R. Kontchakov
C. Lutz
F. Wolter
M. Zakharyaschev

Temporalising tableaux

Abstract. As a remedy for the bad computational behaviour of first-order temporal logic (FOTL), it has recently been proposed to restrict the application of temporal operators to formulas with at most one free variable thereby obtaining so-called *monodic* fragments of FOTL. In this paper, we are concerned with constructing tableau algorithms for monodic fragments based on decidable fragments of first-order logic like the two-variable fragment or the guarded fragment. We present a general framework that shows how existing decision procedures for first-order fragments can be used for constructing a tableau algorithm for the corresponding monodic fragment of FOTL. Some example instantiations of the framework are presented.

Keywords: first-order temporal logic, monodic fragment, tableau algorithm.

1. Introduction

First-order temporal logic (FOTL) based on the flow of time $\langle \mathbb{N}, < \rangle$ is notorious for its bad computational behaviour: even the two-variable monadic fragment of this logic is not recursively enumerable (see e.g. [11] and references therein). A certain breakthrough has recently been achieved in [11], where the so-called *monodic fragment* of FOTL is introduced by restricting applications of temporal operators to formulas with at most one free variable. The full monodic fragment (containing full first-order logic) turns out to be axiomatisable [26]. Moreover, by restricting its first-order part to decidable fragments, we obtain decidable monodic FOTLs, say, the monodic guarded, monodic two-variable, and monodic monadic fragments. This opens a way to various applications of the monodic FOTL in knowledge representation, temporal databases, program specification and verification, and other fields. For example, many temporal description logics and spatio-temporal logics can be regarded as fragments of monodic FOTL [12, 7, 27]. Unfortunately, the decision procedures provided in [11] are of model-theoretic character and cannot be used as a basis for implementations. In [1] and quite recently in [3], a resolution-based approach has been first developed for certain subfragments of the monodic fragment and then for the full monodic fragment.

Presented by **Heinrich Wansing**; *Received* December 1, 2002

Studia Logica **68**: 1–46, 2001.

A *tableau-based analysis* of the decision problem for monodic FOTL has been missing. In this paper we are trying to fill in this gap. More specifically, our aims are as follows:

1. *to develop a general framework for devising tableau-based decision procedures for decidable monodic FOTLs and then,*

2. *within this framework, to construct tableau systems for a number of concrete monodic fragments.*

We consider monodic FOTLs interpreted in models with both expanding and constant domains. The former case is technically much easier, but the latter one is more general: reasoning with expanding domains can be reduced to reasoning with constant domains, but not vice versa (see e.g., [7]).

Our approach is based on the following ideas:

- *modularity*—a decision procedure for a given fragment of first-order logic is combined with Wolper's tableaux [20] for propositional temporal logic (PTL);

- *finite quasimodel* representations of temporal models with potentially infinite first-order domains—elements indistinguishable by the subformulas (of a given formula) with at most one free variable are represented by the same *type*;

- the *minimal type* technique for dealing with constant domains in temporal models [16]

To describe the proposed framework in some more depth, let us assume that the satisfiability of a monodic FOTL formula ϑ has to be decided. The 'temporal' tableau algorithm tries to construct a model for ϑ, i.e., a (one-side) infinite sequence of classical first-order models. To achieve modularity, we separate the temporal and the pure first-order parts of ϑ and treat the former using Wolper's tableau for PTL and the latter using available decision procedures for fragments of first-order logic. More precisely, the temporal tableau algorithm first replaces all subformulas of ϑ that start with temporal operators by their 'surrogates,' i.e., by unary predicates. *Unary* predicates are sufficient here, since we are dealing with *monodic* FOTLs. The proper 'temporal behaviour' of the surrogates is ensured by some auxiliary *surrogate axioms*, which are passed to the first-order decision procedure along with the surrogated version of ϑ. This decision procedure is expected to provide us with descriptions of possible models for its input. We then choose an appropriate model $\mathfrak{M}$ for the current time point and make one 'step in time' by omitting the 'next-time' operator (as in Wolper's tableaux) and adding new surrogate axioms. This way we build up a temporal tableau. When such a tableau is completed, the *pruning technique*—which is also used in Wolper's tableau for PTL—is employed to check whether all eventualities are fulfilled, i.e., whether the tableau represents a temporal model of the input formula.

Additional effort is needed to preserve the representation of tableaux finite and to guarantee termination. For example, first-order models are represented by finite sets of *types*, each representing a possibly infinite number of domain elements. *Quasimodels*, which are well-known from e.g. [11], are used to encode temporal models by associating a finite set of types with each time instant. To avoid constructing an infinite number of (finite representations of) first-order models, we use *blocking* to detect and avoid duplicates.

Two rather general theorems, one for expanding domains and one for constant domains, provide conditions under which a first-order decision procedure can be combined with Wolper's tableaux to yield a tableau-based decision procedure for the corresponding monodic FOTL. The price we have to pay for this level of generality is that the resulting combined tableaux are far from optimal. In particular, in many concrete cases new tableau rules can be used instead of surrogate axioms. Thus, our general framework for combining tableaux is not supposed for direct applications or implementations, but rather as a *guide* for considering more specific cases.

The paper is organised as follows. In Section 2 we define the syntax and semantics of first-order temporal logic and introduce the monodic fragment of FOTL. We start Section 3 with characterising decision procedures for fragments of first-order logic that can be used as building blocks in tableau calculi for monodic fragments (so-called saturation rules). Then we prove that quasimodels are a proper abstraction of temporal models. In Section 4 we show how to obtain a tableau procedure for a monodic fragment based on an existing decision procedure for the corresponding FO fragment. We prove termination, soundness and completeness of the algorithm for both expanding and constant domains. In Section 5, two example instantiations of our framework are presented: we describe two standard first-order tableau algorithms (for the one-variable fragment and the modal logic $\mathsf{S4}_u$—i.e., Lewis's $\mathsf{S4}$ with the universal modality) and prove that they can be considered as saturation rules. Then we present some applications of the tableau algorithm for the temporalisation of the one-variable fragment of first-order logic. We conclude in Section 6.

2. First-order temporal logic

In this section, we introduce first-order temporal logic and its monodic fragment. Then we consider monodic fragments as 'temporalisations' of certain fragments of first-order logic and show how the monodic formulas can be split up into temporal and first-order parts.

Let $\mathcal{QTL}$ be the *first-order* (or *quantified*) *temporal language* based on the following vocabulary:

- predicate symbols $P_0, P_1, \ldots$, each of which is of some fixed arity ≥ 0;
- a countably infinite set $\mathcal{V}$ of individual variables $x_0, x_1, \ldots$;
- the Boolean connectives $\wedge$ and $\neg$;
- the universal quantifier $\forall x$ for every individual variable x;
- the temporal operators $\mathcal{U}$ ('until') and $\bigcirc$ ('next-time').

Remark 2.1. Note that our language contains neither constant symbols nor equality. The reason for omitting the constants is to simplify presentation by avoiding unnecessary technical details. The reader should not have any problems to extend the method developed in the paper to the language with constant symbols. Equality and/or function symbols may ruin good algorithmic properties of the monodic fragment by making it not recursively enumerable [11]. Moreover, it is shown in [2] that the monodic monadic two-variable fragment with equality is undecidable; see, however, [10] where it is shown that the monodic packed fragment with equality is decidable.

The set of $\mathcal{QTL}$-*formulas* is defined as follows:

- if P is an m-ary predicate symbol and $x_1, \ldots, x_m$ are variables, then $P(x_1, \ldots, x_m)$ is an (atomic) formula;
- if φ and ψ are formulas, then so are $\varphi \wedge \psi$ and $\neg\varphi$;
- if φ is a formula and x a variable, then $\forall x\, \varphi$ is a formula;
- if φ and ψ are formulas, then so are $\varphi\, \mathcal{U}\, \psi$ and $\bigcirc\varphi$.

We use the standard abbreviations $\vee$, $\rightarrow$, and

$$\top = \tau, \quad \bot = \neg\top, \quad \exists x\, \varphi = \neg\forall x\, \neg\varphi, \quad \Diamond\varphi = \top\, \mathcal{U}\, \varphi, \quad \Box\varphi = \neg\Diamond\neg\varphi,$$

where τ is some fixed tautology. Intuitively, $\Diamond$ means 'now or sometime in the future' and $\Box$ means 'from now on.'

For a given formula φ, $sub(\varphi)$ denotes the set of subformulas of φ and $free(\varphi)$ the set of variables occurring free in φ. We write $\varphi(x_1, \ldots, x_m)$ to indicate that all free variables of φ are in the set $\{x_1, \ldots, x_m\}$; in particular, $\varphi(x)$ has at most one free variable x. The pure (non-temporal) first-order fragment of $\mathcal{QTL}$ is denoted by $\mathcal{QL}$.

Let us now define the semantics of $\mathcal{QTL}$: in principle, we just have to fix a flow of time and then relate each moment of time with some first-order model. Since in this paper we are concerned with the flow of time $\langle \mathbb{N}, < \rangle$, it thus suffices to associate with each moment $n \in \mathbb{N}$ a first-order model. Thus we obtain $\mathcal{QTL}$-models, in which domains of first-order structures can vary along the time axis. However, a more natural (and more powerful) semantics is obtained by additional restrictions on the domains. In what follows, we consider two kinds of temporal models: with expanding and constant domains. The former class of models is much easier to be dealt with by tableau decision procedures (as well as by resolution [3]), whereas the latter one is more general, since reasoning with expanding (or, in general, varying) domains can be reduced to reasoning with constant domains; see e.g., [5, 25, 7].

DEFINITION 2.2 (model). A *$\mathcal{QTL}$-model* is a triple $\mathfrak{M} = \langle \mathbb{N}, <, I \rangle$, where $\langle \mathbb{N}, < \rangle$ is the set of natural numbers equipped with the usual strict order $<$, and I is a function associating with each $n \in \mathbb{N}$ some first-order model

$$I(n) = \left\langle \Delta_n, P_0^{I(n)}, P_1^{I(n)}, \ldots \right\rangle,$$

where Δ_n is a non-empty set and each $P_i^{I(n)}$ is a relation on Δ_n of the same arity as P_i. $\mathfrak{M}$ is said to be a *model with expanding domains* if $\Delta_i \subseteq \Delta_j$ whenever $i < j$, and $\mathfrak{M}$ is called a *model with constant domains* if $\Delta_i = \Delta_j$ for all $i, j \in \mathbb{N}$.

From now on by a $\mathcal{QTL}$-model we mean a $\mathcal{QTL}$-model with expanding or constant domains.

There are different approaches to defining truth in $\mathcal{QTL}$-models; see e.g., [13]. We take the following version due to [14]:

DEFINITION 2.3 (truth). Let $\mathfrak{M} = \langle \mathbb{N}, <, I \rangle$ be a $\mathcal{QTL}$-model. An *assignment* $\mathfrak{a}$ in $\mathfrak{M}$ is a function from the set $\mathcal{V}$ of individual variables to $\bigcup_{n \in \mathbb{N}} \Delta_n$. Given a $\mathcal{QTL}$-formula ϑ, the *truth-relation* $(\mathfrak{M}, n) \models^{\mathfrak{a}} \vartheta$ ('ϑ is *true* at moment n in model $\mathfrak{M}$ under assignment $\mathfrak{a}$') is defined inductively on the construction of ϑ *for only those assignments* $\mathfrak{a}$ *that satisfy the condition* $\mathfrak{a}(x) \in \Delta_n$ for all $x \in \mathit{free}(\vartheta)$:

- $(\mathfrak{M}, n) \models^{\mathfrak{a}} P(x_1, \ldots, x_m)$ iff $\langle \mathfrak{a}(x_1) \ldots, \mathfrak{a}(x_m) \rangle \in P^{I(n)}$;

- $(\mathfrak{M}, n) \models^{\mathfrak{a}} \neg\varphi$ iff $(\mathfrak{M}, n) \not\models^{\mathfrak{a}} \varphi$;
- $(\mathfrak{M}, n) \models^{\mathfrak{a}} \varphi \wedge \psi$ iff $(\mathfrak{M}, n) \models^{\mathfrak{a}} \varphi$ and $(\mathfrak{M}, n) \models^{\mathfrak{a}} \psi$;
- $(\mathfrak{M}, n) \models^{\mathfrak{a}} \forall x\, \varphi$ iff $(\mathfrak{M}, n) \models^{\mathfrak{b}} \varphi$ for every assignment $\mathfrak{b}$ that may differ from $\mathfrak{a}$ only on x, provided that $\mathfrak{b}(x) \in \Delta_n$;
- $(\mathfrak{M}, n) \models^{\mathfrak{a}} \varphi\, \mathcal{U}\, \psi$ iff there is $m \geq n$ such that $(\mathfrak{M}, m) \models^{\mathfrak{a}} \psi$ and $(\mathfrak{M}, k) \models^{\mathfrak{a}} \varphi$ for all $k \in [n, m)$, where $[n, m) = \{k \mid n \leq k < m\}$;
- $(\mathfrak{M}, n) \models^{\mathfrak{a}} \bigcirc\varphi$ iff $(\mathfrak{M}, n+1) \models^{\mathfrak{a}} \varphi$.

A $\mathcal{QTL}$-formula φ is said to be *satisfiable in expanding domains* (or *satisfiable*, for short) if $(\mathfrak{M}, n) \models^{\mathfrak{a}} \varphi$ holds for some model $\mathfrak{M}$ with expanding domains, moment n and assignment $\mathfrak{a}$ in $\mathfrak{M}$. If $\mathfrak{M}$ is a model with constant domains, we say that φ is *satisfiable in constant domains*. The notions of *validity* and *validity in constant domains* are defined in the dual way. It is not hard to see that satisfiability in constant domains implies satisfiability in expanding domains, but not vice versa: the formula

$$\forall x \bigcirc\neg C(x) \wedge \bigcirc\exists x\, C(x)$$

is satisfiable in expanding domains, but not in constant domains. Note that both $\mathcal{QTL}$ with expanding domains and $\mathcal{QTL}$ with constant domains are conservative extensions of classical first-order logic in the language $\mathcal{QL}$.

Throughout this paper, we will not be distinguishing between a finite set Γ of formulas and the conjunction $\bigwedge\Gamma$ of formulas in it. In particular, we write $(\mathfrak{M}, n) \models^{\mathfrak{a}} \Gamma$ to say that $(\mathfrak{M}, n) \models^{\mathfrak{a}} \varphi$ for every $\varphi \in \Gamma$. Instead of $(\mathfrak{M}, n) \models^{\mathfrak{a}} \varphi(x_1, \dots, x_m)$ we often write $(\mathfrak{M}, n) \models \varphi[a_1, \dots, a_m]$, where $\mathfrak{a} = \{x_1 \mapsto a_1, \dots, x_m \mapsto a_m\}$.

2.1. The monodic fragment

As is known too well, first-order temporal logic and even its 'small' fragments such as the two-variable monadic fragment are not recursively enumerable (see [6] and references therein). The maximal 'well-behaved' sublanguage of $\mathcal{QTL}$ that has been discovered so far [11] consists of so-called monodic formulas.

DEFINITION 2.4 (monodic fragment). A $\mathcal{QTL}$-formula is said to be *monodic* if it contains no subformula of the form $\varphi\, \mathcal{U}\, \psi$ or $\bigcirc\varphi$ with more than one free variable. The set of all monodic formulas will be denoted by $\mathcal{QTL}_{\boxed{1}}$.

Two important results concerning the monodic fragment are relevant here. First, the set of valid (in constant domains) monodic formulas is finitely axiomatisable [26], and so there exists a semi-decision procedure (as $\mathcal{QTL}_{\boxdot}$ clearly contains full $\mathcal{QL}$, it is undecidable). The second result obtained in [11] states (roughly) that, if we take a fragment of $\mathcal{QTL}_{\boxdot}$ the underlying first-order (non-temporal) part of which is decidable, then this fragment itself is decidable as well. Examples of decidable monodic fragments are:

- the two-variable monodic fragment $\mathcal{QTL}^2_{\boxdot}$;
- the monadic monodic fragment $\mathcal{QTL}^{mo}_{\boxdot}$;
- the guarded monodic fragment $\mathcal{TGF}_{\boxdot}$ (in which quantification is restricted to patterns $\forall \bar{y}\,(\gamma \to \varphi)$, where $\bar{y}$ is a tuple of variables, every free variable in φ is free in γ as well, and the 'guard' γ is an atomic formula).

2.2. Temporalisation

These and other similar fragments $\mathcal{QTL}' \subseteq \mathcal{QTL}_{\boxdot}$ can be regarded as *temporalisations* of the corresponding first-order fragments $\mathcal{QL}' \subseteq \mathcal{QL}$ (two-variable, monadic, guarded, etc.) by extending their formula-formation rules with the following one:

$$\text{if } \varphi(x) \text{ and } \psi(x) \text{ are } \mathcal{QTL}'\text{-formulas, then so are } \bigcirc\varphi(x) \text{ and } \varphi(x)\,\mathcal{U}\,\psi(x). \qquad (\dagger)$$

Various temporalisations of expressive propositional modal (say, epistemic, description, or dynamic) logics [17, 21, 23, 22, 24, 7] can also be viewed as fragments of $\mathcal{QTL}_{\boxdot}$. However, we have to be careful here because not all constructors of these logics are first-order definable, for instance, the transitive reflexive closure of binary relations used in some description logics and PDL.

To include such logics in our general framework, we first define a *fragment theory* (or an *f-theory*, for short) as a pair $(\mathcal{QL}', \mathsf{M})$, where $\mathcal{QL}' \subseteq \mathcal{QL}$ and M is a class of models in the signature of $\mathcal{QL}'$ (or its extension). For instance, the two-variable fragment $\mathcal{QL}^2$ of $\mathcal{QL}$ can be considered as the f-theory $(\mathcal{QL}^2, \mathsf{FO})$, where FO is the class of all first-order models.

As another example take the propositional bimodal logic $\mathsf{S4}_u$, i.e., Lewis's $\mathsf{S4}$ with the universal modality $\boxed{\forall}$ (see [8]). Let $\mathcal{ST}$ be the set of *standard* *translations* of $\mathsf{S4}_u$-formulas defined by taking

$$\begin{aligned}
(p)^* &= P(x), \quad \text{for a propositional variable } p,\\
(\varphi \wedge \psi)^* &= \varphi^* \wedge \psi^*,\\
(\neg\varphi)^* &= \neg\varphi^*,\\
(\Box\varphi)^* &= \forall y\,\big(R(x,y) \to \varphi^*\{y/x\}\big), \quad \text{for a fresh variable } y,\\
(\boxed{\forall}\varphi)^* &= \forall x\,\varphi^*,
\end{aligned}$$

where $\varphi\{y/x\}$ denotes the result of replacing the free variable x in φ with y. Denote by TR the class of all first-order models of the form

$$\mathfrak{M} = \left\langle W, R^{\mathfrak{M}}, P_0^{\mathfrak{M}}, P_1^{\mathfrak{M}}, \dots \right\rangle,$$

where $R^{\mathfrak{M}}$ is a transitive and reflexive relation on W and the $P_i^{\mathfrak{M}}$ are arbitrary subsets of W. As is well-known, for every $\mathsf{S4}_u$-formula φ, φ^* is satisfiable in a model from TR iff φ is satisfiable. Thus, we can consider the modal logic $\mathsf{S4}_u$ as the f-theory $(\mathcal{ST}, \mathsf{TR})$.

Now we can define a *temporal monodic fragment theory* (or *tmf-theory*) as a pair $(\mathcal{QTL}', \mathsf{TM})$, where $\mathcal{QTL}' \subseteq \mathcal{QTL}_{\boxdot}$ and TM is a class of temporal models (with either expanding or constant domains) in the signature of $\mathcal{QTL}'$.

Definition 2.5 (temporalisation). A tmf-theory $(\mathcal{QTL}', \mathsf{TM})$ is called the *expanding* (*constant*) *domain temporalisation* of an f-theory $(\mathcal{QL}', \mathsf{M})$ if

- $\mathcal{QTL}'$ is obtained from $\mathcal{QL}'$ by extending its formula formation rules with (†),
- TM consists of models of the form $\mathfrak{M} = \langle \mathbb{N}, <, I\rangle$ with expanding (respectively, constant) domains such that $I(n) \in \mathsf{M}$ for all $n \in \mathbb{N}$.

For instance, the constant domain temporalisation of $(\mathcal{ST}, \mathsf{TR})$ is the tmf-theory $(\mathcal{TST}, \mathsf{TTR})$ such that the language $\mathcal{TST}$ consists of formulas of the form

$$\varphi \ ::= \ P_i(x) \ \mid \ \neg\varphi \ \mid \ \varphi_1 \wedge \varphi_2 \ \mid \ \forall x\,\varphi(x) \ \mid \ \forall y\,(R(x,y) \to \varphi(y)) \ \mid \ \bigcirc\varphi \ \mid \ \varphi_1\,\mathcal{U}\,\varphi_2,$$

where the P_i are unary predicates and R is the only binary predicate (note that $\mathcal{TST}$-formulas contain at most one free variable), and TTR consists of models with constant domains of the form $\mathfrak{M} = \langle \mathbb{N}, <, I\rangle$, where $I(n) \in \mathsf{TR}$ for every $n \in \mathbb{N}$.

2.3. Surrogates (detemporalisation)

Our approach to devising tableau decision procedures for decidable monodic fragments is based on a simple principle: we want to separate the temporal and the first-order parts of formulas and treat them using available procedures for propositional temporal logic and the corresponding first-order fragment. With this in mind, we introduce the following notion of 'surrogates' for temporal formulas.

DEFINITION 2.6 (surrogates). With every formula $\vartheta(x)$ of the form $\varphi \mathcal{U} \psi$ or $\bigcirc\varphi$ having x as its only free variable we associate a fresh unary predicate $Q_\vartheta(x)$. Similarly, with every sentence ϑ of the form $\varphi \mathcal{U} \psi$ or $\bigcirc\varphi$ we associate a fresh propositional variable q_ϑ (i.e., a predicate of arity 0). $Q_\vartheta(x)$ and q_ϑ are called the *surrogates* of $\vartheta(x)$ and ϑ, respectively. Given a $\mathcal{QTL}_{\boxed{1}}$-formula ϑ, denote by $\overline{\vartheta}$ the formula obtained by replacing all its subformulas of the form $\varphi \mathcal{U} \psi$ and $\bigcirc\varphi$ that are not in the scope of another temporal operator with their surrogates. $\overline{\vartheta}$ is called the *first-order reduct* of ϑ.

The first-order reduct $\overline{\vartheta}$ of a $\mathcal{QTL}_{\boxed{1}}$-formula ϑ does not contain temporal operators at all—they are replaced with their surrogates. To ensure the proper 'temporal behavior' of these surrogates, we use the following formulas.

For all formulas of the form $\varphi \mathcal{U} \psi$ and $\bigcirc\varphi$ with one free variable x, let

$$
\begin{aligned}
(\varphi \mathcal{U} \psi)^+ &= \forall x \left(Q_{\varphi\mathcal{U}\psi}(x) \to \overline{\psi} \vee (\overline{\varphi} \wedge Q_{\bigcirc(\varphi\mathcal{U}\psi)}(x)) \right), \\
(\varphi \mathcal{U} \psi)^- &= \forall x \left(\neg Q_{\varphi\mathcal{U}\psi}(x) \to \neg\overline{\psi} \wedge (\neg\overline{\varphi} \vee Q_{\bigcirc\neg(\varphi\mathcal{U}\psi)}(x)) \right), \\
(\varphi \mathcal{U} \psi)^\neg &= \forall x \left(\neg Q_{\bigcirc(\varphi\mathcal{U}\psi)}(x) \to Q_{\bigcirc\neg(\varphi\mathcal{U}\psi)}(x) \right), \\
(\bigcirc\varphi)^\neg &= \forall x \left(\neg Q_{\bigcirc\varphi}(x) \to Q_{\bigcirc\neg\varphi}(x) \right).
\end{aligned}
$$

Similarly, for all sentences of the form $\varphi \mathcal{U} \psi$ and $\bigcirc\varphi$, let

$$
\begin{aligned}
(\varphi \mathcal{U} \psi)^+ &= q_{\varphi\mathcal{U}\psi} \to \overline{\psi} \vee (\overline{\varphi} \wedge q_{\bigcirc(\varphi\mathcal{U}\psi)}), \\
(\varphi \mathcal{U} \psi)^- &= \neg q_{\varphi\mathcal{U}\psi} \to \neg\overline{\psi} \wedge (\neg\overline{\varphi} \vee q_{\bigcirc\neg(\varphi\mathcal{U}\psi)}), \\
(\varphi \mathcal{U} \psi)^\neg &= \neg q_{\bigcirc(\varphi\mathcal{U}\psi)} \to q_{\bigcirc\neg(\varphi\mathcal{U}\psi)}, \\
(\bigcirc\varphi)^\neg &= \neg q_{\bigcirc\varphi} \to q_{\bigcirc\neg\varphi}.
\end{aligned}
$$

DEFINITION 2.7 (surrogate axioms). The set $\mathbf{Ax}_\vartheta(x)$ of *surrogate axioms* for a $\mathcal{QTL}_{\boxed{1}}$-formula ϑ consists of $\top(x)$, where x is a fresh variable, and all the formulas $(\varphi \mathcal{U} \psi)^+$, $(\varphi \mathcal{U} \psi)^-$, $(\varphi \mathcal{U} \psi)^\neg$, and $(\bigcirc\varphi)^\neg$ for $\varphi \mathcal{U} \psi \in sub(\vartheta)$ and $\bigcirc\varphi \in sub(\vartheta)$.

The formula $\top(x)$ will be important for dealing with constant domains as is explained later on.

EXAMPLE 2.8. Consider, for instance, the sentence

$$\Box\Big(\exists y\,\big(C(y)\land\neg\bigcirc C(y)\big)\land\forall y\,\big(\neg C(y)\to\bigcirc\neg C(y)\big)\Big)$$

which is equivalent to

$$\vartheta=\neg\Big(\top\,\mathcal{U}\,\neg\Big(\underbrace{\exists y\,\big(C(y)\land\neg\bigcirc C(y)\big)\land\forall y\,\big(\neg C(y)\to\bigcirc\neg C(y)\big)}_{\psi}\Big)\Big)$$

Then $\overline{\vartheta}=\neg q_{\top\mathcal{U}\neg\psi}$ and $\mathbf{Ax}_{\vartheta}(x)$ consists of the formulas

$$\begin{gathered}\top(x),\\ q_{\top\mathcal{U}\neg\psi}\to\neg\Big(\exists y\,(C(y)\land\neg Q_{\bigcirc C}(y))\land\forall y\,(\neg C(y)\to Q_{\bigcirc\neg C}(y))\Big)\lor q_{\bigcirc(\top\mathcal{U}\neg\psi)},\\ \neg q_{\top\mathcal{U}\neg\psi}\to\Big(\exists y\,(C(y)\land\neg Q_{\bigcirc C}(y))\land\forall y\,(\neg C(y)\to\neg Q_{\bigcirc\neg C}(y))\Big)\land q_{\bigcirc\neg(\top\mathcal{U}\neg\psi)},\\ \neg q_{\bigcirc(\top\mathcal{U}\neg\psi)}\to q_{\bigcirc\neg(\top\mathcal{U}\neg\psi)},\\ \forall y\,\big(\neg Q_{\bigcirc C}(y)\to Q_{\bigcirc\neg C}(y)\big),\\ \forall y\,\big(\neg Q_{\bigcirc\neg C}(y)\to Q_{\bigcirc C}(y)\big).\end{gathered}$$

DEFINITION 2.9 (reduct). Let $\mathcal{QTL}'$ be a fragment of $\mathcal{QTL}_{\boxdot}$ and TM a class of first-order temporal models (in the signature of $\mathcal{QTL}'$). Say that an f-theory $(\mathcal{QL}',\mathsf{M})$ is a *reduct* of the tmf-theory $(\mathcal{QTL}',\mathsf{TM})$ if the following holds:

- $\mathcal{QL}'$ is the smallest sublanguage of $\mathcal{QL}$ containing all first-order reducts of $\mathcal{QTL}'$-formulas, closed under the Boolean connectives, and such that $\forall x\,\varphi(x)\in\mathcal{QL}'$ whenever $\varphi(x)\in\mathcal{QL}'$;
- M consists of models of the form

$$\Big\langle\Delta,P_0^{\mathfrak{M}},\dots,Q_0^{\mathfrak{M}},\dots,q_0^{\mathfrak{M}},\dots\Big\rangle,$$

  where $\langle\Delta,P_0^{\mathfrak{M}},\dots\rangle=I(n)$ for some $\langle\mathbb{N},<,I\rangle\in\mathsf{TM}$, $n\in\mathbb{N}$, and $Q_0,Q_1,\dots$ are the predicates used as surrogates for temporal formulas from $\mathcal{QTL}'$ and $q_0,q_1,\dots$ are the propositional variables surrogating temporal sentences from $\mathcal{QTL}'$.

It should be noted that the surrogate symbols Q_j and q_j can be interpreted in an arbitrary way. We illustrate this definition with a number of examples.

1. The two-variable fragment $(\mathcal{QL}^2, \mathsf{FO})$ of first-order logic can be regarded as a reduct of $(\mathcal{QTL}^2_{\boxdot}, \mathsf{TFO})$, where TFO is the class of all temporal first-order models with constant domains. Note, that in this case, as well as in all following cases, without loss of generality we can assume that the language $\mathcal{QL}^2$ already contains infinitely many unary predicates and propositional variables for surrogates of all $\mathcal{QTL}^2_{\boxdot}$-formulas.

2. The monadic fragment $(\mathcal{QL}^{mo}, \mathsf{FO})$ of first-order logic is a reduct of $(\mathcal{QTL}^{mo}_{\boxdot}, \mathsf{TFO})$.

3. The one-variable fragment $(\mathcal{QL}^1, \mathsf{FO})$ of predicate logic can be considered as a reduct of the temporalised modal logic $\mathsf{S5}$, i.e., of the one-variable fragment of $\mathcal{QTL}$.

4. The propositional modal logic $\mathsf{S4}_u$ (i.e., the f-theory $(\mathcal{ST}, \mathsf{TR})$) can be viewed as a reduct of its own temporalisation $(\mathcal{TST}, \mathsf{TTR})$.

5. Strictly speaking, the first-order guarded fragment $\mathcal{GF}$ *is not* a reduct of the monodic guarded fragment $\mathcal{TGF}_{\boxdot}$, because $\mathcal{GF}$ does not contain arbitrary formulas of the form $\forall x\, \varphi(x)$. To get round this problem we can introduce a 'dummy' guard $\top(x)$ and use $\forall x\, (\top(x) \to \varphi(x))$ instead of $\forall x\, \varphi(x)$. Thus, the extended language can be regarded as a reduct of $\mathcal{TGF}_{\boxdot}$.

3. Quasimodels

In this section, we introduce the two core notions underlying our framework for tableau calculi presented in Section 4. First, we develop a general condition that first-order decision procedures must satisfy to be a useful building block in tableau calculi for fragments of monodic FOTL. Second, we define an abstraction of temporal models called 'quasimodels.' To keep tableaux finite and guarantee termination, the tableau procedure to be devised tries to construct a quasimodel for the input formula, rather than a temporal model itself.

3.1. First-order decision procedure

We require first-order decision procedures not only to return 'true' or 'false,' but rather to compute finite representations of all possible models for the input formula. The reason for this is as follows: first, we need an *explicit* representation of models to make a step in time, i.e., to take all temporal formulas realised in a model and then dropping a single occurrence of the next operator from them. Second, we need representations of *all* models since some of them may be appropriate for participating in a temporal model and others may not—which we will usually find out much later in the construction of the temporal tableau. The requirement of returning finite representations of models is much less exotic than it seems on first sight: indeed, most decision procedures for fragments of first-order logic satisfy it or can be easily modified to do so. This includes, for example, most tableau- and resolution-based algorithms (see Section 5).

The finite representation of models is type-based. Hence, fix some f-theory $(\mathcal{QL}', \mathsf{M})$ and a $\mathcal{QL}'$-formula η. Let x be a variable not occurring in η. Then we put

$$sub_x(\eta) = \{\varphi\{x/y\}, \neg\varphi\{x/y\} \mid \varphi(y) \in sub(\eta)\}$$

and call a non-empty subset of $sub_x(\eta)$ a *type for* a $\mathcal{QL}'$-formula η (usually denoted by $\mathbf{t}(x)$).

Important kinds of types for η are given by models $\mathfrak{M} \in \mathsf{M}$ and their elements a:

$$\mathbf{t}_a^{\mathfrak{M}}(x) = \{\varphi(x) \mid \varphi(x) \in sub_x(\eta) \text{ and } \mathfrak{M} \models \varphi[a]\}.$$

In what follows we will identify a type $\mathbf{t}(x)$ with the conjunction of all formulas in it and write $\mathfrak{M} \models \mathbf{t}[a]$ instead of '$\mathfrak{M} \models \varphi[a]$ for all $\varphi(x) \in \mathbf{t}(x)$.'

Definition 3.1 (flock). By a *flock* for a $\mathcal{QL}'$-formula η we mean any non-empty set $\mathbf{T}$ of types for η. Such a flock is called *saturated* if

- for every $\mathbf{t}(x) \in \mathbf{T}$,
  - if $\varphi \wedge \psi \in \mathbf{t}(x)$ then $\varphi \in \mathbf{t}(x)$ and $\psi \in \mathbf{t}(x)$;
  - if $\neg(\varphi \wedge \psi) \in \mathbf{t}(x)$ then $\neg\varphi \in \mathbf{t}(x)$ or $\neg\psi \in \mathbf{t}(x)$;
  - if $\forall z\, \varphi(z) \in \mathbf{t}(x)$ then $\varphi(x) \in \mathbf{t}(x)$;
- all types in $\mathbf{T}$ contain precisely the same sentences.

Because of the latter item, we may write $\varphi \in \mathbf{T}$ to say that a sentence φ belongs to some (every) type in the saturated flock $\mathbf{T}$.

Let $\mathbf{T}$ be a flock for η and $\mathfrak{M} \in \mathsf{M}$ a model with domain Δ. A $\mathbf{T}$-*assignment in* Δ is a map $\mathfrak{a} : \mathbf{T} \to \Delta$. We write $\mathfrak{M} \models \mathbf{T}[\mathfrak{a}]$ if $\mathfrak{M} \models \mathbf{t}[\mathfrak{a}(\mathbf{t})]$, for all $\mathbf{t}(x) \in \mathbf{T}$. A flock $\mathbf{T}$ is called *satisfiable* in M if there are $\mathfrak{M} \in \mathsf{M}$ with domain Δ and a $\mathbf{T}$-assignment $\mathfrak{a}$ in Δ such that $\mathfrak{M} \models \mathbf{T}[\mathfrak{a}]$.

We are now ready to give a formal account of decision procedures for fragments of first-order logic that can be used for constructing temporal tableau algorithms. We call such decision procedures *saturation rules* since, in the temporal tableau algorithm, they play the role of a tableau rule that 'saturates' a set of first-order types associated with a single time point: they take a flock and return a set of *saturated* flocks, each describing a class of models from M.

DEFINITION 3.2 (saturation rule). A *saturation rule* for $(\mathcal{QL}', \mathsf{M})$ is a computable function $\mathcal{A}$ which takes a flock $\mathbf{T}'$ for a $\mathcal{QL}'$-formula η and returns either '**clash**' if $\mathbf{T}'$ is not satisfiable in M, or a (finite) set $\mathcal{A}(\mathbf{T}')$ of saturated flocks for η such that the following holds:

(TR) for every $\mathbf{t}'(x) \in \mathbf{T}'$, each flock $\mathbf{T} \in \mathcal{A}(\mathbf{T}')$ contains a type $\mathbf{t}(x) \supseteq \mathbf{t}'(x)$, in which case we write $\mathbf{t}'(x) \to_{\mathcal{A}} \mathbf{t}(x)$;

(CO) for every $\mathfrak{M} \in \mathsf{M}$ with domain Δ, every type $\mathbf{t}'(x) \in \mathbf{T}'$ and every $\mathbf{T}'$-assignment $\mathfrak{a}'$ in Δ, if $\mathfrak{M} \models \mathbf{T}'[\mathfrak{a}']$ then there is a flock $\mathbf{T} \in \mathcal{A}(\mathbf{T}')$ such that

- there exist a type $\mathbf{t}(x) \in \mathbf{T}$ and a $\mathbf{T}$-assignment $\mathfrak{a}$ in Δ for which $\mathfrak{M} \models \mathbf{T}[\mathfrak{a}]$, $\mathbf{t}'(x) \to_{\mathcal{A}} \mathbf{t}(x)$ and $\mathfrak{a}'(\mathbf{t}') = \mathfrak{a}(\mathbf{t})$;

(SO) there is a cardinal $\kappa \geq \aleph_0$ such that for every $\kappa' \geq \kappa$ and every $\mathbf{T} \in \mathcal{A}(\mathbf{T}')$, there exists a model

$$\mathfrak{M} = \left\langle \Delta, P_0^{\mathfrak{M}}, \ldots, Q_0^{\mathfrak{M}}, \ldots, q_0^{\mathfrak{M}}, \ldots \right\rangle \in \mathsf{M}$$

in which

- $\Delta = \bigcup_{\mathbf{t} \in \mathbf{T}} \Delta^{\mathbf{t}}$, where $\Delta^{\mathbf{t}}$ are pairwise disjoint sets of cardinality κ',
- the q_i are all of the propositional variables, and $q_i^{\mathfrak{M}}$ is true iff $q_i \in \mathbf{T}$,
- the Q_i are all of the unary predicates, and $a \in Q_i^{\mathfrak{M}}$ iff there is a type $\mathbf{t}(x) \in \mathbf{T}$ such that $Q_i(x) \in \mathbf{t}(x)$ and $a \in \Delta^{\mathbf{t}}$,

such that $\mathfrak{M} \models \mathbf{t}[a]$ holds for all $\mathbf{t}(x) \in \mathbf{T}$ and all $a \in \Delta^{\mathbf{t}}$.

Intuitively, (SO) corresponds to the *soundness* of the decision procedure and (CO) to its *completeness*. In more details this connection will be illustrated in Section 5, where we show that standard tableau algorithms for fragments of first-order logic can be viewed as saturation rules.

3.2. Quasimodels: expanding domains

We abstract temporal models to the more manageable quasimodels. Both in the definition of quasimodels and in the tableau procedure to be devised, we consider sentences rather than formulas which obviously does not sacrifice generality. Let $(\mathcal{QTL}', \mathsf{TM})$ be a tmf-theory and $(\mathcal{QL}', \mathsf{M})$ its first-order reduct. Fix a $\mathcal{QTL}'$-*sentence* ϑ.

By a *type* for the *temporal* sentence ϑ we mean any subset of

$$\{\overline{\varphi}(x) \mid \varphi(x) \in sub_x(\vartheta)\} \cup sub(\mathbf{Ax}_\vartheta(x)),$$

where x is a variable not occurring in ϑ. It should be noted that every type $\mathbf{t}(x)$ for ϑ can be considered as a type for a first-order formula. In particular, $\mathbf{Ax}_\vartheta(x)$ is a type for

$$\top(x) \wedge \bigwedge_{\varphi\,\mathcal{U}\,\psi \in sub(\vartheta)} \Big((\varphi\,\mathcal{U}\,\psi)^{+} \wedge (\varphi\,\mathcal{U}\,\psi)^{-} \wedge (\varphi\,\mathcal{U}\,\psi)^{\neg}\Big) \wedge \bigwedge_{\bigcirc\varphi \in sub(\vartheta)} (\bigcirc\varphi)^{\neg}.$$

Then a *flock* for ϑ is a non-empty set of types for ϑ (which again can be treated as a flock for a first-order formula).

DEFINITION 3.3 (quasimodel). Let $\mathcal{A}$ be a saturation rule for $(\mathcal{QL}', \mathsf{M})$. An $\mathcal{A}$-*quasistate* for ϑ is a flock $\mathbf{T} \in \mathcal{A}(\mathbf{T}')$, where $\mathbf{T}'$ is a flock for ϑ containing $\mathbf{Ax}_\vartheta(x)$. Let $Q = (\mathbf{T}_n \mid n \in \mathbb{N})$ be a sequence of $\mathcal{A}$-quasistates for ϑ. A *run* in Q is a function r with domain $dom(r) = \{n \in \mathbb{N} \mid n \geq n_0\}$, for some $n_0 \in \mathbb{N}$, which for every $n \in dom(r)$ returns a type $\mathbf{t}_n(x) \in \mathbf{T}_n$ for ϑ such that the following two conditions hold:

- for every $\mathcal{QTL}'$-formula $\bigcirc\varphi(x)$, if $\overline{\bigcirc\varphi}(x) \in r(n)$ then $\overline{\varphi}(x) \in r(n+1)$;
- for every $\mathcal{QTL}'$-formula $(\varphi\,\mathcal{U}\,\psi)(x)$, if $\overline{(\varphi\,\mathcal{U}\,\psi)}(x) \in r(n)$ then there is $k \geq n$ such that $\overline{\psi}(x) \in r(k)$ and $\overline{\varphi}(x) \in r(i)$ for every $i \in [n, k)$.

The sequence Q of $\mathcal{A}$-quasistates is called an $\mathcal{A}$-*quasimodel* if for every $n \in \mathbb{N}$ and every type $\mathbf{t}_n(x) \in \mathbf{T}_n$, there is a run r in Q such that $r(n) = \mathbf{t}_n(x)$. We say that ϑ is $\mathcal{A}$-*satisfiable* if there are an $\mathcal{A}$-quasimodel $Q = (\mathbf{T}_n \mid n \in \mathbb{N})$ and some $n \in \mathbb{N}$ such that $\overline{\vartheta} \in \mathbf{T}_n$.

Quasimodels are defined such that every $\mathcal{QTL}'$-*sentence* ϑ has a model iff it has a quasimodel. However, for the correctness proof of our tableau calculus, we will only make use of the "if" direction of this claim.

THEOREM 3.4. *Let* $(\mathcal{QTL}', \mathsf{TM})$ *be a tmf-theory, where* TM *is a class of models with expanding domains. Let* $(\mathcal{QL}', \mathsf{M})$ *be a first-order reduct of* $(\mathcal{QTL}', \mathsf{TM})$ *and* $\mathcal{A}$ *a saturation rule for* $(\mathcal{QL}', \mathsf{M})$. *If a* $\mathcal{QTL}'$*-sentence* ϑ *is* $\mathcal{A}$*-satisfiable, then it is satisfiable in a model from* TM.

PROOF. Take an $\mathcal{A}$-quasimodel $Q = (\mathbf{T}_n \mid n \in \mathbb{N})$ satisfying ϑ. Denote by Ω the set of all runs in Q and take a cardinal κ' exceeding the cardinality of the set Ω and the cardinal κ supplied by (SO). For each $n \in \mathbb{N}$, we set

$$\Delta_n = \{\langle r, \xi\rangle \mid r \in \Omega,\, n \in dom(r),\, \xi < \kappa'\}.$$

By the definition of quasimodel, we have $\Delta_n \subseteq \Delta_m$ if $n \leq m$. By (SO), for every $\mathbf{T}_n$, $n \in \mathbb{N}$, we can find a model

$$\mathfrak{M}_n = \left\langle \Delta_n, P_0^{\mathfrak{M}_n}, \dots, Q_0^{\mathfrak{M}_n}, \dots, q_0^{\mathfrak{M}_n}, \dots \right\rangle \in \mathsf{M}$$

where

- $\Delta_n = \bigcup_{\mathbf{t}(x) \in \mathbf{T}_n} \Delta_n^{\mathbf{t}}$, with $\Delta_n^{\mathbf{t}}$ being pairwise disjoint sets of cardinality κ',
- $q_i^{\mathfrak{M}_n}$ is true iff $q_i \in \mathbf{T}_n$, and
- $a \in Q_i^{\mathfrak{M}_n}$ iff there is a type $\mathbf{t}(x) \in \mathbf{T}_n$ with $Q_i(x) \in \mathbf{t}(x)$ and $a \in \Delta_n^{\mathbf{t}}$,

such that $\mathfrak{M}_n \models \mathbf{t}[a]$ for all types $\mathbf{t}(x) \in \mathbf{T}_n$ and all $a \in \Delta_n^{\mathbf{t}}$. This means, in particular, that for all sentences φ we have $\mathfrak{M}_n \models \varphi$ whenever $\varphi \in \mathbf{T}_n$. Without loss of generality we can assume that

$$\Delta_n^{\mathbf{t}} = \{\langle r, \xi\rangle \in \Delta_n \mid r(n) = \mathbf{t}(x)\}. \qquad (\ddagger)$$

Let $\mathfrak{M} = \langle \mathbb{N}, <, I\rangle$, where $I(n) = \left\langle \Delta_n, P_0^{\mathfrak{M}_n}, \dots \right\rangle$, for all $n \in \mathbb{N}$.

CLAIM. *For every* $n \in \mathbb{N}$, *every assignment* $\mathfrak{a}$ *in* Δ_n *and every formula* $\chi \in \{\varphi, \neg\varphi \mid \varphi \in sub(\vartheta)\}$, *if* $\mathfrak{a}(x) \in \Delta_n$ *for all* $x \in free(\chi)$, *then*

$$\mathfrak{M}_n \models^{\mathfrak{a}} \overline{\chi} \quad \textit{implies} \quad (\mathfrak{M}, n) \models^{\mathfrak{a}} \chi.$$

Suppose for a moment that the claim holds. Since Q is an $\mathcal{A}$-quasimodel of ϑ, there exists an $n \in \mathbb{N}$ such that $\overline{\vartheta} \in \mathbf{T}_n$. By the choice of $\mathfrak{M}_n$, we have $\mathfrak{M}_n \models \overline{\vartheta}$, whence $(\mathfrak{M}, n) \models \vartheta$, which proves our theorem.

PROOF OF CLAIM. The proof is by induction on the construction of χ.

Case $\chi = P_i(x_1, \dots, x_m)$ follows from the definition of $\mathfrak{M}$.

Case $\chi = \neg P_i(x_1, \dots, x_m)$. This means that $\langle \mathfrak{a}(x_1), \dots, \mathfrak{a}(x_m)\rangle \notin P_i^{\mathfrak{M}_n}$. As $\mathfrak{a}(x_j) \in \Delta_n$ for $j \in [1, m]$, we then obtain $(\mathfrak{M}, n) \models^{\mathfrak{a}} \neg P_i(x_1, \dots, x_m)$.

Cases $\chi = \neg\neg\varphi$, $\chi = \varphi \wedge \psi$, $\chi = \neg(\varphi \wedge \psi)$ follow from the obvious equivalences

$$\overline{\neg\neg\varphi} = \neg\neg\overline{\varphi} \qquad \overline{\varphi \wedge \psi} = \overline{\varphi} \wedge \overline{\psi}, \qquad \overline{\neg(\varphi \wedge \psi)} = \neg(\overline{\varphi} \wedge \overline{\psi}).$$

Case $\chi = \forall x\, \varphi$. We need to show that, for all assignments $\mathfrak{b}$ that may differ from $\mathfrak{a}$ only on x and such that $\mathfrak{b}(x) \in \Delta_n$, we have $(\mathfrak{M}, n) \models^{\mathfrak{b}} \varphi$. Fix such an assignment $\mathfrak{b}$. Since $\mathfrak{M}_n \models^{\mathfrak{a}} \overline{\forall x\, \varphi}$ and $\overline{\forall x\, \varphi} = \forall x\, \overline{\varphi}$, we have $\mathfrak{M}_n \models^{\mathfrak{b}} \overline{\varphi}$. By IH, $(\mathfrak{M}, n) \models^{\mathfrak{b}} \varphi$, as required.

Case $\chi = \neg\forall x\, \varphi$. Since $\overline{\neg\forall x\, \varphi} = \neg\forall x\, \overline{\varphi}$, we have $\mathfrak{M}_n \models^{\mathfrak{a}} \neg\forall x\, \overline{\varphi}$. Then there exists an assignment $\mathfrak{b}$ that may differ from $\mathfrak{a}$ only on x and such that $\mathfrak{M}_n \models^{\mathfrak{b}} \neg\overline{\varphi}$. Since the domain of $\mathfrak{M}_n$ is Δ_n, we have $\mathfrak{b}(x) \in \Delta_n$. By IH, $(\mathfrak{M}, n) \models^{\mathfrak{b}} \neg\varphi$ and so $(\mathfrak{M}, n) \models^{\mathfrak{a}} \neg\forall x\, \varphi$.

Case $\chi = \bigcirc\varphi$. Let $\mathfrak{a}(x) = \langle r, \xi\rangle$. By the choice of $\mathfrak{M}_n$, there exists a type $\mathbf{t}(x)$ such that $\overline{\bigcirc\varphi}(x) \in \mathbf{t}(x)$ and $\langle r, \xi\rangle \in \Delta_n^{\mathbf{t}}$. By (‡), we have $r(n) = \mathbf{t}(x)$ and $\overline{\bigcirc\varphi}(x) \in r(n)$. Then by the definition of runs, $\overline{\varphi}(x) \in r(n+1)$. Let $r(n+1) = \mathbf{t}'(x)$. By using again (‡), we obtain $\langle r, \xi\rangle \in \Delta_{n+1}^{\mathbf{t}'}$, and therefore $\mathfrak{M}_{n+1} \models \overline{\varphi}[\langle r, \xi\rangle]$. By IH, $(\mathfrak{M}, n+1) \models \varphi[\langle r, \xi\rangle]$. Then $(\mathfrak{M}, n) \models \bigcirc\varphi[\langle r, \xi\rangle]$, which means that $(\mathfrak{M}, n) \models^{\mathfrak{a}} \bigcirc\varphi$.

Case $\chi = \neg\bigcirc\varphi$. Since $\mathbf{T}_n$ is an $\mathcal{A}$-quasistate and $\mathcal{A}$ satisfies (TR), we have $\forall x\, (\neg Q_{\bigcirc\varphi}(x) \to Q_{\bigcirc\neg\varphi}(x)) \in \mathbf{T}_n$. Therefore, $\mathfrak{M}_n \models^{\mathfrak{a}} \neg\overline{\bigcirc\varphi}$ implies $\mathfrak{M}_n \models^{\mathfrak{a}} \overline{\bigcirc\neg\varphi}$. As in the previous case, we obtain $(\mathfrak{M}, n) \models^{\mathfrak{a}} \bigcirc\neg\varphi$, and so $(\mathfrak{M}, n) \models^{\mathfrak{a}} \neg\bigcirc\varphi$.

Case $\chi = \varphi\, \mathcal{U}\, \psi$. The proof is similar to the case $\chi = \bigcirc\varphi$: we use the definition of runs and (‡).

Case $\chi = \neg(\varphi\, \mathcal{U}\, \psi)$. As $\mathbf{T}_j$ is an $\mathcal{A}$-quasistate,

$$\forall x\, \big(\neg Q_{\varphi\mathcal{U}\psi}(x) \to \neg\overline{\psi}(x) \wedge (\neg\overline{\varphi}(x) \vee Q_{\bigcirc\neg(\varphi\mathcal{U}\psi)}(x))\big) \in \mathbf{T}_j. \tag{i}$$

Suppose $\mathfrak{M}_n \models^{\mathfrak{a}} \neg\overline{(\varphi\, \mathcal{U}\, \psi)}(x)$. First we show that

(A) for all $k \geq n$, either $\mathfrak{M}_k \models^{\mathfrak{a}} \neg\overline{\psi}(x) \wedge \neg Q_{\varphi\mathcal{U}\psi}(x)$ or there is $i \in [n, k)$ such that $\mathfrak{M}_i \models^{\mathfrak{a}} \neg\overline{\varphi}(x)$.

The proof is by induction on k. The basis of induction, i.e., $k = n$ follows from (i).

Assume now that the claim has already been proved for $k = m$. If there is some $i \in [n, m)$ such that $\mathfrak{M}_i \models^{\mathfrak{a}} \neg\overline{\varphi}(x)$, then we are clearly done. So suppose that there is no such an i. Then, by IH, we have $\mathfrak{M}_m \models^{\mathfrak{a}} \neg\overline{\psi}(x)$ and $\mathfrak{M}_m \models^{\mathfrak{a}} \neg Q_{\varphi\mathcal{U}\psi}(x)$. (i) gives us either $\mathfrak{M}_m \models^{\mathfrak{a}} \neg\overline{\varphi}(x)$ or $\mathfrak{M}_m \models^{\mathfrak{a}} Q_{\bigcirc\neg(\varphi\mathcal{U}\psi)}(x)$. In the former case we are done. Consider the latter. As in the case $\chi = \bigcirc\varphi$, we then have $\mathfrak{M}_{m+1} \models^{\mathfrak{a}} \neg Q_{\varphi\mathcal{U}\psi}(x)$. Using (i), we obtain $\mathfrak{M}_{m+1} \models^{\mathfrak{a}} \neg\overline{\psi}$, as required. This completes the induction step, and hence the proof of (A).

By the induction hypothesis of the main proof and (A), we then have:

(B) for all $k \geq n$, either $(\mathfrak{M}, k) \models^{\mathfrak{a}} \neg\psi(x)$ or there is $i \in [n, k)$ such that $(\mathfrak{M}, i) \models^{\mathfrak{a}} \neg\varphi(x)$.

This means that $(\mathfrak{M}, n) \models^{\mathfrak{a}} \neg(\varphi\,\mathcal{U}\,\psi)$. ∎

3.3. Quasimodels: constant domains

To define quasimodels which give rise to first-order temporal models with constant domains, we should obviously require all runs to be total functions on $\mathbb{N}$. We also need the following refinement of the definition of saturation rules.

Let $(\mathcal{QTL}', \mathsf{TM})$ be a tmf-theory and $(\mathcal{QL}', \mathsf{M})$ its first-order reduct.

Definition 3.5 (exhaustive saturation rule). Say that a flock $\mathbf{T}$ for a $\mathcal{QL}'$-formula η is *exhaustive* for a model $\mathfrak{M} \in \mathsf{M}$ with domain Δ if for each $a \in \Delta$ there is a type $\mathbf{t}(x) \in \mathbf{T}$ such that $\mathfrak{M} \models \mathbf{t}[a]$. A saturation rule $\mathcal{A}$ for $(\mathcal{QL}', \mathsf{M})$ is called *exhaustive* if the following strengthening of (CO) holds:

(CO$'$) for every $\mathfrak{M} \in \mathsf{M}$ with domain Δ, every type $\mathbf{t}'(x) \in \mathbf{T}'$ and every $\mathbf{T}'$-assignment $\mathfrak{a}'$ in Δ, if $\mathfrak{M} \models \mathbf{T}'[\mathfrak{a}']$ then there is a flock $\mathbf{T} \in \mathcal{A}(\mathbf{T}')$ such that

- $\mathbf{T}$ is exhaustive for $\mathfrak{M}$;
- there exist a type $\mathbf{t}(x) \in \mathbf{T}$ and a $\mathbf{T}$-assignment $\mathfrak{a}$ in Δ for which $\mathfrak{M} \models \mathbf{T}[\mathfrak{a}]$, $\mathbf{t}'(x) \to_{\mathcal{A}} \mathbf{t}(x)$ and $\mathfrak{a}'(\mathbf{t}') = \mathfrak{a}(\mathbf{t})$.

As in the case of expanding domains, exhaustive saturation rules can be obtained from standard tableau algorithms by making some rather minor modifications. More details are provided in Section 5.3.

Let $\mathcal{A}$ be an exhaustive saturation rule for $(\mathcal{QL}', \mathsf{M})$ and fix some $\mathcal{QTL}'$-sentence ϑ. We now define exhaustive flocks for ϑ.

Definition 3.6 (constant domain quasimodel). An $\mathcal{A}$-*quasistate* for ϑ is an exhaustive flock $\mathbf{T}$ for ϑ such that $\mathbf{T} \in \mathcal{A}(\mathbf{T}')$ for some flock $\mathbf{T}'$ containing the type $\mathbf{Ax}_\vartheta(x)$. A sequence $Q = (\mathbf{T}_n \mid n \in \mathbb{N})$ of $\mathcal{A}$-quasistates for ϑ is called a *constant domain $\mathcal{A}$-quasimodel* if for every $n \in \mathbb{N}$ and every type $\mathbf{t}_n(x) \in \mathbf{T}_n$ for ϑ there is a *total* run r in Q (i.e., $dom(r) = \mathbb{N}$) such that $r(n) = \mathbf{t}_n(x)$.

We say that a sentence ϑ is *$\mathcal{A}$-satisfiable in constant domains* if there is a constant domain $\mathcal{A}$-quasimodel $Q = (\mathbf{T}_n \mid n \in \mathbb{N})$ such that $\overline{\vartheta} \in \mathbf{T}_n$ for some $n \in \mathbb{N}$.

Following the proof of Theorem 3.4, one can readily show the following:

THEOREM 3.7. *Let $(\mathcal{QTL}', \mathsf{TM})$ be a tmf-theory, with TM being a class of models with constant domains. Let $(\mathcal{QL}', \mathsf{M})$ be a first-order reduct of $(\mathcal{QTL}', \mathsf{TM})$ and $\mathcal{A}$ an exhaustive saturation rule for $(\mathcal{QL}', \mathsf{M})$. If a $\mathcal{QTL}'$-sentence ϑ is $\mathcal{A}$-satisfiable in constant domains, then it is satisfiable in a model from TM.*

4. Tableaux

We are in a position now to define temporal tableaux for decidable monodic fragments. Let us start with the expanding domain case. Fix a tmf-theory $(\mathcal{QTL}', \mathsf{TM})$, its first-order reduct $(\mathcal{QL}', \mathsf{M})$, a saturation rule $\mathcal{A}$ for $(\mathcal{QL}', \mathsf{M})$, and a $\mathcal{QTL}'$-sentence ϑ.

To decide the satisfiability of ϑ, the tableau algorithm tries to construct an $\mathcal{A}$-quasimodel for ϑ by applying the saturation rule $\mathcal{A}$ to the reduct of ϑ, then making a step in time, then again applying $\mathcal{A}$, and so on. Let us start its presentation with defining the basic data structure.

DEFINITION 4.1 (temporal tableau). A *temporal tableau* for ϑ is a labelled directed graph $\mathcal{G} = \langle S, s_r, \to, \ell, \ell_{\bigcirc} \rangle$, where S is a set of *states* containing the *root state* s_r, $\to$ is a binary relation on S, and ℓ, $\ell_{\bigcirc}$ are state labelling functions such that $\ell(s)$ is a saturated flock for each state $s \in S \setminus \{s_r\}$, and $\ell_{\bigcirc}(s)$ is a flock for ϑ for each state $s \in S$.

Intuitively, tableaux can be understood as follows: apart from the root state, each state s is associated with a time point n in the sense that the saturated flock $\ell(s)$ is a candidate for the quasistate $\mathbf{T}_n$ for time point n of the quasimodel to be constructed. Distinct states may be associated with the same time point n describing different possible choices for the quasistate $\mathbf{T}_n$. If a state s describes time point n, then any state s' with $s \to s'$ describes time point $n+1$. It remains to explain the second labelling $\ell_{\bigcirc}(s)$: it's purpose is to list those types that have to be included in the quasistate $\mathbf{T}_{n+1}$ due to temporal formulas appearing (in surrogated form) in $\ell(s)$. Let us formally define how $\ell_{\bigcirc}(s)$ can be obtained from $\ell(s)$.

DEFINITION 4.2 (transition rule). If $\mathbf{t}'(x) = \{\overline{\varphi}(x) \mid \overline{\bigcirc\varphi}(x) \in \mathbf{t}(x)\} \cup \{\top(x)\}$ for a type $\mathbf{t}(x)$ for ϑ, then we write

$$\mathbf{t}(x) \to_{\bigcirc} \mathbf{t}'(x).$$

The *transition rule* for $\mathcal{QTL}'$ is the map $\mathcal{N}$ that takes a flock $\mathbf{T}$ for ϑ and returns the flock

$$\mathcal{N}(\mathbf{T}) = \{\mathbf{t}'(x) \mid \mathbf{t}(x) \in \mathbf{T} \text{ and } \mathbf{t}(x) \to_{\bigcirc} \mathbf{t}'(x)\} \cup \{\mathbf{Ax}_\vartheta(x)\}.$$

Suppose now that the satisfiability of ϑ is to be decided. The algorithm starts with the initial temporal tableau

$$\mathcal{G}_\vartheta = \langle \{s_r\}, s_r, \emptyset, \ell, \ell_{\bigcirc} \rangle ,$$

where

$$\ell(s_r) = \emptyset \quad \text{and} \quad \ell_{\bigcirc}(s_r) = \{\{\overline{\vartheta}, \top(x)\}, \mathbf{Ax}_\vartheta(x)\}$$

for some variable x not to occur in ϑ. Note that the root state s_r is not associated with a point in time but only serves the purpose of getting started with the tableau construction. The flock $\ell_{\bigcirc}(r_s)$ consists of two types: one of them, $\{\overline{\vartheta}, \top(x)\}$, ensures that ϑ is satisfied in the first quasistate of a quasimodel to be constructed and the other one, $\mathbf{Ax}_\vartheta(x)$, contains the surrogate axioms. Then we apply the saturation rule $\mathcal{A}$ to $\ell_{\bigcirc}(s_r)$ and obtain new $\to$-successor states $s_{\mathbf{T}}$ of s_r, for every $\mathbf{T} \in \mathcal{A}(\ell_{\bigcirc}(s_r))$, labelled with

$$\ell(s_{\mathbf{T}}) = \mathbf{T} \quad \text{and} \quad \ell_{\bigcirc}(s_{\mathbf{T}}) = \mathcal{N}(\mathbf{T}).$$

We continue by applying $\mathcal{A}$ to the $\ell_{\bigcirc}(s_{\mathbf{T}})$, and so forth (see Section 5.4 for detailed examples). Here is a more precise definition.

DEFINITION 4.3 (tableau rule). Say that a tableau $\mathcal{G}'$ for ϑ *is obtained by an application of rule* $\Longrightarrow$ from a tableau $\mathcal{G} = \langle S, s_r, \to, \ell, \ell_{\bigcirc} \rangle$ for ϑ and write $\mathcal{G} \Longrightarrow \mathcal{G}'$ if there is a state $s_0 \in S$ such that $\ell_{\bigcirc}(s_0) = \mathbf{T}_0$, $\mathcal{A}(\mathbf{T}_0)$ is not a clash, and there is a saturated flock $\mathbf{T}_1 \in \mathcal{A}(\mathbf{T}_0)$ such that either

- there is no state $s_1 \in S$ for which $\ell(s_1) = \mathbf{T}_1$, and

$$\mathcal{G}' = \langle S \cup \{s_1\},\ s_r,\ \to \cup \{\langle s_0, s_1 \rangle\},\ \ell',\ \ell'_{\bigcirc} \rangle ,$$

$$\text{where} \quad \ell'(s) = \begin{cases} \ell(s) & \text{if } s \in S \\ \mathbf{T}_1 & \text{if } s = s_1 \end{cases} \quad \text{and} \quad \ell'_{\bigcirc}(s) = \begin{cases} \ell_{\bigcirc}(s) & \text{if } s \in S \\ \mathcal{N}(\mathbf{T}_1) & \text{if } s = s_1, \end{cases}$$

- or there is $s_1 \in S$ for which $\ell(s_1) = \mathbf{T}_1$, $s_0 \not\to s_1$, and

$$\mathcal{G}' = \langle S,\ s_r,\ \to \cup \{\langle s_0, s_1\rangle\},\ \ell,\ \ell_{\bigcirc}\rangle .$$

A tableau $\mathcal{G}$ for ϑ is called *complete* if rule $\Longrightarrow$ is not applicable to it.

The second item in Definition 4.3 is the so-called *blocking condition* which is crucial for ensuring termination.

THEOREM 4.4 (termination). *The process of completing a tableau $\mathcal{G}$ for ϑ terminates. In other words, there is no infinite sequence*

$$\mathcal{G} \Longrightarrow \mathcal{G}_1 \Longrightarrow \ldots \Longrightarrow \mathcal{G}_n \Longrightarrow \ldots$$

of tableaux for ϑ.

PROOF. Any complete tableau $\mathcal{G}$ for ϑ contains at most

$$2^{2^{p(|\vartheta|)}}$$

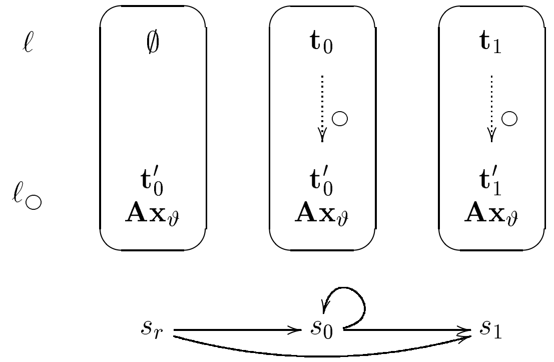
states, where p is a polynomial function of the length $|\vartheta|$ of ϑ. Indeed, the number of subformulas of

$$\{\overline{\varphi} \mid \varphi \in sub(\vartheta)\} \cup \mathbf{Ax}_{\vartheta}(x)$$

is linear in $|\vartheta|$, the number of different types for ϑ is bounded by $2^{p(|\vartheta|)}$, and the number of states in $\mathcal{G}$ does not exceed $2^{2^{p(|\vartheta|)}}$. ■

Suppose we have constructed a complete tableau $\mathcal{G}$ for ϑ. Due to the presence of the temporal until operator, such tableaux do *not* necessarily give rise to a quasimodel satisfying ϑ. As an example, we present a complete tableau for the obviously unsatisfiable sentence $\vartheta = \top\,\mathcal{U}\,\bot$ on Fig. 1. The state s_1 has no successors because the type $\mathbf{t}_1'(x)$ is not satisfiable, and the saturation rule returns **clash**. Nevertheless, the tableau contains the loop $s_0 \to s_0$, and so one could have tried to extract a quasimodel from this infinite path. It follows from Theorem 3.4 that the extracted sequence of flocks cannot be a quasimodel.

How can we identify tableaux that do not describe quasimodels? Here we come to the second component of Wolper's [20] tableau procedure. Having built up a complete tableau for ϑ, one has to eliminate those states that have no successors or contain so-called *unrealised eventualities*. Wolper proved (in the propositional case) that ϑ is satisfiable iff the root state of the complete tableau is not eliminated.



$$\mathbf{t}_0(x) = \{q_\vartheta, q_{\bigcirc\vartheta}\} \cup \mathbf{Ax}_\vartheta(x) \qquad \mathbf{t}'_0(x) = \{q_\vartheta, \top(x)\}$$

$$\mathbf{t}_1(x) = \{q_\vartheta, q_{\bigcirc\vartheta}, q_{\bigcirc\neg\vartheta}\} \cup \mathbf{Ax}_\vartheta(x) \qquad \mathbf{t}'_1(x) = \{q_\vartheta, \neg q_\vartheta, \top(x)\}$$

$$\mathbf{Ax}_\vartheta(x) = \{\top(x),\ q_\vartheta \to q_{\bigcirc\vartheta},\ \neg q_\vartheta \to q_{\bigcirc\neg\vartheta},\ \neg q_{\bigcirc\vartheta} \to q_{\bigcirc\neg\vartheta}\}$$

Figure 1. Complete tableau for $\bot\,\mathcal{U}\,\top$.

DEFINITION 4.5 (eventuality). Formulas of the form $\overline{\varphi\,\mathcal{U}\,\psi}(x)$ are called *eventualities*. Let $\mathcal{G} = \langle S, s_r, \to, \ell, \ell_\bigcirc\rangle$ be a tableau for ϑ. A sequence

$$s_0 \to \cdots \to s_n$$

of states in $\mathcal{G}$, where $n \geq 0$, is said to *realise* $\overline{\varphi\,\mathcal{U}\,\psi}(x) \in \mathbf{t}_0(x) \in \ell(s_0)$ if there exists a sequence

$$\mathbf{t}_0(x) \to_\bigcirc \mathbf{t}'_1(x) \to_\mathcal{A} \mathbf{t}_1(x) \to_\bigcirc \mathbf{t}'_2(x) \to_\mathcal{A} \ldots \to_\bigcirc \mathbf{t}'_n(x) \to_\mathcal{A} \mathbf{t}_n(x)$$

of types such that $\mathbf{t}_i(x) \in \ell(s_i)$, $\mathbf{t}'_i(x) \in \ell_\bigcirc(s_i)$ for i, $0 \leq i \leq n$, and $\overline{\psi}(x) \in \mathbf{t}_n(x)$.

DEFINITION 4.6 (elimination rules). We use the following rules to eliminate states in $\mathcal{G}$:

(E2) if a state $s \in S$ has no $\to$-successor, eliminate it;

(E3) if $\ell(s)$, $s \in S$, contains an eventuality having no realising sequence starting from s, eliminate s.

Elimination rules (E2) and (E3) are very similar to those in [20]. However, we do not need rule (E1) from [20], since the flock $\ell(s)$, for every state

s, contains no contradiction (it is the result of applying the saturation rule $\mathcal{A}$). In the complete tableau for $\vartheta = \top\,\mathcal{U}\,\bot$ from above, the state s_1 is eliminated since it has no successor. Then s_0 is eliminated since the eventuality q_ϑ has no realising sequence. Finally, we eliminate the root s_r due to (E2).

THEOREM 4.7. *Let* $(\mathcal{QTL}', \mathsf{TM})$ *be a tmf-theory,* $(\mathcal{QL}', \mathsf{M})$ *its first-order reduct,* $\mathcal{A}$ *a saturation rule for* $(\mathcal{QL}', \mathsf{M})$*, and* $\mathcal{N}$ *a transition rule for* $\mathcal{QTL}'$*. Then for every* $\mathcal{QTL}'$*-sentence* ϑ *the following conditions are equivalent:*

(1) ϑ *is satisfiable in a model from* TM*;*

(2) *the root of a complete tableau for* ϑ *cannot be eliminated using rules* (E2) *and* (E3)*.*

In the following two subsections we will prove the implications (1) ⇒ (2) (completeness) and (2) ⇒ (1) (soundness).

4.1. Completeness

We require a number of lemmas.

Suppose $\mathfrak{M} = \langle \mathbb{N}, <, I \rangle \in \mathsf{TM}$, where $I(n) = \left\langle \Delta_n, P_0^{I(n)}, \dots \right\rangle$. For every $n \in \mathbb{N}$, define a first-order model

$$\mathfrak{M}_n = \left\langle \Delta_n, P_0^{\mathfrak{M}_n}, \dots, Q_{\alpha_0}^{\mathfrak{M}_n}, \dots, q_{\beta_0}^{\mathfrak{M}_n}, \dots \right\rangle \in \mathsf{M},$$

where $\alpha_0, \alpha_1, \dots$ is an enumeration of all formulas of the form $\psi_1\,\mathcal{U}\,\psi_2$ and $\bigcirc\psi$ with one free variable x and $\beta_0, \beta_1, \dots$ is an enumeration of all sentence of the form $\psi_1\,\mathcal{U}\,\psi_2$ and $\bigcirc\psi$. Namely, we set $P_i^{\mathfrak{M}_n} = P_i^{I(n)}$ and define the $Q_\varphi^{\mathfrak{M}_n}$ and $q_\varphi^{\mathfrak{M}_n}$ as follows:

- If $Q_\varphi(x) \in sub(\mathbf{Ax}_\vartheta(x))$, then $a \in Q_\varphi^{\mathfrak{M}_n}$ iff $(\mathfrak{M}, n) \models \varphi[a]$, for every $a \in \Delta_n$; otherwise put, say, $Q_\varphi^{\mathfrak{M}_n} = \emptyset$.
- If $q_\varphi \in sub(\mathbf{Ax}_\vartheta(x))$, then $q_\varphi^{\mathfrak{M}_n}$ is true iff $(\mathfrak{M}, n) \models \varphi$; otherwise let, say, $q_\varphi^{\mathfrak{M}_n}$ be false.

LEMMA 4.8. *For every subformula* $\varphi(y)$ *of* ϑ*, every* $n \in \mathbb{N}$*, and every* $a \in \Delta_n$*,*

$$(\mathfrak{M}, n) \models \varphi[a] \qquad \textit{iff} \qquad \mathfrak{M}_n \models \overline{\varphi}[a];$$

besides, $\mathfrak{M}_n \models \mathbf{Ax}_\vartheta(x)$*.*

PROOF. The former claim follows immediately from the definition of $\mathfrak{M}_n$. As to the latter, we show only that

$$\mathfrak{M}_n \models q_{\varphi\mathcal{U}\psi} \to \overline{\psi} \vee (\overline{\varphi} \wedge q_{\bigcirc(\varphi\mathcal{U}\psi)}).$$

Suppose otherwise. Then $\mathfrak{M}_n \models q_{\varphi\mathcal{U}\psi}$, $\mathfrak{M}_n \not\models \overline{\psi}$ and $\mathfrak{M}_n \not\models \overline{\varphi} \wedge q_{\bigcirc(\varphi\mathcal{U}\psi)}$. It follows that $(\mathfrak{M}, n) \models \varphi\,\mathcal{U}\,\psi$ and $(\mathfrak{M}, n) \not\models \psi$. Moreover, we must also have either $(\mathfrak{M}, n) \not\models \varphi$ or $(\mathfrak{M}, n) \not\models \bigcirc(\varphi\,\mathcal{U}\,\psi)$, i.e., $(\mathfrak{M}, n+1) \not\models \varphi\,\mathcal{U}\,\psi$, contrary to the truth-definition of $\mathcal{U}$. ■

Suppose now that $\mathcal{G} = \langle S, s_r, \to, \ell, \ell_\bigcirc \rangle$ is a complete tableau for ϑ and that ϑ is satisfiable in a model $\mathfrak{M} = \langle \mathbb{N}, <, I \rangle$ from the class TM, where $I(n) = \left\langle \Delta_n, P_0^{I(n)}, \dots \right\rangle$.

LEMMA 4.9. *Let $n \in \mathbb{N}$, $s' \in S$, $\ell_\bigcirc(s') = \mathbf{T}'$, let $\mathfrak{a}'$ be a $\mathbf{T}'$-assignment in Δ_n and $\mathbf{t}_0'(x) \in \mathbf{T}'$. If $\mathfrak{M}_n \models \mathbf{T}'[\mathfrak{a}']$ then there are a state $s \in S$ with $\ell(s) = \mathbf{T}$, a $\mathbf{T}$-assignment $\mathfrak{a}$ in Δ_n and a type $\mathbf{t}_0(x) \in \mathbf{T}$ such that*

$$s' \to s, \qquad \mathfrak{M}_n \models \mathbf{T}[\mathfrak{a}], \qquad \mathfrak{a}'(\mathbf{t}_0') = \mathfrak{a}(\mathbf{t}_0),$$

and $\mathbf{t}_0'(x) \to_{\mathcal{A}} \mathbf{t}_0(x)$.

PROOF. Suppose $\mathfrak{M}_n \models \mathbf{T}'[\mathfrak{a}']$. Then, by (CO), we can find $\mathbf{T} \in \mathcal{A}(\mathbf{T}')$, $\mathbf{t}_0(x) \in \mathbf{T}$ and a $\mathbf{T}$-assignment $\mathfrak{a}$ such that $\mathfrak{M}_n \models \mathbf{T}[\mathfrak{a}]$, $\mathfrak{a}'(\mathbf{t}_0') = \mathfrak{a}(\mathbf{t}_0)$ and $\mathbf{t}_0'(x) \to_{\mathcal{A}} \mathbf{t}_0(x)$. Since $\mathcal{G}$ is complete, there is a state $s \in S$ such that $s' \to s$ and $\ell(s) = \mathbf{T}$. ■

LEMMA 4.10. *Let $n \in \mathbb{N}$, $s \in S$, $\ell(s) = \mathbf{T}$, let $\mathfrak{a}$ be a $\mathbf{T}$-assignment in Δ_n and $\mathbf{t}_0(x) \in \mathbf{T}$. If $\mathfrak{M}_n \models \mathbf{T}[\mathfrak{a}]$ then there are a state $s'' \in S$ with $\ell(s'') = \mathbf{T}''$, a $\mathbf{T}''$-assignment $\mathfrak{a}''$ in Δ_{n+1}, and a type $\mathbf{t}_0''(x) \in \mathbf{T}''$ such that*

$$s \to s'', \qquad \mathfrak{M}_{n+1} \models \mathbf{T}''[\mathfrak{a}''], \qquad \mathfrak{a}(\mathbf{t}_0) = \mathfrak{a}''(\mathbf{t}_0''),$$

and $\mathbf{t}_0(x) \to_\bigcirc \mathbf{t}_0'(x) \to_{\mathcal{A}} \mathbf{t}_0''(x)$ for some type $\mathbf{t}_0'(x)$.

PROOF. Let $\ell_\bigcirc(s) = \mathbf{T}'$. Take the type $\mathbf{t}_0'(x) \in \mathbf{T}'$ such that $\mathbf{t}_0(x) \to_\bigcirc \mathbf{t}_0'(x)$ and define a $\mathbf{T}'$-assignment $\mathfrak{a}'$ in Δ_{n+1} so that $\mathfrak{a}'(\mathbf{t}_0') = \mathfrak{a}(\mathbf{t}_0)$. For every other type $\mathbf{t}'(x) \in \mathbf{T}'$, either there is a type $\mathbf{t}(x) \in \mathbf{T}$ with $\mathbf{t}(x) \to_\bigcirc \mathbf{t}'(x)$ or $\mathbf{t}'(x) = \mathbf{Ax}_\vartheta(x)$. In the former case put $\mathfrak{a}'(\mathbf{t}') = \mathfrak{a}(\mathbf{t})$ and in the latter one $\mathfrak{a}'(\mathbf{t}') = \mathfrak{a}(\mathbf{t}_0)$. Clearly, $\mathfrak{M}_{n+1} \models \mathbf{T}'[\mathfrak{a}']$ holds whenever $\mathfrak{M}_n \models \mathbf{T}[\mathfrak{a}]$ holds. By Lemma 4.9, we then have a required state s''. ■

LEMMA 4.11. *There exists an infinite sequence*

$$s_r \to s_0 \to s_1 \to \dots$$

of states in $\mathcal{G}$ such that every eventuality $\overline{\varphi\,\mathcal{U}\,\psi}$ in every type of $\ell(s_n)$, $n \geq 0$, is realised in the sequence $s_n, \dots, s_m$, for some $m \geq n$.

PROOF. Without loss of generality we may assume that $(\mathfrak{M}, 0) \models \vartheta$. By Lemma 4.8, we then have $\mathfrak{M}_0 \models \overline{\vartheta} \wedge \mathbf{Ax}_\vartheta(x)$.

We construct the required sequence by induction. We begin with s_r and take an arbitrary $\mathbf{T}_0'$-assignment $\mathfrak{a}_0'$ in Δ_0, where $\ell_\bigcirc(s_r) = \mathbf{T}_0'$. Then clearly $\mathfrak{M}_0 \models \mathbf{T}_0'[\mathfrak{a}_0']$. By Lemma 4.9, we obtain a state s_0 with $\ell(s_0) = \mathbf{T}_0$ and a $\mathbf{T}_0$-assignment $\mathfrak{a}_0$ in Δ_0 such that $s_r \to s_0$ and $\mathfrak{M}_0 \models \mathbf{T}_0[\mathfrak{a}_0]$. Denote the beginning of our sequence by

$$s_r \to s_0 \mid \mathfrak{a}_0$$

(we will always need to remember the last assignment).

Suppose now that we have constructed a sequence

$$s_r \to s_0 \to \dots \to s_n \mid \mathfrak{a}_n \tag{ii}$$

such that $\ell(s_n) = \mathbf{T}_n$ and $\mathfrak{M}_n \models \mathbf{T}_n[\mathfrak{a}_n]$. Two cases are possible.

Case 1. Every eventuality in every type of $\ell(s_i)$, $0 \leq i \leq n$, is realised in (ii). In this case we take an arbitrary type $\mathbf{t}_n(x) \in \mathbf{T}_n$ and, by Lemma 4.10, find a state s_{n+1} with $\ell(s_{n+1}) = \mathbf{T}_{n+1}$ and a $\mathbf{T}_{n+1}$-assignment $\mathfrak{a}_{n+1}$ in Δ_{n+1} such that $s_n \to s_{n+1}$ and $\mathfrak{M}_{n+1} \models \mathbf{T}_{n+1}[\mathfrak{a}_{n+1}]$. So we can extend (ii) with $s_{n+1} \mid \mathfrak{a}_{n+1}$:

$$s_r \to s_0 \to \dots \to s_n \to s_{n+1} \mid \mathfrak{a}_{n+1}.$$

Case 2. Suppose that Case 1 does not hold. Take a minimal $k \leq n$ such that some eventuality $\overline{\varphi\,\mathcal{U}\,\psi}(x)$ in some $\mathbf{t}_k(x) \in \mathbf{T}_k$ is not realised in (ii). As all $\mathbf{T}_i$ are $\mathcal{A}$-quasistates, $(\varphi\,\mathcal{U}\,\psi)^+ \in \mathbf{T}_i$. As $\mathbf{T}_k$ is saturated and satisfiable, either $\overline{\psi}(x) \in \mathbf{t}_k(x)$ or $\overline{\varphi}(x), \overline{\bigcirc(\varphi\,\mathcal{U}\,\psi)}(x) \in \mathbf{t}_k(x)$. And as $\overline{\varphi\,\mathcal{U}\,\psi}(x)$ is not realised in (ii), only the latter case is possible. It follows that there are $\mathbf{t}_{k+1}'(x)$ and $\mathbf{t}_{k+1}(x)$ such that

$$\mathbf{t}_k(x) \to_\bigcirc \mathbf{t}_{k+1}'(x) \to_\mathcal{A} \mathbf{t}_{k+1}(x) \quad \text{and} \quad \overline{\varphi\,\mathcal{U}\,\psi}(x) \in \mathbf{t}_{k+1}(x).$$

Thus we can choose a sequence

$$\mathbf{t}_k(x) \to_\bigcirc \mathbf{t}_{k+1}'(x) \to_\mathcal{A} \dots \to_\bigcirc \mathbf{t}_n'(x) \to_\mathcal{A} \mathbf{t}_n(x)$$

such that $\mathbf{t}_i(x) \in \mathbf{T}_i$,

$$\overline{\varphi\,\mathcal{U}\,\psi}(x) \in \mathbf{t}_i(x), \quad \overline{\bigcirc(\varphi\,\mathcal{U}\,\psi)}(x) \in \mathbf{t}_i(x) \quad \text{and} \quad \overline{\psi}(x) \notin \mathbf{t}_i(x) \tag{iii}$$

for all $i \in [k, n]$.

Let $a = \mathfrak{a}_n(\mathbf{t}_n)$. We have $\mathfrak{M}_n \models \mathbf{T}_n[\mathfrak{a}_n]$, and so, by (iii), $\mathfrak{M}_n \models \overline{\varphi\,\mathcal{U}\,\psi}[a]$ and $\mathfrak{M}_n \models \neg\overline{\psi}[a]$. Then by Lemma 4.8,

$$(\mathfrak{M}, n) \models (\varphi\,\mathcal{U}\,\psi)[a] \quad \text{and} \quad (\mathfrak{M}, n) \models \neg\psi[a]. \tag{iv}$$

Now we construct a sequence of states realising our eventuality. By applying Lemma 4.10 to s_n, $\mathfrak{a}_n$ and $\mathbf{t}_n(x) \in \mathbf{T}_n$ such that $\mathfrak{M}_n \models \mathbf{T}_n[\mathfrak{a}_n]$, $\mathfrak{a}_n(\mathbf{t}_n) = a$ and

$$\overline{\varphi\,\mathcal{U}\,\psi}(x) \in \mathbf{t}_n(x), \quad \overline{\bigcirc(\varphi\,\mathcal{U}\,\psi)}(x) \in \mathbf{t}_n(x), \quad \overline{\psi} \notin \mathbf{t}_n(x)$$

we find a state s_{n+1} with $\ell(s_{n+1}) = \mathbf{T}_{n+1}$ and a $\mathbf{T}_{n+1}$-assignment $\mathfrak{a}_{n+1}$ in Δ_{n+1} with

$$s_n \to s_{n+1} \quad \text{and} \quad \mathfrak{M}_{n+1} \models \mathbf{T}_{n+1}[\mathfrak{a}_{n+1}],$$

and $\mathbf{t}_{n+1}(x) \in \mathbf{T}_{n+1}$ such that $\mathfrak{a}_{n+1}(\mathbf{t}_{n+1}) = a$ and $\overline{\varphi\,\mathcal{U}\,\psi}(x) \in \mathbf{t}_{n+1}(x)$. So we can extend (ii) with $s_{n+1} \mid \mathfrak{a}_{n+1}$.

Note that $(\varphi\,\mathcal{U}\,\psi)^+ \in \mathbf{T}_{n+1}$ and $\mathbf{T}_{n+1}$ is a saturated flock, so either $\overline{\psi}(x) \in \mathbf{t}_{n+1}(x)$ or $\overline{\bigcirc(\varphi\,\mathcal{U}\,\psi)}(x) \in \mathbf{t}_{n+1}(x)$ and $\overline{\psi}(x) \notin \mathbf{t}_{n+1}(x)$. In the former case the eventuality is realised by

$$s_k \to \cdots \to s_n \to s_{n+1}.$$

In the latter case we again apply the above procedure to the state s_{n+1}, the assignment $\mathfrak{a}_{n+1}$ and the type $\mathbf{t}_{n+1}(x)$. It follows from (iv) that there must exist $m > n$ such that $(\mathfrak{M}, m) \models \psi[a]$, and so, by Lemma 4.8, $\mathfrak{M}_m \models \overline{\psi}[a]$. Thus, we will find a realising sequence in at most $m - n$ steps.

In the limit we obtain an infinite sequence

$$s_r \to s_0 \to s_1 \to \dots \tag{v}$$

satisfying the requirements of the lemma. ■

We are in a position now to prove the completeness part of Theorem 4.7.

Proof. Suppose that ϑ is satisfiable in a model from the class TM, and let $\mathcal{G} = \langle S, s_r, \to, \ell, \ell_{\bigcirc}\rangle$ be a complete tableau for ϑ. By Lemma 4.11, we have an infinite sequence

$$s_r \to s_0 \to s_1 \to \dots \tag{vi}$$

of states in $\mathcal{G}$ such that every eventuality $\overline{\varphi\,\mathcal{U}\,\psi}$ in every type of $\ell(s_n)$, $n \geq 0$, is realised in the sequence $s_n, \dots, s_m$, for some $m \geq n$.

To prove completeness, it suffices to show that no s_i from the sequence is eliminated. Let

$$S = S_0 \supseteq S_1 \supseteq \dots$$

be the sequence produced by the elimination procedure. We show by induction on n that, for all $n \in \mathbb{N}$,

$$\{s_r\} \cup \{s_i \mid i \in \mathbb{N}\} \subseteq S_n.$$

The basis of induction ($n = 0$) is clear. Suppose $\{s_r\} \cup \{s_i \mid i \in \mathbb{N}\} \subseteq S_k$. Since every state s_i has a successor, rule (E2) is not applicable to it. As all eventualities in the sequence (vi) are realised, rule (E3) is not applicable either. ■

4.2. Soundness

LEMMA 4.12. *Let $\mathcal{G} = \langle S, s_r, \rightarrow, \ell, \ell_{\bigcirc}\rangle$ be a complete tableau for ϑ and let S' be the set of states that remains after execution of the elimination procedure. If $s_r \in S'$ then there is an infinite sequence*

$$s_r \rightarrow s_0 \rightarrow s_1 \rightarrow \dots$$

of states in S' such that every eventuality $\overline{\varphi\,\mathcal{U}\,\psi}$ in every type of $\ell(s_n)$, $n \geq 0$, is realised by the sequence $s_n, \dots, s_m$ for some $m \geq n$.

PROOF. Suppose that we have constructed a sequence

$$s_r \rightarrow s_0 \rightarrow s_1 \rightarrow \dots \rightarrow s_n \tag{vii}$$

sitting entirely in S'. Two cases are possible.

Case 1. Every eventuality in every type of $\ell(s_i)$, $i \geq 0$, is realised in (vii). As (E2) is not applicable to s_n, we can extend (vii) by a state $s_{n+1} \in S'$ with $s_n \rightarrow s_{n+1}$.

Case 2. Suppose Case 1 does not hold. Take a minimal $k \in [0, n]$ such that an eventuality $\overline{\varphi\,\mathcal{U}\,\psi}(x)$ in a type of $\ell(s_k)$ is not realised in (vii). Then $\overline{\varphi\,\mathcal{U}\,\psi}(x)$ belongs to some type in $\ell(s_n)$. Since (E3) is not applicable to s_n, $\overline{\varphi\,\mathcal{U}\,\psi}(x)$ is realised by a sequence

$$s_n \rightarrow s_{n+1} \rightarrow \dots \rightarrow s_m$$

for some $m \geq n$ such that $s_i \in S'$ for $i \in [n, m]$. Then we extend (vii) with the states $s_{n+1}, \dots, s_m$.

In the limit we obtain a sequence satisfying the conditions of the lemma. ∎

We can now complete the proof of Theorem 4.7.

PROOF. Suppose that $\mathcal{G} = \langle S, s_r, \rightarrow, \ell, \ell_{\bigcirc} \rangle$ is a complete tableau for ϑ and S' is the set of states which remains after execution of the elimination procedure and that the root s_r was not eliminated. By Theorem 3.4, it is enough to prove that there exists a quasimodel satisfying ϑ. Lemma 4.12 provides us with an infinite sequence

$$s_r \rightarrow s_0 \rightarrow s_1 \rightarrow \dots$$

of states in S' realising all eventualities. The reader can readily check that $(\ell(s_i) \mid i \in \mathbb{N})$ is a quasimodel satisfying ϑ. ∎

4.3. Tableaux: constant domains

Let us consider now the case of constant domains. Tableaux for this case can be obtained by a simple modification of tableaux for the case of expanding domains. The major difference is that we use *exhaustive* saturation and transition rules.

DEFINITION 4.13 (exhaustive transition rule). The *exhaustive transition rule* for $\mathcal{QTL}'$ is the map $\mathcal{N}$ that takes a flock $\mathbf{T}$ for ϑ and returns the flock

$$\mathcal{N}(\mathbf{T}) = \big\{\mathbf{t}'(x) \mid \mathbf{t}(x) \in \mathbf{T} \text{ and } \mathbf{t}(x) \rightarrow_{\bigcirc} \mathbf{t}'(x)\big\} \cup \{\mathbf{Ax}_{\vartheta}(x) \cup \{\mathbf{CDA_T}\}\},$$

where

$$\mathbf{CDA_T} = \forall x \bigvee_{\substack{\mathbf{t}(x) \in \mathbf{T} \\ \mathbf{t}(x) \rightarrow_{\bigcirc} \mathbf{t}'(x)}} \mathbf{t}'(x).$$

The formula $\mathbf{CDA_T}$ is used to deal with constant domains: we must now construct a quasimodel in which the domains of runs are total. In other words, this means that for every quasistate $\mathbf{T}_{n+1}$ and every $\mathbf{t}(x) \in \mathbf{T}_{n+1}$, there must exist a type $\mathbf{t}'(x) \in \mathbf{T}_n$ such that the two conditions formulated in Definition 3.3 are satisfied. However, this is precisely what the joint use of $\mathbf{CDA_T}$ and the surrogate axioms ensures. Together with the modified Property (CO′) of exhaustive saturation rules, this approach resembles the 'minimal types' technique developed in [16].

THEOREM 4.14. *Let* $(\mathcal{QTL}', \mathsf{TM})$ *be a tmf-theory, where* TM *is a class of models with constant domains,* $(\mathcal{QL}', \mathsf{M})$ *a first-order reduct of* $(\mathcal{QTL}', \mathsf{TM})$, $\mathcal{A}$ *an exhaustive saturation rule for* $(\mathcal{QL}', \mathsf{M})$ *and* $\mathcal{N}$ *an exhaustive transition rule for* $\mathcal{QTL}'$. *Then for every* $\mathcal{QTL}'$*-sentence* ϑ *the following conditions are equivalent:*

(1) ϑ *is satisfiable in a model from* TM;

(2) *the root of a complete tableau for* ϑ *cannot be eliminated using rules* (E2) *and* (E3).

A close inspection of the proofs for the case of expanding domains shows that it is sufficient to prove reformulations of the lemmas above in which

- the model $\mathfrak{M}$ is assumed to have constant domains,
- the $\ell(s_n)$ are exhaustive flocks for the corresponding $\mathfrak{M}_n$.

The only non-trivial changes are in Lemmas 4.9 and 4.10.

Suppose that $\mathcal{G} = \langle S, s_r, \rightarrow, \ell, \ell_{\bigcirc} \rangle$ is a complete tableau for ϑ and that ϑ is satisfiable in a model $\mathfrak{M} = \langle \mathbb{N}, <, I \rangle \in \mathsf{TM}$, where $I(n) = \left\langle \Delta, P_0^{I(n)}, \dots \right\rangle$ for all $n \in \mathbb{N}$.

LEMMA 4.15. *Let* $n \in \mathbb{N}$, $s' \in S$, $\ell_{\bigcirc}(s') = \mathbf{T}'$, *let* $\mathfrak{a}'$ *be a* $\mathbf{T}'$*-assignment in* Δ *and* $\mathbf{t}_0'(x) \in \mathbf{T}'$. *If* $\mathfrak{M}_n \models \mathbf{T}'[\mathfrak{a}']$ *and* $\mathbf{T}'$ *is exhaustive for* $\mathfrak{M}_n$ *then there are a state* $s \in S$ *with* $\ell(s) = \mathbf{T}$, *a* $\mathbf{T}$*-assignment* $\mathfrak{a}$ *in* Δ *and a type* $\mathbf{t}_0(x) \in \mathbf{T}$ *such that*

$$s' \rightarrow s, \qquad \mathfrak{M}_n \models \mathbf{T}[\mathfrak{a}] \text{ and } \mathbf{T} \text{ is exhaustive for } \mathfrak{M}_n, \qquad \mathfrak{a}'(\mathbf{t}_0') = \mathfrak{a}(\mathbf{t}_0)$$

and $\mathbf{t}_0'(x) \rightarrow_{\mathcal{A}} \mathbf{t}_0(x)$.

PROOF. The proof is analogous to the proof of Lemma 4.9 and follows immediately from (CO$'$). ■

LEMMA 4.16. *Let* $n \in \mathbb{N}$, $s \in S$, $\ell(s) = \mathbf{T}$, *let* $\mathfrak{a}$ *be a* $\mathbf{T}$*-assignment in* Δ, *and* $\mathbf{t}_0(x) \in \mathbf{T}$. *If* $\mathfrak{M}_n \models \mathbf{T}[\mathfrak{a}]$ *and* $\mathbf{T}$ *is exhaustive for* $\mathfrak{M}_n$ *then there are a state* $s'' \in S$ *with* $\ell(s'') = \mathbf{T}''$, *a* $\mathbf{T}''$*-assignment* $\mathfrak{a}''$ *in* Δ, *and a type* $\mathbf{t}_0''(x) \in \mathbf{T}''$ *such that*

$$s \rightarrow s'', \quad \mathfrak{M}_{n+1} \models \mathbf{T}''[\mathfrak{a}''] \text{ and } \mathbf{T}'' \text{ is exhaustive for } \mathfrak{M}_{n+1}, \quad \mathfrak{a}(\mathbf{t}_0) = \mathfrak{a}''(\mathbf{t}_0'')$$

and $\mathbf{t}_0(x) \rightarrow_{\bigcirc} \mathbf{t}_0'(x) \rightarrow_{\mathcal{A}} \mathbf{t}_0''(x)$ *for some type* $\mathbf{t}_0'(x)$.

PROOF. The proof is analogous to the proof of Lemma 4.10. The only difference is that in the proof of $\mathfrak{M}_{n+1} \models \mathbf{T}'[\mathfrak{a}']$ we have to show additionally that

$$\mathfrak{M}_{n+1} \models \mathbf{CDA_T}.$$

Suppose $a \in \Delta$. As $\mathbf{T}$ is exhaustive for $\mathfrak{M}_n$, there is a type $\mathbf{t}(x) \in \mathbf{T}$ such that $\mathfrak{M}_n \models \mathbf{t}[a]$, and we choose $\mathbf{t}'(x) \in \mathbf{T}'$ with $\mathbf{t}(x) \to_{\bigcirc} \mathbf{t}'(x)$. Then $\mathfrak{M}_{n+1} \models \mathbf{t}'[a]$ and $\mathfrak{M}_{n+1} \models \mathbf{CDA_T}$.

It should be clear that the flock $\mathbf{T}'$ is exhaustive for $\mathfrak{M}_{n+1}$, and then Lemma 4.15 supplies a state s'' and an assignment $\mathfrak{a}''$ as required. ■

5. Instantiating the framework

The purpose of this section is to illustrate the generality of our approach by presenting example instantiations of the framework. To keep the presentation succinct, we stick to simple yet useful fragments of first-order logic: exhaustive and non-exhaustive saturation rules are presented for

1. the f-theory $(\mathcal{QL}^1, \mathsf{FO})$ induced by the one-variable fragment of first-order logic (which is a notational variant of propositional modal logic $\mathsf{S5}$ [19]) and
2. the f-theory $(\mathcal{ST}, \mathsf{TR})$ corresponding to the propositional bimodal logic $\mathsf{S4}_u$ introduced in Section 2.2.

In fact, we show that the well-known, *existing* tableau decision procedures for these fragments of first-order logic—for the corresponding modal logics, to be more precise (see e.g., [4, 9])—can be regarded as saturation rules for the case of expanding domains, whereas some additional efforts are needed to obtain exhaustive saturation rules. The technique described in Section 4 then yields 'temporal' tableau algorithms for the one-variable fragment $\mathcal{QTL}^1$ of $\mathcal{QTL}$ and the tmf-theory $(\mathcal{TST}, \mathsf{TTR})$ from Section 2.2, i.e., the temporalisation of $(\mathcal{ST}, \mathsf{TR})$—both for expanding and constant domains. We finish this section with presenting some example runs of the tableau algorithm for $\mathcal{QTL}^1$.

5.1. S5

As a (non-exhaustive) saturation rule for the one-variable fragment of first-order logic, we use a slight variant of the well-known prefixed tableaux for $\mathsf{S5}$, as presented e.g. in [4]. The main difference between our presentation of this algorithm and the one given in [4] is that we write formulas in the syntax of first-order logic rather than modal logic.

A *labelled formula* is of the from $\sigma :: \varphi$, where σ is a label and φ a formula (in our examples φ is a $\mathcal{QL}$-formula with at most one free variable). A *label* σ is a nonempty sequence of natural numbers separated by dots. For example, 1.21 and 1.2.1 are labels. Labels allow us to distinguish between formulas that belong to one world and those belonging to another one. Moreover, the structure of labels describes the accessibility relation between the worlds. Although in the definition of the tableau algorithm for $\mathsf{S5}$ we use only natural numbers as labels (i.e., all labels are of length 1), we still denote these sequences by Greek letters (σ, τ, etc.) because the same tableau rules are used for $\mathsf{S4}_u$ in Section 5.2, where the structure of labels is essential.

A *tableau* T (for both f-theories we consider in this section) is a finite tree, where each node contains a single labelled formula. A *tableau branch* $\mathcal{B}$ of T is a path starting at the root node and ending at a leaf node.

Suppose we are given a flock $\mathbf{T} = \{\mathbf{t}_1(x), \dots, \mathbf{t}_k(x)\}$. The tableau algorithm starts with an initial tableau T_0 consisting of a single branch such that its nodes contain all labelled formulas of the set

$$\{n :: \varphi(x) \mid \varphi(x) \in \mathbf{t}_n(x) \text{ and } 1 \leq n \leq k\}.$$

Thus, for every type $\mathbf{t}_i(x)$ we introduce a unique label i which denotes a new world for this type (in the modal logic setting), or a set of domain elements indistinguishable by formulas of $\mathbf{t}_i(x)$ (in the first-order logic setting).

Then the algorithm exhaustively applies the *tableau rules* given in Fig. 2 to nodes on each branch $\mathcal{B}$ of the tableau as follows.

($l\neg$) If a node contains $\sigma :: \neg\neg\varphi(x)$ then an application of ($l\neg$) appends a node containing $\sigma :: \varphi(x)$ to $\mathcal{B}$.

($l\wedge$) If a node contains $\sigma :: \varphi(x) \wedge \psi(x)$ then an application of ($l\wedge$) appends two consecutive nodes to $\mathcal{B}$, one containing $\sigma :: \varphi(x)$ and the other $\sigma :: \psi(x)$.

($l\vee$) If a node contains $\sigma :: \neg(\varphi(x) \wedge \psi(x))$ then an application of ($l\vee$) splits the end of $\mathcal{B}$ and extends the left fork with $\sigma :: \neg\varphi(x)$ and the right one with $\sigma :: \neg\psi(x)$.

($l\exists$) If a node contains $\sigma :: \neg\forall x\, \varphi(x)$ then an application of ($l\exists$) extends $\mathcal{B}$ with $\tau :: \neg\varphi(x)$, where τ is a new label on $\mathcal{B}$.

($l\forall$) If a node contains $\sigma :: \forall x\, \varphi(x)$ then an application of ($l\forall$) extends $\mathcal{B}$ with $\tau :: \varphi(x)$, where the label τ already exists on $\mathcal{B}$.

($l\forall^*$) If a node contains $\sigma :: \varphi$, where φ is a sentence, then an application of ($l\forall^*$) extends $\mathcal{B}$ with $\tau :: \varphi$, where the label τ already exists on $\mathcal{B}$.

$$(l\neg)\quad \frac{\sigma :: \neg\neg\varphi(x)}{\sigma :: \varphi(x)} \qquad (l\wedge)\quad \frac{\sigma :: \varphi(x) \wedge \psi(x)}{\begin{array}{c}\sigma :: \varphi(x)\\ \sigma :: \psi(x)\end{array}}$$

$$(l\vee)\quad \frac{\sigma :: \neg(\varphi(x) \wedge \psi(x))}{\sigma :: \neg\varphi(x) \quad \mid \quad \sigma :: \neg\psi(x)}$$

$$(l\exists)\quad \frac{\sigma :: \neg\forall x\, \varphi(x)}{\tau :: \neg\varphi(x)} \qquad (l\forall)\quad \frac{\sigma :: \forall x\, \varphi(x)}{\tau :: \varphi(x)} \qquad (l\forall^*)\quad \frac{\sigma :: \varphi}{\tau :: \varphi}$$

τ is new for $\mathcal{B}$ $\qquad$ τ is used on $\mathcal{B}$ $\qquad$ τ is used on $\mathcal{B}$

Figure 2. Tableau rules for S5.

Observe that every branch $\mathcal{B}$ of a tableau can be converted into a flock $\mathbf{T}_\mathcal{B}$ by setting

$$\mathbf{T}_\mathcal{B} = \{\mathbf{t}_\sigma^\mathcal{B}(x) \mid \sigma \text{ is a label on } \mathcal{B}\},$$

where

$$\mathbf{t}_\sigma^\mathcal{B}(x) = \{\varphi(x) \mid \sigma :: \varphi(x) \text{ occurs on } \mathcal{B}\}$$

(note that types $\mathbf{t}_\sigma^\mathcal{B}(x)$ and $\mathbf{t}_\tau^\mathcal{B}(x)$ may coincide for $\sigma \neq \tau$; in this case they are identified in $\mathbf{T}_\mathcal{B}$).

We generally assume tableau rules to be applied in such a way that no labelled formula appears twice on the same branch and that $(l\exists)$ is never applied twice to the same labelled formula. A branch $\mathcal{B}$ is *complete* if no rule can be applied to it. A branch $\mathcal{B}$ is called *contradictory* if both $\sigma :: \varphi$ and $\sigma :: \neg\varphi$ occur on $\mathcal{B}$, for some formula φ and label σ. A tableau T is *complete* if each branch in T is complete.

To simplify further considerations, we fix an order of rule applications. We assume that there is an ordering on pairs $((l), \sigma :: \varphi)$, with tableau rule (l) and labelled formula $\sigma :: \varphi$, and that a tableau rule (l) is applied to a formula $\sigma :: \varphi$ only if $((l), \sigma :: \varphi)$ is minimal with respect to the ordering. In this way, the tableau constructed by the algorithm for a given input $\mathbf{T}$ is completely determined. We call this tableau the *canonical tableau* for $\mathbf{T}$. Note that canonical tableaux are complete by definition. We use $\mathfrak{B}_\mathbf{T}$ to denote the set of non-contradictory branches in the canonical tableau for $\mathbf{T}$.

If started on a flock $\mathbf{T}$, the tableau algorithm constructs the canonical tableau for $\mathbf{T}$. It is a standard task to prove that this construction terminates. If $\mathfrak{B}_\mathbf{T}$ is the empty set, then $\mathcal{A}_{\mathsf{S5}}(\mathbf{T}) = \textbf{clash}$ is returned. Otherwise, each element of $\mathfrak{B}_{\mathbf{T}}$ represents a flock as explained above. Since every branch in $\mathfrak{B}_{\mathbf{T}}$ is complete, the corresponding flocks are saturated. Thus, if $\mathfrak{B}_{\mathbf{T}}$ is nonempty, then the tableau algorithm returns the set

$$\mathcal{A}_{\mathsf{S5}}(\mathbf{T}) = \{\mathbf{T}_{\mathcal{B}} \mid \mathcal{B} \in \mathfrak{B}_{\mathbf{T}}\}.$$

It is easy to see that this algorithm satisfies property (TR) of saturation rules. Hence, let us proceed to property (CO).

LEMMA 5.1. *Let* $\mathbf{T}$ *be a flock. For every model* $\mathfrak{M} \in \mathsf{FO}$ *with domain* Δ, *every type* $\mathbf{t}(x) \in \mathbf{T}$, *and every* $\mathbf{T}$*-assignment* $\mathfrak{a}$ *in* Δ, *if* $\mathfrak{M} \models \mathbf{T}[\mathfrak{a}]$ *then there are a branch* $\mathcal{B} \in \mathfrak{B}_{\mathbf{T}}$, *a type* $\mathbf{t}'(x) \in \mathbf{T}_{\mathcal{B}}$, *and a* $\mathbf{T}_{\mathcal{B}}$*-assignment* $\mathfrak{a}'$ *in* Δ *such that* $\mathbf{t}(x) \subseteq \mathbf{t}'(x)$, $\mathfrak{a}(\mathbf{t}) = \mathfrak{a}'(\mathbf{t}')$, *and* $\mathfrak{M} \models \mathbf{T}_{\mathcal{B}}[\mathfrak{a}']$.

PROOF. Let $\mathbf{T} = \{\mathbf{t}_1(x), \dots, \mathbf{t}_k(x)\}$ be a flock, $\mathfrak{M} \in \mathsf{FO}$ with domain Δ and let $\mathfrak{a}$ be a $\mathbf{T}$-assignment in Δ such that $\mathfrak{M} \models \mathbf{T}[\mathfrak{a}]$. Without loss of generality we may assume that $\mathbf{t} = \mathbf{t}_1$. Suppose that the canonical tableau for $\mathbf{T}$ is the last tableau of the sequence

$$T_0, T_1, \dots, T_n,$$

where T_0 is the initial tableau for $\mathbf{T}$ and, for every $i < n$, T_{i+1} is obtained from T_i by an application of a tableau rule.

We define, for each tableau T_i, $0 \leq i \leq n$, a branch $\mathcal{B}_i$ of T_i and a $\mathbf{T}_{\mathcal{B}_i}$-assignment $\mathfrak{a}_i$ in Δ as follows. Let $\mathcal{B}_0$ be the single branch of T_0. Set $\mathfrak{a}_0 = \mathfrak{a}$ (recall that formulas of type $\mathbf{t}_j(x)$ are labelled by j on $\mathcal{B}_0$). Clearly we have $\mathfrak{M} \models \mathbf{T}_{\mathcal{B}_0}[\mathfrak{a}_0]$. Then we proceed in such a way that the following conditions are satisfied for every i, $0 < i \leq n$:

$$\mathfrak{M} \models \mathbf{T}_{\mathcal{B}_i}[\mathfrak{a}_i] \quad \text{and} \quad \mathfrak{a}_i(\mathbf{t}_1^{\mathcal{B}_i}) = \mathfrak{a}_{i-1}(\mathbf{t}_1^{\mathcal{B}_{i-1}}). \tag{viii}$$

Without loss of generality we can always assume that $\mathbf{t}_1^{\mathcal{B}_i} \in \mathbf{T}_{\mathcal{B}_i}$ for every i, $0 \leq i \leq n$ (recall that in a flock $\mathbf{T}_{\mathcal{B}_i}$ types can be identified).

Suppose that we have already constructed $\mathcal{B}_i$ and $\mathfrak{a}_i$, $0 \leq i < n$, and T_{i+1} is obtained from T_i by an application of (l). Consider all possible cases.

$(l\vee)$ is applied to $\sigma :: \neg(\varphi(x) \wedge \psi(x))$ on $\mathcal{B}_i$. The rule application splits $\mathcal{B}_i$ into two branches $\mathcal{B}_\varphi$ and $\mathcal{B}_\psi$, where $\mathcal{B}_\varphi$ contains a new node with $\sigma :: \neg\varphi(x)$ and $\mathcal{B}_\psi$ contains a new node with $\sigma :: \neg\psi(x)$. Let a be the value assigned to the type of σ at step i, i.e., $\mathfrak{a}_i(\mathbf{t}_\sigma^{\mathcal{B}_i})$. Since, by IH, $\mathfrak{M} \models \neg(\varphi(x) \wedge \psi(x))[a]$, we have either $\mathfrak{M} \models \neg\varphi(x)[a]$ or $\mathfrak{M} \models \neg\psi(x)[a]$. In the former case, set $\mathcal{B}_{i+1} = \mathcal{B}_\varphi$ and in the latter one $\mathcal{B}_{i+1} = \mathcal{B}_\psi$. In either case, let $\mathfrak{a}_{i+1}(\mathbf{t}_\sigma^{\mathcal{B}_{i+1}}) = \mathfrak{a}_i(\mathbf{t}_\sigma^{\mathcal{B}_i})$ for every type $\mathbf{t}_\sigma^{\mathcal{B}_{i+1}}(x) \in \mathbf{T}_{\mathcal{B}_{i+1}}$.

$(l\exists)$ is applied to $\tau' :: \neg\forall x\,\varphi(x)$ on $\mathcal{B}_i$, introducing a label τ and creating a node with $\tau :: \neg\varphi(x)$. Since $\mathfrak{M} \models \neg\forall x\,\varphi(x)$, there exists an element $a \in \Delta$ such that $\mathfrak{M} \models \varphi[a]$. Let $\mathcal{B}_{i+1}$ be the extension of $\mathcal{B}_i$ with the new node and $\mathfrak{a}_{i+1}(\mathbf{t}_\sigma^{\mathcal{B}_{i+1}}) = \mathfrak{a}_i(\mathbf{t}_\sigma^{\mathcal{B}_i})$ for every type $\mathbf{t}_\sigma^{\mathcal{B}_{i+1}}(x) \in \mathbf{T}_{\mathcal{B}_{i+1}}$ and $\mathfrak{a}_{i+1}(\mathbf{t}_\tau^{\mathcal{B}_{i+1}}) = a$.

In all other cases $\mathcal{B}_{i+1}$ is the extension of $\mathcal{B}_i$ with the new nodes and $\mathfrak{a}_{i+1}(\mathbf{t}_\sigma^{\mathcal{B}_{i+1}}) = \mathfrak{a}_i(\mathbf{t}_\sigma^{\mathcal{B}_i})$ for every type $\mathbf{t}_\sigma^{\mathcal{B}_{i+1}}(x) \in \mathbf{T}_{\mathcal{B}_{i+1}}$.

It is easy to show that in every case $\mathcal{B}_{i+1}$ and $\mathfrak{a}_{i+1}$ satisfy (viii). Since $\mathfrak{M} \models \mathbf{T}_{\mathcal{B}_n}[\mathfrak{a}_n]$, the branch $\mathcal{B}_n$ is not contradictory, i.e., $\mathcal{B}_n \in \mathfrak{B}_{\mathbf{T}}$. Now set $\mathcal{B} = \mathcal{B}_n$, $\mathfrak{a}'$ to $\mathfrak{a}_n$ and $\mathbf{t}'(x) = \mathbf{t}_1^{\mathcal{B}_n}(x)$. It is readily checked that $\mathcal{B}$, $\mathbf{t}'$ and $\mathfrak{a}'$ are as required. ■

Note that the proof of the lemma above resembles the standard completeness proof for the described tableau algorithm: an existing model is used to 'guide' the application of the tableau rules. We now come to property (SO), i.e., to soundness.

LEMMA 5.2. *For every flock $\mathbf{T}$, every cardinal $\kappa' \geq \aleph_0$, and every branch $\mathcal{B} \in \mathfrak{B}_{\mathbf{T}}$, there exists a model*

$$\mathfrak{M} = \left\langle \Delta, Q_0^{\mathfrak{M}}, \dots q_0^{\mathfrak{M}}, \dots \right\rangle \in \mathsf{FO},$$

in which

- $\Delta = \bigcup_{\mathbf{t} \in \mathbf{T}_{\mathcal{B}}} \Delta^{\mathbf{t}}$, *where $\Delta^{\mathbf{t}}$ are pairwise disjoint sets of cardinality κ',*
- $q_i^{\mathfrak{M}}$ *is true iff $q_i \in \mathbf{T}_{\mathcal{B}}$,*
- $a \in Q_i^{\mathfrak{M}}$ *iff there is a type $\mathbf{t}(x) \in \mathbf{T}_{\mathcal{B}}$ such that $Q_i(x) \in \mathbf{t}(x)$ and $a \in \Delta^{\mathbf{t}}$,*

such that $\mathfrak{M} \models \mathbf{t}[a]$ holds for all $\mathbf{t}(x) \in \mathbf{T}_{\mathcal{B}}$ and $a \in \Delta^{\mathbf{t}}$.

PROOF. Fix a branch $\mathcal{B} \in \mathfrak{B}_{\mathbf{T}}$ and a cardinal $\kappa' \geq \aleph_0$. Define a model $\mathfrak{M}$ by taking

$$\begin{array}{rcl} \Delta^{\mathbf{t}} & = & \{\langle \mathbf{t}, \xi\rangle \mid \xi < \kappa'\} \text{ for } \mathbf{t}(x) \in \mathbf{T}_{\mathcal{B}}, \\ Q_i^{\mathfrak{M}} & = & \{\langle \mathbf{t}, \xi\rangle \mid Q_i(x) \in \mathbf{t}(x) \text{ and } \xi < \kappa'\}, \\ q_i^{\mathfrak{M}} & = & \{\langle \mathbf{t}, \xi\rangle \mid q_i \in \mathbf{t}(x) \text{ and } \xi < \kappa'\}. \end{array}$$

Using the fact that $\mathcal{B}$ is complete and non-contradictory, by induction on the structure of formulas one can easily show that $\mathfrak{M}$ is as required. ■

We thus obtain the following lemma.

LEMMA 5.3. *$\mathcal{A}_{\mathsf{S5}}$ is a saturation rule for $(\mathcal{QL}^1, \mathsf{FO})$.*

$$(l\pi)\quad \frac{\sigma :: \neg\forall v\,(R(x,v) \to \varphi(v))}{\sigma.m :: \neg\varphi(x)} \qquad \sigma.m \text{ is new to } \mathcal{B}$$

$$(lK)\quad \frac{\sigma :: \forall v\,(R(x,v) \to \varphi(v))}{\sigma.w :: \varphi(x)} \qquad (lT)\quad \frac{\sigma :: \forall v\,(R(x,v) \to \varphi(v))}{\sigma :: \varphi(x)}$$

$$(l4)\quad \frac{\sigma :: \forall v\,(R(x,v) \to \varphi(v))}{\sigma.w :: \forall v\,(R(x,v) \to \varphi(v))}$$

$\sigma.w$ is already exists on $\mathcal{B}$

Figure 3. Additional tableau rules for $\mathsf{S4}_u$.

5.2. $\mathsf{S4}_u$

Let us now extend the previous example to a saturation rule for the propositional modal logic $\mathsf{S4}$ with the universal modality, that is to the f-theory $(\mathcal{ST}, \mathsf{TR})$ defined in Section 2.2.

The tableau algorithm for $\mathsf{S4}_u$ is similar to that for $\mathsf{S5}$, so we concentrate on the differences. The set of tableau rules is comprised of those in Fig. 2 (for the Booleans and the universal modality) and Fig. 3 (for the transitive and reflexive modal operator $\Box$ of $\mathsf{S4}$—its first-order translation, to be more precise). Again we assume that the rules $(l\exists)$ and $(l\pi)$ are applied at most once for every node.

To ensure termination of rule application, some additional efforts are required. We say that a label σ is *reduced* if no rule different from $(l\exists)$ and $(l\pi)$ can be applied to nodes containing $\sigma :: \varphi$. A label σ is called *fully reduced* if no tableau rule is applicable to nodes containing $\sigma :: \varphi$. Now, a branch $\mathcal{B}$ is *complete* if

- all labels on $\mathcal{B}$ are reduced and
- for every σ that is not fully reduced, there exists a fully reduced label τ such that $\mathbf{t}_\sigma^{\mathcal{B}}(x) = \mathbf{t}_\tau^{\mathcal{B}}(x)$.

To guarantee termination, tableau rules must not be applied to complete branches. The tableau algorithm works as the one from the previous section: it constructs the canonical tableau, returns $\mathcal{A}_{\mathsf{S4}_u} = \textbf{clash}$ if $\mathfrak{B}_{\mathbf{T}}$ is empty and the set of saturated flocks

$$\mathcal{A}_{\mathsf{S4}_u}(\mathbf{T}) = \{\mathbf{T}_{\mathcal{B}} \mid \mathcal{B} \in \mathfrak{B}_{\mathbf{T}}\},$$

otherwise.

We now show that the extended algorithm is a (non-exhaustive) saturation rule for $(\mathcal{ST}, \mathsf{TR})$. As it is easy to prove that (TR) is satisfied, we again start with (CO).

LEMMA 5.4. *Let* $\mathbf{T}$ *be a flock. For every model* $\mathfrak{M} \in \mathsf{TR}$ *with domain* Δ, *every type* $\mathbf{t}(x) \in \mathbf{T}$, *and every* $\mathbf{T}$*-assignment* $\mathfrak{a}$ *in* Δ, *if* $\mathfrak{M} \models \mathbf{T}[\mathfrak{a}]$ *then there are a branch* $\mathcal{B} \in \mathfrak{B}_{\mathbf{T}}$, *a type* $\mathbf{t}'(x) \in \mathbf{T}_{\mathcal{B}}$, *and a* $\mathbf{T}_{\mathcal{B}}$*-assignment* $\mathfrak{a}'$ *in* Δ *such that* $\mathbf{t}(x) \subseteq \mathbf{t}'(x)$, $\mathfrak{a}(\mathbf{t}) = \mathfrak{a}'(\mathbf{t}')$, *and* $\mathfrak{M} \models \mathbf{T}_{\mathcal{B}}[\mathfrak{a}']$.

PROOF. Let $\mathbf{T} = \{\mathbf{t}_1(x), \dots, \mathbf{t}_k(x)\}$ be a flock,

$$\mathfrak{M} = \left\langle \Delta, R^{\mathfrak{M}}, Q_0^{\mathfrak{M}}, \dots, q_0^{\mathfrak{M}}, \dots, \right\rangle \in \mathsf{TR},$$

and let $\mathfrak{a}$ be a $\mathbf{T}$-assignment in Δ such that $\mathfrak{M} \models \mathbf{T}[\mathfrak{a}]$. Without loss of generality we may assume that $\mathbf{t} = \mathbf{t}_1$. Suppose that the canonical tableau for $\mathbf{T}$ is the last tableau of the sequence

$$T_0, T_1, \dots, T_n,$$

where T_0 is the initial tableau for $\mathbf{T}$ and, for every $i < n$, T_{i+1} is obtained from T_i by an application of a tableau rule. For a branch $\mathcal{B}$ of a tableau and an $\mathbf{T}_{\mathcal{B}}$-assignment $\mathfrak{a}$, we write $\mathfrak{M} \models^{\mathfrak{a}} R^{\mathcal{B}}$ to say that $(\mathfrak{a}(\mathbf{t}_\sigma^{\mathcal{B}}), \mathfrak{a}(\mathbf{t}_{\sigma.\tau}^{\mathcal{B}})) \in R^{\mathfrak{M}}$, for all labels σ and $\sigma.\tau$ on the branch (both σ and τ are sequences of natural numbers).

We define, for each tableau T_i, $0 \le i \le n$, a branch $\mathcal{B}_i$ of T_i and a $\mathbf{T}_{\mathcal{B}_i}$-assignment $\mathfrak{a}_i$. Let $\mathcal{B}_0$ be the single branch of T_0. Set $\mathfrak{a}_0 = \mathfrak{a}$. Then we proceed in such a way that the following conditions are satisfied for every i, $0 < i \le n$,

$$\mathfrak{M} \models \mathbf{T}_{\mathcal{B}_i}[\mathfrak{a}_i], \qquad \mathfrak{a}_i(\mathbf{t}_1^{\mathcal{B}_i}) = \mathfrak{a}_{i-1}(\mathbf{t}_1^{\mathcal{B}_{i-1}}) \quad \text{and} \quad \mathfrak{M} \models^{\mathfrak{a}_i} R^{\mathcal{B}_i}.$$

Assume that we have already constructed $\mathcal{B}_i$ and $\mathfrak{a}_i$, for $0 \le i < n$, and T_{i+1} is obtained from T_i by an application of (l). Since the rules in Fig. 2 can be treated in precisely the same way as in Lemma 5.1, we concentrate only on the rules in Fig. 3.

$(l\pi)$ is applied to $\sigma :: \neg\forall v\,(R(x,v) \to \varphi(v))$ on $\mathcal{B}_i$, introducing a new label $\sigma.m$ and creating a node $\sigma.m :: \neg\varphi(x)$. Let a be the value assigned to the type of σ at step i, i.e., $\mathfrak{a}_i(\mathbf{t}_\sigma^{\mathcal{B}_i})$. Since, by IH, $\mathfrak{M} \models (\neg\forall v\,(R(x,v) \to \varphi(v)))[a]$, there exists an element $a' \in \Delta$ such that $(a, a') \in R^{\mathfrak{M}}$ and $\mathfrak{M} \models \neg\varphi[a']$. Let $\mathcal{B}_{i+1}$ be the extension of $\mathcal{B}_i$ with the new node, $\mathfrak{a}_{i+1}(\mathbf{t}_{\sigma.m}^{\mathcal{B}_{i+1}}) = a'$ and $\mathfrak{a}_{i+1}(\mathbf{t}_\tau^{\mathcal{B}_{i+1}}) = \mathfrak{a}_i(\mathbf{t}_\tau^{\mathcal{B}_i})$ for all other types $\mathbf{t}_\tau^{\mathcal{B}_{i+1}}(x) \in \mathbf{T}_{\mathcal{B}_{i+1}}$.

In all other cases $\mathcal{B}_{i+1}$ is the extension of $\mathcal{B}_i$ with the new nodes and $\mathfrak{a}_{i+1}(\mathbf{t}_\sigma^{\mathcal{B}_{i+1}}) = \mathfrak{a}_i(\mathbf{t}_\sigma^{\mathcal{B}_i})$ for every type $\mathbf{t}_\sigma^{\mathcal{B}_{i+1}}(x) \in \mathbf{T}_{\mathcal{B}_{i+1}}$.

Again it is straightforward to show that $\mathcal{B}_{i+1}$ and $\mathfrak{a}_{i+1}$ are as required (for (lK), (lT) and $(l4)$ we need to use the fact that $\mathfrak{M} \models^{\mathfrak{a}_i} R^{\mathcal{B}_i}$) and that $\mathcal{B}_n$ and $\mathfrak{a}_n$ induce a branch $\mathcal{B} \in \mathfrak{B}_{\mathbf{T}}$, a type $\mathbf{t}'(x) \in \mathbf{T}_{\mathcal{B}}$, and a $\mathbf{T}_{\mathcal{B}}$-assignment $\mathfrak{a}'$, as required by the lemma. ■

It remains to prove that the soundness property (SO) holds.

LEMMA 5.5. *For every flock $\mathbf{T}$, every cardinal $\kappa' \geq \aleph_0$, and every branch $\mathcal{B} \in \mathfrak{B}_{\mathbf{T}}$ there exists a model*

$$\mathfrak{M} = \left\langle \Delta, R^{\mathfrak{M}}, Q_0^{\mathfrak{M}}, \dots q_0^{\mathfrak{M}}, \dots \right\rangle \in \mathsf{TR},$$

in which

- $\Delta = \bigcup_{\mathbf{t} \in \mathbf{T}_{\mathcal{B}}} \Delta^{\mathbf{t}}$, *where $\Delta^{\mathbf{t}}$ are pairwise disjoint sets of cardinality κ',*
- $q_i^{\mathfrak{M}}$ *is true iff $q_i \in \mathbf{T}_{\mathcal{B}}$,*
- $a \in Q_i^{\mathfrak{M}}$ *iff there is a type $\mathbf{t}(x) \in \mathbf{T}_{\mathcal{B}}$ such that $Q_i(x) \in \mathbf{t}(x)$ and $a \in \Delta^{\mathbf{t}}$,*

such that $\mathfrak{M} \models \mathbf{t}[a]$ holds for all $\mathbf{t}(x) \in \mathbf{T}_{\mathcal{B}}$ and $a \in \Delta^{\mathbf{t}}$.

PROOF. Fix a branch $\mathcal{B} \in \mathfrak{B}_{\mathbf{T}}$ and a cardinal $\kappa' \geq \aleph_0$. Define a model $\mathfrak{M}$ by taking

$$\begin{array}{rcl}
\Delta^{\mathbf{t}} & = & \{\langle \mathbf{t}, \xi\rangle \mid \xi < \kappa'\} \ \text{ for } \mathbf{t}(x) \in \mathbf{T}_{\mathcal{B}}, \\
R^{\mathfrak{M}} & = & \{(\langle \mathbf{t}, \xi\rangle, \langle \mathbf{t}', \xi'\rangle) \mid \mathbf{t} = \mathbf{t}_\sigma^{\mathcal{B}} \text{ and } \mathbf{t}' = \mathbf{t}_\tau^{\mathcal{B}} \text{ for } \sigma \leq \tau\}, \\
Q_i^{\mathfrak{M}} & = & \{\langle \mathbf{t}, \xi\rangle \mid Q_i(x) \in \mathbf{t}(x) \text{ and } \xi < \kappa'\}, \\
q_i^{\mathfrak{M}} & = & \{\langle \mathbf{t}, \xi\rangle \mid q_i \in \mathbf{t}(x) \text{ and } \xi < \kappa'\},
\end{array}$$

where $\sigma \leq \tau$ iff σ is a (not necessarily proper) prefix of τ. Clearly, $R^{\mathfrak{M}}$ is reflexive and transitive. Since distinct labels θ and θ' may describe the same type $\mathbf{t}_\theta^{\mathcal{B}} = \mathbf{t}_{\theta'}^{\mathcal{B}}$, in general $R^{\mathfrak{M}}$ is not necessarily antisymmetric, i.e., it is a quasi-order. Using the fact that $\mathcal{B}$ is complete and non-contradictory, by induction on the structure of formulas one can easily show that $\mathfrak{M}$ is as required. ■

Summing up, we obtain the following:

LEMMA 5.6. *$\mathcal{A}_{\mathsf{S4}_u}$ is a saturation rule for $(\mathcal{ST}, \mathsf{TR})$.*

$$
\begin{array}{c|c|c}
\multicolumn{3}{c}{k :: \neg(\varphi(x) \wedge \psi(x)) \qquad k \text{ is marked}} \\
\hline
 & k :: \neg\varphi(x) & \\
k :: \neg\varphi(x) & m :: \neg\psi(x) \quad \text{for } m \text{ new to } \mathcal{B} & k :: \neg\psi(x) \\
 & m :: \eta(x) \quad \text{for every } k :: \eta(x) \text{ on } \mathcal{B} &
\end{array}
$$

Figure 4. The disjunction rule ($l\vee^*$) for marked labels.

5.3. Constant domains

With minor modifications, the tableau algorithms presented in Sections 5.1 and 5.2 also give rise to exhaustive saturation rules for $(\mathcal{QL}^1, \mathsf{FO})$ and $(\mathcal{ST}, \mathsf{TR})$, respectively. Here we consider only the latter, more general case.

In the constant domain tableau algorithm, there exist two types of labels: marked and unmarked ones, where marked labels are always of length one (i.e. contain no dots). We assume that each input flock contains the distinguished type $\mathbf{t}_z(x) = \{\top(x)\}$. In the initial tableau for a flock $\mathbf{T}$, a marked label is used for $\mathbf{t}_z(x)$. All other labels in the initial tableau are unmarked. For the application of tableau rules, sentences and formulas with unmarked labels are treated precisely as in the expanding domain case.

The only difference for marked labels is that a modified version of the disjunction rule is used, which can be found in Fig. 4: if $k :: \neg(\varphi(x) \vee \psi(x))$ is found on a branch $\mathcal{B}$ with k marked, then we split the end of the branch *into three* and do the following: the left fork is extended with the labelled formula $k :: \neg\varphi(x)$, the right one with $k :: \psi(x)$, and the middle fork is extended with formulas

$$\{k :: \neg\varphi(x),\ m :: \neg\psi(x)\} \cup \{m :: \eta(x) \mid k :: \eta(x) \text{ is on } \mathcal{B}\},$$

where m is a new marked label of length 1 (a 'copy' of k). Intuitively, we are constructing a set of 'minimal types' corresponding to the marked labels as proposed in [16]: if $\mathcal{B}$ is a non-contradictory branch of the canonical tableau and $\mathfrak{M}$ a model with domain Δ such that $\mathfrak{M} \models \mathbf{T}_{\mathcal{B}}[\mathfrak{a}]$ for some $\mathbf{T}_{\mathcal{B}}$-assignment $\mathfrak{a}$, then for each $d \in \Delta$ we find a marked label σ on $\mathcal{B}$ such that $\mathfrak{M} \models \mathbf{t}^{\mathcal{B}}_{\sigma}[d]$.[1] This obviously corresponds to the 'exhaustiveness' property required by the strengthened completeness condition (CO$'$) for constant domains.

[1] The type $\mathbf{t}^{\mathcal{B}}_{\sigma}(x)$ is called a *minimal* type, since there is no type $\mathbf{t}(x) \in \mathbf{T}_{\mathcal{B}}$ such that $\mathbf{t}(x) \subset \mathbf{t}^{\mathcal{B}}_{\sigma}(x)$.

The result returned by the constant domain tableau algorithm $\mathcal{A}'_{\mathsf{S4}_u}$ is obtained from the canonical tableau in the very same way as for expanding domains. Let us now prove the constant domain completeness property (CO$'$):

LEMMA 5.7. *Let* $\mathbf{T}$ *be a flock. For every model* $\mathfrak{M} \in \mathsf{TR}$ *with domain* Δ*, every type* $\mathbf{t}(x) \in \mathbf{T}$*, and every* $\mathbf{T}$*-assignment* $\mathfrak{a}$ *in* Δ*, if* $\mathfrak{M} \models \mathbf{T}[\mathfrak{a}]$ *then there is a branch* $\mathcal{B} \in \mathfrak{B}_{\mathbf{T}}$ *such that*

1. $\mathbf{T}_{\mathcal{B}}$ *is exhaustive for* $\mathfrak{M}$*,*
2. *there exist a type* $\mathbf{t}'(x) \in \mathbf{T}_{\mathcal{B}}$ *and a* $\mathbf{T}_{\mathcal{B}}$*-assignment* $\mathfrak{a}'$ *in* Δ *such that* $\mathbf{t}(x) \subseteq \mathbf{t}'(x)$*,* $\mathfrak{a}(\mathbf{t}) = \mathfrak{a}'(\mathbf{t}')$*, and* $\mathfrak{M} \models \mathbf{T}_{\mathcal{B}}[\mathfrak{a}']$*.*

PROOF. Let $\mathbf{T}$ be a flock, $\mathfrak{M} = \langle \Delta, R^{\mathfrak{M}}, Q_0^{\mathfrak{M}}, \dots, q_0^{\mathfrak{M}}, \dots \rangle \in \mathsf{TR}$, and $\mathfrak{a}$ a $\mathbf{T}$-assignment in Δ such that $\mathfrak{M} \models \mathbf{T}[\mathfrak{a}]$. Suppose that the canonical tableau for $\mathbf{T}$ is the last tableau of the sequence

$$T_0, T_1, \dots, T_n,$$

where T_0 is the initial tableau for $\mathbf{T}$ and, for every $i < n$, T_{i+1} is obtained from T_i by an application of a tableau rule.

We define, for each tableau T_i, $0 \leq i \leq n$, a branch $\mathcal{B}_i$ of T_i, an $\mathbf{T}_{\mathcal{B}_i}$-assignment $\mathfrak{a}_i$, and a surjective map π_i from Δ to the set of marked labels on $\mathcal{B}_i$. Let $\mathcal{B}_0$ be the single branch of T_0. Set $\mathfrak{a}_0 = \mathfrak{a}$, and let π_0 be the function mapping every element of Δ to the single marked label on $\mathcal{B}_0$ (recall that $\top(x)$ is the only formula labelled by it). We proceed in such a way that the following conditions are satisfied:

$$\mathfrak{M} \models \mathbf{T}_{\mathcal{B}_i}[\mathfrak{a}_i], \qquad \mathfrak{a}_i(\mathbf{t}_1^{\mathcal{B}_i}) = \mathfrak{a}_{i-1}(\mathbf{t}_1^{\mathcal{B}_{i-1}}), \qquad \mathfrak{M} \models^{\mathfrak{a}_i} R^{\mathcal{B}_i},$$

and

$$\pi_i(d) = \sigma \quad \text{implies} \quad \mathfrak{M} \models \mathbf{t}_\sigma^{\mathcal{B}_i}[d] \quad \text{for every } d \in \Delta. \tag{ix}$$

Assume that we have already constructed $\mathcal{B}_i$, $\mathfrak{a}_i$, and π_i for $0 \leq i < n$, and T_{i+1} is obtained from T_i by an application of (l). All rules except $(l\vee^*)$ are treated as in Lemmas 5.1 and 5.4 with the addition that $\pi_{i+1} = \pi_i$ for any of these rules.

$(l\vee^*)$ is applied to $\sigma.k :: \neg(\varphi(x) \wedge \psi(x))$, where $\sigma.k$ is a marked label. The rule application splits $\mathcal{B}_i$ into three branches $\mathcal{B}_\varphi$, $\mathcal{B}_\psi$ and $\mathcal{B}_*$, where $\mathcal{B}_\varphi$ has a new node containing $\sigma.k :: \neg\varphi(x)$, $\mathcal{B}_\psi$ has a node containing $\sigma.k :: \neg\psi(x)$, and $\mathcal{B}_*$ has new nodes containing

$$\{\sigma.k :: \neg\varphi(x),\ \sigma.m :: \neg\psi(x)\} \cup \{\sigma.m :: \eta(x) \mid \sigma.k :: \eta(x) \text{ is on } \mathcal{B}_i\},$$

where $\sigma.m$ is a new marked label. Define two sets

$$\Delta_\varphi = \{d \in \Delta \mid \pi_i(d) = \sigma.k \text{ and } \mathfrak{M} \models \neg\varphi[d]\},$$
$$\Delta_\psi = \{d \in \Delta \mid \pi_i(d) = \sigma.k \text{ and } \mathfrak{M} \models \neg\psi[d]\}.$$

Due to the surjectivity of π_i, we have either $\Delta_\varphi \neq \emptyset$ or $\Delta_\psi \neq \emptyset$. So we have to consider three cases:

1. If $\Delta_\varphi = \emptyset$, then $\mathcal{B}_{i+1} = \mathcal{B}_\psi$.

2. If $\Delta_\psi = \emptyset$, then $\mathcal{B}_{i+1} = \mathcal{B}_\varphi$.

3. If $\Delta_\varphi \neq \emptyset$ and $\Delta_\psi \neq \emptyset$, then $\mathcal{B}_{i+1} = \mathcal{B}_*$, $\mathfrak{a}_{i+1}(\mathbf{t}^{\mathcal{B}_{i+1}}_{\sigma.m}) = \mathfrak{a}_i(\mathbf{t}^{\mathcal{B}_i}_{\sigma.k})$ and $\mathfrak{a}_{i+1}(\mathbf{t}^{\mathcal{B}_{i+1}}_\tau) = \mathfrak{a}_i(\mathbf{t}^{\mathcal{B}_i}_\tau)$ for every other $\mathbf{t}^{\mathcal{B}_{i+1}}_\tau(x) \in \mathcal{B}_{i+1}$, and

$$\pi_{i+1}(d) = \begin{cases} \sigma.k, & \text{if } d \in \Delta_\varphi, \\ \sigma.m, & \text{if } d \in \Delta_\psi \setminus \Delta_\varphi, \\ \pi_i(d), & \text{otherwise.} \end{cases}$$

Finally, in the first two cases we let $\pi_{i+1} = \pi_i$ and $\mathfrak{a}_{i+1}(\mathbf{t}^{\mathcal{B}_{i+1}}_\tau) = \mathfrak{a}_i(\mathbf{t}^{\mathcal{B}_i}_\tau)$ for every $\mathbf{t}^{\mathcal{B}_{i+1}}_\tau(x) \in \mathcal{B}_{i+1}$.

In the same way as in the proof of Lemma 5.4 we can use $\mathcal{B}_n$ and $\mathfrak{a}_n$ to fix a branch $\mathcal{B} = \mathcal{B}_n \in \mathfrak{B}_{\mathbf{T}}$, a type $\mathbf{t}' \in \mathbf{T}_{\mathcal{B}}$, and a $\mathbf{T}_{\mathcal{B}}$-assignment $\mathfrak{a}'$ such that condition 2 from the formulation of the lemma is satisfied. It remains to note that exhaustiveness of $\mathbf{T}_{\mathcal{B}}$ is obviously an immediate consequence of (ix). ■

Since soundness (SO) can be proved precisely as in the expanding domain case, we obtain the following:

LEMMA 5.8. *$\mathcal{A}'_{\mathsf{S4}_u}$ is an exhaustive saturation rule for $(\mathcal{ST}, \mathsf{TR})$.*

In general, it seems that all tableau algorithms which may serve as an (expanding domain) saturation rule can be converted into an exhaustive saturation rule by modifying all non-deterministic tableau rules in the way we modified the $(l\vee^*)$ rule: instead of considering each non-deterministic outcome separately, we must also consider arbitrary combinations of such outcomes. More details on this issue can be found in [16].

5.4. Temporal tableaux at work

In the following, we exemplarily apply the temporal tableau calculus from Section 4 to some $\mathcal{QTL}^1$-formulas using the tableau algorithm for $\mathcal{QL}^1$ as a saturation rule as shown in Section 5.1.

REMARK 5.9. Let ϑ be a $\mathcal{QTL}_{\boxed{1}}$-sentence and $\mathcal{G}$ a complete tableau for ϑ after execution of the elimination procedure. Then clearly no type in the tableau contains both $\overline{\bigcirc\varphi}(x)$ and $\overline{\bigcirc\neg\varphi}(x)$ (otherwise the node has no successor). On the other hand, it follows from $(\bigcirc\varphi)^{\neg}$ that every type of every saturated flock $\ell(s)$, where s is a state in $\mathcal{G}$, contains at least one of $\overline{\bigcirc\varphi}(x)$ and $\overline{\bigcirc\neg\varphi}(x)$, for every subformula $\bigcirc\varphi(x)$ of ϑ. So in the final (completed and pruned) tableaux we can identify $\overline{\bigcirc\varphi}(x)$ and $\neg\overline{\bigcirc\neg\varphi}(x)$ and consider only one of them, since the truth value of the other can easily be restored. Similarly, axiom $(\psi_1\,\mathcal{U}\,\psi_2)^{\neg}$ guarantees that every type in every saturated flock $\ell(s)$ contains precisely one of $\overline{\bigcirc(\psi_1\,\mathcal{U}\,\psi_2)}(x)$ and $\overline{\bigcirc\neg(\psi_1\,\mathcal{U}\,\psi_2)}(x)$, for every subformula $\psi_1\,\mathcal{U}\,\psi_2(x)$ of ϑ. So by the same argument we can identify $\overline{\bigcirc(\psi_1\,\mathcal{U}\,\psi_2)}(x)$ and $\neg\overline{\bigcirc\neg(\psi_1\,\mathcal{U}\,\psi_2)}(x)$.

EXAMPLE 5.10. Consider the formula

$$\Box\Big(\exists y\,\big(C(y)\wedge\neg\bigcirc C(y)\big)\wedge\forall y\,\big(\neg C(y)\to\bigcirc\neg C(y)\big)\Big),$$

from Example 2.8. As was shown above, we can identify $q_{\bigcirc(\top\mathcal{U}\neg\psi)}$ with $\neg q_{\bigcirc\neg(\top\mathcal{U}\neg\psi)}$ and $Q_{\bigcirc C}(x)$ with $\neg Q_{\bigcirc\neg C}(x)$ (and consider only one representative of each pair). This is done to simplify tableau in the example by avoiding construction of dead ends. Then the set $\mathbf{Ax}_\vartheta(x)$ of surrogate axioms consists of the following formulas:

$$\begin{gathered}\top(x),\\ q_{\top\mathcal{U}\neg\psi}\to\neg\exists y\,\big(C(y)\wedge\neg Q_{\bigcirc C}(y)\big)\vee\neg\forall y\,\big(C(y)\vee\neg Q_{\bigcirc C}(y)\big)\vee q_{\bigcirc(\top\mathcal{U}\neg\psi)},\\ \neg q_{\top\mathcal{U}\neg\psi}\to\exists y\,\big(C(y)\wedge\neg Q_{\bigcirc C}(y)\big)\wedge\forall y\,\big(C(y)\vee\neg Q_{\bigcirc C}(y)\big)\wedge\neg q_{\bigcirc(\top\mathcal{U}\neg\psi)},\\ \neg q_{\bigcirc(\top\mathcal{U}\neg\psi)}\to\neg q_{\bigcirc(\top\mathcal{U}\neg\psi)},\\ \forall x\,\big(\neg Q_{\bigcirc C}(x)\to\neg Q_{\bigcirc C}(x)\big).\end{gathered}$$

We begin constructing a tableau for ϑ with a state s_r such that

$$\ell(s_r)=\emptyset,\quad \ell_{\bigcirc}(s_r)=\mathbf{T}'_0\quad\text{and}\quad\mathbf{T}'_0=\{\mathbf{t}'_0(x),\ \mathbf{Ax}_\vartheta(x)\}.$$

The flock $\mathbf{T}'_0$ consists of only two types, namely, $\mathbf{t}'_0(x)=\{\neg q_{\top\mathcal{U}\neg\psi},\top(x)\}$ and $\mathbf{Ax}_\vartheta(x)$. As a saturation rule we use the tableau procedure for the one-variable fragment from Section 5.1. The complete tableau for $\mathbf{T}'_0$ contains

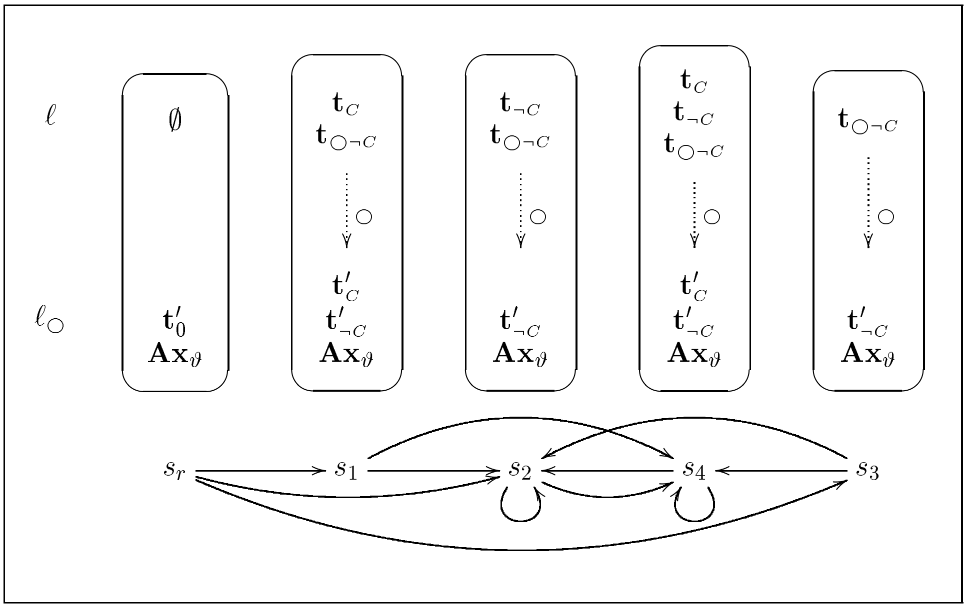


Figure 5. Complete tableau for Example 2.8.

then three non-contradictory branches, each of which represents a class of models for $\mathbf{T}'_0$. These branches give us the following saturated flocks

$$\mathbf{T}_1 = \{\mathbf{t}_{\bigcirc\neg C}(x),\ \mathbf{t}_C(x)\}, \quad \mathbf{T}_2 = \{\mathbf{t}_{\bigcirc\neg C}(x),\ \mathbf{t}_{\neg C}(x)\}, \quad \mathbf{T}_3 = \{\mathbf{t}_{\bigcirc\neg C}(x)\}$$

consisting of three distinct types

$$\begin{aligned} \mathbf{t}_{\bigcirc\neg C}(x) &= \Gamma_0(x) \cup \{C(x),\ \neg Q_{\bigcirc C}(x),\ C(x) \wedge \neg Q_{\bigcirc C}(x)\}, \\ \mathbf{t}_C(x) &= \Gamma_0(x) \cup \{C(x),\ Q_{\bigcirc C}(x)\}, \\ \mathbf{t}_{\neg C}(x) &= \Gamma_0(x) \cup \{\neg C(x),\ \neg Q_{\bigcirc C}(x)\}, \end{aligned}$$

where $\Gamma_0(x) = \mathbf{Ax}_\vartheta(x) \cup \{\neg q_{\top\mathcal{U}\neg\psi},\ \neg q_{\bigcirc(\top\mathcal{U}\neg\psi)}\}$. Thus, the result of saturation is

$$\mathcal{A}(\mathbf{T}'_0) = \{\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3\},$$

and so we create three new states s_1, s_2 and s_3 labelled by $\mathbf{T}_1$, $\mathbf{T}_2$ and $\mathbf{T}_3$, respectively.

Now we take one step in time and obtain $\mathcal{N}(\mathbf{T}_i) = \mathbf{T}'_i$, for $i = 1, 2, 3$, with

$$\begin{aligned}
\mathbf{T}_1' &= \left\{\mathbf{t}_{\neg C}'(x),\ \mathbf{t}_C'(x),\ \mathbf{Ax}_\vartheta(x)\right\},\\
\mathbf{T}_2' &= \left\{\mathbf{t}_{\neg C}'(x),\ \mathbf{Ax}_\vartheta(x)\right\},\\
\mathbf{T}_3' &= \left\{\mathbf{t}_{\neg C}'(x),\ \mathbf{Ax}_\vartheta(x)\right\},
\end{aligned}$$

where $\mathbf{t}_{\neg C}'(x) = \{\neg q_{\top\mathcal{U}\neg\psi},\ \neg C(x),\ \top(x)\}$, $\mathbf{t}_C'(x) = \{\neg q_{\top\mathcal{U}\neg\psi},\ C(x),\ \top(x)\}$.

An application of the saturation rule to the flocks $\mathbf{T}_1'$, $\mathbf{T}_2'$ and $\mathbf{T}_3'$ gives

$$\mathcal{A}(\mathbf{T}_1') = \{\mathbf{T}_2, \mathbf{T}_4\},\ \mathcal{A}(\mathbf{T}_2') = \{\mathbf{T}_2, \mathbf{T}_4\} \text{ and } \mathcal{A}(\mathbf{T}_3') = \{\mathbf{T}_2, \mathbf{T}_4\},$$

where $\mathbf{T}_4 = \{\mathbf{t}_{\bigcirc\neg C}(x),\ \mathbf{t}_{\neg C}(x),\ \mathbf{t}_C(x)\}$. As in the tableau we already have a state, s_2, labelled by $\mathbf{T}_2$, we create one new state s_4 and label it by $\mathbf{T}_4$. Having taken the second step in time, we obtain $\mathcal{N}(\mathbf{T}_4) = \mathbf{T}_4'$, where

$$\mathbf{T}_4' = \left\{\mathbf{t}_{\neg C}'(x),\ \mathbf{t}_C'(x),\ \mathbf{Ax}_\vartheta(x)\right\}.$$

An application of the saturation rule to $\mathbf{T}_4'$ gives no new states, so this step completes the tableau. Wolper's elimination rules will not reduce the number of states, since our formula contains no eventualities. The resulting tableau is depicted on Fig. 5.

EXAMPLE 5.11. Consider now the formula $\vartheta = \forall y \bigcirc\neg C(y) \wedge \bigcirc\exists y\, C(y)$. Its first-order reduct is $\overline{\vartheta} = \forall y\, Q_{\bigcirc\neg C}(y) \wedge q_{\bigcirc\exists y C}$ and the set $\mathbf{Ax}_\vartheta(x)$ of surrogate axioms consists of three formulas (modulo the simplifications above):

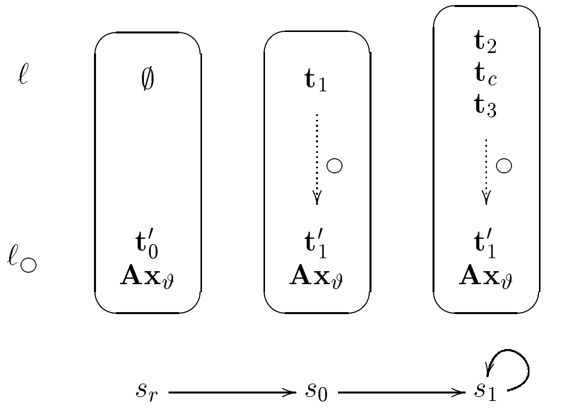
$$\begin{gathered}
\top(x),\\
\forall x\, (\neg Q_{\bigcirc\neg C}(x) \to \neg Q_{\bigcirc\neg C}(x)),\\
\neg q_{\bigcirc\exists y C} \to \neg q_{\bigcirc\exists y C}.
\end{gathered}$$

In the case of expanding domains (using again the saturation rule for the one-variable fragment from Section 5.1) we obtain then the infinite path of the complete temporal tableau for ϑ (see Fig. 5.4). It should be remarked that the complete tableau contains more than 30 states, however the displayed path is enough to construct a quasimodel satisfying ϑ.

In the case of constant domains, the type $\mathbf{Ax}_\vartheta(x)$ in s_0 contains the formula

$$\mathbf{CDA}_\mathbf{T} = \forall x\, (\neg C(x) \wedge \exists y\, C(y)),$$

which is clearly not satisfiable. Therefore, s_0 has no successors, and the elimination procedure removes both s_r and s_0, so that the resulting tableau is empty. By Theorem 4.14, ϑ is not satisfiable in constant domains.



$$
\begin{aligned}
\mathbf{t}_0'(x) &= \{\overline{\vartheta},\ \top(x)\}\\
\mathbf{t}_1(x) &= \{\overline{\vartheta},\ \forall y\, Q_{\bigcirc\neg C}(y),\ Q_{\bigcirc\neg C}(x),\ q_{\bigcirc\exists y C}, \top(x)\}\\
\mathbf{t}_1'(x) &= \{\neg C(x),\ \exists y\, C(y),\ \top(x)\}\\
\mathbf{t}_2(x) &= \{\neg C(x),\ \exists y\, C(y),\ Q_{\bigcirc\neg C}(x),\ q_{\bigcirc\exists y C},\ \top(x)\}\\
\mathbf{t}_c(x) &= \{C(x),\ \exists y\, C(y),\ Q_{\bigcirc\neg C}(x),\ q_{\bigcirc\exists y C},\ \top(x)\}\\
\mathbf{t}_3(x) &= \{\exists y\, C(y),\ Q_{\bigcirc\neg C}(x),\ q_{\bigcirc\exists y C},\ \top(x)\}
\end{aligned}
$$

Figure 6. An infinite path in the tableau for $\forall y \bigcirc \neg C(y) \wedge \bigcirc \exists y\, C(y)$.

6. Conclusion

We have presented a general framework for constructing tableau algorithms for monodic fragments of first-order temporal logic from Wolper's tableau algorithm for PTL and decision procedures for fragments of first-order logic. In both the expanding domain and the constant domain case, we can use existing decision procedures for first-order fragments. However, for constant domains we need more than a single application of the algorithm.

As example instantiations of our framework, we have developed tableau algorithms for the one-variable fragment of monodic FOTL and for the temporalisation of the modal logic $\mathsf{S4}_u$. These logics are sufficiently simple to serve as examples but also have some rather serious applications:

- The tableau system for the one-variable fragment $\mathcal{QTL}^1$ of FOTL can be used for various spatio-temporal reasoning tasks, see [27] for an embedding of spatio-temporal logics in this fragment.

- In the case of constant domains, the tableau for $\mathcal{QTL}^1$ actually yields a tableau decision procedure for the Cartesian product of propositional linear temporal logic PTL and S5 (see e.g. [7]).
- The tableau system for the temporalised $\mathsf{S4}_u$ can be generalised in a straightforward way to tableaux for various temporal description logics (see, e.g., [24, 18, 15]).

It should be obvious that the presented framework can also be used to develop tableau algorithms for more powerful fragments of monodic FOTL such as the monodic two-variable fragment and the monodic guarded fragment.

Acknowledgements

The work of the first and fourth author was partially supported by UK EPSRC grant GR/R45369/01 "Analysis and mechanisation of decidable first-order temporal logics." The work of the second author was partially supported by Deutsche Forschungsgemeinschaft (DFG) grant Ba1122/3-1. The work of the third author was partially supported by Deutsche Forschungsgemeinschaft (DFG) grant Wo583/3-1.

References

[1] A. Degtyarev and M. Fisher. Towards first-order temporal resolution. In F. Baader, G. Brewka, and T. Eiter, editors, *Advances in Artificial Intelligence (KI'2001)*, volume 2174 of LNAI, pages 18–32. Springer-Verlag, 2001.

[2] A. Degtyarev, M. Fisher, and A. Lisitsa. Equality and monodic first-order temporal logic. *Studia Logica*, 72(2):147–156, 2002.

[3] A. Degtyarev, M. Fisher, and B. Konev. Monodic temporal resolution. Submitted, 2003. Available as Technical report ULCS-03-001 from `http://www.csc.liv.ac.uk/research/techreports`.

[4] M. Fitting. *Proof Methods for Modal and Intuitionistic Logics*. Reidel, Dordrecht, 1983.

[5] M. Fitting and R. Mendelson. *First-Order Modal Logic*. Kluwer Academic Publishers, Dordrecht, 1998.

[6] D. Gabbay, I. Hodkinson, and M. Reynolds. *Temporal Logic: Mathematical Foundations and Computational Aspects, Volume 1*. Oxford University Press, 1994.

[7] D. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-Dimensional Modal Logics: Theory and Applications*. Elsevier, 2003.

[8] V. Goranko and S. Passy. Using the universal modality: Gains and questions. *Journal of Logic and Computation*, 2:5–30, 1992.

[9] R. Goré. Tableau algorithms for modal and temporal logic. In D'Agostino et al., editors, *Handbook of Tableau Methods*. Kluwer Academic Publishers, Dordrecht, 1999.

[10] I. Hodkinson. Monodic packed fragment with equality is decidable. *Studia Logica*, 72(2):185–197, 2002.

[11] I. Hodkinson, F. Wolter, and M. Zakharyaschev. Decidable fragments of first-order temporal logics. *Annals of Pure and Applied Logic*, 106:85–134, 2000.

[12] I. Hodkinson, F. Wolter, and M. Zakharyaschev. Monodic fragments of first-order temporal logics: 2000–2001 A.D. In *Logic for Programming, Artificial Intelligence and Reasoning*, volume 2250 of LNAI, pages 1–23. Springer-Verlag, 2001.

[13] G.E. Hughes and M.J. Cresswell. *A New Introduction to Modal Logic*. Methuen, London, 1996.

[14] S.A. Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94, 1963.

[15] C. Lutz, H. Sturm, F. Wolter, and M. Zakharyaschev. Tableaux for temporal description logic with constant domain. In R. Goré, A. Leitsch, and T. Nipkow, editors, *Proceedings of the First International Joint Conference on Automated Reasoning (IJCAR'01)*, volume 2083 of LNAI, pages 121–136. Springer-Verlag, 2001.

[16] C. Lutz, H. Sturm, F. Wolter, and M. Zakharyaschev. A tableau decision algorithm for modalized $\mathcal{ALC}$ with constant domains. *Studia Logica*, 72(2):199–232, 2002.

[17] K. Schild. Combining terminological logics with tense logic. In Miguel Filgueiras and Luís Damas, editors, *Progress in Artificial Intelligence – 6th Portuguese Conference on Artificial Intelligence, EPIA'93*, volume 727 of LNAI, pages 105–120. Springer-Verlag, 1993.

[18] H. Sturm and F. Wolter. A tableau calculus for temporal description logic: The expanding domain case. *Journal of Logic and Computation*, 2002.

[19] M. Wajsberg. Ein erweiterter Klassenkalkül. *Monatsh Math. Phys.*, 40:113–126, 1933.

[20] P. Wolper. The tableau method for temporal logic: An overview. *Logique et Analyse*, 28:119–152, 1985.

[21] F. Wolter and M. Zakharyaschev. Satisfiability problem in description logics with modal operators. In A. Cohn, L. Schubert, and S. Shapiro, editors, *KR'98: Principles of Knowledge Representation and Reasoning*, pages 512–523. Morgan Kaufmann Publishers, 1998.

[22] F. Wolter and M. Zakharyaschev. Multi-dimensional description logics. In D. Thomas, editor, *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 104–109. Morgan Kaufmann Publishers, 1999.

[23] F. Wolter and M. Zakharyaschev. Dynamic description logic. In K. Segerberg, M. de Rijke, H. Wansing, and M. Zakharyaschev, editors, *Advances in Modal Logic, Volume 2*, pages 431–445. CSLI Publications, 2000.

[24] F. Wolter and M. Zakharyaschev. Temporalizing description logics. In D. Gabbay and M. de Rijke, editors, *Frontiers of Combining Systems II*, pages 379–401. Studies Press/Wiley, 2000.

[25] F. Wolter and M. Zakharyaschev. Decidable fragments of first-order modal logics. *Journal of Symbolic Logic*, 66:1415–1438, 2001.

[26] F. Wolter and M. Zakharyaschev. Axiomatizing the monodic fragment of first-order temporal logic. *Annals of Pure and Applied Logic*, 118(1–2):133–145, 2002.

[27] F. Wolter and M. Zakharyaschev. Qualitative spatio-temporal representation and reasoning: a computational perspective. In *Exploring Artificial Intelligence in the New Millenium*, pages 175–215. Morgan Kaufmann Publishers, 2002.

Roman Kontchakov
Department of Computer Science
King's College London
Strand
London WC2R 2LS, U.K.
`romanvk@dcs.kcl.ac.uk`

Carsten Lutz
Institut für Theoretische Informatik
TU Dresden, Fakultät Informatik
01062 Dresden, Germany
`lutz@tcs.inf.tu-dresden.de`

Frank Wolter
Department of Computing
Universty of Liverpool
Liverpool L69 7ZF, U.K.
`frank@csc.liv.ac.uk`

Michael Zakharyaschev
Department of Computer Science
King's College London
Strand
London WC2R 2LS, U.K.
`mz@dcs.kcl.ac.uk`