcal{F}_V(\mathrm{Tr}_{n-k+1}(s'_{k-1})) = \left(\bigwedge_{(s'_{k-1},s'_k)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_{n-k}(s'_k))\right)\wedge$ $\mathrm{AX}\left(\bigvee_{(s'_{k-1},s'_k)\in R} \mathcal{F}_V(\mathrm{Tr}_{n-k}(s'_k))\right) \wedge \mathcal{F}_V(\mathrm{Tr}_0(s'_{k-1}))$ by the definition of characterizing formula of the computation tree. Then we have for any $(s_{k-1}, s_k) \in R$ there is $(s'_{k-1}, s'_k) \in R'$ such that $\mathrm{Tr}_{n-k}(s_k) \leftrightarrow_{\overline{V}} \mathrm{Tr}_{n-k}(s'_k)$ by $\mathrm{Tr}_{n-k+1}(s_{k-1}) \leftrightarrow_{\overline{V}} \mathrm{Tr}_{n-k+1}(s'_{k-1})$. Besides, for any $(s'_{k-1}, s'_k) \in R'$ there is $(s_{k-1}, s_k) \in R$ such that $\mathrm{Tr}_{n-k}(s_k) \leftrightarrow_{\overline{V}} \mathrm{Tr}_{n-k}(s'_k)$ by $\mathrm{Tr}_{n-k+1}(s_{k-1}) \leftrightarrow_{\overline{V}} \mathrm{Tr}_{n-k+1}(s'_{k-1})$. Therefore, we have $\mathcal{F}_V(\mathrm{Tr}_{n-k+1}(s_{k-1})) \equiv \mathcal{F}_V(\mathrm{Tr}_{n-k+1}(s'_{k-1}))$ by induction hypothesis. ■

Theorem 2 Given $V \subseteq \mathcal{A}$, let $\mathcal{M} = (S, R, L, s_0)$ and $\mathcal{M}' = (S', R', L', s'_0)$ be two model structures. Then,

(i) $(\mathcal{M}', s'_0) \models \mathcal{F}_V(\mathcal{M}, s_0)$ iff $(\mathcal{M}, s_0) \leftrightarrow_{\overline{V}} (\mathcal{M}', s'_0)$;

(ii) $s_0 \leftrightarrow_{\overline{V}} s'_0$ implies $\mathcal{F}_V(\mathcal{M}, s_0) \equiv \mathcal{F}_V(\mathcal{M}', s'_0)$.

In order to prove Theorem 2, we prove the following two lemmas at first.

Lemma 7 *Let $V \subseteq \mathcal{A}$, $\mathcal{M} = (S, R, L, s_0)$ and $\mathcal{M}' = (S', R', L', s'_0)$ be two model structures, $s \in S$, $s' \in S'$ and $n \geq 0$.*

(i) $(\mathcal{M}, s) \models \mathcal{F}_V(Tr_n(s))$.

(ii) If $(\mathcal{M}, s) \models \mathcal{F}_V(Tr_n(s'))$ then $Tr_n(s) \leftrightarrow_{\overline{V}} Tr_n(s')$.

Proof: (i) It is apparent from the definition of $\mathcal{F}_V(\mathrm{Tr}_n(s))$.
Base. It is apparent that $(\mathcal{M}, s) \models \mathcal{F}_V(\mathrm{Tr}_0(s))$.
Step. For $k \geq 0$, supposing the result talked in (i) is correct in $k-1$, we will show that $(\mathcal{M}, s) \models \mathcal{F}_V(\mathrm{Tr}_{k+1}(s))$, i.e.,:

$$(\mathcal{M}, s) \models \left(\bigwedge_{(s,s')\in R} \mathrm{EX}T(s')\right) \wedge \mathrm{AX}\left(\bigvee_{(s,s')\in R} T(s')\right) \wedge \mathcal{F}_V(\mathrm{Tr}_0(s)).$$

Where $T(s') = \mathcal{F}_V(\mathrm{Tr}_k(s'))$. It is apparent that $(\mathcal{M}, s) \models \mathcal{F}_V(\mathrm{Tr}_0(s))$ by Base. It is apparent that for any $(s, s') \in R$, there is $(\mathcal{M}, s') \models \mathcal{F}_V(\mathrm{Tr}_k(s'))$ by inductive assumption. Then we have $(\mathcal{M}, s) \models \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_k(s')$, and then $(\mathcal{M}, s) \models \left(\bigwedge_{(s,s')\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_k(s'))\right)$. Similarly, we have that for any $(s, s') \in R$, there is $(\mathcal{M}, s') \models \bigvee_{(s,s'')\in R} \mathcal{F}_V(\mathrm{Tr}_k(s''))$. Therefore, $(\mathcal{M}, s) \models \mathrm{AX}\left(\bigvee_{(s,s'')\in R} \mathcal{F}_V(\mathrm{Tr}_k(s''))\right)$.

(ii) **Base**. If $n = 0$, then $(\mathcal{M}, s) \models \mathcal{F}_V(\mathrm{Tr}_0(s'))$ implies $L(s) - \overline{V} = L'(s') - \overline{V}$. Hence, $\mathrm{Tr}_0(s) \leftrightarrow_{\overline{V}} \mathrm{Tr}_0(s')$.
Step. Supposing $n > 0$ and the result talked in (ii) is correct in $n-1$.
(a) It is easy to see that $L(s) - \overline{V} = L'(s') - \overline{V}$.
(b) We will show that for each $(s, s_1) \in R$, there is a $(s', s'_1) \in R'$ such that $\mathrm{Tr}_{n-1}(s_1) \leftrightarrow_{\overline{V}} \mathrm{Tr}_{n-1}(s'_1)$. Since $(\mathcal{M}, s) \models \mathcal{F}_V(\mathrm{Tr}_n(s'))$, then $(\mathcal{M}, s) \models \mathrm{AX}\left(\bigvee_{(s',s'_1)\in R} \mathcal{F}_V(\mathrm{Tr}_{n-1}(s'_1))\right)$. Therefore, for each $(s, s_1) \in R$ there is a $(s', s'_1) \in R'$ such that $(\mathcal{M}, s_1) \models \mathcal{F}_V(\mathrm{Tr}_{n-1}(s'_1))$. Hence, $\mathrm{Tr}_{n-1}(s_1) \leftrightarrow_{\overline{V}} \mathrm{Tr}_{n-1}(s'_1)$ by inductive hypothesis.
(c) We will show that for each $(s', s'_1) \in R'$ there is a $(s, s_1) \in R$ such that $\mathrm{Tr}_{n-1}(s'_1) \leftrightarrow_{\overline{V}} \mathrm{Tr}_{n-1}(s_1)$. Since $(\mathcal{M}, s) \models \mathcal{F}_V(\mathrm{Tr}_n(s'))$, then $(\mathcal{M}, s) \models \bigwedge_{(s',s'_1)\in R'} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_{n-1}(s'_1))$. Therefore, for each $(s', s'_1) \in R'$ there is a $(s, s_1) \in R$ such that $(\mathcal{M}, s_1) \models \mathcal{F}_V(\mathrm{Tr}_{n-1}(s'_1)$. Hence, $\mathrm{Tr}_{n-1}(s_1) \leftrightarrow_{\overline{V}} \mathrm{Tr}_{n-1}(s'_1)$ by inductive hypothesis. ■

A consequence of the previous lemma is:

Lemma 8 *Let $V \subseteq \mathcal{A}$, $\mathcal{M} = (S, R, L, s_0)$ a model structure, $k = ch(\mathcal{M}, V)$ and $s \in S$.*

- *$(\mathcal{M}, s) \models \mathcal{F}_V(Tr_k(s))$, and*
- *for each $s' \in S$, $(\mathcal{M}, s) \leftrightarrow_{\overline{V}} (\mathcal{M}, s')$ if and only if $(\mathcal{M}, s') \models \mathcal{F}_V(Tr_k(s))$.*

Proof: Let $\phi = \mathcal{F}_V(\mathrm{Tr}_k(s))$, where k is the V-characteristic number of $\mathcal{M}$. $(\mathcal{M}, s) \models \phi$ by the definition of $\mathcal{F}$, and then $\forall s' \in S$, if $s \leftrightarrow_{\overline{V}} s'$ there is $(\mathcal{M}, s') \models \phi$ by Theorem 1 due to $\mathrm{IR}(\phi, \mathcal{A} - V)$. Supposing $(\mathcal{M}, s') \models \phi$, if $s \not\leftrightarrow_{\overline{V}} s'$, then $\mathrm{Tr}_k(s) \not\leftrightarrow_{\overline{V}} \mathrm{Tr}_k(s')$, and then $(\mathcal{M}, s') \not\models \phi$ by Lemma 7, a contradiction. ■

Now we are in the position of proving Theorem 2.
Proof: (i) Let $\mathcal{F}_V(\mathcal{M}, s_0)$ be the characterizing formula of $(\mathcal{M}, s_0)$ on V. It is apparent that $\mathrm{IR}(\mathcal{F}_V(\mathcal{M}, s_0), \overline{V})$. We will show that $(\mathcal{M}, s_0) \models \mathcal{F}_V(\mathcal{M}, s_0)$ at first.

It is apparent that $(\mathcal{M}, s_0) \models \mathcal{F}_V(\mathrm{Tr}_c(s_0))$ by Lemma 7. We must show that $(\mathcal{M}, s_0) \models \bigwedge_{s\in S} G(\mathcal{M}, s)$. Let $\mathcal{X} = \mathcal{F}_V(\mathrm{Tr}_c(s)) \rightarrow \left(\bigwedge_{(s,s_1)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_1))\right) \wedge \mathrm{AX}\left(\bigvee_{(s,s_1)\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_1))\right)$, we will show $\forall s \in S$, $(\mathcal{M}, s_0) \models G(\mathcal{M}, s)$. Where $G(\mathcal{M}, s) = \mathrm{AG}\mathcal{X}$. There are two cases we should consider:

- If $(\mathcal{M}, s_0) \not\models \mathcal{F}_V(\mathrm{Tr}_c(s))$, it is apparent that $(\mathcal{M}, s_0) \models \mathcal{X}$;
- If $(\mathcal{M}, s_0) \models \mathcal{F}_V(\mathrm{Tr}_c(s))$:
  $(\mathcal{M}, s_0) \models \mathcal{F}_V(\mathrm{Tr}_c(s))$
  $\Rightarrow s_0 \leftrightarrow_{\overline{V}} s$ by the definition of characteristic number and Lemma 8.
  For each $(s, s_1) \in R$ there is:
  $(\mathcal{M}, s_1) \models \mathcal{F}_V(\mathrm{Tr}_c(s_1))$ $(s_1 \leftrightarrow_{\overline{V}} s_1)$
  $\Rightarrow (\mathcal{M}, s) \models \bigwedge_{(s,s_1)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_1))$
  $\Rightarrow (\mathcal{M}, s_0) \models \bigwedge_{(s,s_1)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_1))$ (by $\mathrm{IR}(\bigwedge_{(s,s_1)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_1)), \overline{V})$, $s_0 \leftrightarrow_{\overline{V}} s$).
  For each (s, s_1) there is:
  $\mathcal{M}, s_1 \models \bigvee_{(s,s_2)\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_2))$
  $\Rightarrow (\mathcal{M}, s) \models \mathrm{AX}\left(\bigvee_{(s,s_2)\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_2))\right)$
  $\Rightarrow (\mathcal{M}, s_0) \models \mathrm{AX}\left(\bigvee_{(s,s_2)\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_2))\right)$ (by $\mathrm{IR}(\mathrm{AX}\left(\bigvee_{(s,s_2)\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_2))\right), \overline{V})$, $s_0 \leftrightarrow_{\overline{V}} s$)
  $\Rightarrow (\mathcal{M}, s_0) \models \mathcal{X}$.

For any other states s' which can reach from s_0 can be proved similarly, i.e.,, $(\mathcal{M}, s') \models \mathcal{X}$. Therefore, $\forall s \in S$, $(\mathcal{M}, s_0) \models G(\mathcal{M}, s)$, and then $(\mathcal{M}, s_0) \models \mathcal{F}_V(\mathcal{M}, s_0)$.

We will prove this theorem from the following two aspects:

($\Leftarrow$) If $s_0 \leftrightarrow_{\overline{V}} s'_0$, then $(\mathcal{M}', s'_0) \models \mathcal{F}_V(M, s_0)$. Since $(\mathcal{M}, s_0) \models \mathcal{F}_V(\mathcal{M}, s_0)$ and $\mathrm{IR}(\mathcal{F}_V(\mathcal{M}, s_0), \overline{V})$, hence $(\mathcal{M}', s'_0) \models \mathcal{F}_V(M, s_0)$ by Theorem 1.

($\Rightarrow$) If $(\mathcal{M}', s'_0) \models \mathcal{F}_V(M, s_0)$, then $s_0 \leftrightarrow_{\overline{V}} s'_0$. We will prove this by showing that $\forall n \geq 0, Tr_n(s_0) \leftrightarrow_{\overline{V}} Tr_n(s'_0)$.

Base. It is apparent that $Tr_0(s_0) \equiv Tr_0(s'_0)$.

Step. Supposing $\mathrm{Tr}_k(s_0) \leftrightarrow_{\overline{V}} \mathrm{Tr}_k(s'_0)$ $(k > 0)$, we will prove $\mathrm{Tr}_{k+1}(s_0) \leftrightarrow_{\overline{V}} \mathrm{Tr}_{k+1}(s'_0)$. We should only show that $\mathrm{Tr}_1(s_k) \leftrightarrow_{\overline{V}} \mathrm{Tr}_1(s'_k)$. Where $(s_0, s_1), (s_1, s_2), \ldots, (s_{k-1}, s_k) \in R$ and $(s'_0, s'_1), (s'_1, s'_2), \ldots, (s'_{k-1}, s'_k) \in R'$, i.e., s_{i+1} (s'_{i+1}) is an immediate successor of s_i (s'_i) for all $0 \leq i \leq k-1$.

(a) It is apparent that $L(s_k) - \overline{V} = L'(s'_k) - \overline{V}$ by inductive assumption.

Before talking about the other points, note the following fact that:
$(\mathcal{M}', s'_0) \models \mathcal{F}_V(\mathcal{M}, s_0)$
$\Rightarrow \forall s' \in S', (\mathcal{M}', s') \models \mathcal{F}_V(\mathrm{Tr}_c(s)) \rightarrow$
$\left(\bigwedge_{(s,s_1)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_1))\right) \wedge$
$\mathrm{AX}\left(\bigvee_{(s,s_1)\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_1))\right)$ for any $s \in S$. **(fact)**
(I) $(\mathcal{M}', s'_0) \models \mathcal{F}_V(\mathrm{Tr}_c(s_0)) \rightarrow$
$\left(\bigwedge_{(s_0,s_1)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_1))\right) \wedge$
$\mathrm{AX}\left(\bigvee_{(s_0,s_1)\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_1))\right)$ **(fact)**
(II) $(\mathcal{M}', s'_0) \models \mathcal{F}_V(\mathrm{Tr}_c(s_0))$ (known)
(III) $(\mathcal{M}', s'_0) \models \left(\bigwedge_{(s_0,s_1)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_1))\right) \wedge$
$\mathrm{AX}\left(\bigvee_{(s_0,s_1)\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_1))\right)$ ((I),(II))

(b) We will show that for each $(s_k, s_{k+1}) \in R$ there is a $(s'_k, s'_{k+1}) \in R'$ such that $L(s_{k+1}) - \overline{V} = L'(s'_{k+1}) - \overline{V}$.
(1) $(\mathcal{M}', s'_0) \models \bigwedge_{(s_0,s_1)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_1))$ (III)
(2) $\forall (s_0, s_1) \in R$, $\exists (s'_0, s'_1) \in R'$ *s.t.* $(\mathcal{M}', s'_1) \models \mathcal{F}_V(\mathrm{Tr}_c(s_1))$ (2)
(3) $\mathrm{Tr}_c(s_1) \leftrightarrow_{\overline{V}} \mathrm{Tr}_c(s'_1)$ ((2), Lemma 7)
(4) $L(s_1) - \overline{V} = L'(s'_1) - \overline{V}$ ((3), $c \geq 0$)
(5) $(\mathcal{M}', s'_1) \models \mathcal{F}_V(\mathrm{Tr}_c(s_1)) \rightarrow$
$\left(\bigwedge_{(s_1,s_2)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_2))\right) \wedge$
$\mathrm{AX}\left(\bigvee_{(s_1,s_2)\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_2))\right)$ **(fact)**
(6) $(\mathcal{M}', s'_1) \models \left(\bigwedge_{(s_1,s_2)\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_2))\right) \wedge$
$\mathrm{AX}\left(\bigvee_{(s_1,s_2)\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_2))\right)$ ((2), (5))
(7) .....
(8) $(\mathcal{M}', s'_k) \models \left(\bigwedge_{(s_k,s_{k+1})\in R} \mathrm{EX}\mathcal{F}_V(\mathrm{Tr}_c(s_{k+1}))\right) \wedge$
$\mathrm{AX}\left(\bigvee_{(s_k,s_{k+1})\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_{k+1}))\right)$ (similar with (6))
(9) $\forall (s_k, s_{k+1}) \in R$, $\exists (s'_k, s'_{k+1}) \in R'$ *s.t.* $(\mathcal{M}', s'_{k+1}) \models \mathcal{F}_V(\mathrm{Tr}_c(s_{k+1}))$ (8)
(10) $\mathrm{Tr}_c(s_{k+1}) \leftrightarrow_{\overline{V}} \mathrm{Tr}_c(s'_{k+1})$ ((9), Lemma 7)
(11) $L(s_{k+1}) - \overline{V} = L'(s'_{k+1}) - \overline{V}$ ((10), $c \geq 0$)

(c) We will show that for each $(s'_k, s'_{k+1}) \in R'$ there is a $(s_k, s_{k+1}) \in R$ such that $L(s_{k+1}) - \overline{V} = L'(s'_{k+1}) - \overline{V}$.
(1) $(\mathcal{M}', s'_k) \models \mathrm{AX}\left(\bigvee_{(s_k,s_{k+1})\in R} \mathcal{F}_V(\mathrm{Tr}_c(s_{k+1}))\right)$ (by (8) talked above)
(2) $\forall (s'_k, s'_{k+1}) \in R'$, $\exists (s_k, s_{k+1}) \in R$ *s.t.* $(\mathcal{M}', s'_{k+1}) \models \mathcal{F}_V(\mathrm{Tr}_c(s'_{k+1}))$ (1)
(3) $\mathrm{Tr}_c(s_{k+1}) \leftrightarrow_{\overline{V}} \mathrm{Tr}_c(s'_{k+1})$ ((2), Lemma 7)
(4) $L(s_{k+1}) - \overline{V} = L'(s'_{k+1}) - \overline{V}$ ((3), $c \geq 0$)

(ii) This is following Lemma 2 and the definition of the characterizing formula of initial K-structure $\mathcal{K}$ on V. ■

Lemma 3 Let φ be a formula. We have

$$\varphi \equiv \bigvee_{(\mathcal{M},s_0)\in \mathit{Mod}(\varphi)} \mathcal{F}_{\mathcal{A}}(\mathcal{M}, s_0). \quad (4)$$

Proof: Let $(\mathcal{M}', s'_0)$ be a model of φ. Then $(\mathcal{M}', s'_0) \models \bigvee_{(\mathcal{M},s_0)\in \mathit{Mod}(\varphi)} \mathcal{F}_{\mathcal{A}}(\mathcal{M}, s_0)$ due to $(\mathcal{M}', s'_0) \models \mathcal{F}_{\mathcal{A}}(\mathcal{M}', s'_0)$. On the other hand, suppose that $(\mathcal{M}', s'_0)$ is a model of $\bigvee_{(\mathcal{M},s_0)\in \mathit{Mod}(\varphi)} \mathcal{F}_{\mathcal{A}}(\mathcal{M}, s_0)$. Then there is a $(\mathcal{M}, s_0) \in \mathit{Mod}(\varphi)$ such that $(\mathcal{M}', s'_0) \models \mathcal{F}_{\mathcal{A}}(\mathcal{M}, s_0)$. And then $(\mathcal{M}, s_0) \leftrightarrow_{\emptyset} (\mathcal{M}', s'_0)$ by Theorem 2. Therefore, $(\mathcal{M}, s_0)$ is also a model of φ by Theorem 1. ■

Theorem 3 **(Representation theorem)** Let φ, φ' and ϕ be CTL formulas and $V \subseteq \mathcal{A}$. Then the following statements are equivalent:

(i) $\varphi' \equiv \mathrm{F}_{\mathrm{CTL}}(\varphi, V)$,

(ii) $\varphi' \equiv \{\phi | \varphi \models \phi \text{ and } \mathrm{IR}(\phi, V)\}$,

(iii) Postulates (**W**), (**PP**), (**NP**) and (**IR**) hold.

Proof: $(i) \Leftrightarrow (ii)$. To prove this, we will show that:

$$\begin{aligned} &\mathit{Mod}(\mathrm{F}_{\mathrm{CTL}}(\varphi, V)) = \mathit{Mod}(\{\phi | \varphi \models \phi, \mathrm{IR}(\phi, V)\}) \\ &= \mathit{Mod}(\bigvee_{\mathcal{M},s_0\in \mathit{Mod}(\varphi)} \mathcal{F}_{\mathcal{A}-V}(\mathcal{M}, s_0)). \end{aligned}$$

Firstly, suppose that $(\mathcal{M}', s'_0)$ is a model of $\mathrm{F}_{\mathrm{CTL}}(\varphi, V)$. Then there exists an an initial K-structure $(\mathcal{M}, s_0)$ such that $(\mathcal{M}, s_0)$ is a model of φ and $(\mathcal{M}, s_0) \leftrightarrow_V (\mathcal{M}', s'_0)$. By Theorem 1, we have $(\mathcal{M}', s'_0) \models \phi$ for all ϕ that $\varphi \models \phi$ and $\mathrm{IR}(\phi, V)$. Thus, $(\mathcal{M}', s'_0)$ is a model of $\{\phi | \varphi \models \phi, \mathrm{IR}(\phi, V)\}$.

Secondly, suppose that $(\mathcal{M}', s'_0)$ is a models of $\{\phi | \varphi \models \phi, \mathrm{IR}(\phi, V)\}$. Thus, $(\mathcal{M}', s'_0) \models \bigvee_{(\mathcal{M},s_0)\in \mathit{Mod}(\varphi)} \mathcal{F}_{\mathcal{A}-V}(\mathcal{M}, s_0)$ due to $\bigvee_{(\mathcal{M},s_0)\in \mathit{Mod}(\varphi)} \mathcal{F}_{\mathcal{A}-V}(\mathcal{M}, s_0)$ is irrelevant to V.

Finally, suppose that $(\mathcal{M}', s'_0)$ is a model of $\bigvee_{\mathcal{M},s_0\in \mathit{Mod}(\varphi)} \mathcal{F}_{\mathcal{A}-V}(\mathcal{M}, s_0)$. Then there exists $(\mathcal{M}, s_0) \in \mathit{Mod}(\varphi)$ such that $(\mathcal{M}', s'_0) \models \mathcal{F}_{\mathcal{A}-V}(\mathcal{M}, s_0)$.

Hence, $(\mathcal{M}, s_0) \leftrightarrow_V (\mathcal{M}', s_0')$ by Theorem 2. Thus $(\mathcal{M}', s_0')$ is also a model of $\mathrm{F}_{\mathrm{CTL}}(\varphi, V)$.

$(ii) \Rightarrow (iii)$. It is not difficult to prove it.

$(iii) \Rightarrow (ii)$. Suppose that all postulates hold. By Positive Persistence, we have $\varphi' \models \{\phi | \varphi \models \phi, \mathrm{IR}(\phi, V)\}$. Now we show that $\{\phi | \varphi \models \phi, \mathrm{IR}(\phi, V)\} \models \varphi'$. Otherwise, there exists formula ϕ' such that $\varphi' \models \phi'$ but $\{\phi | \varphi \models \phi, \mathrm{IR}(\phi, V)\} \not\models \phi'$. There are three cases:

- ϕ' is relevant to V. Thus, φ' is also relevant to V, a contradiction to Irrelevance.
- ϕ' is irrelevant to V and $\varphi \models \phi'$. This contradicts to our assumption.
- ϕ' is irrelevant to V and $\varphi \not\models \phi'$. By Negative Persistence, $\varphi' \not\models \phi'$, a contradiction.

Thus, φ' is equivalent to $\{\phi | \varphi \models \phi, \mathrm{IR}(\phi, V)\}$. ■

Lemma 4 Let φ and α be two CTL formulae and $q \in \overline{Var(\varphi) \cup Var(\alpha)}$. Then $\mathrm{F}_{\mathrm{CTL}}(\varphi \wedge (q \leftrightarrow \alpha), q) \equiv \varphi$.

Proof: Let $\varphi' = \varphi \wedge (q \leftrightarrow \alpha)$. For any model $(\mathcal{M}, s)$ of $\mathrm{F}_{\mathrm{CTL}}(\Gamma', q)$ there is an initial K-structure $(\mathcal{M}', s')$ s.t. $(\mathcal{M}, s) \leftrightarrow_{\{q\}} (\mathcal{M}', s')$ and $(\mathcal{M}', s') \models \varphi'$. It's apparent that $(\mathcal{M}', s') \models \varphi$, and then $(\mathcal{M}, s) \models \varphi$ since $\mathrm{IR}(\varphi, \{q\})$ and $(\mathcal{M}, s) \leftrightarrow_{\{q\}} (\mathcal{M}', s')$ by Theorem 1.

Let $(\mathcal{M}, s) \in \mathit{Mod}(\varphi)$ with $\mathcal{M} = (S, R, L, s)$. We construct $(\mathcal{M}', s)$ with $\mathcal{M}' = (S, R, L', s)$ as follows:

$L' : S \to \mathcal{A}\ and\ \forall s^* \in S, L'(s^*) = L(s^*)\ if\ (\mathcal{M}, s^*) \not\models \alpha,$
$else\ L'(s^*) = L(s^*) \cup \{q\},$
$L'(s) = L(s) \cup \{q\}\ if\ (\mathcal{M}, s) \models \alpha,\ and\ L'(s) = L(s)$
$otherwise.$

It is clear that $(\mathcal{M}', s) \models \varphi$, $(\mathcal{M}', s) \models q \leftrightarrow \alpha$ and $(\mathcal{M}', s) \leftrightarrow_{\{q\}} (\mathcal{M}, s)$. Therefore $(\mathcal{M}', s) \models \varphi \wedge (q \leftrightarrow \alpha)$, and then $(\mathcal{M}, s) \models \mathrm{F}_{\mathrm{CTL}}(\varphi \wedge (q \leftrightarrow \alpha), q)$ by $(\mathcal{M}', s) \leftrightarrow_{\{q\}} (\mathcal{M}, s)$. ■

Proposition 5 Given a formula $\varphi \in$ CTL, V a set of atoms and p an atom such that $p \notin V$. Then,

$$\mathrm{F}_{\mathrm{CTL}}(\varphi, \{p\} \cup V) \equiv \mathrm{F}_{\mathrm{CTL}}(\mathrm{F}_{\mathrm{CTL}}(\varphi, p), V).$$

Proof: Let $(\mathcal{M}_1, s_1)$ with $\mathcal{M}_1 = (S_1, R_1, L_1, s_1)$ be a model of $\mathrm{F}_{\mathrm{CTL}}(\varphi, \{p\} \cup V)$. By the definition, there exists a model $(\mathcal{M}, s)$ with $\mathcal{M} = (S, R, L, s)$ of φ, such that $(\mathcal{M}_1, s_1) \leftrightarrow_{\{p\} \cup V} (\mathcal{M}, s)$. We construct an initial K-structure $(\mathcal{M}_2, s_2)$ with $\mathcal{M}_2 = (S_2, R_2, L_2, s_2)$ as follows:

(1) for s_2: let s_2 be the state such that:
- $p \in L_2(s_2)$ iff $p \in L_1(s_1)$,
- for all $q \in V$, $q \in L_2(s_2)$ iff $q \in L(s)$,
- for all other atoms q', $q' \in L_2(s_2)$ iff $q' \in L_1(s_1)$ iff $q' \in L(s)$.

(2) for another:
- (i) for all pairs $w \in S$ and $w_1 \in S_1$ such that $w \leftrightarrow_{\{p\} \cup V} w_1$, let $w_2 \in S_2$ and
  - $p \in L_2(w_2)$ iff $p \in L_1(w_1)$,
  - for all $q \in V$, $q \in L_2(w_2)$ iff $q \in L(w)$,
  - for all other atoms q', $q' \in L_2(w_2)$ iff $q' \in L_1(w_1)$ iff $q' \in L(w)$.
- (ii) if $(w_1', w_1) \in R_1$, w_2 is constructed based on w_1 and $w_2' \in S_2$ is constructed based on w_1', then $(w_2', w_2) \in R_2$.

(3) delete duplicated states in S_2 and pairs in R_2.

Then we have $(\mathcal{M}, s) \leftrightarrow_{\{p\}} (\mathcal{M}_2, s_2)$ and $(\mathcal{M}_2, s_2) \leftrightarrow_V (\mathcal{M}_1, s_1)$. Thus, $(\mathcal{M}_2, s_2) \models \mathrm{F}_{\mathrm{CTL}}(\varphi, p)$. And therefore $(\mathcal{M}_1, s_1) \models \mathrm{F}_{\mathrm{CTL}}(\mathrm{F}_{\mathrm{CTL}}(\varphi, p), V)$.

On the other hand, suppose that $(\mathcal{M}_1, s_1)$ be a model of $\mathrm{F}_{\mathrm{CTL}}(\mathrm{F}_{\mathrm{CTL}}(\varphi, p), V)$, then there exists an initial K-structure $(\mathcal{M}_2, s_2)$ such that $(\mathcal{M}_2, s_2) \models \mathrm{F}_{\mathrm{CTL}}(\varphi, p)$ and $(\mathcal{M}_2, s_2) \leftrightarrow_V (\mathcal{M}_1, s_1)$, and there exists $(\mathcal{M}, s)$ such that $(\mathcal{M}, s) \models \varphi$ and $(\mathcal{M}, s) \leftrightarrow_{\{p\}} (\mathcal{M}_2, s_2)$. Therefore, $(\mathcal{M}, s) \leftrightarrow_{\{p\} \cup V} (\mathcal{M}_1, s_1)$ by Proposition 1, and consequently, $(\mathcal{M}_1, s_1) \models \mathrm{F}_{\mathrm{CTL}}(\varphi, \{p\} \cup V)$. ■

Proposition 6 Let φ, φ_i, ψ_i $(i = 1, 2)$ be formulas and $V \subseteq \mathcal{A}$. We have

(i) $\mathrm{F}_{\mathrm{CTL}}(\varphi, V)$ is satisfiable iff φ is;

(ii) If $\varphi_1 \equiv \varphi_2$, then $\mathrm{F}_{\mathrm{CTL}}(\varphi_1, V) \equiv \mathrm{F}_{\mathrm{CTL}}(\varphi_2, V)$;

(iii) If $\varphi_1 \models \varphi_2$, then $\mathrm{F}_{\mathrm{CTL}}(\varphi_1, V) \models \mathrm{F}_{\mathrm{CTL}}(\varphi_2, V)$;

(iv) $\mathrm{F}_{\mathrm{CTL}}(\psi_1 \vee \psi_2, V) \equiv \mathrm{F}_{\mathrm{CTL}}(\psi_1, V) \vee \mathrm{F}_{\mathrm{CTL}}(\psi_2, V)$;

(v) $\mathrm{F}_{\mathrm{CTL}}(\psi_1 \wedge \psi_2, V) \models \mathrm{F}_{\mathrm{CTL}}(\psi_1, V) \wedge \mathrm{F}_{\mathrm{CTL}}(\psi_2, V)$;

Proof: (i) ($\Rightarrow$) Supposing $(\mathcal{M}, s)$ is a model of $\mathrm{F}_{\mathrm{CTL}}(\varphi, V)$, then there is a model $(\mathcal{M}', s')$ of φ s.t. $(\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}', s')$ by the definition of $\mathrm{F}_{\mathrm{CTL}}$.

($\Leftarrow$) Supposing $(\mathcal{M}, s)$ is a model of φ, then there is an initial Kripke structure $(\mathcal{M}', s')$ s.t. $(\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}', s')$, and then $(\mathcal{M}', s') \models \mathrm{F}_{\mathrm{CTL}}(\varphi, V)$ by the definition of $\mathrm{F}_{\mathrm{CTL}}$.

The (ii) and (iii) can be proved similarly.

(iv) ($\Rightarrow$) $\forall (\mathcal{M}, s) \in \mathit{Mod}(\mathrm{F}_{\mathrm{CTL}}(\psi_1 \vee \psi_2, V)), \exists (\mathcal{M}', s') \in \mathit{Mod}(\psi_1 \vee \psi_2)$ s.t. $(\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}', s')$ and $(\mathcal{M}', s') \models \psi_1$ or $(\mathcal{M}', s') \models \psi_2$
$\Rightarrow \exists (\mathcal{M}_1, s_1) \in \mathit{Mod}(\mathrm{F}_{\mathrm{CTL}}(\psi_1, V))$ s.t. $(\mathcal{M}', s') \leftrightarrow_V (\mathcal{M}_1, s_1)$ or $\exists (\mathcal{M}_2, s_2) \in \mathit{Mod}(\mathrm{F}_{\mathrm{CTL}}(\psi_2, V))$ s.t. $(\mathcal{M}', s') \leftrightarrow_V (\mathcal{M}_2, s_2)$
$\Rightarrow (\mathcal{M}, s) \models \mathrm{F}_{\mathrm{CTL}}(\psi_1, V) \vee \mathrm{F}_{\mathrm{CTL}}(\psi_2, V)$ by Theorem 1.

($\Leftarrow$) $\forall (\mathcal{M}, s) \in \mathit{Mod}(\mathrm{F}_{\mathrm{CTL}}(\psi_1, V) \vee \mathrm{F}_{\mathrm{CTL}}(\psi_2, V))$
$\Rightarrow (\mathcal{M}, s) \models \mathrm{F}_{\mathrm{CTL}}(\psi_1, V)$ or $(\mathcal{M}, s) \models \mathrm{F}_{\mathrm{CTL}}(\psi_2, V)$
$\Rightarrow$ there is an initial K-structure $(\mathcal{M}_1, s_1)$ s.t. $(\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}_1, s_1)$ and $(\mathcal{M}_1, s_1) \models \psi_1$ or $(\mathcal{M}_1, s_1) \models \psi_2$
$\Rightarrow (\mathcal{M}_1, s_1) \models \psi_1 \vee \psi_2$
$\Rightarrow$ there is an initial K-structure $(\mathcal{M}_2, s_2)$ s.t. $(\mathcal{M}_1, s_1) \leftrightarrow_V (\mathcal{M}_2, s_2)$ and $(\mathcal{M}_2, s_2) \models \mathrm{F}_{\mathrm{CTL}}(\psi_1 \vee \psi_2, V)$
$\Rightarrow (\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}_2, s_2)$ and $(\mathcal{M}, s) \models \mathrm{F}_{\mathrm{CTL}}(\psi_1 \vee \psi_2, V)$.

The (v) can be proved as (iv). ■

Proposition 7 **(Homogeneity)** Let $V \subseteq \mathcal{A}$ and $\phi \in$ CTL,

(i) $\mathrm{F}_{\mathrm{CTL}}(\mathrm{AX}\phi, V) \equiv \mathrm{AXF}_{\mathrm{CTL}}(\phi, V)$.

(ii) $\mathrm{F}_{\mathrm{CTL}}(\mathrm{EX}\phi, V) \equiv \mathrm{EXF}_{\mathrm{CTL}}(\phi, V)$.

(iii) $\mathrm{F}_{\mathrm{CTL}}(\mathrm{AF}\phi, V) \equiv \mathrm{AFF}_{\mathrm{CTL}}(\phi, V)$.

(iv) $\mathrm{F}_{\mathrm{CTL}}(\mathrm{EF}\phi, V) \equiv \mathrm{EFF}_{\mathrm{CTL}}(\phi, V)$.

Proof: Let $\mathcal{M} = (S, R, L, s_0)$ with initial state s_0 and $\mathcal{M}' = (S', R', L', s'_0)$ with initial state s'_0, then we call $\mathcal{M}', s'_0$ be a sub-structure of $\mathcal{M}, s_0$ if:

- $S' \subseteq S$ and $S' = \{s' | s'$ is reachable from $s'_0\}$,
- $R' = \{(s_1, s_2) | s_1, s_2 \in S'$ and $(s_1, s_2) \in R\}$,
- $L' : S' \to \mathcal{A}$ and $\forall s_1 \in S'$ there is $L'(s_1) = L(s_1)$, and
- s'_0 is s_0 or a state reachable from s_0.

(i) In order to prove $\mathrm{F}_{\mathrm{CTL}}(\mathrm{AX}\phi, V) \equiv \mathrm{AX}(\mathrm{F}_{\mathrm{CTL}}(\phi, V))$, we only need to prove $\mathit{Mod}(\mathrm{F}_{\mathrm{CTL}}(\mathrm{AX}\phi, V)) = \mathit{Mod}(\mathrm{AXF}_{\mathrm{CTL}}(\phi, V))$:

($\Rightarrow$) $\forall(\mathcal{M}', s') \in \mathit{Mod}(\mathrm{F}_{\mathrm{CTL}}(\mathrm{AX}\phi, V))$ there exists an initial K-structure $(\mathcal{M}, s)$ s.t. $(\mathcal{M}, s) \models \mathrm{AX}\phi$ and $(\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}', s')$
$\Rightarrow$ for any sub-structure $(\mathcal{M}_1, s_1)$ of $(\mathcal{M}, s)$ there is $(\mathcal{M}_1, s_1) \models \phi$, where s_1 is a directed successor of s
$\Rightarrow$ there is an initial K-structure $(\mathcal{M}_2, s_2)$ s.t. $(\mathcal{M}_2, s_2) \models \mathrm{F}_{\mathrm{CTL}}(\phi, V)$ and $(\mathcal{M}_2, s_2) \leftrightarrow_V (\mathcal{M}_1, s_1)$
$\Rightarrow$ it is easy to construct an initial K-structure $(\mathcal{M}_3, s_3)$ by $(\mathcal{M}_2, s_2)$ s.t. $(\mathcal{M}_2, s_2)$ is a sub-structure of $(\mathcal{M}_3, s_3)$ with s_2 is a direct successor of s_3 and $(\mathcal{M}_3, s_3) \leftrightarrow_V (\mathcal{M}, s)$
$\Rightarrow (\mathcal{M}_3, s_3) \models \mathrm{AX}(\mathrm{F}_{\mathrm{CTL}}(\phi, V))$ and $(\mathcal{M}_3, s_3) \leftrightarrow_V (\mathcal{M}', s')$
$\Rightarrow (\mathcal{M}', s') \models \mathrm{AX}(\mathrm{F}_{\mathrm{CTL}}(\phi, V))$.

($\Leftarrow$) $\forall\ (\mathcal{M}_3, s_3) \in \mathit{Mod}(\mathrm{AX}(\mathrm{F}_{\mathrm{CTL}}(\phi, V)))$, then for any sub-structure $(\mathcal{M}_2, s_2)$ with s_2 is a directed successor of s_3 there is $(\mathcal{M}_2, s_2) \models \mathrm{F}_{\mathrm{CTL}}(\phi, V)$
$\Rightarrow$ for any $(\mathcal{M}_2, s_2)$ there is an initial K-structure $(\mathcal{M}_1, s_1)$ s.t. $(\mathcal{M}_1, s_1) \models \phi$ and $(\mathcal{M}_1, s_1) \leftrightarrow_V (\mathcal{M}_2, s_2)$
$\Rightarrow$ it is easy to construct an initial structure $(\mathcal{M}, s)$ by $(\mathcal{M}_1, s_1)$ s.t. $(\mathcal{M}_1, s_1)$ is a sub-structure of $(\mathcal{M}, s)$ with s_1 is a direct successor of s and $(\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}_3, s_3)$
$\Rightarrow (\mathcal{M}, s) \models \mathrm{AX}\phi$ and then $(\mathcal{M}_3, s_3) \models \mathrm{F}_{\mathrm{CTL}}(\mathrm{AX}\phi, V)$.

(ii) In order to prove $\mathrm{F}_{\mathrm{CTL}}(\mathrm{EX}\phi, V) \equiv \mathrm{EXF}_{\mathrm{CTL}}(\phi, V)$, we only need to prove $\mathit{Mod}(\mathrm{F}_{\mathrm{CTL}}(\mathrm{EX}\phi, V)) = \mathit{Mod}(\mathrm{EXF}_{\mathrm{CTL}}(\phi, V))$:

($\Rightarrow$) $\forall \mathcal{M}', s' \in \mathit{Mod}(\mathrm{F}_{\mathrm{CTL}}(\mathrm{EX}\phi, V))$ there exists an initial K-structure $(\mathcal{M}, s)$ s.t. $(\mathcal{M}, s) \models \mathrm{EX}\phi$ and $(\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}', s')$
$\Rightarrow$ there is a sub-structure $(\mathcal{M}_1, s_1)$ of $(\mathcal{M}, s)$ s.t. $(\mathcal{M}_1, s_1) \models \phi$, where s_1 is a directed successor of s
$\Rightarrow$ there is an initial K-structure $(\mathcal{M}_2, s_2)$ s.t. $(\mathcal{M}_2, s_2) \models \mathrm{F}_{\mathrm{CTL}}(\phi, V)$ and $(\mathcal{M}_2, s_2) \leftrightarrow_V (\mathcal{M}_1, s_1)$
$\Rightarrow$ it is easy to construct an initial K-structure $(\mathcal{M}_3, s_3)$ by $(\mathcal{M}_2, s_2)$ s.t. $(\mathcal{M}_2, s_2)$ is a sub-structure of $(\mathcal{M}_3, s_3)$ that s_2 is a direct successor of s_3 and $(\mathcal{M}_3, s_3) \leftrightarrow_V (\mathcal{M}, s)$
$\Rightarrow (\mathcal{M}_3, s_3) \models \mathrm{EX}(\mathrm{F}_{\mathrm{CTL}}(\phi, V))$
$\Rightarrow (\mathcal{M}', s') \models \mathrm{EX}(\mathrm{F}_{\mathrm{CTL}}(\phi, V))$.

($\Leftarrow$) $\forall\ (\mathcal{M}_3, s_3) \in \mathit{Mod}(\mathrm{EX}(\mathrm{F}_{\mathrm{CTL}}(\phi, V)))$, then there exists a sub-structure $(\mathcal{M}_2, s_2)$ of $(\mathcal{M}_3, s_3)$ s.t. $(\mathcal{M}_2, s_2) \models \mathrm{F}_{\mathrm{CTL}}(\phi, V)$
$\Rightarrow$ there is an initial K-structure $(\mathcal{M}_1, s_1)$ s.t. $(\mathcal{M}_1, s_1) \models \phi$ and $(\mathcal{M}_1, s_1) \leftrightarrow_V (\mathcal{M}_2, s_2)$
$\Rightarrow$ it is easy to construct an initial K-structure $(\mathcal{M}, s)$ by $(\mathcal{M}_1, s_1)$ s.t. $(\mathcal{M}_1, s_1)$ is a sub-structure of $(\mathcal{M}, s)$ that s_1 is a direct successor of s and $(\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}_3, s_3)$
$\Rightarrow (\mathcal{M}, s) \models \mathrm{EX}\phi$ and then $(\mathcal{M}_3, s_3) \models \mathrm{F}_{\mathrm{CTL}}(\mathrm{EX}\phi, V)$.

(iii) and (iV) can be proved as (i) and (ii) respectively. ■

Proposition8 **(Model Checking on Forgetting)** Let $(\mathcal{M}, s_0)$ be an initial K-structure, φ be a CTL formula and V a set of atoms. Deciding whether $(\mathcal{M}, s_0)$ is a model of $\mathrm{F}_{\mathrm{CTL}}(\varphi, V)$ is NP-complete.

Proof: The problem can be determined by the following two things: (1) guessing an initial K-structure $(\mathcal{M}', s'_0)$ satisfying φ; and (2) checking if $(\mathcal{M}, s_0) \leftrightarrow_V (\mathcal{M}', s'_0)$. Both two steps can be done in polynomial time. Hence, the problem is in NP. The hardness follows that the model checking for propositional variable forgetting is NP-hard (Zhang and Zhou 2008). ■

Theorem 5 **(Entailment on Forgetting)** Let φ and ψ be two $\mathrm{CTL}_{\mathrm{AF}}$ formulas and V a set of atoms. Then, results:

(i) deciding $\mathrm{F}_{\mathrm{CTL}}(\varphi, V) \models^? \psi$ is co-NP-complete,

(ii) deciding $\psi \models^? \mathrm{F}_{\mathrm{CTL}}(\varphi, V)$ is Π_2^P-complete,

(iii) deciding $\mathrm{F}_{\mathrm{CTL}}(\varphi, V) \models^? \mathrm{F}_{\mathrm{CTL}}(\psi, V)$ is Π_2^P-complete.

Proof: (1) It is proved that deciding whether ψ is satisfiable is NP-Complete (Meier et al. 2015). The hardness is easy to see by setting $\mathrm{F}_{\mathrm{CTL}}(\varphi, \mathit{Var}(\varphi)) \equiv \top$, i.e., deciding whether ψ is valid. For membership, from Theorem 3, we have $\mathrm{F}_{\mathrm{CTL}}(\varphi, V) \models \psi$ iff $\varphi \models \psi$ and $IR(\psi, V)$. Clearly, in $\mathrm{CTL}_{\mathrm{AF}}$, deciding $\varphi \models \psi$ is in co-NP. We show that deciding whether $IR(\psi, V)$ is also in co-NP. Without loss of generality, we assume that ψ is satisfiable. We consider the complement of the problem: deciding whether ψ is not irrelevant to V. It is easy to see that ψ is not irrelevant to V iff there exist a model $(\mathcal{M}, s_0)$ of ψ and an initial K-structure $(\mathcal{M}', s'_0)$ such that $(\mathcal{M}, s_0) \leftrightarrow_V (\mathcal{M}', s'_0)$ and $(\mathcal{M}', s'_0) \not\models \psi$. So checking whether ψ is not irrelevant to V can be achieved in the following steps: (1) guess two initial K-structures $(\mathcal{M}, s_0)$ and $(\mathcal{M}', s'_0)$, (2) check if $(\mathcal{M}, s_0) \models \psi$ and $(\mathcal{M}', s'_0) \not\models \psi$, and (3) check $(\mathcal{M}, s_0) \leftrightarrow_V (\mathcal{M}', s'_0)$. Obviously (1) can be done in polynomial time and also (2) and (3) can be done in polynomial time.

(2) Membership. We consider the complement of the problem. We may guess an initial K-structure $(\mathcal{M}, s_0)$ and check whether $(\mathcal{M}, s_0) \models \psi$ and $(\mathcal{M}, s_0) \not\models \mathrm{F}_{\mathrm{CTL}}(\varphi, V)$. From Proposition 8, we know that this is in Σ_2^P. So the original problem is in Π_2^P. Hardness. Let $\psi \equiv \top$. Then the problem is reduced to decide $\mathrm{F}_{\mathrm{CTL}}(\varphi, V)$'s validity. Since a propositional variable forgetting is a special case temporal forgetting, the hardness is directly followed from the proof of Proposition 24 in (Lang, Liberatore, and Marquis 2003).

(3) Membership. If $\mathrm{F}_{\mathrm{CTL}}(\varphi, V) \not\models \mathrm{F}_{\mathrm{CTL}}(\psi, V)$ then there exist an initial K-structure $(\mathcal{M}, s)$ such that $(\mathcal{M}, s) \models \mathrm{F}_{\mathrm{CTL}}(\varphi, V)$ but $(\mathcal{M}, s) \not\models \mathrm{F}_{\mathrm{CTL}}(\psi, V)$, i.e.,, there is $(\mathcal{M}_1, s_1) \leftrightarrow_V (\mathcal{M}, s)$ with $(\mathcal{M}_1, s_1) \models \varphi$ but $(\mathcal{M}_2, s_2) \not\models \psi$ for every $(\mathcal{M}_2, s_2)$ with $(\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}_2, s_2)$. It is evident that guessing such $(\mathcal{M}, s)$, $(\mathcal{M}_1, s_1)$ with $(\mathcal{M}_1, s_1) \leftrightarrow_V (\mathcal{M}, s)$ and checking $(\mathcal{M}_1, s_1) \models \varphi$ are feasible while checking $(\mathcal{M}_2, s_2) \not\models \psi$ for every $(\mathcal{M}, s) \leftrightarrow_V (\mathcal{M}_2, s_2)$ can be done in polynomial time. Thus the problem is in Π_2^P.

Hardness. It follows from (2) due to the fact that $\mathrm{F}_{\mathrm{CTL}}(\varphi, V) \models \mathrm{F}_{\mathrm{CTL}}(\psi, V)$ iff $\varphi \models \mathrm{F}_{\mathrm{CTL}}(\psi, V)$ thanks to $IR(\mathrm{F}_{\mathrm{CTL}}(\psi, V), V)$.

■

Proposition 9 **(dual)** Let V, q, φ and ψ are like in Definition 5. The ψ is a SNC (WSC) of q on V under φ iff $\neg\psi$ is a WSC (SNC) of $\neg q$ on V under φ.

Proof: (i) Suppose ψ is the SNC of q. Then $\varphi \models q \rightarrow \psi$. Thus $\varphi \models \neg\psi \rightarrow \neg q$. So $\neg\psi$ is a SC of $\neg q$. Suppose ψ' is any other SC of $\neg q$: $\varphi \models \psi' \rightarrow \neg q$. Then $\varphi \models q \rightarrow \neg\psi'$, this means $\neg\psi'$ is a NC of q on P under φ. Thus $\varphi \models \psi \rightarrow \neg\psi'$ by assumption. So $\varphi \models \psi' \rightarrow \neg\psi$. This proves that $\neg\psi$ is the WSC of $\neg q$. The proof of the other part of the proposition is similar.

(ii) The WSC case can be proved similarly with SNC case. ■

Proposition 10 Let Γ and α be two formulas, $V \subseteq \mathit{Var}(\alpha) \cup \mathit{Var}(\phi)$ and q is a new proposition not in Γ and α. Then, a formula φ of V is the SNC (WSC) of α on V under Γ iff it is the SNC (WSC) of q on V under $\Gamma' = \Gamma \cup \{q \leftrightarrow \alpha\}$.

Proof: We prove this for SNC. The case for WSC is similar. Let $\mathit{SNC}(\varphi, \alpha, V, \Gamma)$ denote that φ is the SNC of α on V under Γ, and $\mathit{NC}(\varphi, \alpha, V, \Gamma)$ denote that φ is the NC of α on V under Γ.

($\Rightarrow$) We will show that if $\mathit{SNC}(\varphi, \alpha, V, \Gamma)$ holds, then $\mathit{SNC}(\varphi, q, V, \Gamma')$ will be true. According to $\mathit{SNC}(\varphi, \alpha, V, \Gamma)$ and $\alpha \equiv q$, we have $\Gamma' \models q \rightarrow \varphi$, which means φ is a NC of q on V under Γ'. Suppose φ' is any NC of q on V under Γ', then $\mathrm{F}_{\mathrm{CTL}}(\Gamma', q) \models \alpha \rightarrow \varphi'$ due to $\alpha \equiv q$, $\mathit{IR}(\alpha \rightarrow \varphi', \{q\})$ and (**PP**), i.e., $\Gamma \models \alpha \rightarrow \varphi'$ by Lemma 4, this means $\mathit{NC}(\varphi', \alpha, V, \Gamma)$. Therefore, $\Gamma \models \varphi \rightarrow \varphi'$ by the definition of SNC and $\Gamma' \models \varphi \rightarrow \varphi'$. Hence, $\mathit{SNC}(\varphi, q, V, \Gamma')$ holds.

($\Leftarrow$) We will show that if $\mathit{SNC}(\varphi, q, V, \Gamma')$ holds, then $\mathit{SNC}(\varphi, \alpha, V, \Gamma)$ will be true. According to $\mathit{SNC}(\varphi, q, V, \Gamma')$, it's not difficult to know that $\mathrm{F}_{\mathrm{CTL}}(\Gamma', \{q\}) \models \alpha \rightarrow \varphi$ due to $\alpha \equiv q$, $\mathit{IR}(\alpha \rightarrow \varphi, \{q\})$ and (**PP**), i.e., $\Gamma \models \alpha \rightarrow \varphi$ by Lemma 4, this means $\mathit{NC}(\varphi, \alpha, V, \Gamma)$. Suppose φ' is any NC of α on V under Γ. Then $\Gamma' \models q \rightarrow \varphi'$ since $\alpha \equiv q$ and $\Gamma' = \Gamma \cup \{q \equiv \alpha\}$, which means $\mathit{NC}(\varphi', q, V, \Gamma')$. According to $\mathit{SNC}(\varphi, q, V, \Gamma')$, $\mathit{IR}(\varphi \rightarrow \varphi', \{q\})$ and (**PP**), we have $\mathrm{F}_{\mathrm{CTL}}(\Gamma', \{q\}) \models \varphi \rightarrow \varphi'$, and $\Gamma \models \varphi \rightarrow \varphi'$ by Lemma 4. Hence, $\mathit{SNC}(\varphi, \alpha, V, \Gamma)$ holds. ■

Theorem 7 Let φ be a formula, $V \subseteq \mathit{Var}(\varphi)$ and $q \in \mathit{Var}(\varphi) - V$.

(i) $\mathrm{F}_{\mathrm{CTL}}(\varphi \wedge q, (\mathit{Var}(\varphi) \cup \{q\}) - V)$ is a SNC of q on V under φ.

(ii) $\neg\mathrm{F}_{\mathrm{CTL}}(\varphi \wedge \neg q, (\mathit{Var}(\varphi) \cup \{q\}) - V)$ is a WSC of q on V under φ.

Proof: We will prove the SNC part, while it is not difficult to prove the WSC part according to Proposition 9. Let $\mathcal{F} = \mathrm{F}_{\mathrm{CTL}}(\varphi \wedge q, (\mathit{Var}(\varphi) \cup \{q\}) - V)$.

The "NC" part: It's easy to see that $\varphi \wedge q \models \mathcal{F}$ by **(W)**. Hence, $\varphi \models q \rightarrow \mathcal{F}$, this means $\mathcal{F}$ is a NC of q on P under φ.

The "SNC" part: for all ψ', ψ' is the NC of q on V under φ, s.t. $\varphi \models \mathcal{F} \rightarrow \psi'$. Suppose that there is a NC ψ of q on V under φ and ψ is not logic equivalence with $\mathcal{F}$ under φ, s.t. $\varphi \models \psi \rightarrow \mathcal{F}$. We know that $\varphi \wedge q \models \psi$ iff $\mathcal{F} \models \psi$ by **(PP)**, since $\mathit{IR}(\psi, (\mathit{Var}(\varphi) \cup \{q\}) - V)$. Hence, $\varphi \wedge \mathcal{F} \models \psi$ by $\varphi \wedge q \models \psi$ (by suppose). We can see that $\varphi \wedge \psi \models \mathcal{F}$ by suppose. Therefore, $\varphi \models \psi \leftrightarrow \mathcal{F}$, which means ψ is logic equivalence with $\mathcal{F}$ under φ. This is contradict with the suppose. Then $\mathcal{F}$ is the SNC of q on P under φ. ■

Theorem 8 Let $\mathcal{K} = (\mathcal{M}, s)$ be an initial K-structure with $\mathcal{M} = (S, R, L, s_0)$ on the set $\mathcal{A}$ of atoms, $V \subseteq \mathcal{A}$ and $q \in V' = \mathcal{A} - V$. Then:

(i) the SNC of q on V under $\mathcal{K}$ is $\mathrm{F}_{\mathrm{CTL}}(\mathcal{F}_{\mathcal{A}}(\mathcal{K}) \wedge q, V')$.

(ii) the WSC of q on V under $\mathcal{K}$ is $\neg\mathrm{F}_{\mathrm{CTL}}(\mathcal{F}_{\mathcal{A}}(\mathcal{K}) \wedge \neg q, V')$.

Proof: (i) As we know that any initial K-structure $\mathcal{K}$ can be described as a characterizing formula $\mathcal{F}_{\mathcal{A}}(\mathcal{K})$, then the SNC of q on V under $\mathcal{F}_{\mathcal{A}}(\mathcal{K})$ is $\mathrm{F}_{\mathrm{CTL}}(\mathcal{F}_{\mathcal{A}}(\mathcal{K}) \wedge q, \mathcal{A} - V)$.

(ii) This is proved by the dual property. ■

Proposition?? Let φ be a CTL formula and $V \subseteq \mathcal{A}$ with $|\mathcal{A}| = m$ and $|V| = n$. The time and space complexity of Algorithm **??** are $O(2^{m*2^m})$.

Proof: The time and space spent by Algorithm **??** is mainly the **for** cycles between lines 4 and 18. Under a given number i of states, there are i^i number of relations, i^{2^m} number of label functions and i number of possible initial states. In this case, we need the memory for the initial K-model in each time is $(i + i^i + i^{2^m} + 1)$.

For each $1 \leq i \leq 2^m$, there is at most $i * i^i * i^{2^m} * i = i^2 * i^{(i+2^m)}$ possible initial K-models. Suppose that we can obtain an initial K-models in unit time (at each step), then we require $(2^m)^2 * (2^m)^{(2^m+2^m)} = (2^m)^{(2+2*2^m)}$ steps in the worst case.

Let $k = m - n$, for any initial K-structure $\mathcal{K} = (\mathcal{M}, s_0)$ with $i \geq 1$ nodes, in the worst, i.e., $ch(\mathcal{M}, V) = i$, we will spend $N(i)$ space to store the characterizing formula.

$$\begin{aligned} N(i) &= (k + (\cdots + (k + 2ik) * (2i)) \cdots * (2i)) \\ &= (2i)^0 k + 2ik + (2i)^2 k + \cdots + (2i)^{(i-1)} k \\ &= \frac{(2i)^i - 1}{2i - 1} k. \end{aligned}$$

In the worst case, i.e., there is $i = 2^m$ initial K-structure with 2^m nodes, we will spent $2^m * N(i)$ space to store the result of forgetting.

It is obvious that computing the V-characterization number of any initial K-structure $\mathcal{K}$ dose not more than $O(i^2 P)$ with P expressing a polynomial function. Therefore, the time and space complexity are $O(2^{(m*2^m)})$. ■

References

Baier, C., and Katoen, J. 2008. *Principles of Model Checking*. The MIT Press.

Bolotov, A. 1999. A clausal resolution method for ctl branching-time temporal logic. *Journal of Experimental & Theoretical Artificial Intelligence* 11(1):77–93.

Bolotov, A. 2000. *Clausal resolution for branching-time temporal logic*. Ph.D. Dissertation, Manchester Metropolitan University.

Browne, M. C.; Clarke, E. M.; and Grumberg, O. 1988. Characterizing finite kripke structures in propositional temporal logic. *Theor. Comput. Sci.* 59:115–131.

Clarke, E. M., and Emerson, E. A. 1981. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Workshop on Logic of Programs*, 52–71. Springer.

Clarke, E. M.; Emerson, E. A.; and Sistla, A. P. 1986. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. Program. Lang. Syst.* 8(2):244–263.

Dailler, S.; Hauzar, D.; Marché, C.; and Moy, Y. 2018. Instrumenting a weakest precondition calculus for counterexample generation. *Journal of logical and algebraic methods in programming* 99:97–113.

Dijkstra, E. W. 1978. Guarded commands, nondeterminacy, and formal derivation of programs. In *Programming Methodology*. Springer. 166–175.

Eiter, T., and Kern-Isberner, G. 2019. A brief survey on forgetting from a knowledge representation and reasoning perspective. *KI-Künstliche Intelligenz* 33(1):9–33.

Eiter, T., and Wang, K. 2008. *Semantic forgetting in answer set programming*. Elsevier Science Publishers Ltd.

Fang, L.; Liu, Y.; and Van Ditmarsch, H. 2019. Forgetting in multi-agent modal logics. *Artificial Intelligence* 266:51–80.

Lang, J., and Marquis, P. 2008. On propositional definability. *Artificial Intelligence* 172(8):991–1017.

Lang, J., and Marquis, P. 2010. *Reasoning under inconsistency: a forgetting-based approach*. Elsevier Science Publishers Ltd.

Lang, J.; Liberatore, P.; and Marquis, P. 2003. Propositional independence: Formula-variable independence and forgetting. *J. Artif. Intell. Res.* 18:391–443.

Lin, F., and Reiter, R. 1994. Forget it. In *Working Notes of AAAI Fall Symposium on Relevance*, 154–159.

Lin, F. 2001. On strongest necessary and weakest sufficient conditions. *Artif. Intell.* 128(1-2):143–159.

Lin, F. 2003. Compiling causal theories to successor state axioms and strips-like systems. *Journal of Artificial Intelligence Research* 19:279–314.

Liu, Y., and Wen, X. 2011. On the progression of knowledge in the situation calculus. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 976–982. Barcelona, Catalonia, Spain: IJCAI/AAAI.

Lutz, C., and Wolter, F. 2011. Foundations for uniform interpolation and forgetting in expressive description logics. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 989–995. Barcelona, Catalonia, Spain: IJCAI/AAAI.

Meier, A.; Thomas, M.; Vollmer, H.; and Mundhenk, M. 2015. Erratum: The complexity of satisfiability for fragments of CTL and ctl*. *Int. J. Found. Comput. Sci.* 26(8):1189.

Su, K.; Sattar, A.; Lv, G.; and Zhang, Y. 2009. Variable forgetting in reasoning about knowledge. *Journal of Artificial Intelligence Research* 35:677–716.

Wang, Z.; Wang, K.; Topor, R. W.; and Pan, J. Z. 2010. Forgetting for knowledge bases in DL-Lite. *Annuals of Mathematics and Artificial Intelligence* 58(1-2):117–151.

Wang, Y.; Zhang, Y.; Zhou, Y.; and Zhang, M. 2012. Forgetting in logic programs under strong equivalence. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Thirteenth International Conference*, 643–647. Rome, Italy: AAAI Press.

Wang, Y.; Wang, K.; and Zhang, M. 2013. Forgetting for answer set programs revisited. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, 1162–1168. Beijing, China: IJCAI/AAAI.

Wong, K.-S. 2009. *Forgetting in Logic Programs*. Ph.D. Dissertation, The University of New South Wales.

Woodcock, J. C., and Morgan, C. 1990. Refinement of state-based concurrent systems. In *International Symposium of VDM Europe*, 340–351. Springer.

Zhang, Y., and Foo, N. Y. 2006. Solving logic program conflict through strong and weak forgettings. *Artificial Intelligence* 170(8-9):739–778.

Zhang, Y., and Zhou, Y. 2008. Properties of knowledge forgetting. In Pagnucco, M., and Thielscher, M., eds., *Proceedings of the Twelfth International Workshop on Non-Monotonic Reasoning*, 68–75.

Zhang, Y., and Zhou, Y. 2009. Knowledge forgetting: Properties and applications. *Artificial Intelligence* 173(16-17):1525–1537.

Zhang, Y.; Foo, N. Y.; and Wang, K. 2005. Solving logic program conflict through strong and weak forgettings. In *Ijcai-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, Uk, July 30-August*, 627–634.

Zhang, L.; Hustadt, U.; and Dixon, C. 2008. First-order resolution for ctl. Technical report, Citeseer.

Zhang, L.; Hustadt, U.; and Dixon, C. 2009. A refined resolution calculus for ctl. In *International Conference on Automated Deduction*, 245–260. Springer.

Zhao, Y., and Schmidt, R. A. 2017. Role forgetting for alcoqh (δ)-ontologies using an ackermann-based approach. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 1354–1361. AAAI Press.