A decision procedure for alternation-free modal μ-calculi

YOSHINORI TANABE, KOICHI TAKAHASHI AND MASAMI HAGIYA

ABSTRACT. The alternation-free fragment of the propositional modal μ-calculus (AFμ) allows less complex decision procedure for satisfiability judgment than the full μ-calculus, yet it still has strong expressive power. In this paper, we present a concrete decision procedure with its complexity for AFμ enriched by features of nominals, backward modalities, and functional modalities. While AFμ with all three features is undecidable, AFμ with two out of the three features are decidable and the procedure is sound and complete for these combinations. The procedure is suitable for implementation with BDDs. An application of the decision procedure for program analysis is also reported.

Keywords: modal μ-calculus, satisfiability, decision procedure

1 Introduction

The propositional modal μ-calculus has been studied extensively, and has been applied to verification problems. The authors have proposed to apply its variants for analyzing graph transformation systems [1, 2]. One of the key operations for such analysis is determine whether a given formula is satisfiable.

Known decision procedures for the propositional modal μ-calculus are complex. To reduce the complexity, we have restricted ourselves to its alternation-free fragment. Although this restriction does not reduce the theoretical complexity (both are EXPTIME-complete), it does lead to efficient implementation using binary decision diagrams (BDDs). Meanwhile, the restricted fragments are still sufficiently powerful, and we can apply them to the analysis. In previous research, we established decision procedures for the alternation-free modal μ-calculus (AFμ) [3] and its extension with backward modalities [4]. They are used to analyze programs that handle XML documents. To apply them to other areas such as shape analysis [5], we require logics with more features, nominals and functional modalities, as well as backward modalities.

A nominal is a type of atomic formula, which is satisfied by one and only one node in a Kripke structure. A functional modality is interpreted in the Kripke structures as a (partial) function on the set of states, whereas an ordinary modality is interpreted as a relation on it. A backward modality

m^{-1}, where m is an ordinary (forward) modality, follows the transition relation of a Kripke structure in the reverse direction.

In this paper, we extend the above-mentioned decision procedure to AFμ that has (1) nominals, (2) backward modalities, and (3) functional modalities. Unfortunately, the logic with all three features is undecidable [6]. However, the logics with any two features out of the three are decidable, and we prove that the procedure is sound and complete for these combinations. Moreover, the procedure is sound for the logic with all three features. The complexity of the procedure is $2^{\mathcal{O}(n \log n)}$ for AFμ + (2) + (3) and $2^{\mathcal{O}(n^2 \log n)}$ for the other two combinations. We have an application of the decision procedure for shape analysis. We developed an experimental tool for analyzing programs that manipulate pointers. The procedure was implemented in the tool using JavaBDD [7] with a small modification to fulfill requirements of the tool. Some properties of programs were successfully verified, including the partial correctness of the Deutsch-Schorr-Waite marking algorithm.

The propositional modal μ-calculus was introduced by Kozen [8], which he proved to be decidable. Emerson and Jutla proved that the complexity of its satisfiability problem is EXPTIME-complete [9]. Bonatti and Peron showed that the satisfiability of the modal μ-calculus with nominals, backward modalities, and graded modalities is undecidable [6]. By checking their proof, one can see that AFμ + (1) + (2) + (3) is also undecidable. Bonatti *et al.* [10] proved that the satisfiability problems of the modal μ-calculus extended with any two features out of nominals, backward modalities, and graded modalities are decidable, and their complexity is EXPTIME-complete. Since a functional modality is a type of graded modality, and the satisfiability problem of AFμ is already EXPTIME-hard, it is apparent from their results that "AFμ + any two of (1), (2), and (3)" are also EXPTIME-complete.

The decision procedure given in [10] is based on alternating tree automata. From an application point of view, it is extremely complex and no running implementations are reported. On the other hand, our procedure consists of set operations, which are easily encoded in BDDs for efficient implementation owing to the restriction to the alternation-free fragment. To the best of our knowledge, our decision procedure is the first one that covers all the above-mentioned features, and has been actually applied to solve some concrete problems. Its applicability is also guaranteed by the accurate computational complexity obtained by our analysis.

Various implementations exist for variants of modal logics. Emerson provided a decision procedure for CTL based on the tableau method [11]. Pan *et al.* provided efficient implementations of the decision procedure using BDDs for the minimal modal logic **K** [12, 13]. MONA [14] is a famous tool that implements decision procedures for WS1S and WS2S. This tool also utilizes BDDs for its implementation. Kupferman and Vardi [15] showed an efficient decision procedure for tree automata which leads to implementation for the modal μ-calculus. Eijck developed a theorem prover for hybrid logics [16] based on the tableau method. The decision procedure presented in this paper cannot be replaced with any of the above-mentioned studies. The first four do not contain nominals, and the last does not contain fixed-point operators.

The remainder of the paper is organized as follows. In Section 2, we define syntax and semantics of the logics. In Section 3, we describe the decision procedure. In Section 4, its correctness is proved. The complexity is discussed in Section 5. In Section 6, we present an application that uses a variant of the procedure. Finally, future work is described in Section 7.

2 Preliminaries

2.1 Syntax

Let PS, Nom, PV, GMS, and FMS be countable sets of 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.

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

Form $\ni \varphi ::= p \mid x \mid X \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle m \rangle \varphi \mid \mu X \varphi \mid @x\, \varphi$

where $p \in$ PS, $x \in$ Nom, $X \in$ PV, $g \in$ GMS, and $f \in$ FMS. In $\mu X \varphi$, any free occurrence of X in φ (i.e., an occurrence of X in φ that is not bound by another μX or νX) must be positive (i.e., the number of negation symbols whose scope contains the occurrence is even). A modality in the form of m^{-1} is called a *backward modality.* We denote the set of formulas by $\mathcal{L}$. We define Atom = PS $\cup$ Nom, and MS = GMS $\cup$ FMS. For $m \in$ Mod, the set of the formulas in the form of $\langle m \rangle \varphi$ and $[m]\varphi$ are denoted by $\mathcal{L}_{\langle m \rangle}$ and $\mathcal{L}_{[m]}$, respectively. When $\varphi = \langle m \rangle \varphi'$ or $\varphi = [m]\varphi'$, we denote φ' by $\vec{\varphi}$

We define $(m^{-1})^{-1} = m$ for $m \in$ MS, therefore, $(m^{-1})^{-1} = m$ for any $m \in$ Mod.

The following standard abbreviations are used: **false** $= p \wedge \neg p$ for some fixed $p \in$ PS, **true** $= \neg$**false**, $\varphi_1 \wedge \varphi_2 = \neg(\neg\varphi_1 \vee \neg\varphi_2)$, $\varphi_1 \rightarrow \varphi_2 = \neg\varphi_1 \vee \varphi_2$, $[m]\varphi = \neg\langle m \rangle \neg\varphi$, $\nu X \varphi = \neg\mu\neg\varphi[\neg X/X]$.

2.2 Semantics

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

- S is a set. An element of S is called a *state.*
- R : MS $\rightarrow \mathcal{P}(S \times S)$. $R(f)$ is a (graph of partial) function if $f \in$ FMS.
- L : Atom $\rightarrow \mathcal{P}(S)$. $L(x)$ is a singleton if $x \in$ Nom.

For $x \in$ Nom, we denote the unique element of $L(x)$ by $L'(x)$, i.e., $L(x) = \{L'(x)\}$. If $f \in$ FMS and $s \in \mathrm{dom}(f)$, we express $R(f, s)$ for the unique $s' \in S$ such that $(s, s') \in R(f)$. We also consider that $\mathrm{dom}(R) =$ Mod by defining $R(m^{-1}) = (R(m))^{-1}$ for $m \in$ MS.

A function ρ : PV $\rightarrow \mathcal{P}(S)$ is called a *valuation* for $\mathcal{K}$. The *interpretation* $[\![\varphi]\!]^{\mathcal{K},\rho} \subseteq S$ of $\varphi \in \mathcal{L}$ is defined in the standard manner as follows. Symbols

$\mathcal{K}$ and/or ρ are omitted if there is no possibility of confusion. For a function F, we denote by $F[a \mapsto b]$ a function G defined by $\mathrm{dom}(G) = \mathrm{dom}(F) \cup \{a\}$, $G(a) = b$, and $G(x) = F(x)$ for $x \in \mathrm{dom}(F) \setminus \{a\}$.

$$
\begin{array}{ll}
[\![a]\!] = L(a) \quad \text{for } a \in \text{Atom} & [\![X]\!]^{\rho} = \rho(X) \quad \text{for } X \in \text{PV} \\
[\![\neg\varphi]\!] = S \setminus [\![\varphi]\!] & [\![\varphi_1 \vee \varphi_2]\!] = [\![\varphi_1]\!] \cup [\![\varphi_2]\!] \\
\multicolumn{2}{l}{[\![\langle m\rangle\varphi]\!] = \{s \in S \mid \exists s' \in S.\ (s, s') \in R(m) \text{ and } s' \in [\![\varphi]\!]\}} \\
\multicolumn{2}{l}{[\![\mu X\varphi]\!]^{\rho} = \bigcap\{T \subseteq S \mid [\![\varphi]\!]^{\rho[X \mapsto T]} \subseteq T\}} \\
[\![@x\, \varphi]\!] = S \quad \text{if } L'(x) \in [\![\varphi]\!] & [\![@x\, \varphi]\!] = \varnothing \quad \text{if } L'(x) \notin [\![\varphi]\!]
\end{array}
$$

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 state s. Formulas φ and φ' are *equivalent* ($\varphi \equiv \varphi'$) if $[\![\varphi]\!]^{\mathcal{K},\rho} = [\![\varphi']\!]^{\mathcal{K},\rho}$ for any Kripke structure $\mathcal{K}$ and valuation ρ. A formula φ is *valid* if it is equivalent to **true**. It is *satisfiable* if its negation is not valid.

2.3 Closures

A formula φ is in *positive normal form (PNF)*, if φ satisfies the following conditions:

- All negation operators ($\neg$) in φ appear immediately before propositional symbols, nominals, or propositional variables.
- All propositional variables in φ are bound at most once.

It is easy to see that any formula is equivalent to a formula in PNF.

Symbol λ is used to express either μ or ν. Thus $\lambda X\varphi$ is either $\mu X\varphi$ or $\nu X\varphi$. For the formula $\lambda X\varphi$, we denote by $\exp(\lambda X\varphi)$ the formula $\varphi[\lambda X\varphi/X]$, which is obtained from φ by replacing all free occurrences of X with $\lambda X\varphi$, and call it the *expansion* of $\lambda X\varphi$. For example, $\exp(\mu X(p \vee \langle m\rangle X)) = p \vee \langle m\rangle(\mu X(p \vee \langle m\rangle X))$. It is easy to see that $\exp(\lambda X\varphi) \equiv \lambda X\varphi$.

We define relation F on the set of all formulas in PNF that satisfies the following:

$$
\begin{array}{ll}
\multicolumn{2}{l}{(\varphi_1 \vee \varphi_2, \varphi_j) \in F \text{ and } (\varphi_1 \wedge \varphi_2, \varphi_j) \in F \quad \text{for } j = 1, 2} \\
(\langle m\rangle\varphi, \varphi) \in F & ([m]\varphi, \varphi) \in F \\
(\lambda X\varphi, \exp(\lambda X\varphi)) \in F & (@n\, \varphi, \varphi) \in F
\end{array}
$$

and that no other pairs belong to F.

For a formula φ in PNF, the *closure* of φ is the least set of formulas that contains φ and is closed under the relation F, i.e., if φ is in the closure and $(\varphi, \psi) \in F$ then ψ is in the closure. We denote the closure of φ by $\mathrm{cl}(\varphi)$. The *lean* of φ is a subset of $\mathrm{cl}(\varphi)$ defined by $\{\psi \in \mathrm{cl}(\varphi) \mid \psi \in \text{Atom or } \psi \text{ is in the form of } \langle m\rangle\chi, [m]\chi, \text{ or } @n\, \chi\}$.

An occurrence of a propositional variable X in a formula φ is *guarded* if there exists a formula ψ such that (1) ψ contains the occurrence, (2) ψ is in the form of $\langle m\rangle\psi'$, $[m]\psi'$, or $@n\, \psi'$, and (3) ψ is a subformula of the formula $\lambda X \chi$ that binds the occurrence in φ. A formula is *guarded* if all occurrences of bound propositional variables are guarded. For example, $\mu X(p \vee \langle m \rangle X \vee @n\, X)$ is guarded, but $\mu X(p \vee X)$ is not.

PROPOSITION 1. (Kozen) *For every formula φ, there is a guarded formula ψ such that $\varphi \equiv \psi$.*

Refer to [17] for a proof of the proposition. With a little modification of the proof, we see that ψ can be taken so that $|\,\mathrm{cl}(\psi)| \leq |\,\mathrm{cl}(\varphi)|$ if φ is in PNF.

Hereafter in this paper, we assume that formulas are guarded and are in PNF unless explicitly stated otherwise.

2.4 Alternation-freeness

A formula φ is *alternation-free* if the following conditions are satisfied:

- For any subformula ψ of φ in the form of $\mu X \psi'$ and for any subformula χ of ψ' in the form of $\nu Y \chi'$, X does not occur freely in χ'.
- For any subformula ψ of φ in the form of $\nu Y \psi'$ and for any subformula χ of ψ' in the form of $\mu X \chi'$, Y does not occur freely in χ'.

For example, $\varphi_1 = \mu X(\nu Y(p \wedge \langle m \rangle Y) \vee [m]X)$ is an alternation-free formula, whereas $\varphi_2 = \mu X(\nu Y(p \wedge \langle m \rangle (X \wedge Y)) \vee [m]X)$ is not.

We denote the set of alternation-free formulas by $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{back}, \mathrm{func})$. Also, $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{back})$, $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{func})$, and $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{back}, \mathrm{func})$ are the sets of alternation-free formulas that do not contain functional modalities, backward modalities, and nominals, respectively.

We consider $(\mathrm{cl}(\varphi), F)$ as a graph and denote the set of strongly connected components (SCCs) of the graph by $\mathcal{D}(\varphi)$. Also, we denote by $\mathcal{D}_\mu(\varphi)$ and $\mathcal{D}_\nu(\varphi)$ the set of strongly connected components of the graph that contains a formula in the form of $\mu X \psi$ and $\nu X \psi$, respectively. If φ is clear from the context, they are written as $\mathcal{D}_\mu$ and $\mathcal{D}_\nu$, respectively. Also we write $D_\mu = \bigcup \mathcal{D}_\mu$ and $D_\nu = \bigcup \mathcal{D}_\nu$. If $\psi \in \mathrm{cl}(\varphi)$ is an element of $\bigcup \mathcal{D}(\varphi)$, we denote by $D(\psi)$ the strong connected component $D \in \mathcal{D}(\varphi)$ that contains ψ.

For example, consider the above formula φ_1. By letting $\psi_1 = \nu Y(p \wedge \langle m \rangle Y)$, $D_1 = \{\varphi_1, \psi_1 \vee [m]\varphi_1, [m]\varphi_1\}$, $D_2 = \{\psi_1, p \wedge \langle m \rangle \psi_1, \langle m \rangle \psi_1\}$, and $D_3 = \{p\}$, we have $\mathcal{D}(\varphi_1) = \{D_1, D_2, D_3\}$, $\mathcal{D}_\mu(\varphi_1) = \{D_1\}$, and $\mathcal{D}_\nu(\varphi_1) = \{D_2\}$.

It is easy to show the following lemma.

LEMMA 2. *$\mathcal{D}_\mu(\varphi) \cap \mathcal{D}_\nu(\varphi) = \varnothing$ if and only if φ is alternation-free.*

2.5 Rank and choice functions

In this section, we prepare lemmas to be used to show the correctness of the decision procedure. We denote the class of ordinal numbers by On.

Let φ_{I} be a closed alternation-free formula. Let $\mathcal{K} = (S, R, L)$ be a Kripke structure and $D \in \mathcal{D}_\mu(\varphi_{\mathrm{I}})$. We define $U_{D,\alpha} \subseteq \mathrm{cl}(\varphi_{\mathrm{I}}) \times S$ for $\alpha \in \mathrm{On}$ as the least set that satisfies the following conditions. We omit D and write U_α if no confusion occurs.

- $U_0 = \{(\varphi, s) \mid \varphi \in \mathrm{cl}(\varphi_\mathrm{I}) \setminus D,\ s \in S,\ s \models \varphi\}$.
- If $\varphi \in \mathrm{cl}(\varphi_\mathrm{I})$ and one of the following conditions holds, $(\varphi, s) \in U_\alpha$.
  - $\varphi = \varphi_1 \vee \varphi_2$ and either $(\varphi_1, s) \in U_\alpha$ or $(\varphi_2, s) \in U_\alpha$.
  - $\varphi = \varphi_1 \wedge \varphi_2$ and both $(\varphi_1, s) \in U_\alpha$ and $(\varphi_2, s) \in U_\alpha$.
  - $\varphi = \lambda X \varphi_1$ and $(\exp(\varphi), s) \in U_\alpha$.
  - $\varphi = \langle m \rangle \varphi_1$ and there is $s' \in S$ such that $(s, s') \in R(m)$ and $(\varphi_1, s') \in U_\beta$ for some $\beta < \alpha$.
  - $\varphi = [m]\varphi_1$ and for all $s' \in S$ if $(s, s') \in R(m)$ then there exists $\beta < \alpha$ such that $(\varphi_1, s') \in U_\beta$.
  - $\varphi = @n\, \varphi_1$ and $(\varphi_1, L'(n)) \in U_\beta$ for some $\beta < \alpha$.

LEMMA 3. *For any $D \in \mathcal{D}_\mu(\varphi_\mathrm{I})$, we have $\{(\varphi, s) \in \mathrm{cl}(\varphi_\mathrm{I}) \times S \mid s \models \varphi\} = \bigcup_{\alpha \in \mathrm{On}} U_{D,\alpha}$.*

Proof. We use the fact that $s \models \varphi$ holds if and only if (φ, s) belongs to the winning region of Player 0 in the corresponding parity game [18].

Let $U_\infty = \bigcup_{\alpha \in \mathrm{On}} U_\alpha$ and assume $(\varphi, s) \in \mathrm{cl}(\varphi_\mathrm{I}) \times (S \setminus U_\infty)$. It is easy to see that Player 1 can keep the vertex outside of U_∞. Since $\mathrm{cl}(\varphi_\mathrm{I}) \times S \setminus U_\infty \subseteq D \times S$, the winner of the trace is Player 1. Therefore $s \not\models \varphi$.

The other direction can be shown by induction on α. ■

We define the rank function rank_D: its domain is $\{(\varphi, s) \in \mathrm{cl}(\varphi_\mathrm{I}) \times S \mid \mathcal{K}, s \models \varphi\}$, its range is On, and $\mathrm{rank}_D(\varphi, s) = \min\{\alpha \in \mathrm{On} \mid (\varphi, s) \in U_\alpha\}$. The following lemma clearly holds from the definition.

LEMMA 4. *Suppose $(\varphi, s) \in \mathrm{dom}(\mathrm{rank}_D)$. If $\varphi \notin D$, $\mathrm{rank}_D(\varphi, s) = 0$. Otherwise, the following holds:*

- $\mathrm{rank}_D(\varphi_1 \vee \varphi_2, s) = \min(\{\mathrm{rank}_D(\varphi_j, s) \mid s \models \varphi_j,\ j = 1, 2\})$.
- $\mathrm{rank}_D(\varphi_1 \wedge \varphi_2, s) = \max(\mathrm{rank}_D(\varphi_1, s), \mathrm{rank}_D(\varphi_2, s))$.
- $\mathrm{rank}_D(\langle m \rangle \varphi, s) = \min\{\mathrm{rank}_D(\varphi, s') + 1 \mid (s, s') \in R(m),\ s' \models \varphi\}$.
- $\mathrm{rank}_D([m]\varphi, s) = \sup\{\mathrm{rank}_D(\varphi, s') + 1 \mid (s, s') \in R(m)\}$.
- $\mathrm{rank}_D(\lambda X \varphi, s) = \mathrm{rank}_D(\exp(\lambda X \varphi), s)$.
- $\mathrm{rank}_D(@n\, \varphi, s) = \mathrm{rank}_D(\varphi, L'(n)) + 1$.

A *choice function* c for φ_I and $\mathcal{K}$ is a function that satisfies the following conditions:

- $\mathrm{dom}(c) = \Phi \times S$, where Φ is the set of formulas in $\mathrm{cl}(\varphi_\mathrm{I})$ that is in the form of either $\varphi_1 \vee \varphi_2$ or $\langle m \rangle \varphi$ where $m \in \mathrm{Mod}$.
- For $\varphi_1 \vee \varphi_2 \in \mathrm{cl}(\varphi_\mathrm{I})$ and $s \in S$, $c(\varphi_1 \vee \varphi_2, s)$ is either φ_1 or φ_2. If $s \models \varphi_1 \vee \varphi_2$, $s \models c(\varphi_1 \vee \varphi_2, s)$.

- For $\langle m\rangle\varphi \in \mathrm{cl}(\varphi_\mathrm{I})$ and $s \in S$, $s' = c(\langle m\rangle\varphi, s) \in S$. If $s \models \langle m\rangle\varphi$, $(s, s') \in R(m)$ and $s' \models \varphi$.

A *trace* τ is a finite or an infinite sequence $((\varphi_i, s_i) \mid i < \alpha)$, where $\alpha < \omega + 1$, $\varphi_i \in \mathrm{cl}(\varphi_\mathrm{I})$, and $s_i \in S$, satisfying the following conditions:

(1) $(\varphi_i, \varphi_{i+1}) \in F$ for $i + 1 < \alpha$.

(2) If $\varphi_i = \langle m\rangle\varphi_{i+1}$ or $\varphi_i = [m]\varphi_{i+1}$, $(s_i, s_{i+1}) \in R(m)$.

(3) If $\varphi_i = @n\,\varphi_{i+1}$, $s_{i+1} = L'(n)$.

(4) In cases other than (2) and (3), $s_{i+1} = s_i$.

Trace $\tau = ((\varphi_i, s_i) \mid i < \alpha)$ *conforms with* choice function c if the following conditions are satisfied for $i + 1 < \alpha$:

- If $\varphi_i = \xi \vee \eta$, $\varphi_{i+1} = c(\varphi_i, s_i)$.
- If $\varphi_i = \langle m\rangle\varphi_{i+1}$, $s_{i+1} = c(\varphi_i, s_i)$.

LEMMA 5. *Let c be a choice function for φ_I and $\mathcal{K}$ and assume $Z \subseteq \mathrm{cl}(\varphi_\mathrm{I}) \times S$. If the following conditions are satisfied, $\mathcal{K}, s \models \varphi$ holds for all $(\varphi, s) \in Z$.*

(1) If a is an atom or its negation, $(a, s) \in Z$ implies $\mathcal{K}, s \models a$.

(2) If $((\varphi, s), (\varphi', s'))$ is a trace that conforms with c and $(\varphi, s) \in Z$, then $(\varphi', s') \in Z$.

(3) Assume $((\varphi_i, s_i) \mid i < \omega)$ is an infinite trace that conforms with c, $(\varphi_i, s_i) \in Z$ for all $i < \omega$, $D \in \mathcal{D}_\mu(\varphi_\mathrm{I})$, and $\varphi_0 \in D$. Then, there is $k < \omega$ such that $\varphi_k \notin D$.

Proof. Using the choice function, a strategy σ for Player 0 is defined in an obvious manner. We show that σ is a winning strategy on any $(\varphi, s) \in Z$. By using condition (2), one can show that for any trace τ beginning at (φ, s) and conforming σ and for any $n \in \mathrm{dom}(\tau)$, $\tau(n) = (\varphi_n, s_n) \in Z$. If $\mathrm{dom}(\tau)$ is finite, the winner of τ is Player 0 by condition (1). If $\mathrm{dom}(\tau)$ is infinite, there is $N \in \omega$ and $D \in \mathcal{D}$ such that $\varphi_n \in D$ for all $n \geq N$. By condition (3), $D \in \mathcal{D}_\nu$. This means that all priorities (greater than zero) appearing infinitely often are even and therefore the winner of τ in this case is also Player 0. ■

3 Decision procedure

Now, we describe the decision procedure. In Section 3.1, we define the necessary concepts, and the procedure is defined in Section 3.2.

Let φ_I be the given formula of which we judge satisfiability. We denote the lean of φ_I by Lean. When φ_I has free variables, we replace each of them with a distinct fresh propositional symbol (i.e., one that does not appear in φ_{I}), and denote it by φ'_{I}. It is clear that φ_{I} and φ'_{I} are equi-satisfiable. Therefore, hereafter, we assume that φ_{I} is closed. In what follows, we denote by PS the set of propositional symbols that appear in φ_{I}. Thus, PS is a finite set. The same convention is also applied to Nom, PV, GMS and FMS.

3.1 Tableau

For $D \in \mathcal{D}_\mu$, we denote by BMod_D the set of modality symbol m such that there are formulas $\varphi \in D$ in the form of $\langle m \rangle \varphi'$ or $[m]\varphi'$ and $\psi \in D$ in the form of $\langle m^{-1} \rangle \psi'$ or $[m^{-1}]\psi'$. We denote by BForm_D the subset of D that consists of the formulas in the form of $\langle m \rangle \varphi$, $[m]\varphi$, $\langle m^{-1} \rangle \varphi$, or $[m^{-1}]\varphi$ for some $m \in \mathrm{BMod}_D$.

The decision procedure is a variant of the tableau method. A node of the tableau is a pair (x, y) that satisfies the following conditions:

- x is a function and its domain is Lean. For $\varphi \in$ Lean, $x(\varphi)$ is either a natural number or value "∞". If $\varphi \in D_\mu$ and $\mathrm{BForm}_{D(\varphi)} \neq \varnothing$, then either $x(\varphi) \leq |\mathrm{BForm}_{D(\varphi)}| + 1$ or $x(\varphi) = \infty$; otherwise $x(\varphi)$ is either 0 or ∞. We regard $0 < 1 < \cdots < \infty$.

- y is a function and its domain is $\{\varphi \in D_\mu \cap \mathrm{Lean} \mid x(\varphi) < \infty\}$. For $\varphi \in \mathrm{dom}(y)$, $y(\varphi)$ is a natural number and $y(\varphi) \leq |\mathrm{Nom}| \cdot |D(\varphi)|$.

- If there is $D \in \mathcal{D}_\mu$ such that $\varphi_1, \varphi_2 \in D$ and $x(\varphi_1) < x(\varphi_2) < \infty$, then $y(\varphi_1) \leq y(\varphi_2)$.

Note that there are only finitely many pairs (x, y) that satisfy the conditions. We denote by Tab the set of all pairs that satisfy the above conditions. When $t = (x, y) \in$ Tab, x is denoted by x_t and y is denoted by y_t.

The intention is that at node $t = (x, y)$, $\varphi \in$ Lean is satisfied if $x(\varphi) < \infty$ and is not satisfied if $x(\varphi) = \infty$. In the case $\varphi \in D \in \mathcal{D}_\mu$, due to (3) of Lemma 5, if we can appropriately define a choice function c, there should be a finite trace that conforms with c, starts with (φ, t), and goes outside D. The values $x(\varphi)$ and $y(\varphi)$ both relate to the trace. If $x(\psi) \leq x(\varphi)$, we regard that (ψ, t) can appear in the trace that starts with (φ, t). The value $y(\varphi)$ expresses the number of nominals that appear in the trace. For precise meanings, refer to Section 4.

A function g from Nom to Tab is called a *naming function* when it satisfies the following conditions:

- $x_{g(n)}(n) = 0$ for all $n \in$ Nom.

- $x_{g(n_1)}(n_2) = 0 \implies g(n_1) = g(n_2)$ for any $n_1, n_2 \in$ Nom.

We denote the set of naming functions by NF. Note that NF is a finite set. A naming function designates a node at which each nominal is satisfied. By recalling that $x_t(\varphi) = 0$ means that t satisfies φ, the first condition means that a nominal should be satisfied at the node where the nominal is satisfied. The second condition means that if n_2 is satisfied at the node where n_1 is satisfied, then the node where n_1 is satisfied should be equal to the node where n_2 is satisfied.

Values $x(\varphi)$ are defined only on Lean but we extend its domain to $\mathrm{cl}(\varphi_\mathrm{I})$ by induction in the following manner. Assume $t = (x, y)$. For $a \in \mathrm{Atom}$, $x(\neg a) = 0$ if $x(a) = \infty$ and $x(\neg a) = \infty$ if $x(a) = 0$. For disjunctions, conjunctions, and fixed-points, we define the x-value basically as the minimum values of its subformulas, the maximum values of its subformulas, and the value of its expansion, respectively; however, if the formula goes outside the SCC, the value goes to 0 or ∞. More precisely, $x(\varphi_1 \vee \varphi_2) = \min(\tilde{x}(\varphi_1, D(\varphi_1 \vee \varphi_2)), \tilde{x}(\varphi_2, D(\varphi_1 \vee \varphi_2)))$, $x(\varphi_1 \wedge \varphi_2) = \max(\tilde{x}(\varphi_1, D(\varphi_1 \wedge \varphi_2)), \tilde{x}(\varphi_2, D(\varphi_1 \wedge \varphi_2)))$, and $x(\lambda X \varphi) = x(\exp(\lambda X \varphi))$, where $\tilde{x}(\varphi, D) = 0$ if $x(\varphi) < \infty$ and $\varphi \notin D$ and $\tilde{x}(\varphi, D) = x(\varphi)$ otherwise. Note that the induction is sound since the formulas are guarded.

The domain of y is also extended to $\{\varphi \in D_\mu \mid x(\varphi) < \infty\}$ in a similar manner: $y(\varphi_1 \vee \varphi_2) = \min(\tilde{y}(\varphi_1, D(\varphi_1 \vee \varphi_2)), \tilde{y}(\varphi_2, D(\varphi_1 \vee \varphi_2)))$, $y(\varphi_1 \wedge \varphi_2) = \max(\tilde{y}(\varphi_1, D(\varphi_1 \wedge \varphi_2)), \tilde{y}(\varphi_2, D(\varphi_1 \wedge \varphi_2)))$, and $y(\lambda X \varphi) = y(\exp(\lambda X \varphi))$. If $x(\varphi) = \infty$, then $\tilde{y}(\varphi, D) = \infty$, else if $\varphi \notin D$, then $\tilde{y}(\varphi, D) = 0$, else $\tilde{y}(\varphi, D) = y(\varphi)$.

We write $t \Vdash \varphi$ when $x_t(\varphi) < \infty$. The set $\{\varphi \in \mathrm{Lean} \mid t \Vdash \varphi\}$ is denoted by $\mathrm{sat}(t)$. A node $t \in \mathrm{Tab}$ *has a name* if there is $n \in \mathrm{Nom}$ such that $t \Vdash n$.

For each $m \in \mathrm{MS}$, we define the *transition relation* $\mathrm{Tr}(m)$ on Tab as the conjunction of the following five conditions: $\mathrm{ConBox}(t, t', m)$, $\mathrm{ConBox}(t', t, m^{-1})$, $\mathrm{ConNom}(t, t', m)$, $\mathrm{ConNom}(t', t, m^{-1})$, and $\mathrm{LoopFree}(t, t', m)$, each of which is defined below. For $m \in \mathrm{MS}$, $\mathrm{Tr}(m^{-1})$ is defined as $\mathrm{Tr}(m^{-1}) = \mathrm{Tr}(m)^{-1}$.

For $t \in \mathrm{Tab}$ and $m \in \mathrm{Mod}$, we define $\mathrm{Box}(t, m) = \mathrm{sat}(t) \cap \mathcal{L}_{[m]}$ if $m \in \mathrm{Mod} \setminus \mathrm{FMS}$ and $\mathrm{Box}(t, m) = \mathrm{sat}(t) \cap (\mathcal{L}_{[m]} \cup \mathcal{L}_{\langle m \rangle})$ if $m \in \mathrm{FMS}$. Then, $\mathrm{ConBox}(t, t', m)$ is defined as "for all $\varphi \in \mathrm{Box}(t, m)$, $t' \Vdash \vec{\varphi}$". The intention of this definition should be clear if we consider $\mathrm{Tr}(m)$ as a corresponding relation of $R(m)$ of a Kripke structure. Note that if $m \in \mathrm{FMS}$, diamond formulas behave similarly to box formulas since there is at most one successor.

For $t, t' \in \mathrm{Tab}$ and $\varphi \in \mathrm{dom}(y_t) \cap \mathrm{Lean}$, $\mathrm{ConNom}(t, t', \varphi)$ is defined as follows: "$\vec{\varphi} \in \mathrm{dom}(y_{t'})$ and $y_t(\varphi) \geq y_{t'}(\vec{\varphi})$ hold. Moreover if t' has a name, $y_t(\varphi) > y_{t'}(\vec{\varphi})$ holds." This is also a natural requirement by considering the intuitive meaning of the function y. For $t, t' \in \mathrm{Tab}$ and $m \in \mathrm{Mod}$, we write $\mathrm{ConNom}(t, t', m)$ if $\mathrm{ConNom}(t, t', \varphi)$ holds for any $\varphi \in \mathrm{dom}(y_t) \cap \mathrm{Box}(t, m)$.

For $t, t' \in \mathrm{Tab}$, $m \in \mathrm{Mod}$, and $\varphi \in \mathrm{Box}(t, m)$, $\mathrm{LoopFree}(t, t', \varphi)$ is defined as follows: "there exists *no* formula $\psi \in \mathcal{L}_{[m]} \cap D(\varphi)$ such that both $x_t(\varphi) \leq x_t(\vec{\psi}) < \infty$ and $x_{t'}(\psi) \leq x_{t'}(\vec{\varphi}) < \infty$ hold." We write $\mathrm{LoopFree}(t, t', m)$ if $\mathrm{LoopFree}(t, t', \varphi)$ holds for any $\varphi \in \mathrm{Box}(t, m) \cap D_\mu$. This completes the definition of $\mathrm{Tr}(m)$.

The intuitive reason why we require the loop-freeness for $\mathrm{Tr}(m)$ is as follows: roughly speaking, we will create a Kripke structure from a tableau in the manner that $t \in \mathrm{Tab}$ is a state, the transition relation of modality m is defined by $\mathrm{Tr}(m)$, and φ is satisfied at state t if and only if $t \Vdash \varphi$. Suppose

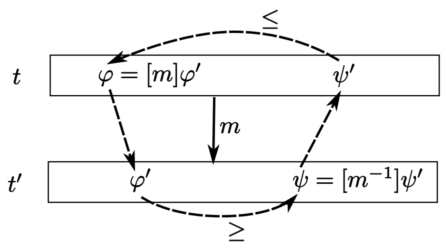

Figure 1. loop-freeness

that there is a transition of modality m from t to t' and it is not loop-free with regard to $\varphi = [m]\varphi'$. Take $\psi = [m^{-1}]\psi'$ as in the definition. Then, (φ, t) may appear in a trace from (ψ', t), and (ψ, t') may appear in a trace from (φ', t'). Furthermore, both $((\varphi, t), (\varphi', t'))$ and $((\psi, t'), (\psi', t))$ can be adjacent pairs in traces. Combining these four parts (see Figure 1), one can construct an infinite trace within D, which contradicts to Lemma 5.

For $m \in \mathrm{Mod}$ and $\varphi \in \mathrm{Lean} \cap \mathcal{L}_{\langle m \rangle}$, we also define the relation $\mathrm{Tr}(\varphi)$ on Tab as the set of tuples (t, t') that satisfies the following three conditions. (1) $(t, t') \in \mathrm{Tr}(m)$. (2) $t \Vdash \varphi$. (3) If $\varphi \in D_\mu$, $\mathrm{ConNom}(t, t', \varphi)$ and $\mathrm{LoopFree}(t, t', \varphi)$ hold.

3.2 Procedure

For given φ_{I}, the decision procedure will try to find a suitable subset T of Tab such that a model for φ_{I} is constructed from T. One of the difficulty is to decide where nominals should be satisfied in T. For example if we know that t should be in T and $t \Vdash \langle m \rangle (n \wedge (\varphi \vee \psi))$. If n is a mere propositional symbol, we can request t_1 or t_2 be a member of T, where t_1 and t_2 are appropriate element of Tab such that $t_1 \Vdash n \wedge \varphi$ and $t_2 \Vdash n \wedge \psi$ and $(t, t_i) \in \mathrm{Tr}(m)$. However, if n is a nominal, and especially if φ and ψ are mutually inconsistent, we need to ask exactly one of t_1 and t_2 exists in T, but which of them should we choose?

Naming functions are used to solve the problem. We fix a naming function g, and describe a subprocedure for g. In the subprocedure, we assume that nominal n is satisfied at $g(n) \in \mathrm{Tab}$. The entire procedure consists of a loop over NF. If there is a $g \in \mathrm{NF}$ for which the subprocedure succeeds, then we consider that φ_{I} is satisfiable. If the procedure fails for all $g \in \mathrm{NF}$, we judge that φ_{I} is not satisfiable. In the rest of this section, we define the subprocedure for g.

We construct a sequence $(T_k)_{k \le K}$ of subsets of Tab so that $T_0 \supseteq T_1 \supseteq \cdots$. The construction is repeated until $T_k = T_{k+1}$ holds. Since Tab is a finite set, there exists such $k \in \omega$. We write this k as K. The procedure succeeds if there is $t \in T_K$ such that $t \Vdash \varphi_{\mathrm{I}}$, and for all $n \in \mathrm{Nom}$, $g(n) \in T_K$.

The initial set T_0 consists of the elements t of Tab that satisfy the following conditions:

- For all $n \in \text{Nom}$, $t \Vdash n \implies t = g(n)$.
- For all $\varphi \in \text{Lean}$ in the form of $@n\,\varphi'$, if $t \Vdash \varphi$, then $g(n) \Vdash \varphi'$. Moreover, if $\varphi \in D_\mu$, $y_t(\varphi) > y_{g(n)}(\varphi')$.

T_{k+1} is obtained from T_k by removing nodes that are $\Diamond$-inconsistent or μ-inconsistent in T_k. Let T be a subset of Tab. Node $t \in T$ is *$\Diamond$-consistent* in T, if:

- for any $m \in \text{Mod} \setminus \text{FMS}$ and $\varphi \in \text{Lean} \cap \mathcal{L}_{\langle m \rangle}$, $t \Vdash \varphi$ implies that there exists $t' \in T$ such that $(t, t') \in \text{Tr}(\varphi)$, and
- for any $m \in \text{FMS}$, if there exists $\varphi \in \text{Lean} \cap \mathcal{L}_{\langle m \rangle}$ such that $t \Vdash \varphi$, then there exists $t' \in T$ such that $(t, t') \in \text{Tr}(m)$.

We say $t \in T$ is *$\Diamond$-inconsistent* if it is not $\Diamond$-consistent.

In order to define μ-consistency, we construct a sequence $(V_j)_{j \le J}$ of subsets of $T \times \mathcal{P}(D_\mu \cap \text{Lean})$ so that $V_0 \subseteq V_1 \subseteq \cdots$. We repeat it until $V_j = V_{j+1}$ holds, and set $J = j$. (J depends on T.) Since $T \times \mathcal{P}(D_\mu \cap \text{Lean})$ is a finite set, there must be such $j \in \omega$. A node $t \in T$ is *μ-consistent* in T if for all $D \in \mathcal{D}_\mu$, $(t, \{\varphi \in D \cap \text{Lean} \mid t \Vdash \varphi\}) \in V_J$, and t is *μ-inconsistent* if it is not μ-consistent.

Let $t = (x, y) \in \text{Tab}$ and $E \subseteq D_\mu$. For $\varphi \in \text{Lean}$, we define $x^E(\varphi)$ by $x^E(\varphi) = \infty$ if $\varphi \in D_\mu \setminus E$, and $x^E(\varphi) = x(\varphi)$ otherwise. Then, we extend the domain of x^E to $\text{cl}(\varphi_\text{I})$ in the same manner as we extended the domain of x. We write $t^E \Vdash \varphi$ when $x^E(\varphi) < \infty$.

The initial set V_0 is defined as $V_0 = \{(t, \varnothing) \mid t \in T\}$. Assume we have V_j. Then V_{j+1} is defined so that $(t, E) \in V_{j+1}$ if and only if $(t, E) \in V_j$ or the following conditions are satisfied:

(a) The following holds for all $\varphi \in E$.
   - $t \Vdash \varphi$.
   - For all $\psi \in D(\varphi) \cap \text{Lean}$, if $x_t(\varphi) > x_t(\psi)$ then $\psi \in E$.
   - For all $\psi \in D(\varphi) \cap \text{Lean} \cap \text{dom}(y_t)$, if $y_t(\varphi) > y_t(\psi)$ then $\psi \in E$.

(b) For any $m \in \text{Mod} \setminus \text{FMS}$ and $\varphi \in \mathcal{L}_{\langle m \rangle} \cap \text{sat}(t)$, there is $(t', E') \in V_j$ such that:
   - $(t, t') \in \text{Tr}(\varphi)$.
   - For any $\psi \in E \cap \mathcal{L}_{[m]}$, $t'^{E'} \Vdash \vec{\psi}$.
   - If $\varphi \in E$, $t'^{E'} \Vdash \vec{\varphi}$.

(c) For any $m \in \text{FMS}$ that satisfies $\mathcal{L}_{\langle m \rangle} \cap \text{sat}(t) \neq \varnothing$, there is $(t', E') \in V_j$ such that:
   - $(t, t') \in \text{Tr}(m)$.
   - For any $\psi \in E \cap (\mathcal{L}_{\langle m \rangle} \cup \mathcal{L}_{[m]})$, $t'^{E'} \Vdash \vec{\psi}$.

This completes the description of the decision procedure.

4 Correctness

In this section we prove that the decision procedure presented in the previous section is sound and complete for $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{back})$, $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{func})$, and $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{back}, \mathrm{func})$; and sound for $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{back}, \mathrm{func})$. We first show the soundness (if there is a Kripke structure that satisfies φ_{I}, the procedure succeeds) in Section 4.1; then, we show the completeness, the other direction, in Section 4.2.

4.1 Soundness

Assume that there is a Kripke structure $\mathcal{K} = (S, R, L)$ and $s_{\mathrm{I}} \in S$ such that $\mathcal{K}, s_{\mathrm{I}} \models \varphi_{\mathrm{I}}$. We define a function $h : S \to \mathrm{Tab}$ and show that $h(s_{\mathrm{I}}) \Vdash \varphi_{\mathrm{I}}$ and all nodes in the range of h remain in T_k. For $\varphi \in D_\mu$ and $s \in S$, we denote the set $\{\mathrm{rank}_D(\psi, s) \mid \psi \in \mathrm{BForm}_D,\ \mathrm{rank}_D(\psi, s) \leq \mathrm{rank}_D(\varphi, s)\}$ by $X(\varphi, s)$, where $D = D(\varphi)$. It is finite since BForm_D is finite. If we write $h(s) = (x, y)$, x and y are defined as follows. If $s \not\models \varphi$, $x(\varphi) = \infty$. Otherwise, if $\varphi \in D_\mu$ then $x(\varphi) = |X(\varphi, s)| + 1$, else $x(\varphi) = 0$. For $\varphi \in \mathrm{dom}(y)$, $y(\varphi) = |Y(\varphi, s)|$, where $Y(\varphi, s) = \{(n, \psi) \in \mathrm{Nom} \times D \mid \mathrm{rank}_D(\psi, L'(n)) < \mathrm{rank}_D(\varphi, s)\}$ and $D = D(\varphi)$.

The following three lemmas can be easily proved and we omit the proofs.

LEMMA 6. *For $\varphi \in \mathrm{cl}(\varphi_{\mathrm{I}})$ and $s \in S$,* $\mathcal{K}, s \models \varphi \iff h(s) \Vdash \varphi$.

LEMMA 7. $(s, s') \in R(m) \implies (h(s), h(s')) \in \mathrm{Tr}(m)$ *holds for $s, s' \in S$ and $m \in \mathrm{Mod}$.*

LEMMA 8. *Suppose $s \in S$, $(x, y) = h(s)$, $\varphi, \psi \in D_\mu$ and $D(\varphi) = D(\psi)$. Let us write $D = D(\varphi) = D(\psi)$.*

(1) $x(\varphi) \leq x(\psi) \iff \mathrm{rank}_D(\varphi, s) \leq \mathrm{rank}_D(\psi, s)$.

(2) If $\varphi, \psi \in \mathrm{dom}(y)$, $y(\varphi) \leq y(\psi) \iff \mathrm{rank}_D(\varphi, s) \leq \mathrm{rank}_D(\psi, s)$.

For $D \in \mathcal{D}_\mu$, $\alpha \in \mathrm{On}$, and $s \in S$, let us denote by $F_D(\alpha, s)$ the set $\{\varphi \in D \cap \mathrm{Lean} \mid s \models \varphi,\ \mathrm{rank}_D(\varphi, s) < \alpha\}$.

LEMMA 9. *For any $\alpha \in \mathrm{On}$ and $D \in \mathcal{D}_\mu$ there exists $j \in \omega$ such that for any $s \in S$, $(h(s), F_D(\alpha, s)) \in V_j$.*

Proof. We prove the lemma by induction on α. The case $\alpha = 0$ is trivial since $F_D(0, s) = \varnothing$. The case in which α is limit is also clear since there exists $J \in \omega$ such that $V_j = V_J$ for all $j \geq J$.

For the remaining case of $\alpha + 1$, we assume that $(h(s), F_D(\alpha, s)) \in V_j$ for all $s \in S$ and prove that $(h(s), F_D(\alpha+1, s)) \in V_{j+1}$ for all $s \in S$. Take $s \in S$ and let $t = h(s)$ and $E = F_D(\alpha+1, s)$. We need to check the conditions (a), (b) and (c) of the definition of V_{j+1}. The condition (a) can be proved without difficulty by using Lemmas 6 and 8. For the condition (b), assume $m \in \mathrm{Mod} \setminus \mathrm{FMS}$, $\varphi = \langle m \rangle \varphi' \in \mathrm{Lean}$, and $t \Vdash \varphi$. Since $s \models \varphi$ by Lemma 6, we can take $s' \in S$ such that $s' \models \varphi$ and $(s, s') \in R(m)$. Moreover, if $\varphi \in D_\mu$, we take s' so that $\mathrm{rank}_D(\varphi', s') < \mathrm{rank}_D(\varphi, s)$ is satisfied. Let $t' = h(s') = (x', y')$ and $E' = F_D(\alpha, s')$. By the induction hypothesis, we have $(t', E') \in V_j$. The three items in the condition (b) can be checked for (t', E'). We skipped the details. For the condition (c): Assume $m \in \mathrm{FMS}$, $\varphi = \langle m \rangle \varphi' \in \mathrm{Lean}$, and $t \Vdash \varphi$. By Lemma 6, we have $s' \in S$ such that $(s, s') \in R(m)$ and $s' \models \varphi'$. Let $t' = h(s')$ and $E' = F_D(\alpha, s')$. The two items in the condition (c) can be checked for (t', E'). ■

LEMMA 10. *If $h[S] \subseteq T \subseteq \mathrm{Tab}$, A node $t \in h[S]$ is both $\Diamond$-consistent and μ-consistent in T.*

Proof. The $\Diamond$-consistency clearly holds by combining Lemmas 6 and 7. For the μ-consistency, note that $F_D(\alpha, s) = \{\varphi \in D \cap \mathrm{Lean} \mid s \models \varphi\}$ if α is sufficiently large, for example if α is a larger cardinal than the cardinality of S. The conclusion follows from Lemmas 6 and 9. ■

Now we prove the soundness.

THEOREM 11. *If there is a Kripke structure $\mathcal{K} = (S, R, L)$ and $s_\mathrm{I} \in S$ such that $\mathcal{K}, s_\mathrm{I} \models \varphi_\mathrm{I}$, the procedure succeeds.*

Proof. Let us define $g(n) = h(L'(n))$ for $n \in \mathrm{Nom}$. It is easy to see that g is a naming function. We claim for this g that $h[S] \subseteq T_k$ holds for all $k \in \omega$. This claim, combined with Lemma 6, is sufficient for the proof.

We prove the claim by induction on k. For $k = 0$, $h[S] \subseteq T_0$ can be checked using Lemma 6 and the definition of the y-part of h.

Assume $h[S] \subseteq T_k$. Then by Lemma 10, $h[S] \subseteq T_{k+1}$. This completes the proof. ■

4.2 Completeness

To prove the completeness, we assume that the procedure succeeds. Let g be the naming function for which the procedure succeeds.

We fix an SCC $D_0 \in \mathcal{D}_\mu$ and a cyclic permutation τ on $\mathcal{D}_\mu$. In case $\mathcal{D}_\mu = \varnothing$, we consider $\mathcal{D}_\mu = \{\varnothing\}$; therefore $D_0 = \varnothing$ and τ is the identity.

Let Nom' be a set of representatives of the equivalence class induced by the equivalence relation $\{(n, n') \in \mathrm{Nom} \mid g(n') \Vdash n\}$. For $t \in \mathrm{Tab}$ and $D \in \mathcal{D}_\mu$, the set $\mathrm{sat}(t) \cap D$ is denoted by $\mathrm{sat}_D(t)$.

We construct a (possibly infinite) forest (W, R^W), that is, a disjoint union of trees, where W is the underlying set and R^W is the forest relation on W, together with functions $t : W \to T_K$, $l : R^W \to \mathrm{Mod}$, $D : W \to \mathcal{D}_\mu$, $E : W \to \mathcal{P}(D_\mu \cap \mathrm{Lean})$, and $j : W \to \omega$. During the construction, we will keep the following invariant: for $w, w' \in W$, $j(w) > 0 \implies (t(w), E(w)) \in V_{j(w)} \setminus V_{j(w)-1}$.

At the first stage of the construction, we create elements w_I and w_n for $n \in \mathrm{Nom}'$ and let $W = \{w_\mathrm{I}\} \cup \{w_n \mid n \in \mathrm{Nom}'\}$ and $R^W = \varnothing$. Also, we define $t(w_\mathrm{I}) = t_\mathrm{I}$ and $t(w_n) = g(n)$ for $n \in \mathrm{Nom}'$. And for $w \in W$, we define $D(w) = D_0$, $E(w) = \mathrm{sat}_{D_0}(t(w))$, and $j(w) = \min\{j \in \omega \mid (t(w), E(w)) \in V_j\}$. The invariant holds for the first stage: since $t(w) \in T_K$, $t(w)$ is μ-consistent, therefore, the set in the right hand side of the definition of $j(w)$ is not empty.

At the second and succeeding stages, we pick a leaf node w of W which has not yet been processed and is located at the shallowest level of the forest among such nodes. If there is $n \in \mathrm{Nom}'$ such that $t(w) = t(w_n)$ but $w \neq w_n$, nothing needs to be done. In this case, w is a leaf of the forest.

Otherwise, for each $m \in \mathrm{Mod} \setminus \mathrm{FMS}$ and $\varphi \in \mathrm{sat}(t(w)) \cap \mathcal{L}_{\langle m \rangle}$, we create a node w' and add it to W. Also for each $m \in \mathrm{FMS}$ that satisfies (1) $\mathrm{sat}(t(w)) \cap \mathcal{L}_{\langle m \rangle} \neq \varnothing$, and (2) there is no $\hat{w} \in W$ such that $(\hat{w}, w) \in R^W$ and $l(\hat{w}, w) = m^{-1}$, we create a node w' and add it to W. In both cases, a pair (w, w') is added to R^W and we define $l(w, w') = m$.

We define $t(w')$ depending on the value of $j(w)$. If $j(w) = 0$, since $t(w) \in T_K$, it is $\Diamond$-consistent in T_K. When $m \in \mathrm{Mod} \setminus \mathrm{FMS}$, take $t' \in T_K$ such that $(t(w), t') \in \mathrm{Tr}(\varphi)$, where φ is the formula that corresponds to w'. When $m \in \mathrm{FMS}$, take $t' \in T_K$ such that $(t(w), t') \in \mathrm{Tr}(m)$. In both cases we define $t(w') = t'$. Also, we define $D(w') = \tau(D(w))$, $E(w') = \mathrm{sat}_{D(w')}(t(w'))$, and $j(w') = \min\{j \in \omega \mid (t(w'), E(w')) \in V_j\}$. The invariant can be checked as before.

If $j(w) > 0$, from the invariant there is $(t', E') \in V_{j(w)} \setminus V_{j(w)-1}$ that satisfies the conditions in the description of the procedure. We define $t(w') = t'$, $D(w') = D(w)$, $E(w') = E'$ and $j(w') = \min\{j \in \omega \mid (t', E') \in V_j\}$. The invariant trivially holds. This completes the construction of the forest.

We define an equivalence relation $\sim$ on W: $w_1 \sim w_2$ if and only if $w_1 = w_2$ or there exists $n \in \mathrm{Nom}'$ such that $t(w_1) = t(w_2) = g(n)$. Note that in each equivalence class there is at most one element that has successors. Using this, we define a tuple $\mathcal{K} = (S, R, L)$ as follows:

- The underlying set S is W divided by the equivalence relation $\sim$. For brevity, we also write w for the equivalence class that contains w.
- $R(m) = \{(w_1, w_2) \in R^W \mid l(w_1, w_2) = m \text{ or } l(w_2, w_1) = m^{-1}\}$.
- $L(p) = \{w \in S \mid t(w) \Vdash p\}$.

Tuple $\mathcal{K}$ is almost a Kripke structure, but $R(m)$ might not be a function for $m \in \mathrm{FMS}$. For example if two new nodes are added with $\langle m^{-1} \rangle n$ for $n \in \mathrm{Nom}$ at two different nodes, they are equivalent and the equivalence class has two successors in $\mathcal{K}$. However, one can easily check that the following lemma holds.

LEMMA 12. *If φ_{I} is in either $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{back})$, $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{back}, \mathrm{func})$, or $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{func})$, tuple $\mathcal{K}$ is a Kripke structure.*

To prove the completeness using Lemma 5, we take a choice function c as follows. Let $w \in S$, $(x, y) = t = t(w)$, $D = D(w)$, and $E = E(w)$. When $\varphi = \varphi_0 \vee \varphi_1 \in \mathrm{cl}(\varphi_{\mathrm{I}})$, $c(\varphi, w)$ is determined in the following order. (1) If $x(\varphi) = \infty$, either φ_0 or φ_1 can be chosen. (2) If only one of $x(\varphi_i) < \infty$ $(i = 0, 1)$ holds, choose φ_i. (3) If $\varphi \notin D$, either φ_0 or φ_1 can be chosen. (4) If only one of $\varphi_i \notin D$ $(i = 0, 1)$ holds, take φ_i. (5) If $y(\varphi_0)$ and $y(\varphi_1)$ differs, choose the smaller of the two. (6) If $x(\varphi_0)$ and $x(\varphi_1)$ differs, choose the smaller of the two. (7) If $x^E(\varphi_0)$ and $x^E(\varphi_1)$ differs, choose the smaller of the two. (8) Either φ_0 or φ_1 can be taken. When $\varphi = \langle m\rangle\varphi'$, (1) if $t \not\Vdash \varphi$, any $w' \in S$ can be chosen. (2) If $m \in \mathrm{FMS}$ and there is $w' \in S$ such that $l(w', w) = m^{-1}$, choose it. (3) In this case, a successor w' of w for φ was created during the construction. Choose it.

Let $Z = \{(\varphi, w) \in \mathrm{cl}(\varphi_{\mathrm{I}}) \times S \mid t(w) \Vdash \varphi\}$. We will show that the choice function c and Z satisfies the conditions of Lemma 5. We start with a preliminary lemma.

LEMMA 13. *Assume* $t = (x, y) \in \mathrm{Tab}$, $D \in \mathcal{D}_\mu$, $E \subseteq D \cap \mathrm{Lean}$, *and* E *is closed under downward order of* x, *that is,* $\psi \in D \cap \mathrm{Lean}$, $\psi' \in E$, *and* $x(\psi) < x(\psi')$ *implies* $\psi \in E$. *Then, for* φ_1, $\varphi_2 \in D$, *if* $x(\varphi_1) < x(\varphi_2)$ *then* $x^E(\varphi_1) < x^E(\varphi_2)$ *or* $x^E(\varphi_1) = x^E(\varphi_2) = \infty$.

Proof. This lemma can be shown by double induction, first on φ_1, then on φ_2. We omit details, which are tedious but not difficult. ■

Next, let us summarize the properties of the forest as a lemma.

LEMMA 14. *The following holds for all* $w \in W$*:*

(1) $j(w) > 0 \implies (t(w), E(w)) \in V_{j(w)} \setminus V_{j(w)-1}$.

(2) $E(w) \subseteq D(w)$

(3) Assume $m \in \mathrm{Mod}$, $\varphi \in \mathrm{sat}(t(w)) \cap \mathcal{L}_{\langle m\rangle}$, *and* $w' = c(\varphi)$.

   (a) Either $(w, w') \in R^W$ *or* $(w', w) \in R^W$ *holds. If* $m \notin \mathrm{FMS}$, *then* $(w, w') \in R^W$ *holds.*

   (b) $(t(w), t(w')) \in \mathrm{Tr}(\varphi)$ *holds.*

   (c) If $m \notin \mathrm{FMS}$ *and* $\varphi \in E(w)$, $t(w')^{E(w')} \Vdash \vec{\varphi}$ *holds.*

(4) For any $w' \in W$, $(w, w') \in R^W$ *implies* $(t(w), t(w')) \in \mathrm{Tr}(l(w, w'))$ *and* $(t(w'), t(w)) \in \mathrm{Tr}(l(w, w')^{-1})$.

(5) For any $w' \in W$, $(w, w') \in R^W$ *implies the following:*

   (a) If $j(w) > 0$, *then we have* $j(w) > j(w')$, $D(w') = D(w)$, *and* $t(w')^{E(w')} \Vdash \vec{\psi}$ *for any* $\psi \in E(w) \cap \mathcal{L}_{[l(w,w')]}$.

   (b) If $j(w) = 0$, *then we have* $E(w) = \varnothing$, $D(w') = \tau(D(w))$ *and* $E(w') = \mathrm{sat}_{D(w')}(t(w'))$.

(6) For $m \in \mathrm{FMS}$, w *has at most one* $w' \in W$ *such that either* $(w, w') \in R^W$ *and* $l(w, w') = m$ *or* $(w', w) \in R^W$ *and* $l(w', w) = m^{-1}$.

Proof. (1) is the invariant we keep through the construction and the others can also be checked easily. ■

LEMMA 15. *Suppose that* $(w_i \mid i \in \omega)$ *is a sequence of elements of* W *and* $(w_i, w_{i+1}) \in R^W$ *for all* $i \in \omega$. *Also suppose* $D \in \mathcal{D}_\mu$ *and* $l \in \omega$.

(1) There exists $k \in \omega$ such that $l \leq k$ and $E(w_k) = \varnothing$.

(2) There exists $k \in \omega$ such that $l < k$, $D(w_k) = D$, and $E(w_k) = \mathrm{sat}_D(t(w_k))$.

Proof. (1) is clear from Lemma 14 (5).
(2) By Lemma 14 (5), there is $k' \geq l$ such that $D(w_{k'}) = \tau(D(w_l))$. Since $\mathcal{D}_\mu$ is finite and τ is a cyclic permutation on $\mathcal{D}_\mu$, by repeating this finitely many times, we have $k > l$ such that $j(w_{k-1}) = 0$, $E(w_{k-1}) = 0$, $D(w_k) = \tau(D(w_{k-1})) = D$, and $E(w_k) = \mathrm{sat}_D(t(w_k))$. ■

Now we prove the key lemma of this section.

LEMMA 16. *The choice function c and the set Z satisfies the conditions of Lemma 5. Therefore, $t(w) \Vdash \varphi$ implies $\mathcal{K}, w \models \varphi$.*

Proof. The conditions (1) and (2) of Lemma 5 can be checked easily from the definitions.

To show the condition (3), we use the reduction to absurdity and assume that there exist $D \in \mathcal{D}_\mu$ and an infinite trace $((\varphi_k, w_k) \mid k < \omega)$ that conforms with c such that $t(w_k) \Vdash \varphi_k$ and $\varphi_k \in D$ for all $k < \omega$.

CLAIM 17. For all $k \in \omega$, $y_{t(w_k)}(\varphi_k) \geq y_{t(w_{k+1})}(\varphi_{k+1})$. If $w_k \neq w_{k+1}$ and $t(w_{k+1})$ has a name, $y_{t(w_k)}(\varphi_k) > y_{t(w_{k+1})}(\varphi_{k+1})$.

Proof. When $\varphi_k = \varphi \vee \psi$ or $\varphi_k = \langle m \rangle \varphi$, the claim holds by the definition of c. (If $m \in$ FMS, we need Lemma 14 (3) as well.) When φ_k is in the form of $\varphi \wedge \psi$ or $\mu X \varphi$, the claim follows from the definition of y. For $\varphi_k = [m]\varphi$, use Lemma 14 (4) and for $\varphi_k = @n\, \varphi$, use the definition of T_0. And φ_k cannot be an atomic formula, its negation, or $\nu X \varphi$ since $\varphi_k \in D \in \mathcal{D}_\mu$. ■

From this claim, it is clear that named nodes appear only finitely many times in the trace. (Note also that formulas are guarded. Therefore there cannot exist $K \in \omega$ such that $w_k = w_K$ for all $k \geq K$.) More precisely, we can take $k_0 \in \omega$ such that $t(w_k) \not\Vdash n$ for all $k \geq k_0$ and $n \in$ Nom.

CLAIM 18. If $K, L \in \omega$, $k_0 \leq K \leq L$, and $w_K = w_L$ (we denote it by w), then $x_{t(w)}(\varphi_K) \geq x_{t(w)}(\varphi_L)$. Moreover, if there is k' such that $K < k' < L$ and $w_{k'} \neq w$, then $x_{t(w)}(\varphi_K) > x_{t(w)}(\varphi_L)$.

Proof. We prove the claim by induction on $L - K$. The claim becomes trivial when $L - K = 0$, so we assume $K < L$.

If φ_K is in the form of $\varphi \vee \psi$, $\varphi \wedge \psi$, or $\mu X \varphi$, then $w_K = w_{K+1}$ and $x_{t(w)}(\varphi_K) \geq x_{t(w)}(\varphi_{K+1})$. Therefore the induction hypothesis leads to the conclusion. Since $k_0 \leq K$, φ_K cannot be in the form of $@n, \varphi$. Also, because $\varphi_K \in D$, it is neither an atomic formula or its negation.

The remaining cases are $\varphi_K = \langle m \rangle \varphi_{K+1}$ and $\varphi_K = [m]\varphi_{K+1}$, where $m \in$ Mod. In these cases, $w_{K+1} \neq w_K$ since W is a forest. Let M be the least index such that $K < M$ and $w_M = w_K$. Clearly $K + 1 \leq M -$

1 and $w_{K+1} = w_{M-1}$ since W is a forest. Let $w' = w_{K+1} = w_{M-1}$. By the induction hypothesis, we have (1) $x_{t(w')}(\varphi_{K+1}) \geq x_{t(w')}(\varphi_{M-1})$ and (2) $x_{t(w)}(\varphi_M) \geq x_{t(w)}(\varphi_L)$. Therefore if we show (3) $x_{t(w)}(\varphi_K) > x_{t(w)}(\varphi_M)$, we have $x_{t(w)}(\varphi_K) > x_{t(w)}(\varphi_L)$ as desired by combining (2) and (3). Note that either $\varphi_{M-1} = \langle m^{-1}\rangle\varphi_M$ or $\varphi_{M-1} = [m^{-1}]\varphi_M$ holds since w_{M-1} is different from w_M. In order to prove (3), it is sufficient to show either $\mathrm{LoopFree}(t(w), t(w'), \varphi_K)$ or $\mathrm{LoopFree}(t(w'), t(w), \varphi_{M-1})$ since we have (2).

First assume $\varphi_K = \langle m\rangle\varphi_{K+1}$. By Lemma 14 (3), $(t(w), t(w')) \in \mathrm{Tr}(\varphi_K)$. Therefore if $\varphi_{M-1} = [m^{-1}]\varphi_M$, $\mathrm{LoopFree}(t(w), t(w'), \varphi_K)$ holds. In the other case, i.e. $\varphi_{M-1} = \langle m^{-1}\rangle\varphi_M$, if we assume $m \notin \mathrm{FMS}$, again by Lemma 14 (3), we have $(w, w') \in R^W$ and $(w', w) \in R^W$, which is impossible since W is a forest. Thus, we have $m \in \mathrm{FMS}$, and then, $\mathrm{LoopFree}(t(w)$, $t(w'), \varphi_K)$ holds from the definition of $\mathrm{Tr}(m)$ and $\mathrm{Tr}(\langle m\rangle\varphi)$. The same argument can be done if we assume $\varphi_{M-1} = \langle m\rangle\varphi_M$.

The only case remained is $\varphi_K = [m]\varphi_{K+1}$ and $\varphi_{M-1} = [m^{-1}]\varphi_M$. Since $((\varphi_K, w_K), (\varphi_{K+1}, w_{K+1}))$ is a trace, $(w_K, w_{K+1}) \in R(m)$. Therefore either $(w_K, w_{K+1}) \in R^W$ and $l(w_K, w_{K+1}) = m$ or $(w_{K+1}, w_K) \in R^W$ and $l(w_{K+1}, w_K) = m^{-1}$. Then by Lemma 14 (4), $(t(w_K), t(w_{K+1})) \in \mathrm{Tr}(m)$ holds, and therefore we have $\mathrm{LoopFree}(t(w)$, $t(w')$, $\varphi_K)$ as desired. ■

From this lemma, it follows that the set $\{k \in \omega \mid w = w_k\}$ is finite for any $w \in W$, We define $m(w)$ by $\max\{k \in \omega \mid w = w_k\}$, and a sequence $(L(i) \mid i \in \omega)$ of natural numbers by $L(0) = m(w_{k_0})$ and $L(i{+}1) = m(w_{L(i)+1})$. It is clear that $w_{L(i+1)} = w_{L(i)+1}$. Note that $\varphi_{L(i)} \in \mathrm{Lean}$ since $w_{L(i)+1} \neq w_{L(i)}$.

CLAIM 19. There is a natural number $i_0 \in \omega$ such that for all $i \in \omega$, $i \geq i_0 \implies (w_{L(i)}, w_{L(i+1)}) \in R^W$.

Proof. Since $w_{L(i)} \neq w_{L(i)+1}$ and $L(i) \geq k_0$, $\varphi_{L(i)}$ is either $\langle m\rangle\varphi_{L(i)+1}$ or $[m]\varphi_{L(i)+1}$ for some $m \in \mathrm{Mod}$. Therefore $(w_{L(i)}, w_{L(i)+1}) \in R(m)$, that is, either $(w_{L(i)}, w_{L(i)+1}) \in R^W$ or $(w_{L(i)+1}, w_{L(i)}) \in R^W$ holds. Since W is a forest, the relation R^W is well-founded, so there is at least one $i_0 \in \omega$ that $(w_{L(i_0)}, w_{L(i_0)+1}) \in R^W$. We show by induction on i, $(w_{L(i)}, w_{L(i+1)}) \in R^W$ for all $i \geq i_0$. The case $i = i_0$ is trivial. The case $i + 1$: recall that either $(w_{L(i+1)}, w_{L(i+1)+1}) \in R^W$ or $(w_{L(i+1)+1}, w_{L(i+1)}) \in R^W$ holds. If the latter is the case, since $(w_{L(i)}, w_{L(i)+1}) \in R^W$, $w_{L(i)+1} = w_{L(i+1)}$, and W is a forest, we have $w_{L(i)} = w_{L(i+1)+1}$, which contradicts the fact that $L(i)$ is the largest index since $L(i) < L(i+1) + 1$. Therefore the former should be the case, i.e., $(w_{L(i+1)}, w_{L(i+1)+1}) \in R^W$. ■

The reflexive transitive closure of R^W is denoted by R^{W+}.

CLAIM 20. For any $j \in \omega$ and $k > L(i_j)$, $(w_{L(i_j)}, w_k) \in R^{W+}$ holds.

Proof. We prove the claim by induction on k. The base case $k = L(i_j) + 1$ is clear from Claim 19. For the general case, assume $(w_{L(i_j)}, w_k) \in R^{W+}$. If we further assume $(w_{L(i_j)}, w_{k+1}) \notin R^{W+}$, w_{k+1} must be identical to $w_{L(i_j)}$

since $\varphi_k = \langle m \rangle \varphi_{k+1}$ or $\varphi_k = [m]\varphi_{k+1}$ for some $m \in \mathrm{Mod}$ because $k \geq k_0$. It again contradicts the fact that $L(i_j)$ is the maximum index. ■

By applying Lemma 15 (2) to the sequence $(w_{L(i)} \mid i_0 \leq i < \omega)$, we can take $i_1 \in \omega$ such that $i_1 \geq i_0$, $D(w_{L(i_1)}) = D$, and $E(w_{L(i_1)}) = \mathrm{sat}_D(t(w_{L(i_1)}))$. In particular, $\varphi_{L(i_1)} \in E_{L(i_1)}$.

CLAIM 21. For all $k \geq L(i_1)$, $D(w_k) = D$ and $t(w_k)^{E(w_k)} \Vdash \varphi_k$ holds.

Proof. We use induction on k. The base step $k = L(i_1)$ is trivial.

For the general step, we first consider the case $\varphi_k = \varphi_{k+1} \vee \psi$. Since $w_k = w_{k+1}$, $D(w_{k+1}) = D$. Note that $\psi \in D$, otherwise c should have chosen ψ. Let $(x, y) = t(w_k)$ and $E = E(w_k)$. By the definition of c, either $x(\varphi_{k+1}) < x(\psi)$ or $x(\varphi_{k+1}) = x(\psi)$ and $x^E(\varphi_{k+1}) \leq x^E(\psi)$ holds. In the former case, by Lemma 13, $x^E(\varphi_{k+1}) < x^E(\psi)$ or $x^E(\varphi_{k+1}) = x^E(\psi) = \infty$. Here, $x^E(\varphi_{k+1}) = x^E(\psi) = \infty$ implies $x^E(\varphi_k) = \infty$, which is impossible. Therefore, in both cases, $x^E(\varphi_{k+1}) \leq x^E(\psi)$, which implies $x^E(\varphi_{k+1}) = x^E(\varphi_k) < \infty$.

The cases $\varphi_k = \varphi_{k+1} \wedge \psi$ and $\mu X \psi$ are easy and we omit them.

The remaining cases are $\varphi_k = \langle m \rangle \varphi_{k+1}$ and $[m]\varphi_{k+1}$. In these two cases, either $(w_{k+1}, w_k) \in R^W$ or $(w_k, w_{k+1}) \in R^W$ holds. Assume first $(w_{k+1}, w_k) \in R^W$. Clearly $D(w_{k+1}) = D(w_k) = D$. Since for all j such that $L(i_1) \leq j < k$, one of $(w_{j+1}, w_j) \in R^W$, $(w_j, w_{j+1}) \in R^W$, or $w_j = w_{j+1}$ holds, so by Claim 20, there is i such that $L(i_1) \leq i < k+1$ such that $w_i = w_{k+1}$. Let $E = E(w_i)$ and $(x, y) = t(w_i)$. By induction hypothesis we have $x(\varphi_i) < \infty$. On the other hand we have $x(\varphi_i) > x(\varphi_{k+1})$ by Lemma 18. Therefore by Lemma 13, $x^E(\varphi_{k+1}) < x^E(\varphi_i) < \infty$.

Assume next $(w_k, w_{k+1}) \in R^W$ holds. In the case of $\varphi_k = \langle m \rangle \varphi_{k+1}$, we have $t(w_{k+1})^{E(w_{k+1})} \Vdash \varphi_{k+1}$ by the definition of c and the induction hypothesis. In the case of $\varphi_k = [m]\varphi_{k+1}$, since $\varphi_k \in E(w_k) \neq \varnothing$, $j(w_k)$ must be positive by Lemma 14 (5). Then again by Lemma 14 (5), $D(w_{k+1}) = D(w_k) = D$ and $t(w_{k+1})^{E(w_{k+1})} \Vdash \varphi_{k+1}$. ■

Claim 21 contradicts Lemma 15 (1), which establishes the condition (3) of Lemma 5. That completes the proof of Lemma 16. ■

THEOREM 22. *Assume that the decision procedure succeeds for* φ_{I} *in either* $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{back})$*,* $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{back}, \mathrm{func})$*, or* $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{func})$*. Then, there is a Kripke structure and its state that satisfies* φ_{I}*.*

Proof. Lemma 12 guarantees that $\mathcal{K}$ is a Kripke structure. Since $t(w_{\mathrm{I}}) = t_{\mathrm{I}} \Vdash \varphi_{\mathrm{I}}$, $\mathcal{K}, w_{\mathrm{I}} \models \varphi$ by Lemma 16. ■

5 Complexity

Our decision procedure solves the satisfiability problems of three logics – $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{back})$, $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{func})$, and $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{back}, \mathrm{func})$. Their complexities have already been known to be EXPTIME-complete [10]. More detailed calculation of the time complexity of our procedure is shown in the following proposition. Let n be the length of the formula φ_{I}.

PROPOSITION 23. *The time complexity of the decision procedure presented in Section 3 is* $2^{\mathcal{O}(n \log n)}$ *for formulas in* $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{back}, \mathrm{func})$ *and* $2^{\mathcal{O}(n^2 \log n)}$ *for formulas in* $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{back})$ *and* $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{func})$.

Proof. Let us count the number of naming functions. The domain of a naming function is Nom and the range is Tab. Such functions exist at most $(n^n \cdot n^{2n})^n = 2^{3n^2 \log n}$. In the case of $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{back}, \mathrm{func})$, the number of functions is 1 since there are no nominals. Therefore it is enough to show that the time complexity of the subprocedure for a naming function is $2^{\mathcal{O}(n \log n)}$.

Next we count the number A of nodes in the tableau. Component x can be regarded as a function from Lean to $\{0, 1, \ldots, n-1\}$. The number of such functions is at most n^n. Component y can be regarded as a function from Lean to $\{0, 1, \ldots, n^2-1\}$. The number of such functions is at most $(n^2)^n$. Therefore, $A \leq n^n \cdot (n^2)^n = 2^{\mathcal{O}(n \log n)}$.

The body of the subprocedure is a double-loop. In the outer loop, $T_0 \supset T_1 \supset \cdots \supset T_K$ are calculated. Since $|T_0| \leq A$, the number of repetition K does not exceed A. The inner loop calculates $V_0 \subset V_1 \subset \cdots \subset V_J$. Since each V_j is a subset of $T_0 \times \mathcal{P}(\mathrm{Lean})$, the number of repetition J does not exceed $B = A \cdot 2^n$.

In each repetition of the inner loop, V_{j+1} is calculated from V_j. This is done by verifying that each (t, E) is an element of V_{j+1}. The number of such (t, E) does not exceed B. Each (t, E) is checked against conditions (a), (b), and (c). They can be performed in polynomial time C once pair (t', E') is fixed and the number of candidates (t', E') does not exceed B.

Therefore, time required to perform the subprocedure is $A \cdot B \cdot B \cdot C \cdot B = C \cdot A^4 \cdot (2^n)^3 = 2^{\mathcal{O}(n \log n)}$. ■

Although we need to transform a given formula into a guarded PNF before applying the decision procedure, we can still presume that n in the proposition refers the length of the given formula, since the proof of Proposition 23 can be done by regarding n as the size of $\mathrm{cl}(\varphi_{\mathrm{I}})$.

6 Application

We apply the decision procedure to verify some properties of programs written in imperative languages that manipulate pointers.

We regard program heaps as Kripke structures. Thus, a property of a heap is expressed by a formula φ of the μ-calculus. Each operation σ on the heap can be regarded as a transformation of Kripke structures, and we compute the weakest precondition $\mathrm{wp}(\sigma, \varphi)$ as a formula. Then, we can determine whether a Kripke structure that satisfies φ can be transformed into a structure that satisfies ψ by verifying that formula $\varphi \wedge \mathrm{wp}(\sigma, \psi)$ is satisfiable. Refer to [19] for details. We built an experimental tool, called

id	len	# cl	# nom	time
regr56	35	11	3	461
listRevNoLeakB	92	25	5	610
listRevSwapA	148	37	6	1199
dswPopC2	188	46	7	2615
dswPushA1	372	54	7	9469

Table 1. Examples of formulas used in the analysis

MLAT [20], based on this method. It is programmed in Java, and the decision procedure is implemented using JavaBDD [7].

There are several versions of MLAT. By using one of the versions, we verified the partial correctness of the Deutsch-Schorr-Waite marking algorithm [21]. Details are reported in [19].

Using this version of the tool, the user constructs an abstract transition system by hand, and the tool checks whether the transition system is a correct abstraction of the concrete transition system corresponding to a given source code. As the logic for describing predicates, we employ $\mathcal{L}(\mu_{\mathrm{AF}}, \mathrm{nom}, \mathrm{back})$ for this version.

We simplify the procedure for efficiency; we do not use the component y at all. When the procedure with this simplification succeeds, a nominal may be satisfied with two different nodes s_1 and s_2 in the resulting "model"; however, such nodes are "indistinguishable" in the sense that $s_1 \models \varphi \iff s_2 \models \varphi$ holds for any $\varphi \in \mathrm{cl}(\varphi_{\mathrm{I}})$.

The modified procedure is still sound. In general, any sound decision procedure can be used in the predicate abstraction technique. The procedure should not necessarily be complete, but using incomplete procedures may affect the precision of the analysis. In our analysis, however, there is another reason for losing the precision: the selection of predicates. Our experiment shows that the second reason affects the results more than the first. We have several cases in which we failed to analyze the properties due to inappropriate selection of predicates, but there is no case in which the analysis failed due to incompleteness of the procedure. Apparently, "indistinguishability" is a good approximation for nominality in our application.

During the verification, a number of formulas were checked for satisfiability. Examples are shown in Table 1. The columns of the table are formula ID, the length of the formula, the size of the closure, the number of nominals in the formula, and elapsed time for the judgment in milliseconds. The machine used for the the measurement is Pentium 4 CPU 2.4GHz, 1GB memory running Microsoft Windows XP.

7 Future work

In the future we would like to implement the entire decision procedure. At the moment, we have only implemented modified versions of the procedure for particular applications, as described in Section 6.

Another possibility is to strengthen our decision procedures for applications to more powerful logics. The loosely guarded fragment (LGF) is a decidable sublogic of the first-order predicate logic, and the extension of the LGF with fixed-point operators is called μ-LGF, which can be regarded as a natural extension of the two-way modal μ-calculus. A natural question is whether our procedure can be extended to the alternation-free part of μ-LGF. A simple adaptation of our procedure to the logic does not work, and further investigations are required.

Acknowledgments. The authors are grateful to the anonymous reviewers for their careful reading. This research was partially supported by the research project "Solving the description explosion problem in verification by means of structure transformation" in CREST (Core Research for Evolution Science and Technology) program of the Japan Science and Technology Agency. This research was partially supported by Grant-in-Aid for Scientific Research by Ministry of Education, Culture, Science and Technology, Scientific Research(C) (2)18500003, "Abstraction of graphs to multisets using temporal logic".

BIBLIOGRAPHY

[1] Takahashi, K., Hagiya, M.: Abstraction of graph transformation using temporal formulas. In: Supplemental Volume of the 2003 International Conference on Dependable Systems and Networks (DNS-2003). (2003) W–65 to W–66

[2] Hagiya, M., Takahashi, K., Yamamoto, M., Sato, T.: Analysis of synchronous and asynchronous cellular automata using abstraction by temporal logic. In: FLOPS2004: The Seventh Functional and Logic Programming Symposium. Volume 2998 of Lecture Notes in Computer Science. (2004) 7–21

[3] Tanabe, Y., Takahashi, K., Yamamoto, M., Sato, T., Hagiya, M.: An implementation of a decision procedure for satisfiability of two-way CTL formulae using BDD (in Japanese). Computer Software **22**(3) (2005) 154–166

[4] Tanabe, Y., Takahashi, K., Yamamoto, M., Tozawa, A., Hagiya, M.: A decision procedure for the alternation-free two-way modal μ-calculus. In Beckert, B., ed.: TABLEAUX. Volume 3702 of Lecture Notes in Computer Science., Springer (2005) 277–291

[5] Sagiv, M., Reps, T., Wilhelm, R.: Parametric shape analysis via 3-valued logic. ACM Transactions on Programming Languages and Systems **24**(3) (2002) 217–298

[6] Bonatti, P.A., Peron, A.: On the undecidability of logics with converse, nominals, recursion and counting. Artificial Intelligence **158** (2004) 75–96

[7] JavaBDD: `http://javabdd.sourceforge.net/`

[8] Kozen, D.: Results on the propositional μ-calculus. Theoretical Computer Science **27**(3) (1983) 333–354

[9] Emerson, E.A., Jutla, C.S.: Tree automata, mu-calculus and determinacy (extended abstract). In: FOCS, IEEE (1991) 368–377

[10] Bonatti, P.A., Lutz, C., Murano, A., Vardi, M.Y.: The complexity of enriched μ-calculi. In Bugliesi, M., Preneel, B., Sassone, V., Wegener, I., eds.: ICALP (2). Volume 4052 of Lecture Notes in Computer Science., Springer (2006) 540–551

[11] Emerson, E.A.: Temporal and modal logic. In: Handbook of theoretical computer science (vol. B): formal models and semantics, Elsevier Science Publishers B.V. (1990) 995–1072

[12] Pan, G., Sattler, U., Vardi, M.Y.: BDD-based decision procedures for K. In: Proceedings of the 18th International Conference on Automated Deduction, Springer-Verlag (2002) 16–30

[13] Pan, G., Vardi, M.Y.: Optimizing a BDD-based modal solvar. In: 19th International Conference on Automated Deduction. Volume 2741 of Lecture Notes in Computer Science. (2003) 75–89

[14] Henriksen, J.G., Jensen, J.L., Jørgensen, M.E., Klarlund, N., Paige, R., Rauhe, T., Sandholm, A.: Mona: Monadic second-order logic in practice. In: Proceedings of the First International Workshop on Tools and Algorithms for Construction and Analysis of Systems. Volume 1019 of Lecture Notes in Computer Science., Springer-Verlag (1995) 89–110

[15] Kupferman, O., Vardi, M.Y.: Safraless decision procedures. In: FOCS, IEEE Computer Society (2005) 531–542

[16] van Eijck, J.: HyLoTab — tableau-based theorem proving for hybrid logics (2002) Manuscript, CWI, available from `http://www.cwi.nl/ jve/hylotab`.

[17] Walukiewicz, I.: Completeness of Kozens axiomatisation of the propositional mu-calculus. Information and Computation **157** (2000) 142–182

[18] Zappe, J.: Modal μ-calculus and alternating tree automata. In Grädel, E., Thomas, W., Wilke, T., eds.: Automata, Logics, and Infinite Games. Volume 2500 of Lecture Notes in Computer Science., Springer (2001) 171–184

[19] Tanabe, Y.: Satisfiability Judgment of Modal Logics and their Application to Verification Problems. PhD thesis, University of Tokyo (2008)

[20] Sekizawa, T., Tanabe, Y., Yuasa, Y., Takahashi, K.: MLAT: A tool for heap analysis based on predicate abstraction by modal logic. In: IASTED International Conference on Software Engineering (SE 2008). (2008) 310–317

[21] Schorr, H., Waite, W.M.: An efficient machine-independent procedure for garbage collection in various list structures. Commun. ACM **10**(8) (1967) 501–506

Yoshinori Tanabe
Graduate School of Information Science and Technology,
University of Tokyo.
Akihabara Daibiru 13F, 1-18-13, Sotokanda, Tokyo, 101-0021,
Japan.
`y-tanabe@ci.i.u-tokyo.ac.jp`

Koichi Takahashi
Research Center for Verification and Semantics (CVS),
National Institute of Advanced Industrial Science and Technology.
1-2-14, Shinsenri Nishimachi, Toyonaka, Osaka, 560-0083,
Japan.
`k.takahashi@aist.go.jp`

Masami Hagiya
Graduate School of Information Science and Technology,
University of Tokyo.
7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-8656,
Japan.
`hagiya@is.s.u-tokyo.ac.jp`