PAPER *Special Section on Formal Approach*

Pre- and Post-Conditions Expressed in Variants of the Modal μ-Calculus*

Yoshinori TANABE[†,†††a)], ***Member***, **Toshifusa SEKIZAWA**[††,†††], **Yoshifumi YUASA**[††††,††], ***and*** **Koichi TAKAHASHI**[††], ***Nonmembers***

SUMMARY Properties of Kripke structures can be expressed by formulas of the modal μ-calculus. Despite its strong expressive power, the validity problem of the modal μ-calculus is decidable, and so are some of its variants enriched by inverse programs, graded modalities, and nominals. In this study, we show that the pre- and post-conditions of transformations of Kripke structures, such as addition/deletion of states and edges, can be expressed using variants of the modal μ-calculus. Combined with decision procedures we have developed for those variants, the properties of sequences of transformations on Kripke structures can be deduced. We show that these techniques can be used to verify the properties of pointer-manipulating programs.

key words: *modal μ-calculus, Kripke structure, precondition, postcondition, pointer*

1. Introduction

In previous studies, we applied temporal logics to verification problems in areas such as concurrent garbage collection [1] and one-dimensional cellular automata [2]. The targets of the studies are considered as graph transformation systems. The basic idea of the analysis is to regard the graphs as Kripke structures and express their properties using formulas of temporal logics such as computational tree logic (CTL). They have sufficient expressive power to describe the properties of the systems and their validity problems are decidable. Although CTL has been successfully applied to the above-mentioned target systems, we need more expressive power to undertake similar approaches for more complicated systems.

First, we use general fixed-point operators, which play a key role in expressing graph properties such as reachability. While CTL has fixed-point operators, which is the main reason we employed this logic as the analysis tool, its usage is restricted to fixed patterns, such as EU or AG. Using general fixed-point operators μ and ν, one can express more complicated properties.

Second, we use *nominals* [3], which are a type of atomic formulas that are satisfied by one and only one state in a Kripke structure. Nominals can be used, for example, to express pointer-type variables of a programming language — when a state of a Kripke structure satisfies a nominal, that state is regarded as the value of the corresponding variable. A propositional symbol cannot be substituted for a nominal since it may be satisfied by two or more states while the value of a variable should be unique.

The third point is with regard to functional modalities. While an ordinary modality m is interpreted in a Kripke structure as a relation $R(m)$, a functional modality f is interpreted as a (partial) function $R(f)$; that is, for each state s, there is at most one s' such that $(s, s') \in R(f)$. They can be used to express, for example, pointer-type fields of a structure in a programming language such as C, just as nominals express pointer-type variables.

The fourth point is *backward modalities*. A backward modality m^{-1}, where m is an ordinary (forward) modality, follows the transition relation of a Kripke structure in the reverse direction. We have already used them in [2]. They are vital for computation of the weakest precondition, as shown in Sect. 3.

Thus, our logic $\mathcal{L}$ has nominals and functional and backward modalities. It can be considered as a variant of *enriched μ-calculi* [4]. Formulas of the logic express properties of Kripke structures.

Pre- and post-conditions play an important role when we reason about the properties of Kripke structures with regard to programs that transform them. We list basic transformations of Kripke structures, such as the addition of states or modification of transition relations, and show that the weakest preconditions can also be expressed in $\mathcal{L}$, extending our previous work [5]. Although $\mathcal{L}$ is not validity-decidable [6], there is a sound (although incomplete) decision procedure [7] for validity if we restrict ourselves to the alternation-free fragment. The expressive power of the fragment is sufficient for our application which we describe later. By combining these two techniques, we can conduct

Manuscript received July 24, 2008.
Manuscript revised November 19, 2008.

[†]The author is with the Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, 101–0021 Japan.
[††]The authors are with the Research Center for Verification and Semantics, National Institute of Advanced Industrial Science and Technology, Toyonaka-shi, 560–0083 Japan.
[†††]The author is with the Graduate School of Information Science and Technology, Osaka University, Suita-shi, 565–0871 Japan.
[††††]The author is with the Graduate School of Information Science and Engineering, Tokyo Institute of Technology, Tokyo, 152–8552 Japan.

*This research was supported by the research project "Solving the description explosion problem in verification by means of structure transformation" in Core Research for Evolution Science and Technology (CREST) program of Japan Science and Technology Agency.

a) E-mail: y-tanabe@ci.i.u-tokyo.ac.jp

DOI: 10.1587/transinf.E92.D.995

backward reasoning on properties expressed in $\mathcal{L}$.

In order to enable forward reasoning, we show that the strongest postconditions can also be expressed in $\mathcal{L}$. Moreover, we show that the strongest postcondition of a property expressed in $\mathcal{L}'$ is also expressed in $\mathcal{L}'$, where $\mathcal{L}'$ is the sublogic of $\mathcal{L}$ obtained by removing the backward modalities. Logic $\mathcal{L}'$ is validity-decidable [4], and our decision procedure [7] for $\mathcal{L}'$ is complete.

As an application of the results, we illustrate how the properties of pointer-manipulating programs are verified. We regard a heap as a Kripke structure. Then each program statement that manipulates pointers can be regarded as a transformation of Kripke structures. The properties of the heap necessary for verification, such as loop invariants, are expressed as formulas in our logic. Then, forward or backward reasoning described above is applied to obtain verification results.

Various studies have analyzed programs that manipulate pointers. In one approach, a three-valued logic is used in addition to the first-order logic, enhanced with an operator to take the transitive closure [8]. Another approach uses Separation Logic [9], which is an extension of Hoare logic, and has operators to handle the status of the heap. Our approach differs in that we use logics with decision procedures for validity testing.

The rest of the paper is organized as follows. In Sect. 2, we define the syntax and semantics of the logic, and introduce transformations of Kripke structures. Preconditions, which are used in backward reasoning, are introduced in Sect. 3, while postconditions for forward reasoning are discussed in Sect. 4. We show examples of verification in Sect. 5. Finally, Sect. 6 concludes the study.

2. Preliminaries

2.1 Syntax

Let PS, Nom, PV, GMS, and FMS be countable sets. Elements of these sets are called *propositional symbols*, *nominals*, *propositional variables*, *general modality symbols*, and *functional modality symbols*, respectively. The set Mod of *modalities* and the set Form of *formulas* are defined as follows.

$$\text{Mod} \ni m ::= \mathbf{o} \mid g \mid f \mid g^{-1} \mid f^{-1}$$

$$\text{Form} \ni \varphi ::= p \mid x \mid X \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle m \rangle \varphi \mid \mu X \varphi$$

where $p \in \text{PS}$, $x \in \text{Nom}$, $X \in \text{PV}$, $g \in \text{GMS}$, and $f \in \text{FMS}$. In $\mu X\varphi$, all free occurrences of X in φ (i.e., occurrences of X that are not bound by μX that occurs in φ) must be positive (i.e., the number of subformulas of $\mu X\varphi$ that contains the occurrence and that is in the form of $\neg\psi$ must be even). We define Atom = PS $\cup$ Nom and MS = GMS $\cup$ FMS, and call their elements *atomic formulas* and *modality symbols*, respectively. Modality $\mathbf{o}$ is called the *global modality*. A modality in the form of m^{-1}, where $m \in \text{MS}$, is called a *backward modality*. We assume that Nom contains an element called **nil**.

This language is denoted by $\mathcal{L}$. The language without backward modalities is denoted by $\mathcal{L}'$. That is, $\mathcal{L}'$ is obtained from $\mathcal{L}$ by replacing the definition of Mod by the following:

$$\text{Mod} \ni m ::= \mathbf{o} \mid g \mid f.$$

The following standard abbreviations are used: **true** = $p \vee \neg p$ for some fixed $p \in \text{PS}$, **false** = $\neg$**true**, $\varphi_1 \wedge \varphi_2 = \neg(\neg\varphi_1 \vee \neg\varphi_2)$, $\varphi_1 \to \varphi_2 = \neg\varphi_1 \vee \varphi_2$, $[m]\varphi = \neg\langle m\rangle\neg\varphi$, and $\nu X\varphi = \neg\mu X\neg\varphi[\neg X/X]$. Here, $\varphi[\psi/\chi]$ is the formula obtained from φ by substituting ψ in place of χ. (The standard restriction for substitution is applied.)

2.2 Semantics

A *Kripke structure* for $\mathcal{L}$ or $\mathcal{L}'$ is a tuple $\mathcal{K} = (S, R, L, \text{nil})$ that satisfies the following conditions. We denote the powerset of S by $\mathcal{P}(S)$.

- nil is an element of set S.
- $R : \text{MS} \to \mathcal{P}(S \times S)$. For $f \in \text{FMS}$ and $s \in S$, there is at most one $s' \in S$ such that $(s, s') \in R(f)$.
- $L : \text{Atom} \to \mathcal{P}(S)$. $L(x)$ is a singleton if $x \in \text{Nom}$.
- $L(\mathbf{nil}) = \{\text{nil}\}$; and $(\text{nil}, s) \notin R(m)$ and $(s, \text{nil}) \notin R(m)$ for $s \in S$ and $m \in \text{MS}$.

For $x \in \text{Nom}$, we denote by $L'(x)$ the unique element of $L(x)$, that is, $L(x) = \{L'(x)\}$. For $f \in \text{FMS}$ and $s \in S$, we define $R(f, s) \in S$ by $R(f, s) = s'$ if there exists $s' \in S$ such that $(s, s') \in R(f)$; otherwise, $R(f, s) = \text{nil}$.

The domain of R is extended to Mod:

- $R(\mathbf{o}) = S \times S$.
- $R(m^{-1}) = (R(m))^{-1}$ for $m \in \text{MS}$.

A function $\rho : \text{PV} \to \mathcal{P}(S)$ is called a *valuation* for $\mathcal{K}$. The *interpretation* $[\![\varphi]\!]^{\mathcal{K},\rho} \subseteq S$ of formula φ is defined as follows. Symbols $\mathcal{K}$ and/or ρ is omitted if no confusion occurs. For function F, we denote by $F[a \mapsto b]$ a function G defined by $\text{dom}(G) = \text{dom}(F) \cup \{a\}$, $G(a) = b$, and $G(x) = F(x)$ for $x \in \text{dom}(F) \setminus \{a\}$.

- $[\![a]\!] = L(a)$ for $a \in \text{Atom}$.
- $[\![X]\!] = \rho(X)$ for $X \in \text{PV}$.
- $[\![\neg\varphi]\!] = S \setminus [\![\varphi]\!]$.
- $[\![\varphi_1 \vee \varphi_2]\!] = [\![\varphi_1]\!] \cup [\![\varphi_2]\!]$.
- $[\![\langle m\rangle\varphi]\!] = \{s \in S \mid \exists s' \in S.\, (s, s') \in R(m) \text{ and } s' \in [\![\varphi]\!]\}$.
- $[\![\mu X\varphi]\!] = \bigcap\{T \subseteq S \mid [\![\varphi]\!]^{\rho[X\mapsto T]} \subseteq T\}$.

We write $\mathcal{K}, \rho, s \models \varphi$ if $s \in [\![\varphi]\!]^{\mathcal{K},\rho}$. Again, $\mathcal{K}$ and/or ρ are often omitted. We write $\mathcal{K} \models \varphi$ if $\mathcal{K}, \rho, s \models \varphi$ holds for any valuation ρ and $s \in S$. Formulas φ and φ' are *equivalent* ($\varphi \equiv \varphi'$) if $[\![\varphi]\!]^{\mathcal{K},\rho} = [\![\varphi']\!]^{\mathcal{K},\rho}$ for any $\mathcal{K}$ and ρ. A formula is *valid* if it is equivalent to **true**.

For nominal x and formulas φ, φ_1, and φ_2, we define $@x\varphi = \langle\mathbf{o}\rangle(x \wedge \varphi)$ and $\varphi_1 \to \varphi_2\,;\varphi_3 = (\varphi_1 \wedge \varphi_2) \vee (\neg\varphi_1 \wedge \varphi_3)$. Intuitive meaning of $@x\varphi$ is "φ holds at the state that satisfies x", and that of $\varphi_1 \to \varphi_2\,;\varphi_3$ is "if φ_1 then φ_2 else φ_3". Obviously, $@x\varphi \equiv [\mathbf{o}](x \to \varphi)$ and $\varphi_1 \to \varphi_2\,;\varphi_3 \equiv (\varphi_1 \to \varphi_2) \wedge (\neg\varphi_1 \to \varphi_3)$ hold.

2.3 Nesting of Fixed-Point Operators and Global Modalities

In this section, we prove a few technical lemmas needed in later sections.

The letters λ and λ' are used to denote fixed-point operator μ or ν. Thus, $\lambda X\varphi$ is either $\mu X\varphi$ or $\nu X\varphi$. The symbol $\{\}$ denotes $\langle\rangle$ or $[]$. Thus, $\{\mathbf{o}\}$ is either $\langle\mathbf{o}\rangle$ or $[\mathbf{o}]$.

A formula is in *positive normal form (PNF)* if the negation symbol ($\neg$) only appears immediately before an atomic formula or a propositional variable. For every formula φ, there is a formula ψ in PNF such that $\varphi \equiv \psi$.

A formula in PNF *alternates* if it has a subformula in the form of $\lambda X\psi$, ψ has a subformula in the form of $\lambda' Y\chi$ and χ has free occurrences of X, where $\lambda \neq \lambda'$ and $X \neq Y$. A formula in PNF is *alternation-free* if it does not alternate.

Lemma 1: Assume that $\lambda X\varphi$ is in PNF and contains $\psi_0 = \{\mathbf{o}\}\psi$ as its subformula. Further assume that there is no subformula of φ in the form of $\lambda' Y\eta$ such that η contains ψ_0 as a subformula. Let $\varphi^{\mathrm{T}} = \varphi[\mathbf{true}/\psi_0]$, $\varphi^{\mathrm{F}} = \varphi[\mathbf{false}/\psi_0]$, $\psi^{\mathrm{T}} = \psi[\lambda X\varphi^{\mathrm{T}}/X]$, and $\psi^{\mathrm{F}} = \psi[\lambda X\varphi^{\mathrm{F}}/X]$. Then, the following holds:

$$\lambda X\varphi \equiv \begin{cases} \{\mathbf{o}\}\psi^{\mathrm{F}} \to \mu X\varphi^{\mathrm{T}}\,;\mu X\varphi^{\mathrm{F}} & \text{if } \lambda = \mu \\ \{\mathbf{o}\}\psi^{\mathrm{T}} \to \nu X\varphi^{\mathrm{T}}\,;\nu X\varphi^{\mathrm{F}} & \text{if } \lambda = \nu \end{cases}$$

Proof : We show only the first half, as the second half can be shown in a similar manner.

Let $\mathcal{K} = (S, R, L, \mathrm{nil})$ be a Kripke structure, v be a valuation for $\mathcal{K}$ and $s \in S$. We show:

(a) When $\mathcal{K}, v \models \{\mathbf{o}\}\psi^{\mathrm{F}}$, $\quad [\![\mu X\varphi^{\mathrm{T}}]\!]^{\mathcal{K},v} = [\![\mu X\varphi]\!]^{\mathcal{K},v}$
(b) When $\mathcal{K}, v \not\models \{\mathbf{o}\}\psi^{\mathrm{F}}$, $\quad [\![\mu X\varphi]\!]^{\mathcal{K},v} = [\![\mu X\varphi^{\mathrm{F}}]\!]^{\mathcal{K},v}$

Note that whether or not $\{\mathbf{o}\}\psi^{\mathrm{F}}$ is satisfied is independent of $s \in S$. In both cases, the right hand side is clearly a subset of the left hand side since φ is in PNF, so we show the other inclusion.

For formula ξ and ordinal number α, we define $S(\xi, \alpha) \subseteq S$ as follows.

- $S(\xi, 0) = \varnothing$
- $S(\xi, \alpha + 1) = [\![\varphi]\!]^{\mathcal{K}, v[X \mapsto S(\xi,\alpha)]}$
- $S(\xi, \alpha) = \bigcup_{\beta<\alpha} S(\xi, \beta)$ (if α is limit)

It is well known that $S(\xi, \alpha) \subseteq S(\xi, \beta)$ if $\alpha < \beta$, $S(\xi, \alpha) \subseteq [\![\mu X\xi]\!]$ for any α, and there is α such that $S(\xi, \alpha) = [\![\mu X\xi]\!]$.

Let κ be the least ordinal such that $S(\varphi, \kappa) = [\![\mu X\varphi]\!]$. Ordinals κ^{T} and κ^{F} are defined similarly for φ^{T} and φ^{F}, respectively.

(a) Assume $\mathcal{K}, v \models \{\mathbf{o}\}\psi^{\mathrm{F}}$. We will show by induction on α that

$$S(\varphi^{\mathrm{T}}, \alpha) \subseteq S(\varphi, \kappa^{\mathrm{F}} + \alpha). \tag{1}$$

Once it is established, by taking $\alpha = \kappa^{\mathrm{T}}$, we have $[\![\mu X\varphi^{\mathrm{T}}]\!] = S(\varphi^{\mathrm{T}}, \kappa^{\mathrm{T}}) \subseteq S(\varphi, \kappa^{\mathrm{F}} + \kappa^{\mathrm{T}}) \subseteq [\![\mu X\varphi]\!]$, which is to be proved in (a).

When $\alpha = 0$ or α is limit, (1) is clearly satisfied. For $\alpha + 1$, we have $S(\varphi^{\mathrm{T}}, \alpha + 1) = [\![\varphi^{\mathrm{T}}]\!]^{v[X\mapsto S(\varphi^{\mathrm{T}},\alpha)]} \subseteq [\![\varphi^{\mathrm{T}}]\!]^{v[X\mapsto S(\varphi,\kappa^{\mathrm{F}}+\alpha)]}$ by induction hypothesis. On the other hand, $S(\varphi, \kappa^{\mathrm{F}} + \alpha + 1) = [\![\varphi]\!]^{v[X\mapsto S(\varphi,\kappa^{\mathrm{F}}+\alpha)]}$ by definition. Therefore it is sufficient to show

$$\mathcal{K}, v[X \mapsto S(\varphi, \kappa^{\mathrm{F}} + \alpha)], s \models \varphi \leftrightarrow \varphi^{\mathrm{T}}. \tag{2}$$

From the assumption $\mathcal{K}, v \models \{\mathbf{o}\}\psi[\mu X\varphi^{\mathrm{F}}/X]$ holds. Therefore $\mathcal{K}, v[X \mapsto S(\varphi^{\mathrm{F}}, \kappa^{\mathrm{F}})] \models \{\mathbf{o}\}\psi$. Since $S(\varphi^{\mathrm{F}}, \kappa^{\mathrm{F}}) \subseteq S(\varphi, \kappa^{\mathrm{F}}) \subseteq S(\varphi, \kappa^{\mathrm{F}} + \alpha)$, we have $\mathcal{K}, v[X \mapsto S(\varphi, \kappa^{\mathrm{F}} + \alpha)] \models \{\mathbf{o}\}\psi$, which means $\mathcal{K}, v[X \mapsto S(\varphi, \kappa^{\mathrm{F}} + \alpha)] \models \{\mathbf{o}\}\psi \leftrightarrow \mathbf{true}$. Now (2) follows from the definition of φ^{T}.

(b) Assume $\mathcal{K}, v \not\models \{\mathbf{o}\}\psi^{\mathrm{F}}$. Then, this implies $\mathcal{K}, v[X \mapsto S(\varphi^{\mathrm{F}}, \kappa^{\mathrm{F}})], s \models \varphi \leftrightarrow \varphi^{\mathrm{F}}$ with a similar argument as in part (a),

We show $S(\varphi, \alpha) \subseteq S(\varphi^{\mathrm{F}}, \alpha)$ by induction on α. We only show the case of successor ordinals since the other cases are trivial. $S(\varphi, \alpha + 1) = [\![\varphi]\!]^{v[X\mapsto S(\varphi,\alpha)]} \subseteq [\![\varphi]\!]^{v[X\mapsto S(\varphi^{\mathrm{F}},\alpha)]} = [\![\varphi^{\mathrm{F}}]\!]^{v[X\mapsto S(\varphi^{\mathrm{F}},\alpha)]} = S(\varphi^{\mathrm{F}}, \alpha + 1)$.

Then, $[\![\mu X\varphi]\!] = S(\varphi, \kappa) \subseteq S(\varphi^{\mathrm{F}}, \kappa) \subseteq [\![\mu X\varphi^{\mathrm{F}}]\!]$ and we are done. □

A formula is *FG-free* if, for all its subformulas lead by *f*ixed-point operators, that is, subformulas in the form of $\lambda X\psi$, the *g*lobal modality does not occur in ψ. A formula is *GV-free* if, for all its subformulas lead by the *g*lobal modality, that is, subformulas in the form of $\{\mathbf{o}\}\psi$, no free *v*ariable occurs in ψ. Clearly, any closed FG-free formula is GV-free.

Lemma 2: If formula φ is FG-free, then there is an FG-free formula that is equivalent to $\lambda X\varphi$.

Proof : Induction on the number of occurrences of the global modality in φ. If the number is zero, the conclusion trivially holds. Assume the number is positive and take a subformula $\psi_0 = \{\mathbf{o}\}\psi$ of φ. Take φ^{T} and φ^{F} as in Lemma 1, which can be applied since φ is FG-free. By the induction hypothesis, there are FG-free formulas ξ^{T} and ξ^{F} that are equivalent to $\lambda X\varphi^{\mathrm{T}}$ and $\lambda X\varphi^{\mathrm{F}}$, respectively. Then, $\lambda X\varphi \equiv \{\mathbf{o}\}\psi[\xi^{\mathrm{F}}/X] \to \xi^{\mathrm{T}}\,;\xi^{\mathrm{F}}$ if $\lambda = \mu$, and $\lambda X\varphi \equiv \{\mathbf{o}\}\psi[\xi^{\mathrm{T}}/X] \to \xi^{\mathrm{T}}\,;\xi^{\mathrm{F}}$ if $\lambda = \nu$. In both cases, the right hand side is FG-free. □

Lemma 3: For any formula φ, there is an FG-free formula that is equivalent to φ.

Proof : Induction on the construction of the formula. In the case of μ and ν, we can use Lemma 2. The other cases are trivial. □

2.4 Transformations of Kripke Structures

In this section, we introduce several transformations of Kripke structures. Formally, a transformation is defined as a relation on the class of all Kripke structures.

We consider the following transformations of Kripke

Table 1 Transformations of Kripke structures and their preconditions.

τ	Condition for $(\mathcal{K}_1, \mathcal{K}_2) \in \tau$	*desc* for $T = \mathrm{pre}'(\tau, \psi)$
$\mathrm{CN}_1(x_1, x_2)$	$L_2 = L_1[x_1 \mapsto L_1(x_2)]$	$x_1 \rightarrowtail x_2$
$\mathrm{CN}_2(x_1, x_2, f)$	$L_2 = L_1[x_1 \mapsto \{R_1(f, L_1'(x_2))\}]$	$x_1 \rightarrowtail (@x_2 \langle f \rangle \mathbf{true} \rightarrow \langle f^{-1} \rangle x_2 \,;\, \mathbf{nil})$
$\mathrm{AP}(x, p)$	$L_2 = L_1[p \mapsto L_1(p) \cup L_1(x)]$	$p \rightarrowtail p \vee x$
$\mathrm{DP}(x, p)$	$L_2 = L_1[p \mapsto L_1(p) \setminus L_1(x)]$	$p \rightarrowtail p \wedge \neg x$
$\mathrm{AT}(m, x_1, x_2)$	$R_2 = R_1[m \mapsto R_1(m) \cup \{(L_1'(x_1), L_1'(x_2))\}]$, $L_1'(x_1) \neq \mathrm{nil}_1$, $L_1'(x_2) \neq \mathrm{nil}_1$	$\langle m \rangle \varphi \rightarrowtail \langle m \rangle T(\varphi) \vee (x_1 \wedge @x_2\, T(\varphi))$; $\langle m^{-1} \rangle \varphi \rightarrowtail \langle m^{-1} \rangle T(\varphi) \vee (x_2 \wedge @x_1\, T(\varphi))$
$\mathrm{DT}_1(m, x_1, x_2)$	$R_2 = R_1[m \mapsto R_1(m) \setminus \{(L_1'(x_1), L_1'(x_2))\}]$	$\langle m \rangle \varphi \rightarrowtail (\neg x_1 \wedge \langle m \rangle T(\varphi)) \vee (\langle m \rangle (\neg x_2 \wedge T(\varphi)))$; $\langle m^{-1} \rangle \varphi \rightarrowtail (\neg x_2 \wedge \langle m^{-1} \rangle T(\varphi)) \vee (\langle m^{-1} \rangle (\neg x_1 \wedge T(\varphi)))$
$\mathrm{DT}_2(f, x)$	$R_2 = R_1[f \mapsto R_1(f) \setminus \{(L_1'(x), R_1(f, L_1'(x)))\}]$	$\langle f \rangle \varphi \rightarrowtail \neg x \wedge \langle f \rangle T(\varphi)$; $\langle f^{-1} \rangle \varphi \rightarrowtail \langle f^{-1} \rangle (\neg x \wedge T(\varphi))$
$\mathrm{AS}(x)$	$\exists s \in S_2.\ S_2 = S_1 \uplus \{s\}, L_2 = L_1[x \mapsto \{s\}]$	$x \rightarrowtail \mathbf{false}$; $\langle \mathbf{o} \rangle \varphi \rightarrowtail \mathrm{ns}(x, \varphi) \vee \langle \mathbf{o} \rangle T(\varphi)$
$\mathrm{DS}(x)$	$S_2 = S_1 \setminus L_1(x)$, $R_2 = R_1 \setminus ((L_1(x) \times S_1) \cup (S_1 \times L_1(x)))$, $L_2 = L_1[y \mapsto \{\mathrm{nil}_1\} \mid y \in \mathrm{Nom}, L_1(y) = L_1(x)]$, $L_1'(x) \neq \mathrm{nil}_1$	$x' \rightarrowtail (@x\, x' \rightarrow \mathbf{nil}\,;x')$ $(x' \in \mathrm{Nom})$; $\langle m \rangle \varphi \rightarrowtail \langle m \rangle (\neg x \wedge T(\varphi))$ $(m \in \mathrm{Mod})$

structures. They are fundamental transformations – adding and removing states and transitions, changing values of the labeling function at a state that can be uniquely identified by a nominal and a functional modality. More complex transformation can be defined by combining them. In the following description, we assume $x, x_1, x_2, y \in \mathrm{Nom}$, $m \in \mathrm{MS}$, $f \in \mathrm{FMS}$, $p \in \mathrm{PS}$, and $\mathcal{K} = (K, R, L, \mathrm{nil})$ is a Kripke structure.

- $\mathrm{CN}_1(x_1, x_2)$: *C*hanges the state that satisfies *n*ominal x_1 to $L'(x_2)$.
- $\mathrm{CN}_2(x_1, x_2, f)$: *C*hanges the state that satisfies *n*ominal x_1 to $R(f, L'(x_2))$.
- $\mathrm{AP}(p, x)$: *A*dds $L'(x)$ to the interpretation of *p*ropositional symbol p.
- $\mathrm{DP}(p, x)$: *D*eletes $L'(x)$ from the interpretation of *p*ropositional symbol p.
- $\mathrm{AT}(m, x_1, x_2)$: *A*dds a *t*ransition for m from $L'(x_1)$ to $L'(x_2)$. Side condition: $L'(x_1)$ and $L'(x_2)$ must not be nil.
- $\mathrm{DT}_1(m, x_1, x_2)$: *D*eletes a *t*ransition for m from $L'(x_1)$ to $L'(x_2)$, if it exists.
- $\mathrm{DT}_2(f, x_1)$: *D*eletes a *t*ransition for f from $L'(x_1)$, if it exists.
- $\mathrm{AS}(x)$: *A*dds a *s*tate and makes it $L'(x)$.
- $\mathrm{DS}(x)$: *D*eletes the *s*tate $L'(x)$. Any transition to and from the state is also deleted. If y is a nominal and $L'(y)$ is the deleted state, then $L'(y)$ becomes nil. Side condition: $L'(x)$ must not be nil.

We denote the set of transformations listed above by Tr. Precise definitions of the transformations are given in the left column of Table 1. We assume that $\mathcal{K}_i = (S_i, R_i, L_i, \mathrm{nil}_i)$ $(i = 1, 2)$ are Kripke structures. For each $\tau \in \mathrm{Tr}$, the condition for $(\mathcal{K}_1, \mathcal{K}_2) \in \tau$ is described in the table. Members of the tuple not explicitly referred to in the table should be identical. For example, let $\tau = \mathrm{CN}_2(x_1, x_2, f)$. Since only L_2 is mentioned in the table, $S_2 = S_1$, $R_2 = R_1$, and $\mathrm{nil}_2 = \mathrm{nil}_1$, i.e., the underlying set, the transition relation, and the nil element is not changed. L_2 is also identical to L_1, except for $L_2(x_1)$ is $\{R_1(f, L_1'(x_2))\}$, i.e., in $\mathcal{K}_2$, x_1 is satisfied by the f-successor state of x_2.

Our main application is analysis of pointer-manipulating programs. As shown in Sect. 5, a program is regarded as a transformation of Kripke structure. We use a small C-like programming language PML [10] to describe such programs. The set Tr covers all the basic statements in PML.

3. Preconditions

In this section, for each closed formula ψ in $\mathcal{L}$ and $\tau \in \mathrm{Tr}$, we introduce a formula $\mathrm{pre}(\tau, \psi)$ that expresses the weakest precondition of ψ with respect to τ.

In order to make definitions concise, we introduce a notation $\mathrm{IDef}\{desc\}$, which defines a formula $T(\varphi)$ from a given formula φ according to *desc*, where *desc* is a semicolon-separated list of entries in the form of $\alpha \rightarrowtail \beta$. If φ appears as α of an entry $\alpha \rightarrowtail \beta$ of *desc*, $T(\varphi) = \beta$. Otherwise, the following rule is applied: $T(a) = a$ $(a \in \mathrm{Atom} \cup \mathrm{PV})$, $T(\neg\varphi) = \neg T(\varphi)$, $T(\varphi_1 \vee \varphi_2) = T(\varphi_1) \vee T(\varphi_2)$, $T(\langle m \rangle \varphi) = \langle m \rangle T(\varphi)$, and $T(\mu X \varphi) = \mu X\, T(\varphi)$. In both cases, if T appears in the right hand side, this procedure is applied recursively. The formula $T(\varphi)$ thus defined is denoted by $\mathrm{IDef}\{desc\}(\varphi)$. For example, let $x \in \mathrm{Nom}$. If $S(\varphi) = \mathrm{IDef}\{x \rightarrowtail \neg x\,;\, \langle f \rangle \varphi \rightarrowtail \langle f \rangle (x \wedge T(\varphi))\ \ (f \in \mathrm{FMS})\}(\varphi)$, then $S(x) = \neg x$, but for $x' \in \mathrm{Nom} \setminus \{x\}$, $S(x') = x'$. $S(\langle f \rangle \varphi) = \langle f \rangle (x \wedge S(\varphi))$ for $f \in \mathrm{FMS}$, but $S(\langle m \rangle \varphi) = \langle m \rangle S(\varphi)$ for $m \in \mathrm{MS} \setminus \mathrm{FMS}$.

Using this notation, we define two auxiliary formulas simultaneously. First, $\mathrm{pre}'(\tau, \psi) = \mathrm{IDef}\{desc\}(\psi)$, where *desc* is defined in the right column of Table 1. Second, $\mathrm{ns}(x, \psi) = \mathrm{IDef}\{x \rightarrowtail \mathbf{true}\ ;\ a \rightarrowtail \mathbf{false}\ \ (a \in \mathrm{Atom} \setminus \{x\})\ ;\ \langle \mathbf{o} \rangle \varphi \rightarrowtail T(\varphi) \vee \langle \mathbf{o} \rangle \mathrm{pre}'(\mathrm{AS}(x), \varphi)\ ;\ \langle m \rangle \varphi \rightarrowtail \mathbf{false}\ \ (m \in \mathrm{Mod} \setminus \{\mathbf{o}\})\}(\psi)$, where $x \in \mathrm{Nom}$.

Intuitively, $\mathrm{pre}'(\tau, \psi)$ claims that the current state satisfies ψ in the Kripke structure $\mathcal{K}'$ obtained by applying τ to the current Kripke structure $\mathcal{K}$; $\mathrm{ns}(x, \psi)$ claims that, when $\tau = \mathrm{AS}(x)$, the newly added state satisfies ψ in $\mathcal{K}'$. Formally, their meaning is expressed in the following lemma. Assume φ is a formula in $\mathcal{L}$, $\mathcal{K}_i = (S_i, R_i, L_i, \mathrm{nil}_i)$ $(i = 1, 2)$ are Kripke structures, ρ_1 and ρ_2 are valuations for $\mathcal{K}_1$ and $\mathcal{K}_2$, respectively, such that $\rho_1(X) \cap S_2 = \rho_2(X) \cap S_1$, $\tau \in \mathrm{Tr}$, and $(\mathcal{K}_1, \mathcal{K}_2) \in \tau$.

Lemma 4:

(1) Let $\tau \in \mathrm{Tr}$ be a transformation other than $\mathrm{AS}(x)$. For any $s \in S_1 \cap S_2$, the following holds.

$$\mathcal{K}_1, \rho_1, s \models \mathrm{pre}'(\tau, \varphi) \iff \mathcal{K}_2, \rho_2, s \models \varphi$$

(2) Assume $x \in \mathrm{Nom}$, $\tau = \mathrm{AS}(x)$, and φ is GV-free. Let $\hat{s} = L_2(x)$, that is, $S_2 = S_1 \uplus \{\hat{s}\}$. We define a valuation ρ'_1 for $\mathcal{K}_1$ by $\rho'_1(X) = S_1$ if $\hat{s} \in \rho_2(X)$ and $\rho'_1(X) = \varnothing$ if $\hat{s} \notin \rho_2(X)$. Then, the following hold.

- Formulas $\mathrm{pre}'(\tau, \varphi)$ and $\mathrm{ns}(x, \varphi)$ are GV-free.
- For $s \in S_1$,

$$\mathcal{K}_1, \rho_1, s \models \mathrm{pre}'(\tau, \varphi) \iff \mathcal{K}_2, \rho_2, s \models \varphi.$$

- $\mathcal{K}_2, \rho_2, \hat{s} \models \varphi \iff [\![\mathrm{ns}(x, \varphi)]\!]^{\mathcal{K}_1, \rho'_1} = S_1$.
- $\mathcal{K}_2, \rho_2, \hat{s} \not\models \varphi \iff [\![\mathrm{ns}(x, \varphi)]\!]^{\mathcal{K}_1, \rho'_1} = \varnothing$.

Proof :
(1) The proof uses induction on the construction of φ. Since all cases can be shown in a straightforward manner, we show only one case, $\tau = \mathrm{AT}(m, x_1, x_2)$.

Let $\varphi = \langle m \rangle \psi$. In this case, $\mathrm{pre}'(\tau, \varphi) = \langle m \rangle \mathrm{pre}'(\tau, \psi) \vee (x_1 \wedge @x_2 \, \mathrm{pre}'(\tau, \psi))$. Assume that $\mathcal{K}_1, \rho_1, s \models \mathrm{pre}'(\tau, \varphi)$ holds. If $\mathcal{K}_1, \rho_1, s \models \langle m \rangle \mathrm{pre}'(\tau, \psi)$ holds, there is $s' \in S_1$ such that $(s, s') \in R_1(m)$ and $\mathcal{K}_1, \rho_1, s' \models \mathrm{pre}'(\tau, \psi)$. By induction hypothesis, $\mathcal{K}_2, \rho_2, s' \models \psi$, and since $R_1(m) \subseteq R_2(m)$, we have $\mathcal{K}_2, \rho_2, s \models \varphi$. If $\mathcal{K}_1, \rho_1, s \models x_1 \wedge @x_2 \, \mathrm{pre}'(\tau, \psi)$, then $s = L_1(x_1) = L_2(x_1)$ and $\mathcal{K}_1, \rho_1, L_1(x_2) \models \mathrm{pre}'(\tau, \psi)$. By induction hypothesis $\mathcal{K}_2, \rho_2, L_2(x_2) \models \psi$. Since $(L_2(x_1), L_2(x_2)) \in R_2(m)$, we have $\mathcal{K}_2, \rho_2, s \models \varphi$. The other direction can be shown similarly.

(2) The first item can be easily checked.

Let $s \in S_1$. We check the second item: $\mathcal{K}_1, \rho_1, s \models \mathrm{pre}'(\tau, \varphi) \iff \mathcal{K}_2, \rho_2, s \models \varphi$. The only non-trivial case is $\langle \mathbf{o} \rangle \psi$. Assume $\varphi = \langle \mathbf{o} \rangle \psi$ and $\mathcal{K}_1, \rho_1, s \models \mathrm{pre}'(\tau, \varphi) = \mathrm{ns}(x, \psi) \vee \langle \mathbf{o} \rangle \mathrm{pre}'(\tau, \psi)$. If $\mathcal{K}_1, \rho_1, s \models \langle \mathbf{o} \rangle \mathrm{pre}'(\tau, \psi)$, then there is $s' \in S_1$ such that $\mathcal{K}_1, \rho_1, s' \models \mathrm{pre}'(\tau, \psi)$. By induction hypothesis, $\mathcal{K}_2, \rho_2, s' \models \psi$, therefore $\mathcal{K}_2, \rho_2, s \models \varphi$. When $\mathcal{K}_1, \rho_1, s \models \mathrm{ns}(x, \psi)$ holds, note that there is no free variable in ψ since φ is GV-free. Therefore $\mathcal{K}_1, \rho'_1, s \models \mathrm{ns}(x, \psi)$ also holds. That means $[\![\mathrm{ns}(x, \psi)]\!]^{\mathcal{K}, \rho'_1} \neq \varnothing$. By induction hypothesis, we have $\mathcal{K}_2, \rho_2, \hat{s} \models \psi$, which implies $\mathcal{K}_2, \rho_2, s \models \varphi$. The other direction can be proved similarly.

The remaining two items on ns can be shown in a similar manner, by using the fact that φ is GV-free, as in the previous paragraph. □

For $\tau \in \mathrm{Tr}$ and *closed* formula φ, we define formula $\mathrm{pre}(\tau, \varphi)$ as follows.

$$\mathrm{pre}(\tau, \varphi) = \begin{cases} \mathrm{pre}'(\tau, \hat{\varphi}) \wedge \mathrm{ns}(x, \hat{\varphi}) & \text{if } \tau = \mathrm{AS}(x) \\ [\mathbf{o}](\neg x \to \mathrm{pre}'(\tau, \varphi)) & \text{if } \tau = \mathrm{DS}(x) \\ \mathrm{pre}'(\tau, \varphi) & \text{otherwise} \end{cases}$$

where $\hat{\varphi}$ is an FG-free formula that is equivalent to φ. Its existence is guaranteed by Lemma 3.

Theorem 1:

$$\mathcal{K}_1 \models \mathrm{pre}(\tau, \varphi) \iff \mathcal{K}_2 \models \varphi$$

Proof : We show only the case where $\tau = \mathrm{AS}(x)$, the others can also be shown without difficulty.

Since $\hat{\varphi}$ is a closed FG-free formula, it is GV-free. Therefore Lemma 4 can be applied. Assume $\mathcal{K}_1 \models \mathrm{pre}(\tau, \varphi)$ and $s \in S_2$. If $s \in S_1$, since $\mathcal{K}_1, s \models \mathrm{pre}'(\tau, \varphi)$, we have $\mathcal{K}_2, s \models \varphi$. If $s \notin S_1$, that is, if $s = \hat{s}$, $\mathcal{K}_2, s \models \varphi$ also holds since $S_1 = [\![\mathrm{ns}(x, \hat{\varphi})]\!]^{\mathcal{K}_1}$ holds. (Note that φ is closed and the right hand side does not depend on a valuation.) Thus, we have $\mathcal{K}_2 \models \varphi$. The other direction is similar. □

Theorem 1 claims that formula $\mathrm{pre}(\tau, \varphi)$ is the weakest precondition in the following sense: let us call ψ a *precondition* of φ with respect to τ if $(\mathcal{K}_1, \mathcal{K}_2) \in \tau$ and $\mathcal{K}_1 \models \psi$ implies $\mathcal{K}_2 \models \varphi$. Then, Theorem 1 implies (1) $\mathrm{pre}(\tau, \varphi)$ is a precondition, and (2) if ψ is a precondition and $\mathcal{K} \models \psi$, then $\mathcal{K} \models \varphi$.

Thus, we can calculate the weakest precondition within the logic $\mathcal{L}$. Although $\mathcal{L}$ is not validity-decidable [6], sound (but incomplete) decision procedures can be built. Combined with such procedures, we can reason about the properties of Kripke structures with respect to transformations.

We have defined a sublogic $\mathcal{L}'$ of $\mathcal{L}$. It is desirable to find a formula in $\mathcal{L}'$ with the property of Theorem 1 because $\mathcal{L}'$ is validity-decidable [4]. Formula $\mathrm{pre}(\tau, \varphi)$ does not always belong to $\mathcal{L}'$ since it may contain backward modalities when $\tau = \mathrm{CN}_2$. The question arises whether an equivalent formula exists within $\mathcal{L}'$, when φ is in $\mathcal{L}'$. Unfortunately, the answer is negative. To see this, let us recall some definitions.

Relation $H \subseteq S_1 \times S_2$ is a *simulation* for $\mathcal{L}'$ from $\mathcal{K}_1$ to $\mathcal{K}_2$ if (1) for any $s_1, s'_1 \in S_1$, $s_2 \in S_2$, $m \in \mathrm{Mod}$ such that $(s_1, s'_1) \in R_1(m)$ and $(s_1, s_2) \in H$, there exists $s'_2 \in S_2$ such that $(s_2, s'_2) \in R_2(m)$ and $(s'_1, s'_2) \in H$, and (2) for any $s_1 \in S_1$, $s_2 \in S_2$ and $a \in \mathrm{Atom}$ such that $(s_1, s_2) \in H$, $s_1 \in L_1(a) \iff s_2 \in L_2(a)$ holds. Relation H is a *bisimulation* for $\mathcal{L}'$ between $\mathcal{K}_1$ and $\mathcal{K}_2$, if H and H^{-1} are simulations. The following lemma is well known.

Lemma 5: Assume $\mathcal{K}_1$ and $\mathcal{K}_2$ are Kripke structures, $s_1 \in S_1$, $s_2 \in S_2$, φ is a closed formula in $\mathcal{L}'$, and H is a bisimulation for $\mathcal{L}'$ between $\mathcal{K}_1$ and $\mathcal{K}_2$. If $(s_1, s_2) \in H$, we have $\mathcal{K}_1, s_1 \models \varphi \iff \mathcal{K}_2, s_2 \models \varphi$.

Let $x, y, z \in \mathrm{Nom}$, $f \in \mathrm{FMS}$, $\varphi_1 = @x \langle f \rangle z$, $\tau = \mathrm{CN}_2(z, y, f)$, and $\varphi_2 = \mathrm{pre}'(\tau, \varphi_1) = @x \langle f \rangle \langle f^{-1} \rangle y$.

Proposition 1: There is no formula in $\mathcal{L}'$ that is equivalent to φ_2.

Proof : Let ψ_2 be a formula equivalent to φ_2. Note that ψ_2 must be closed. We define two Kripke structures $\mathcal{K}_i = (S_i, R_i, L_i, \mathrm{nil}_i)$ $(i = 1, 2)$ as follows: $S_1 = \{s_x, s_y, s_1, \mathrm{nil}_1\}$, $R_1(f) = \{(s_x, s_1), (s_y, s_1)\}$, $L_1(x) = \{s_x\}$, $L_1(y) = \{s_y\}$, $S_2 = \{t_x, t_y, t_1, t_2, \mathrm{nil}_2\}$, $R_2(f) = \{(t_x, t_1), (t_y, t_2)\}$, $L_2(x) = \{t_x\}$, and $L_2(y) = \{t_y\}$. Clearly, $\mathcal{K}_1 \models$

Table 2 Postconditions.

τ	$\mathrm{post}_1(\tau,\psi)$	$\mathrm{post}_2(\tau,\psi)$
$\mathrm{CN}_1(x_1,x_2)$	$@_{x_2}\, x_1$	$\mathrm{pre}'(\mathrm{CN}_1(x_1,y),\psi)$
$\mathrm{CN}_2(x_1,x_2,f)$	$@_{x_2}\langle f\rangle x_1$	$\mathrm{pre}'(\mathrm{CN}_1(x_1,y),\psi)$
$\mathrm{AP}(x,p)$	$@_x\, p$	$\psi \vee \mathrm{pre}'(\mathrm{DP}(x,p),\psi)$
$\mathrm{DP}(x,p)$	$@_x \neg p$	$\psi \vee \mathrm{pre}'(\mathrm{AP}(x,p),\psi)$
$\mathrm{AT}(m,x_1,x_2)$	$@_{x_1}\langle m\rangle x_2$	$\psi \vee \mathrm{pre}'(\mathrm{DT}_1(m,x_1,x_2),\psi)$
$\mathrm{DT}_1(m,x_1,x_2)$	$@_{x_1}[m]\neg x_2$	$\psi \vee \mathrm{pre}'(\mathrm{AT}(m,x_1,x_2),\psi)$
$\mathrm{DT}_2(f,x)$	$@_x[f]\mathbf{false}$	$\psi \vee \mathrm{pre}'(\mathrm{AT}(f,x,y),\psi)$
$\mathrm{AS}(x)$	$@_x\big((\bigwedge_{a\in A_\psi\setminus\{x\}}\neg a)\wedge\bigwedge_{m\in\mathrm{MS}}[m]\mathbf{false}\big)\wedge\bigwedge_{m\in\mathrm{MS}}[\mathbf{o}][m]\neg x$	$\mathrm{pre}'(\mathrm{DS}(x),\mathrm{pre}'(\mathrm{CN}_1(x,y),\psi))$
$\mathrm{DS}(x)$	$@_x\,\mathbf{nil}$	$\bigvee_{Y\subseteq A,Z\subseteq B} T_1(T_2(T_3^Y(T_4^Z(\psi))))$

φ_2 and $\mathcal{K}_2 \not\models \varphi_2$. However, $H = \{(s_x,t_x),(s_y,t_y),(s_1,t_1),(s_1,t_2),(\mathrm{nil}_1,\mathrm{nil}_2)\}$ is a bisimulation between $\mathcal{K}_1$ and $\mathcal{K}_2$. By Lemma 5, ψ_2 cannot be in $\mathcal{L}'$. □

4. Postconditions

In this section, we discuss postconditions of transformations. We cannot take the same approach as preconditions since it is not possible to define $\mathrm{post}'(\tau,\varphi)$ so that the counterpart to Lemma 4 holds. However, by modifying the definition slightly, we can obtain formulas useful for forward deduction.

For $\tau \in \mathrm{Tr}$ and closed formulas φ and ψ, we call φ a *postcondition* of ψ with respect to τ if $(\mathcal{K}_1,\mathcal{K}_2) \in \tau$ and $\mathcal{K}_1 \models \psi$ implies the existence of $\mathcal{K}_2'$ such that $\mathcal{K}_2 \sim_{\tau,\psi} \mathcal{K}_2'$ and $\mathcal{K}_2' \models \varphi$, where $\mathcal{K} \sim_{\tau,\psi} \mathcal{K}'$ means that $\mathcal{K}$ and $\mathcal{K}'$ are identical except for values $L(a)$ for atom a that does not appear in τ or ψ.

We begin by defining two auxiliary formulas $\mathrm{post}_1(\tau,\psi)$ and $\mathrm{post}_2(\tau,\psi)$ for $\tau \in \mathrm{Tr}$ and closed formula ψ. Roughly speaking, the former describes properties that obviously hold in the resulting Kripke structures, and the latter is the weakest precondition of ψ with respect to the "reverse transformation" of τ. Their definitions are given in Table 2. In the table, A_ψ is the set of atoms that appear in ψ, y is a fresh nominal, that is, a nominal that does not occur in ψ and that is not identical to x, x_1, or x_2, A is the set of modality symbols occurring in ψ, and B is the set of nominals and propositional symbols occurring in ψ. Functions T_i $(i = 1,2,3,4)$ are defined as follows.

- $T_1(\psi) = \mathrm{pre}'(\mathrm{AS}(x),\psi)$.
- $T_2(\psi) = \mathrm{IDef}\{\langle m\rangle\varphi \rightarrowtail \langle m\rangle T(\varphi) \vee (x \wedge \langle\mathbf{o}\rangle(y_m \wedge T(\varphi))) \vee (z_m \wedge @_x T(\varphi))\ \ (m \in \mathrm{MS})\}(\psi)$,
  where y_m and z_m for $m \in \mathrm{MS}$ are fresh propositional symbols.
- $T_3^Y(\psi) = \mathrm{IDef}\{\langle m\rangle\varphi \rightarrowtail \langle m\rangle T(\varphi) \vee (x \wedge @_x T(\varphi))\ \ (m \in Y)\}(\psi)$.
- $T_4^Z(\psi) = \mathrm{IDef}\{x' \rightarrowtail x\ \ (x' \in Z \cap \mathrm{Nom})\ ;\ p \rightarrowtail p \vee x\ \ (p \in Z \cap \mathrm{PS})\}(\psi)$.

The meaning of these formulas is explained in the proof of Theorem 2.

We define $\mathrm{post}(\tau,\psi) = \mathrm{post}_1(\tau,\psi) \wedge \mathrm{post}_2(\tau,\psi)$.

Theorem 2:

(1) Formula $\mathrm{post}(\tau,\psi)$ is a postcondition of ψ with respect to τ.
(2) If $\mathcal{K}_2 \models \mathrm{post}(\tau,\psi)$, there exists $\mathcal{K}_1$ such that $\mathcal{K}_1 \models \psi$ and $(\mathcal{K}_1,\mathcal{K}_2) \in \tau$.

Proof (Sketch) : We show only the cases where $\tau = \mathrm{CN}_1(x_1,x_2)$ and $\tau = \mathrm{DS}(x)$.

Case $\tau = \mathrm{CN}_1(x_1,x_2)$. For (1), assume $(\mathcal{K}_1,\mathcal{K}_2) \in \tau$ and $\mathcal{K}_1 \models \psi$. Let $\tau' = \mathrm{CN}_1(x_1,y)$, $s_1 = {L_1}'(x_1)$, and $\mathcal{K}_i'$ be the Kripke structure obtained from $\mathcal{K}_i$ by replacing L_i with $L_i[y \mapsto s_1]$ $(i = 1,2)$. Then, $\mathcal{K}_i \sim_{\tau,\psi} \mathcal{K}_i'$ for $i = 1,2$ and $(\mathcal{K}_2',\mathcal{K}_1') \in \tau'$ hold. Also, we have $\mathcal{K}_1' \models \psi$ since y does not appear in ψ. Therefore, $\mathcal{K}_2' \models \mathrm{pre}'(\tau',\psi) = \mathrm{post}_2(\tau,\psi)$. For (2), assume $\mathcal{K}_2 \models \mathrm{post}(\tau,\psi)$ and let $\mathcal{K}_1 = \tau'(\mathcal{K}_2)$. It is easy to check $\mathcal{K}_1 \models \psi$ and $(\mathcal{K}_1,\mathcal{K}_2) \in \tau$.

Case $\tau = \mathrm{DS}(x)$. We define a transformation τ' using fresh nominals and propositions. It can be regarded as the "reverse" transformation of τ if we are allowed to ignore the interpretation of the fresh symbols. It also satisfies $\mathrm{post}_2(\tau,\psi) = \mathrm{pre}'(\tau',\psi)$. Then, the conclusion follows by the same type of argument as in the case of $\mathrm{CN}_1(x_1,x_2)$.

Note that τ not only removes a state s_1 from S, but also changes R and L as well. We need to define τ' so that all effects of τ are reverted. We define τ' as a composition of four transformations: $\tau' = \tau_4^Z \circ \tau_3^Y \circ \tau_2 \circ \tau_1$. ($Y$ and Z are explained below.) First, $\tau_1 = \mathrm{AS}(x)$, which restores the deleted state s_1 and have the state satisfy x. Second, τ_2 and τ_3^Y reverts the changes on R. More precisely, τ_2 restores the removed transitions between s_1 and other states, while τ_3^Y restores self loops on s_1. To define them, we need to express what the removed transitions are. For this purpose, we use fresh propositional symbols y_m and z_m. Intuitively y_m is true on s if and only if transition from s_1 to s existed in $R(m)$ and was deleted by τ. Similarly, z_m is for transition from s to s_1. With these in mind, we define τ_2 as the transformation that adds (s_1,s') and (s'',s_1) to $R(m)$ for all $s' \in L(y_m)$ and $s'' \in L(z_m)$. Similarly, we introduce a set $Y \subseteq A$, intending the set of modalities m for which a self loop on s_1 existed. Then, τ_3^Y adds (s_1,s_1) to $R(m)$ for all $m \in Y$. Finally, τ_4^Z, where $Z \subseteq B$, restores L by changing $L'(x)$ for all $x \in Z \cap \mathrm{Nom}$ to s_1 and add s_1 to $L(p)$ for all $p \in Z \cap \mathrm{PS}$.

With these definitions, it suffices to prove that $T_1(\psi)$, $T_2(\psi)$, $T_3^Y(\psi)$, and $T_4^Z(\psi)$ are equal to $\mathrm{pre}'(\tau_1,\psi)$, $\mathrm{pre}'(\tau_2,\psi)$, $\mathrm{pre}'(\tau_3^Y,\psi)$, and $\mathrm{pre}'(\tau_4^Z,\psi)$, respectively. It is clear for

$T_1(\psi) = 1$, and the rest of them can be shown in a similar manner to the proof of Theorem 1. □

Using Theorem 2, it is shown that for any postcondition χ, if $\mathcal{K} \models \mathrm{post}(\tau, \psi)$, then there exists $\mathcal{K}'$ such that $\mathcal{K} \sim_{\tau,\psi} \mathcal{K}'$ and $\mathcal{K}' \models \chi$. We can regard $\mathrm{post}(\tau, \psi)$ as the strongest postcondition in this sense.

5. Application to the Analysis of Pointer-Manipulating Programs

In this section, we illustrate how the results of the previous sections can be applied, by proving a property of a pointer-manipulating program.

We have a decision procedure [7] for validity of formulas in the alternation-free fragment of logic $\mathcal{L}$. The procedure is sound although not complete for $\mathcal{L}$, meaning that it may fail to judge correctly when the formula in question is valid, but it always returns the correct answer when the formula is not valid. It is sound and complete for the alternation-free fragment of $\mathcal{L}'$. Due to the following proposition, which is easily shown by induction on the construction of φ, we can combine the decision procedure and the results of this study.

Proposition 2: For $\tau \in \mathrm{Tr}$ and closed alternation-free formula φ in $\mathcal{L}$, formulas $\mathrm{pre}(\tau, \varphi)$ and $\mathrm{post}(\tau, \varphi)$ are closed alternation-free formulas in $\mathcal{L}$. If φ is in $\mathcal{L}'$, $\mathrm{post}(\tau, \varphi)$ is in $\mathcal{L}'$.

Figure 1 shows a program written in a programming language called PML [10]. All variables (`x`, `y`, and `t`) are of pointer-type and `f` is a pointer-type field. The variables correspond to nominals and the field corresponds to a functional modality symbol.

Let $\mathcal{K}_1$ be any given Kripke structure, and $\mathcal{K}_2$ be the Kripke structure obtained from $\mathcal{K}_1$ by applying the program. An example of the pair of $\mathcal{K}_1$ and $\mathcal{K}_2$ is shown in Fig. 1.

```
                { @x EF(a) }
y := nil;       { ψ }
while x != nil do     { ψ ∧ ¬@x nil }
  t := y;       { (@t EU(¬x, a) ∨ @x EU(¬t, a))
                          ∧@x ¬nil ∧ @t y }
  y := x;       { φ1 = (@t EU(¬y, a) ∨ @y EU(¬t, a))
                          ∧@y ¬nil ∧ @y x }
  x := y.f;     { φ2 = (@t EU(¬y, a) ∨ @y EU(¬t, a))
                          ∧@y ⟨f⟩x }
                { φ3 = (@t EU(¬y, a) ∨ @y a
                          ∨@x EU(¬y, a)) ∧ @y ⟨f⟩x  }
  y.f := t      { ψ }
od              { ψ ∧ @x nil }    { @y EF(a) }
```

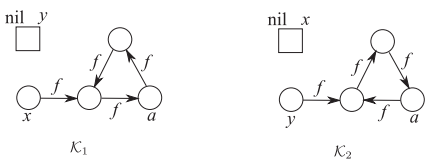

Fig. 1 A pointer-manipulating program.

We verify that every state that is reachable from $L'_1(x)$ in $\mathcal{K}_1$ is reachable from $L'_2(y)$ in $\mathcal{K}_2$. The assertions are written in curly braces. The following abbreviations are used: $\mathrm{EU}(\varphi_1, \varphi_2) = \mu X(\varphi_2 \vee (\varphi_1 \wedge \langle f \rangle X))$, $\mathrm{EF}(\varphi) = \mathrm{EU}(\mathbf{true}, \varphi)$, and $\psi = @x\,\mathrm{EU}(\neg y, a) \vee @y\,\mathrm{EU}(\neg x, a)$.

We introduce a fresh nominal a and put formula $@x\,\mathrm{EF}(a)$, which means "$L(a)$ is reachable from $L(x)$," as the first assertion. The last assertion is $@y\,\mathrm{EF}(a)$. Since a is fresh, this is what we need to deduce.

The weakest preconditions or the strongest postconditions are used to check that each step is correct. For example, the statement `x:=y.f` corresponds to the transformation $\tau = \mathrm{CN}_2(x, y, f)$. To check the triple $\{\varphi_1\}$ `x:=y.f` $\{\varphi_2\}$, we use $\mathrm{post}(\tau, \varphi_1)$. The formula $@y\langle f \rangle x$ in φ_2 comes from $\mathrm{post}_1(\tau, \varphi_1)$, $@t\,\mathrm{EU}(\neg y, a)$ and $@y\,\mathrm{EU}(\neg t, a)$ in φ_2 come from $\mathrm{post}_2(\tau, \varphi_1)$. The formula $@y\,\neg\mathbf{nil}$ can be used to assure that the program does not abort at this point, but for simplicity, we do not further discuss the dangling pointer problem here. The next assertion φ_3 is justified by the fact that $\varphi_2 \to \varphi_3$ is a valid formula, which can be verified using appropriate decision procedures.

While all formulas in the example described above are written in CTL with only forward modalities, general fixed-point operators and backward modalities are also useful for more complex analysis. For example, to verify the correctness of Deutsch-Schorr-Waite marking algorithm, we need formulas with backward modalities that are not in CTL* [11].

We have experimental implementations of a verification tool for pointer-manipulating programs based on the technique described in this section [11], [12].

6. Conclusion and Future Work

We established a method of computing pre- and post-conditions in variants of the modal μ-calculus with regard to transformations of Kripke structures.

An obvious direction for future work is to implement the computation of pre- and post-conditions and combine them with decision procedures of the logic to build a verification system. As already mentioned, this has been partially done; and we plan to extend it to fully cover the contents of this article.

In this study, we choose transformations of Kripke structures based on our intention to apply the results to analyze programs that manipulate single-valued pointers. Analyzing programs with multi-valued pointers should be attempted as future work.

Acknowledgments

This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (C), 21500006, 2009. The authors would like to thank Yoshiki Kinoshita of the National Institute of Advanced Industrial Science and Technology, and Masami Hagiya of the University of Tokyo for their valuable com-

ments and discussions. They are grateful to the anonymous referees for their careful reading and helpful suggestions.

References

[1] K. Takahashi and M. Hagiya, "Abstraction of graph transformation using temporal formulas," Supplemental Volume of the 2003 International Conference on Dependable Systems and Networks, pp.W-65–W-66, 2003.
[2] M. Hagiya, K. Takahashi, M. Yamamoto, and T. Sato, "Analysis of synchronous and asynchronous cellular automata using abstraction by temporal logic," Seventh Functional and Logic Programming Symposium, LNCS 2998, pp.7–21, 2004.
[3] P. Blackburn, "Nominal tense logic," Notre Dame Journal of Formal Logic, vol.34, pp.56–83, 1993.
[4] P.A. Bonatti, C. Lutz, A. Murano, and M.Y. Vardi, "The complexity of enriched μ-calculi," ICALP (2), LNCS 4052, pp.540–551, 2006.
[5] Y. Tanabe, T. Takai, T. Sekizawa, and K. Takahashi, "Preconditions of properties described in CTL for statements manipulating pointers," Supplemental Volume of the 2005 International Conference on Dependable Systems and Networks, pp.228–234, 2005.
[6] P.A. Bonatti and A. Peron, "On the undecidability of logics with converse, nominals, recursion and counting," Artif. Intell., vol.158, pp.75–96, 2004.
[7] Y. Tanabe, K. Takahashi, and M. Hagiya, "A decision procedure for alternation-free modal μ-calculi," Advances in Modal Logic, vol.7, pp.341–362, 2008.
[8] M. Sagiv, T. Reps, and R. Wilhelm, "Parametric shape analysis via 3-valued logic," ACM Trans. Programming Languages and Systems, vol.24, no.3, pp.217–298, 2002.
[9] D. Distefano, P.W. O'Hearn, and H. Yang, "A local shape analysis based on separation logic," TACAS 2006, LNCS 3920, pp.287–302, Springer, 2006.
[10] Y. Kinoshita and K. Nishizawa, "An algebraic semantics of predicate abstraction for PML," Computer Software, JSSST, (to appear).
[11] Y. Yuasa, Y. Tanabe, T. Sekizawa, and K. Takahashi, "Verification of the Deutsch-Schorr-Waite marking algorithm with modal logic," Second International Conference on Verified Software: Theories, Tools, and Experiments, LNCS 5295, pp.115–129, 2008.
[12] T. Sekizawa, Y. Tanabe, Y. Yuasa, and K. Takahashi, "MLAT: A tool for heap analysis based on predicate abstraction by modal logic," IASTED International Conference on Software Engineering, pp.310–317, 2008.

Yoshinori Tanabe is an assistant professor at Graduate School of Information Science and Technology, the University of Tokyo. He received his M.S. degree in mathematics from Tsukuba University in 1987 and Ph.D. in information science and technology from the University of Tokyo in 2008. His research interests include formal methods, verification, and application of modal logic.

Toshifusa Sekizawa received his M.S. degree in physics from Gakushuin University in 1998. He is currently working in the National Institute of Advanced Industrial Science and Technology. He is also a Ph.D. student in the Graduate School of Information Science and Technology, Osaka University. His research interests are in the areas of model checking and its applications.

Yoshifumi Yuasa received his D.Sc. degree in mathematics from Waseda University in 1996. He is currently a Research Associate Professor at the Department of Computer Science, Graduate School of Information Science and Engineering, Tokyo Institute of Technology. His research interests are in software verification, theorem proving, and mathematical logic.

Koichi Takahashi received his M.S. degree in mathematics from Nagoya University in 1988. He received his Ph.D. in information science from the University of Tokyo in 2002. From 1988, he has been working in Electrotechnical Laboratory (currently Advanced Industrial Science and Technology). His research interests include theoretical verifications. He is a member of IPSJ, IEEE, ACM, and JSSST.