Temporal logics over transitive states

Boris Konev[1], Frank Wolter[1], and Michael Zakharyaschev[2]

[1] Department of Computer Science
University of Liverpool
Liverpool L69 7ZF, U.K.
{b.konev,frank@csc.liv.ac.uk}
[2] Department of Computer Science
King's College London,
Strand, London WC2R 2LS, U.K.
mz@dcs.kcl.ac.uk

Abstract. We investigate the computational behaviour of 'two-dimensional' propositional temporal logics over $(\mathbb{N}, <)$ (with and without the next-time operator $\bigcirc$) that are capable of reasoning about states with transitive relations. Such logics are known to be undecidable (even Π^1_1-complete) if the domains of states with those relations are assumed to be constant. Motivated by applications in the areas of temporal description logic and specification & verification of hybrid systems, in this paper we analyse the computational impact of allowing the domains of states to expand. We show that over finite expanding domains (with an arbitrary, tree-like, quasi-order, or linear transitive relation) the logics are recursively enumerable, but undecidable. If these finite domains eventually become constant then the resulting $\bigcirc$-free logics are decidable (but not in primitive recursive time); on the other hand, when equipped with $\bigcirc$ they are not even recursively enumerable. Finally, we show that temporal logics over infinite expanding domains as above are undecidable even for the language with the sole temporal operator 'eventually.' The proofs are based on Kruskal's tree theorem and reductions of reachability problems for lossy channel systems.

1 Introduction

Temporal logics are used in computer science and artificial intelligence to model states (of soft or hardware, data or knowledge bases, spatial regions, multi-agent systems, etc.) changing over time. (For uniformity, we can think of such states as first- or higher-order structures of some fixed signature.) Perhaps the best known example is LTL, the propositional linear temporal logic of infinite sequences $\sigma_0\sigma_1\dots$ of states, equipped with temporal operators like $\Box_F$ 'always in the future' or $\bigcirc$ 'at the next state.' LTL is decidable in PSPACE [28], reasoning with this logic can be mechanised using tableaux [30] or resolution [6], with the existing provers performing reasonably well [15, 27].

However, being *propositional*, LTL is only capable of reasoning about states of some *fixed finite size* which must be known in advance. This restriction seriously limits the scope of applications of LTL in areas where *infinite* or *arbitrarily finite* states are required. Typical examples of such applications are:

– verification and specification of 'infinite state systems' such as real-time systems, hybrid (dynamical) systems, broadcast protocols, and channel systems;
– spatio-temporal representation and reasoning in artificial intelligence (where the states modelling space are usually either unbounded or infinite);
– temporal data or knowledge bases, e.g., 'dynamic' ontologies or temporal entity relationship models (where states are finite, but one cannot impose *a priory* any upper bound on their size);
– distributed multi-agent systems.

The obvious idea to cope with unbounded states by means of 'upgrading' propositional temporal logic to first-order one is extremely expensive: first-order LTL is not recursively enumerable (in fact, it is Π_1^1-hard; see, e.g., [7, 8]), and so we cannot even have a semi-decision procedure.

The attempts to 'tame' first-order temporal logic in the fields of temporal data and knowledge bases, multi-agent systems and spatio-temporal representation and reasoning have led to *semi-decidable* and *decidable* fragments that can be obtained by imposing certain independence and locality restrictions.

The monodic fragment of first-order LTL allows applications of temporal operators to formulas with at most one free variable [14]. Thus, in the framework of this fragment we can only control the temporal change of properties—i.e., *unary predicates*—of states, while binary, ternary, etc. relations can change *arbitrarily*. The full monodic fragment turns out to be semi-decidable [32], and if we restrict the first-order part to a decidable fragment (for example, to the two-variable or guarded fragments), then the resulting monodic fragment is usually decidable as well. The simplest interesting fragment of this sort is the one-variable first-order LTL (Sistla and German [29] considered it in the context of verification). Various spatio-temporal logics based on spatial formalisms like $\mathcal{RCC}$-8, $\mathcal{BRCC}$-8, etc. can be encoded in the one-variable first-order temporal logic [8, 9] and therefore inherit its good computational properties. Monodic fragments of this kind are usually decidable in *elementary* time [8], and both tableau- and resolution-based provers have been developed and implemented for monodic temporal logics [19, 17, 16].

The idea of monodicity is based on two conditions: the 'positive'

Mono$^+$: temporal constraints can be imposed on unary predicates

and the 'negative'

Dya$^-$: no temporal constraint can be imposed on n-nary predicates for $n \geq 2$.

Having in mind possible applications of temporal logic mentioned above, condition **Dya$^-$** appears too restrictive. In fact, in temporal knowledge bases (say, temporal description logics), more sophisticated spatio-temporal formalisms, in particular, dynamic topological logics (designed for reasoning about safety and liveness properties of hybrid systems), or infinite state systems we *do need* to control *binary relations*, for instance, to ensure that some of them do not change in time or can only expand.

Thus, we are facing the problem of weakening $\mathbf{Dya}^-$ without compromising too much the good computational behaviour of the monodic fragment.

Some limits for such a weakening are well-known. For example, one cannot simply replace $\mathbf{Dya}^-$ with $\mathbf{Dya}^+$, for $n = 2$, because even the monadic two-variable fragment of first-order temporal logic with *one constant binary relation* is not recursively enumerable. So it seems that without imposing extra constraints on the language no weakening of $\mathbf{Dya}^-$ can result in new and interesting decidable temporal logics.

$\mathbf{Mono}^+$ and locality. The strongest existing *decidable* temporal logics that are capable of controlling binary relations of unbounded states replace $\mathbf{Dya}^-$ with some *locality conditions* which can be characterised as follows:

(1) over time, binary relations can be constant or expanding, or can change arbitrarily,
(2) within states, these binary relations can satisfy some *local* constraints like reflexivity, symmetry, the triangle inequality (for metric), but not transitivity,
(3) the language referring to binary relations is *local* in the sense that we are only allowed to quantify along these relations like in modal or description logic.

Basically, conditions (2)–(3) mean that every satisfiable formula φ of our language can be satisfied in a model where the length of any strict path of the form $x_0Rx_1 \dots Rx_n$ is bounded by an elementary function depending on φ.

The resulting formalisms can be regarded as extensions of propositional LTL with propositional modal-like (or description logic) operators over states. Typical examples are temporal description logics [25, 31] (where the states are described by the standard $\mathcal{ALC}$ and its extensions), temporal epistemic logics [12, 8] (with state languages like $\mathbf{S5}_m$), and a number of temporal metric logics [18]. The satisfiability problem for such logics is often *non-elementary* (but primitive recursive).

It is worth noting that by adding to the language a *non-local* state operator (such as the universal modality) we immediately obtain an undecidable logic.

Transitive states. The most important example of a non-local constraint on binary relations is *transitivity* which occurs naturally in almost all the examples mentioned above: words in the channel of a channel system are linearly ordered (and therefore based on a transitive structure), expressive description logics allow transitive relations to model, e.g., the part-of relation, quasi-ordered structures representing topological spaces are transitive, common knowledge operators in epistemic logics are interpreted by transitive relations.

Unfortunately, *even a single transitive relation* which does not change over time (and interprets the 'modal' operator from (3) in the standard way) leads to an undecidable (even Π^1_1-complete) temporal logic [8, 11]. This also holds true if we impose on the transitive relation some extra conditions like linearity, reflexivity, being a tree, etc. Undecidability strikes even for the language with sole temporal operator $\Diamond_F$ (without next-time or until) and even if we are only interested in safety properties (that is, interpret the language not over $\mathbb{N}$ but over arbitrary finite initial segments of $\mathbb{N}$).

The proofs of these 'negative' results heavily use the *constant domain assumption* according to which the domains of all states coincide—and, therefore, if uRv holds in some state (where R is the transitive relation and u, v are some state elements) then uRv must hold in *all* states. Being quite natural in many cases, this constant domain assumption may become inadequate for some other applications. For example, in temporal data and knowledge bases new objects may be created, which gives us states with *expanding domains*: that uRv holds in some state σ only means that it also holds in all *subsequent* states, while before σ elements u and v may not exist. Similarly, topological dynamic systems with continuous functions give rise to states with expanding domains [21, 18]. And constructive logics like first-order intuitionistic logic can only be interpreted in models with expanding domains.

It is known that logics with expanding domains are reducible to logics with constant domains; see, e.g., [8]. A major open problem was whether the former can actually have better computational properties than the latter. A partial affirmative answer (for logics with finite flows of time) was obtained in [10]. Here we investigate this problem in full generality.

We show that over finite expanding domains (with an arbitrary, tree-like, quasi-order, or linear transitive relation) the logics are recursively enumerable, but undecidable. If these finite domains eventually become constant then the resulting $\bigcirc$-free logics are decidable (but not in primitive recursive time); on the other hand, when equipped with $\bigcirc$ they are not even recursively enumerable. (Decidability can also be recovered for full LTL if we consider only safety properties, that is, models with finite flows of time [10].) Finally, we show that temporal logics over infinite expanding domains as above are undecidable even for the language with the sole temporal operator 'eventually.' The proofs are based on Kruskal's tree theorem and reductions of reachability problems for lossy channel systems.

2 Temporal models with expanding domains

We begin by introducing the intended semantics for our temporal language discussed above. The only *flow of time* we deal with in this paper is $(\mathbb{N}, <)$. *States* are first-order structures with one transitive binary relation and countably many unary predicates. More precisely, let $\mathfrak{S}$ be a function which associates with every $x \in \mathbb{N}$ a structure

$$\mathfrak{S}(x) = (W_x, R_x, P^1_x, P^2_x, \dots) \tag{1}$$

where W_x is a nonempty set, $R_x \subseteq W_x \times W_x$, and $P_x^i \subseteq W_x$ for all i. We will call $\mathfrak{S}$ a *temporal model with expanding domains*, or an *e-model*, for short, if it satisfies the following conditions: whenever $x < y$ then

- $W_x \subseteq W_y$ and
- for all $u, v \in W_x$, we have uR_xv iff uR_yv.

(see Fig. 1).

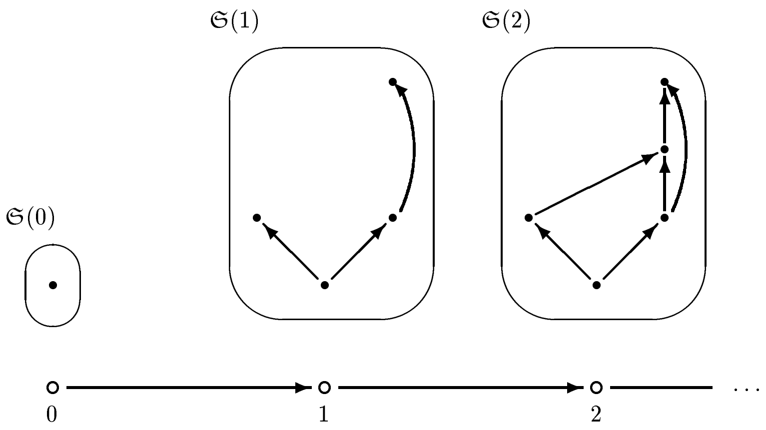

Fig. 1. An e-model $\mathfrak{S}$.

We consider two propositional languages $\mathcal{TL}$ and $\mathcal{TL}_\bigcirc$ for speaking about e-models. The former contains the *temporal operator* $\Diamond_F$ (and its dual $\Box_F$), the modal diamond $\diamondsuit$ (and its dual box $\boxdot$) interpreted over the binary relations in the states, as well as *state variables* (unary predicates) $p_1, p_2, \dots$ and the Booleans. $\mathcal{TL}_\bigcirc$ extends this language with the *next-time* operator $\bigcirc$. Thus, the formulas φ of $\mathcal{TL}_\bigcirc$ can be defined by taking

$$\varphi \quad ::= \quad p_i \quad | \quad \neg\varphi \quad | \quad \varphi_1 \wedge \varphi_2 \quad | \diamondsuit\varphi \quad | \quad \Diamond_F\varphi \quad | \quad \bigcirc\varphi$$

To simplify inductive proofs, we do not include in the language $\mathcal{TL}_\bigcirc$ the 'until' operator $\mathcal{U}$. As there is a satisfiability preserving reduction of formulas with $\mathcal{U}$ to $\mathcal{TL}_\bigcirc$-formulas (see, e.g., [5] or Section 7 below), all our results for $\mathcal{TL}_\bigcirc$ hold for the language with $\mathcal{U}$ as well.

Given an e-model $\mathfrak{S}$ of the form (1), $x \in \mathbb{N}$ and $u \in W_x$, we define the *truth relation* $\mathfrak{S}, (x, u) \models \varphi$ (or simply $(x, u) \models \varphi$, if understood) inductively as follows:

- $(x, u) \models p_i$ iff $u \in P_x^i$,
- $(x, u) \models \diamondsuit\psi$ iff there exists $v \in W_x$ such that uR_xv and $(x, v) \models \psi$,

- $(x,u) \models \Diamond_F \psi$ iff there exists $y \in \mathbb{N}$ such that $x < y$ and $(y,u) \models \psi$,
- $(x,u) \models \bigcirc\psi$ iff $(x+1,u) \models \psi$,

plus the standard clauses for the Boolean connectives.

We say that φ is *satisfied* in $\mathfrak{S}$ if $(x,u) \models \varphi$ for some $x \in \mathbb{N}$ and $u \in W_x$; φ is *valid* in $\mathfrak{S}$ ($\mathfrak{S} \models \varphi$, in symbols) if $(x,u) \models \varphi$ holds for every pair (x,u) with $u \in W_x$. If all formulas from a set $\Sigma \subseteq \mathcal{L}_{\bigcirc}$ are valid in $\mathfrak{S}$ then we write $\mathfrak{S} \models \Sigma$.

In this paper, we consider the following classes of e-models $\mathfrak{S}$ of the form (1):

- $\mathcal{A}$, the class of all e-models,
- $\mathcal{QO}$, the class of e-models with quasi-ordered states, that is, each R_x is transitive and reflexive,
- $\mathcal{T}$, the class of e-models $\mathfrak{S}$ where each (W_x, R_x) is a tree,
- $\mathcal{L}$, the class of e-models $\mathfrak{S}$ where each (W_x, R_x) is a strict linear order.
- the subclasses $\mathcal{A}_{\mathsf{fin}}$, $\mathcal{QO}_{\mathsf{fin}}$, $\mathcal{T}_{\mathsf{fin}}$, $\mathcal{L}_{\mathsf{fin}}$ of the classes above that only have *finite* states,
- the subclasses $\mathcal{C}^{\mathsf{c}}$ of the above classes $\mathcal{C}$ containing only models with *eventually constant* domains in the sense that there exists $n \in \mathbb{N}$ such that $(W_x, R_x) = (W_n, R_n)$ for all $x \geq n$.

Let $\mathcal{C}$ be one of the classes of e-models defined above. Our goal is to investigate the computational properties of the logics

$$\mathsf{Log}\mathcal{C} \quad = \quad \{\varphi \in \mathcal{TL} \mid \forall \mathfrak{S} \in \mathcal{C} \;\; \mathfrak{S} \models \varphi\}$$

and

$$\mathsf{Log}_{\bigcirc}\mathcal{C} \quad = \quad \{\varphi \in \mathcal{TL}_{\bigcirc} \mid \forall \mathfrak{S} \in \mathcal{C} \;\; \mathfrak{S} \models \varphi\}.$$

Our starting point is the results from [11, 8] according to which the corresponding logics under the *constant domain assumption* are not recursively enumerable, with some of them being actually Π^1_1-complete. By allowing models with expanding domains, we hope to obtain more positive results.

Our hopes are not groundless. We will use Kruskal's tree theorem to prove the following:

Theorem 1. *Let $\mathcal{C} \in \{\mathcal{A}_{\mathsf{fin}}, \mathcal{QO}_{\mathsf{fin}}, \mathcal{T}_{\mathsf{fin}}, \mathcal{L}_{\mathsf{fin}}\}$. Then $\mathsf{Log}_{\bigcirc}\mathcal{C}$ (and therefore $\mathsf{Log}\mathcal{C}$) is recursively enumerable.*

It remains an open problem whether the same can be proved for the corresponding classes of models with not necessarily finite states. However, none of these logics is decidable:

Theorem 2. *Let $\mathcal{C} \in \{\mathcal{A}, \mathcal{QO}, \mathcal{T}, \mathcal{L}, \mathcal{A}_{\mathsf{fin}}, \mathcal{QO}_{\mathsf{fin}}, \mathcal{T}_{\mathsf{fin}}, \mathcal{L}_{\mathsf{fin}}\}$. Then $\mathsf{Log}\mathcal{C}$ (and therefore $\mathsf{Log}_{\bigcirc}\mathcal{C}$) is undecidable.*

This result is proved by encoding the undecidable ω-reachability problem for lossy channel systems (see below for definitions).

Consider now the impact of the assumption that eventually the states are constant. In this case we reveal a crucial difference between full LTL and LTL with sole temporal operator $\Diamond_F$:

Theorem 3. *Let* $\mathcal{C} \in \{\mathcal{A}^{\mathsf{c}}_{\mathsf{fin}}, \mathcal{QO}^{\mathsf{c}}_{\mathsf{fin}}, \mathcal{T}^{\mathsf{c}}_{\mathsf{fin}}, \mathcal{L}^{\mathsf{c}}_{\mathsf{fin}}\}$. *Then*
(i) $\mathsf{Log}\mathcal{C}$ *is decidable (but not in primitive recursive time), while*
(ii) $\mathsf{Log}_{\bigcirc}\mathcal{C}$ *is not recursively enumerable.*

The proofs are based on Kruskal's tree theorem, a reduction of the non-primitive recursive reachability problem for lossy channel systems, and a reduction of the undecidable Post correspondence problem (PCP).

Below we only present the proofs of these theorems for the class $\mathcal{L}_{\mathsf{fin}}$ of finite strict linear orders. It is not completely trivial to extend these proofs to arbitrary transitive structures or quasi-orders. For instance, to deal with branching and/or reflexive states, constructions from [10] should be combined with the techniques introduced in the present paper. Also, the applications of Higman's lemma [13] we use here have to be replaced by the corresponding applications of Kruskal's tree theorem [22]. To prove Theorem 3 (ii) in full generality, the undecidable 'master problem' used in this paper (reachability for non-lossy channel systems) should be replaced by PCP as in [8, 18].

3 Recursive enumerability

In this section we prove the following:

Theorem 4. $\mathsf{Log}_{\bigcirc}\mathcal{L}_{\mathsf{fin}}$ *is recursively enumerable.*

Given a $\mathcal{TL}_{\bigcirc}$-formula φ, let $sub\,\varphi$ be the set of all subformulas of φ and their negations. Denote by $\boldsymbol{T}_\varphi$ the set of *Boolean types* $\boldsymbol{t}$ over $sub\,\varphi$, where

- $\neg\psi \in \boldsymbol{t}$ iff $\psi \notin \boldsymbol{t}$, for every $\neg\psi \in sub\,\varphi$, and
- $\chi \wedge \psi \in \boldsymbol{t}$ iff $\chi \in \boldsymbol{t}$ and $\psi \in \boldsymbol{t}$, for every $\chi \wedge \psi \in sub\,\varphi$.

A $\boldsymbol{T}_\varphi$*-word* $\mathfrak{T} = \langle T, <, l\rangle$ is a finite strict linear order $\langle T, <\rangle$ with a *labelling function* l which assigns to every $u \in T$ a type $l(u) \in \boldsymbol{T}_\varphi$. A $\boldsymbol{T}_\varphi$-word $\mathfrak{T} = \langle T, <, l\rangle$ is said to be *coherent* if, for every $\diamondsuit\psi \in sub\,\varphi$ and every $u \in T$, we have $\diamondsuit\psi \in l(u)$ iff there exists a $v \in T$ such that $u < v$ and $\psi \in l(v)$.

Consider a function $\mathfrak{f}$ associating with every natural number x a coherent $\boldsymbol{T}_\varphi$-word $\mathfrak{f}(x) = \langle T_x, <_x, l_x\rangle$. A *run* r through $\mathfrak{f}$ is a function with the domain

$$\mathrm{dom}\, r = \{k \in \mathbb{N} \mid k \geq m\},$$

for some $m \in \mathbb{N}$, such that $r(x) \in T_x$ for all $x \in \mathrm{dom}\, r$ and

- for every $x \in \mathrm{dom}\, r$ and every $\Diamond_F\psi \in sub\,\varphi$, we have $\Diamond_F\psi \in r(x)$ iff there exists $y > x$ such that $\psi \in r(y)$;
- for all $x \in \mathrm{dom}\, r$ and all $\bigcirc\psi \in sub\,\varphi$, we have $\bigcirc\psi \in r(x)$ iff $\psi \in r(x+1)$.

If n is the minimal number of $\mathrm{dom}\, r$ then we say that r *starts at* n.

For a set $\mathfrak{R}$ of runs through $\mathfrak{f}$, we say that the pair $(\mathfrak{f}, \mathfrak{R})$ is a *quasimodel for* φ if the following conditions are satisfied:

(q0) $\varphi \in l_0(w)$ for the minimal w in T_0,

(q1) for all $x \in \mathbb{N}$ and $w \in T_x$, there is a unique run $r \in \mathfrak{R}$ such that $r(x) = w$.

(q2) for all $r, r' \in \mathfrak{R}$ and all $x, y \in \mathrm{dom}\, r \cap \mathrm{dom}\, r'$, $r(x) <_x r'(x)$ iff $r(y) <_y r'(y)$.

Lemma 1. *A $\mathcal{TL}_{\bigcirc}$-formula φ is satisfiable in an e-model from $\mathcal{L}_{\mathsf{fin}}$ iff there exists a quasimodel for φ.*

Proof. We only show the implication ($\Leftarrow$) and leave the (basically trivial) other direction to the reader.

Given a quasimodel $(\mathfrak{f}, \mathfrak{R})$ for φ, define

$$\mathfrak{S}(x) = (W_x, R_x, P_x^1, P_x^2, \dots)$$

by taking, for $x \in \mathbb{N}$,

- $W_x = \{r \in \mathfrak{R} \mid x \in \operatorname{dom} r\}$,
- rR_xr' iff $r(x) <_x r'(x)$, whenever $x \in \operatorname{dom} r \cap \operatorname{dom} r'$,
- $P_x^i = \{r \in \mathfrak{R} \mid x \in \operatorname{dom} r, p_i \in l_x(r(x))\}$.

Clearly, $\mathfrak{S}$ is an e-model from $\mathcal{L}_{\mathsf{fin}}$. By a straightforward induction on the construction of $\psi \in sub\,\varphi$ one can show that $(x, r) \models \psi$ iff $\psi \in r(x)$. The claim of the lemma follows now from **(q0)**.

Of course, the unsurprising Lemma 1 simply reformulates the notion of satisfiability in $\mathcal{L}_{\mathsf{fin}}$ into the language of quasimodels. However, this language will be convenient for showing that actually we can effectively enumerate those formulas that do not have quasimodels.

Suppose we are given a quasimodel $(\mathfrak{f}, \mathfrak{R})$ for φ as above. Formulas of the form $\Diamond_F\psi$ that occur in some $l_x(w)$, $x \in \mathbb{N}$, will be called *eventualities in* $\mathfrak{f}(x)$. We say that an eventuality $\Diamond_F\psi \in l_x(w)$ is *realised at* $y > x$ if y is the minimal number such that $\psi \in l_y(r(y))$, where r is that unique run in $\mathfrak{R}$ for which $r(x) = w$. An eventuality is *realised until* z (or *in the interval* (n, m)) if it is realised at some $y < z$ (at some $y \in (n, m)$, respectively).

We say that $\mathfrak{f}(y) = \langle T_y, <_y, l_y\rangle$ is *embeddable into* $\mathfrak{f}(z) = \langle T_z, <_z, l_z\rangle$, where $y < z$, if there exists an injective map $g : T_y \to T_z$ such that, for all $u, v \in T_y$,

- $u <_y v$ iff $g(u) <_z g(v)$,
- $l_z(g(u)) = l_y(u)$.

If $x < y < z$ and $\mathfrak{f}(y)$ is embeddable into $\mathfrak{f}(z)$ by a map g respecting the runs through $\mathfrak{f}(x)$ in the sense that $g(r(y)) = r(z)$ whenever $x \in \operatorname{dom} r$ then we say that $\mathfrak{f}(y)$ is *x-embeddable* into $\mathfrak{f}(z)$

Let $\ell(\varphi)$ be the *length* of φ, say, $\ell(\varphi) = |sub\,\varphi|$ and let $s(n, \varphi) = (\ell(\varphi)+1)^{n+1}$.

Lemma 2. *A $\mathcal{TL}_{\bigcirc}$-formula φ is satisfiable in an e-model from $\mathcal{L}_{\mathsf{fin}}$ iff there is a quasimodel $(\mathfrak{f}, \mathfrak{R})$ for φ such that*

(A) $|T_n| \le s(n, \varphi)$, *where* $\mathfrak{f}(n) = \langle T_n, <_n, l_n\rangle$, $n \in \mathbb{N}$, *and*

(B) *for the sequence* $0 = k_0 < k_1 < k_2 < \dots$ *of minimal numbers such that all eventualities in* $\mathfrak{f}(k_i)$ *are realised until* k_{i+1}, *if* $k_i < n < m < k_{i+1}$ *and* $\mathfrak{f}(n)$ *is* k_i*-embeddable into* $\mathfrak{f}(m)$, *then some eventuality from* $\mathfrak{f}(k_i)$ *is realised in the interval* (n, m).

Proof. Suppose that a quasimodel $(\mathfrak{h}, \mathfrak{Q})$ for φ is given, with $\mathfrak{h}(n) = \langle T_n, <_n, l_n \rangle$, $n \in \mathbb{N}$. Define two operations shrink and delete on $(\mathfrak{h}, \mathfrak{Q})$.

Shrink makes T_n of size $\le s(n, \varphi)$ provided that $|T_{n-1}| \le s(n-1, \varphi)$ or $n = 0$. If $n = 0$, then set $T = \{w\}$, where w is minimal in $\langle T_0, <_0 \rangle$. If $n > 0$, then let $T \subseteq T_n$ be the set of points $w \in T_n$ such that there exists a run $r \in \mathfrak{Q}$ with $r(n) = w$ and $n - 1 \in \mathrm{dom}\, r$.

Define $T_n' \subseteq T_n$ by adding to T the set of all $<_n$-maximal points $u \in T_n$ such that there is some $w \in T$, $w <_n u$, with $\Diamond\psi \in l_n(w)$ and $\psi \in l(u)$. It should be clear that the size of T_n' is as required. Denote by $<_n'$ and l_n' the restrictions of $<_n$ and l_n to T_n', respectively. Clearly, $\langle T_n', <_n', l_n' \rangle$ is coherent. Now define $\mathfrak{h}'$ by taking, for $m \in \mathbb{N}$,

$$\mathfrak{h}'(m) = \begin{cases} \mathfrak{h}(m) & \text{if} \quad m \neq n, \\ \langle T_n', <_n', l_n' \rangle & \text{if} \quad m = n. \end{cases}$$

Finally, define a set $\mathfrak{Q}'$ of runs as follows: we put r to $\mathfrak{Q}'$ if $r \in \mathfrak{R}$ and $n \notin \mathrm{dom}\, r$, or if $n \in \mathrm{dom}\, r$ and $r(n) \in T_n'$; and if $n \in \mathrm{dom}\, r$ but $r(n) \notin T_n'$ then we put to $\mathfrak{Q}'$ the restriction of r to $\{n+1, \dots\}$. It is easy to see that $(\mathfrak{h}', \mathfrak{Q}')$ is a quasimodel for φ.

Delete removes a part of the quasimodel between $\mathfrak{h}(n)$ and $\mathfrak{h}(m)$, $n < m$, if the former is embeddable in the latter. More precisely, let $x < n < m$ and $\mathfrak{h}(n)$ is x-embeddable in $\mathfrak{h}(m)$ by some injection g. Construct a new quasimodel $(\mathfrak{h}', \mathfrak{Q}')$ as follows. First we set

$$\mathfrak{h}'(k) = \begin{cases} \mathfrak{h}(k) & \text{if} \quad k < n, \\ \mathfrak{h}(k + m - n) & \text{if} \quad k \ge n \end{cases}$$

(that is we 'cut off' the words $\mathfrak{h}(n), \dots, \mathfrak{h}(m-1)$ from the original quasimodel). And then we construct $\mathfrak{Q}'$ by putting into it runs r' defined by taking

- if $r \in \mathfrak{R}$ starts at $k \in [n, m]$ then r' starts at n and $r'(n + y) = r(m + y)$, $y \ge 0$;
- if $r \in \mathfrak{R}$ starts at $k > m$ then r' starts at $n + k - m$ and $r'(n + k - m + y) = r(k + y)$, $y \ge 0$;
- if $r \in \mathfrak{R}$ starts at $k < n$ then there is $r_1 \in \mathfrak{R}$ such that $g(r(n)) = r_1(m)$, and we set
$$r'(k) = \begin{cases} r(k) & \text{if} \quad k < n, \\ r_1(k + m - n) & \text{if} \quad k \ge n. \end{cases}$$

It is not hard to check that $(\mathfrak{h}', \mathfrak{Q}')$ is still a quasimodel for φ.

Using these two operations we can transform any given quasimodel $(\mathfrak{h}, \mathfrak{Q})$ for φ into a quasimodel $(\mathfrak{f}, \mathfrak{R})$ for φ satisfying the conditions of the lemma. We begin by shrinking $\mathfrak{h}(0)$ and finding the minimal k_1 such that all eventualities in the resulting $\mathfrak{h}(0)$ are realised until k_1. Then we shrink the $\mathfrak{h}(i)$, for $0 < i \le k_1$, and delete a part of the quasimodel (if such a part exists) between $\mathfrak{h}(n)$ and $\mathfrak{h}(m)$, $0 < n < m < k_1$, such that $\mathfrak{h}(m)$ is 0-embeddable into $\mathfrak{h}(n)$ and no eventuality from $\mathfrak{h}(0)$ is realised in the interval (n, m). Note that, due to 0-embeddability of

$\mathfrak{h}(n)$ into $\mathfrak{h}(m)$, in the resulting quasimodel every eventuality from $\mathfrak{h}(0)$ is realised until some $k_1' \leq k_1$. Then, we repeat the procedure. After finitely many iterations we end up with a quasimodel for φ with the first segment $[0, k_1]$ satisfying the conditions of the lemma. We then proceed with considering the word k_1, etc.

Now, to conclude the proof of Theorem 4, it is enough to show that there is an algorithm which, when applied to a $\mathcal{TL}_{\bigcirc}$-formula φ, eventually stops iff φ is not satisfiable. The existence of such an algorithm can be proved using Lemma 2, Higman's lemma [13] and König's lemma.

The algorithm explores all possible ways of constructing a quasimodel for a given φ *satisfying* the conditions of Lemma 2. By condition (A), the choice of $\boldsymbol{T}_\varphi$-words for the nth position in such a quasimodel is bounded by some recursive function $s'(n, \varphi)$. We claim that all possible ways of constructing a first segment $[0, k_1]$ satisfying the conditions of Lemma 2 must come to an end (exhaust all possible choices) after some step N_1. Indeed, suppose otherwise, i.e., for every $n \in \mathbb{N}$, we can have a sequence of $\boldsymbol{T}_\varphi$-words $\mathfrak{f}(0), \ldots, \mathfrak{f}(n)$ satisfying (A), (B) and such that not all eventualities in $\mathfrak{f}(0)$ are realised until n. Then, by (A) and König's lemma, there exists an infinite sequence such that condition (A) holds, at least one of the eventualities from $\mathfrak{f}(0)$ is not satisfied, and if $n < m$ and $\mathfrak{f}(n)$ is 0-embeddable into $\mathfrak{f}(m)$ then some eventuality from $\mathfrak{f}(0)$ is realised in the interval (n, m). Let m be the smallest number such that all eventualities in $\mathfrak{f}(0)$ realised in this sequence are actually realised until m (such a number exists because there are only finitely many such eventualities). But then, by Higman's lemma, we must have some i, j, for $m < i < j$, such that $\mathfrak{f}(i)$ is 0-embeddable in $\mathfrak{f}(j)$, contrary to condition (B).

If we fail to construct at least one first segment satisfying Lemma 2, then φ is not satisfiable. Otherwise we try to extend successful first segments to realise the eventualities of their last word, again complying with conditions (A) and (B), and so forth. Clearly, φ is not satisfiable iff this algorithm eventually stops.

4 Decidability

We now show that if we consider satisfiability in models with eventually constant finite domains then we can obtain a decidable logic, provided that its language does not contain the next-time operator.

Theorem 5. $\mathsf{Log}\mathcal{L}^{\mathsf{c}}_{\mathsf{fin}}$ *is decidable, but not in primitive recursive time.*

The crucial difference between $\mathsf{Log}\mathcal{L}^{\mathsf{c}}_{\mathsf{fin}}$ and $\mathsf{Log}\mathcal{L}_{\mathsf{fin}}$is revealed by the following:

Lemma 3. *A $\mathcal{TL}$-formula φ is satisfiable in an e-model from $\mathcal{L}^{\mathsf{c}}_{\mathsf{fin}}$ iff there is a quasimodel $(\mathfrak{f}, \mathfrak{R})$ for φ such that, for some $N \in \mathbb{N}$,*

(a) $|T_n| \leq s(n, \varphi)$, *where* $\mathfrak{f}(n) = \langle T_n, <_n, l_n \rangle$ *and* $n < N$,
(b) *there are no $n < m < N$ such that $\mathfrak{f}(n)$ is embeddable into $\mathfrak{f}(m)$,*
(c) *for all $n \geq N$, $|T_n| = |T_N|$ and there are some $N = n_1 < \cdots < n_k$ such that the set $A_i = \{m \geq N \mid \mathfrak{f}(n_i) = \mathfrak{f}(m)\}$ is infinite for each n_i, and every $\mathfrak{f}(n)$, for $n \geq N$, belongs to some A_i.*

Proof. Since e-models in $\mathcal{L}^{\mathsf{c}}_{\mathsf{fin}}$ have finite states with eventually constant domains, we may assume that φ is satisfied in a quasimodel $(\mathfrak{h}, \mathfrak{Q})$ satisfying condition (c) for some $N \in \mathbb{N}$. By applying operations shrink and delete from the proof of Lemma 2 (with plain 'embeddable' instead of 'x-embeddable') to the $\boldsymbol{T}_\varphi$-words from the segment $\mathfrak{h}(0), \dots, \mathfrak{h}(N-1)$ as many times as possible (the number N will become smaller after each application of delete), we will eventually construct a quasimodel as required.

Now, using the same argument as in the previous section (involving Higman's and König's lemmas), we can effectively construct finitely many initial segments $\mathfrak{f}(0), \dots, \mathfrak{f}(n)$, satisfying (a) and (b) above, of possible quasimodels for φ. For each such segment, take the final state $\mathfrak{f}(n) = (T_n, <_n, l_n)$ and suppose that $w_0 <_n \dots <_n w_m$ are all elements of T_n. Consider the formula

$$\chi_{\mathfrak{f}} = \bar{l}_n(w_0) \wedge \Diamond\Big(\bar{l}_n(w_1) \wedge \Diamond\big(\bar{l}_n(w_2) \wedge \Diamond(\cdots \Diamond \bar{l}_n(w_m) \cdots)\big)\Big),$$

where $\bar{l}_n(w) = \bigwedge\{\psi \mid \psi \in l_n(w)\}$. It should be clear that φ is satisfiable iff, for at least one of the constructed segments $\mathfrak{f}(0), \dots, \mathfrak{f}(n)$, the formula $\chi_{\mathfrak{f}}$ is satisfiable in a quasimodel $\mathfrak{f}(n+1), \mathfrak{f}(n+2), \dots$ (with some set $\mathfrak{R}$ of runs) satisfying condition (c) of Lemma 3.

Observe now that the temporal operators $\Box_F$ and $\Diamond_F$ in such quasimodels behave like **S5** modalities: for all $m > n$ and all $w \in T_m$, we have $\Diamond_F \psi \in l_m(w)$ iff there is $k > n$ such that $\psi \in l_k(r(k))$, where $r(m) = w$. Thus, we can complete the decidability part of the proof of Theorem 5 if we can prove the following.

Let $\mathcal{C}$ be the class of bimodal models of the form $\big(W, R, (\mathfrak{V}_x \mid x \in V)\big)$, where $V \neq \emptyset$, (W, R) is a finite strict linear order, and $\mathfrak{V}_x$, for $x \in V$, is a valuation in W (i.e., a map from the set of propositional variables into the set of subsets of W). In other words, we have $|V|$ (not necessarily distinct) models based on (W, R). Define the truth relation $(x, u) \models \varphi$ for $\mathcal{TL}$-formulas in such a model by taking for $x \in V$ and $u \in W$:

- $(x, u) \models p_i$ iff $u \in \mathfrak{V}_x(p_i)$,
- $(x, u) \models \Diamond_F \psi$ iff there exists $y \in V$ such that $(y, u) \models \psi$,
- $(x, u) \models \Diamond \psi$ iff there exists $v \in W$ such that uRv and $(x, v) \models \psi$,

plus the standard clauses for the Booleans. (In fact, we have defined bimodal models based on product frames of the form $(V, V \times V) \times (W, R)$; see [8] for details.)

Proposition 1. *The satisfiability problem for $\mathcal{TL}$-formulas in models from $\mathcal{C}$ is decidable. (Moreover, if a formula is satisfiable, then it satisfiable in a finite model from $\mathcal{C}$).*

This proposition can be proved using the quasimodel technique from [8]. The second half of Theorem 5 can be proved in the same way as in [10] using a reduction of the non-primitive recursive reachability problem for lossy channel systems from [26].

In the next section we will show that the addition of the 'next-time' operator $\bigcirc$ results in a logic for $\mathcal{L}^{\mathsf{c}}_{\mathsf{fin}}$ that is not even recursively enumerable. Notice that the decidability proof given above breaks down for $\bigcirc$ when we observe that 'on the tail' the temporal operators behave like **S5** modalities: this is not the case for $\bigcirc$. Lemma 3, however, still holds for the language with $\bigcirc$.

5 Undecidable problems for channel systems

Our proofs of undecidability and non-recursive enumerability (Theorems 2 and 3) proceed by reduction of suitable reachability problems for channel systems. We briefly discuss the required problems in this section; for further information on channel systems the reader is referred to [2, 4, 26].

A *single channel system* is a triple $S = \langle Q, \Sigma, \Delta \rangle$, where $Q = \{q_1, \dots, q_n\}$ is a finite set of *control states*, $\Sigma = \{a, b, \dots\}$ is a finite alphabet of *messages*, and $\Delta \subseteq Q \times \{?, !\} \times \Sigma \times Q$ is a finite set of *transitions*.

A *configuration* of S is a pair $\gamma = \langle q, \boldsymbol{w} \rangle$, where $q \in Q$ and $\boldsymbol{w} \in \Sigma^*$. Say that a configuration $\gamma' = \langle q', \boldsymbol{w}' \rangle$ is the result of a *perfect transition* of S from $\gamma = \langle q, \boldsymbol{w} \rangle$ and write $\gamma \to_p \gamma'$ if

- there is $(q, !, u, q') \in \Delta$ such that $\boldsymbol{w}' = u\boldsymbol{w}$, or
- there is $(q, ?, u, q') \in \Delta$ such that $\boldsymbol{w} = \boldsymbol{w}'u$.

The *reachability problem* for channel systems is formulated as follows: given a channel system S and two states q_0 and q_f, decide whether there is a computation starting from $\langle q_0, \epsilon \rangle$ and reaching q_f, where ϵ is the empty word. This reachability problem is obviously *recursively enumerable.* However, similarly to the halting problem for Turing machines we have the following result that was proved in [2]:

Theorem 6. *The reachability problem for channel systems is undecidable.*

We say that γ' is a result of a *lossy transition* from γ and write $\gamma \to_\ell \gamma'$ if

$$\gamma \quad \sqsupseteq \quad \gamma_1 \to_p \gamma_2 \quad \sqsupseteq \quad \gamma'$$

for some γ_1 and γ_2, where $\langle q, \boldsymbol{w} \rangle \sqsupseteq \langle q', \boldsymbol{w}' \rangle$ iff $\boldsymbol{w}'$ is a subword of $\boldsymbol{w}$ and $q = q'$.

The *ω-reachability problem* for (lossy) channel systems is formulated as follows: given a channel system S and two states q_0 and q_f, decide whether for every $n \in \mathbb{N}$ there exists a lossy computation of S starting with $\langle q_0, \epsilon \rangle$ and reaching q_f at least n times. The proof of the next theorem was kindly suggested by Ph. Schnoebelen.

Lemma 4. *The ω-reachability problem for lossy channel systems is undecidable.*

Proof. We prove this lemma by reduction of the undecidable *boundedness problem* [24]: given a channel system S, determine whether the set of configurations of S that are reachable from $\langle q_0, \epsilon \rangle$ is *finite.*

Given a channel system S, we construct a system S' in such a way that S is bounded, that is, has only finitely many configurations reachable from $\langle q_0, \epsilon \rangle$, iff S' has the ω-reachability property. The set of states of S' extends that of S with one new additional state q_{rec}, and the set of transitions of S' is that of S plus non-deterministic transitions from every state of S into q_{rec}. Being in q_{rec}, the system reads one symbol from the channel and stays in q_{rec}. It should be clear now that there exists a (lossy) computation of S' starting with $\langle q_0, \epsilon\rangle$ and reaching q_{rec} arbitrary many times iff S is unbounded.

6 Non-recursive enumerability

Here we show that the addition of the next-time operator to $\mathcal{TL}$ immediately destroys the decidability result of Theorem 5 for $\mathsf{Log}\mathcal{L}^{\mathsf{c}}_{\mathsf{fin}}$.

Theorem 7. $\mathsf{Log}_{\bigcirc}\mathcal{L}^{\mathsf{c}}_{\mathsf{fin}}$ *is not recursively enumerable.*

Proof. Given a channel system S, control states q_0 and q_f, we construct a $\mathcal{TL}_{\bigcirc}$-formula φ_{S,q_0,q_f} which is satisfiable in a model from $\mathcal{L}^{\mathsf{c}}_{\mathsf{fin}}$ iff a computation started from $\langle q_0, \epsilon\rangle$ reaches q_f. Since the reachability problem for channel systems is undecidable, but recursively enumerable, this will show that the set $\mathsf{Log}_{\bigcirc}\mathcal{L}^{\mathsf{c}}_{\mathsf{fin}}$ cannot be recursively enumerable.

With a slight abuse of notation, we use the propositional variables

- δ, for every instruction $\delta \in \Delta$,
- a, for every $a \in \Sigma$,
- q, for every $q \in Q$,
- m, a marker,
- end, a marker for 'end of word' or 'empty word.'

Let w stand for $\bigvee_{a\in\Sigma} a$, and let $\Box^+_F\psi = \psi \wedge \Box_F\psi$, $\boxdot^+\psi = \psi \wedge \boxdot\psi$, and $\Diamond^+_F\psi = \psi \vee \Diamond_F\psi$, $\diamonddot^+\psi = \psi \vee \diamonddot\psi$.

Intuitively, our encoding of the reachability problem works as follows. First we 'mark' infinitely many states by making marker m true everywhere in these states (and false in all others).

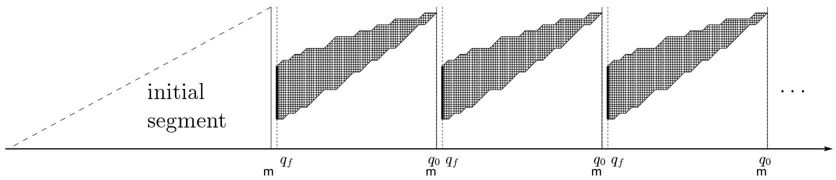

Fig. 2. Encoding of the perfect channel reachability problem.

This can be achieved by using the formulas:

$$\Box^+_F \Diamond^+_F \mathsf{m} \tag{2}$$

$$\Box^+_F\big((\mathsf{m} \to \boxdot\mathsf{m}) \wedge (\neg\mathsf{m} \to \boxdot\neg\mathsf{m})\big) \tag{3}$$

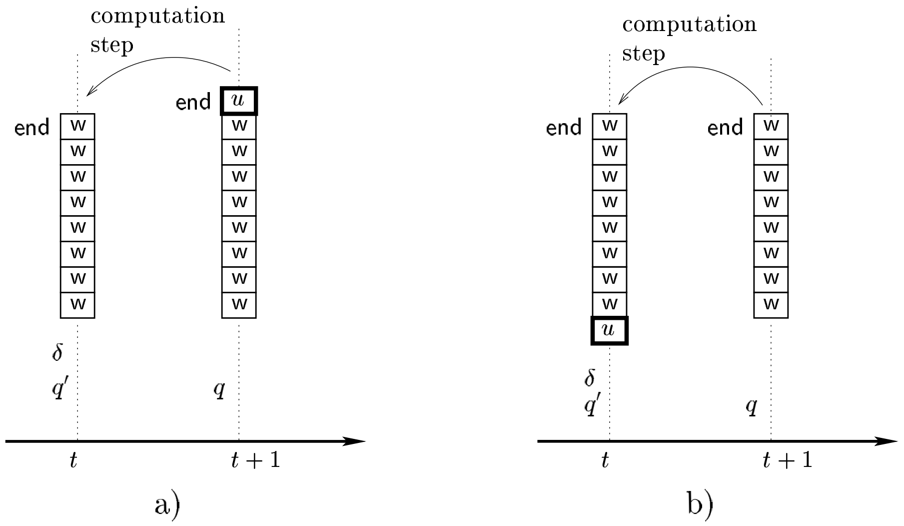

Fig. 3. Encoding one transition of a channel system. a) $\delta = (q, ?, u, q')$: symbol u is read from the end of the channel; b) $\delta = (q, !, u, q')$: symbol u is written at the beginning of the channel.

Between any two markers, we simulate from right to left (that is, from future to past) a computation of the channel system S starting with $\langle q_0, \epsilon\rangle$ and reaching control state q_f; see Fig. 2. At every moment x we write the contents of the channel on the linear order $(W_x, <_x)$ as a word without 'gaps.' We mark its end with end, and if the word is empty then end will hold somewhere:

$$\Box_F^+\big(\Diamonddot^+\mathsf{end} \wedge \boxdot^+(\mathsf{end} \to \boxdot\neg\mathsf{end})\big) \tag{4}$$

$$\Box_F^+\boxdot^+\big((\mathsf{w} \wedge \boxdot\neg\mathsf{w}) \to \mathsf{end}\big) \tag{5}$$

$$\Box_F^+\boxdot^+\neg\big(\mathsf{w} \wedge \Diamonddot(\neg\mathsf{w} \wedge \Diamonddot\mathsf{w})\big) \tag{6}$$

At every marked state, the system is in control state q_0 and the channel is empty. Moreover, this initial configuration is not obtained from any previous state by any instruction δ:

$$\Box_F^+\boxdot^+\big(\mathsf{m} \to (q_0 \wedge \neg\mathsf{w} \wedge \bigwedge_{\delta\in\Delta} \neg\delta)\big) \tag{7}$$

At every non-marked state the system is in a certain control state q which results from the previous state by means of an application of some instruction δ:

$$\Box_F^+\boxdot^+\Big(\bigvee_{q\in Q} q \wedge \bigwedge_{q\neq q'} (q \to \neg q') \wedge \bigwedge_{q\in Q} (q \to \boxdot q)\Big) \tag{8}$$

$$\Box_F^+\boxdot^+\Big(\neg\mathsf{m} \to \big(\bigvee_{\delta\in\Delta} \delta \wedge \bigwedge_{\delta\neq\delta'} (\delta \to \neg\delta') \wedge \bigwedge_{\delta\in\Delta} (\delta \to \boxdot\delta)\big)\Big) \tag{9}$$

The following formula ensures that words are encoded properly and that the contents of channels does not change arbitrarily:

$$\Box_F^+ \boxdot^+ \Big(\bigwedge_{a \in \Sigma} \big(a \to \bigcirc(\mathsf{w} \to a)\big) \wedge \bigwedge_{a \neq a'} (a \to \neg a') \Big) \tag{10}$$

Finally, we encode the effect of instructions δ; see Fig. 3. For every instruction $\delta = (q, !, u, q')$, take

$$\Box_F^+(\delta \to \bigcirc q) \tag{11}$$

$$\Box_F^+ \Big(\delta \to q' \wedge \dot{\Diamond}^+(u \wedge \neg \bigcirc \mathsf{w}) \wedge \boxdot^+ \big(\mathsf{w} \to \boxdot(\mathsf{w} \leftrightarrow \bigcirc \mathsf{w})\big) \Big) \tag{12}$$

This formula says that we add u to the beginning of the word encoded at the next moment of time and that nothing else changes. Similarly, for every instruction $\delta = (q, ?, u, q')$, take

$$\Box_F^+ \Big(\delta \to \bigcirc \big(q \wedge \dot{\Diamond}(u \wedge \mathsf{end})\big) \Big) \tag{13}$$

$$\Box_F^+ \Big(\delta \to \big(q' \wedge \dot{\Diamond}^+(\mathsf{end} \wedge \bigcirc(\dot{\Diamond}\mathsf{end} \wedge \boxdot\boxdot\neg\mathsf{end})) \wedge \\ \boxdot^+(\mathsf{w} \to \bigcirc \mathsf{w}) \wedge \boxdot^+(\bigcirc(\mathsf{w} \wedge \neg\mathsf{end}) \to \mathsf{w})\big) \Big) \tag{14}$$

This formula says that we delete u from the end of the word encoded at the next moment of time and that nothing else changes. To make sure that the final state of the computations is q_f, we need one more formula

$$\Box_F^+(\mathsf{m} \to \bigcirc q_f) \tag{15}$$

Note that (15) together with (7) and (8) also ensure that there cannot be two marked adjacent states.

Let φ_{S,q_0,q_f} be the conjunction of formulas (2)–(15). It is not difficult to show that if there exists a computation of S starting from $\langle q_0, \epsilon \rangle$ and reaching q_f, then φ_{S,q_0,q_f} is satisfied in a model with constant domains such that between any two markers the computation of S is simulated. Conversely, suppose that φ_{S,q_0,q_f} is satisfied in a model from $\mathcal{L}_{\mathsf{fin}}^{\mathsf{c}}$. Take two successive marked states n_1 and n_2 such that $W_{n_1} = W_{n_2}$ (i.e., the domain does not change between n_1 and n_2). Then a computation of S starting with q_0 and reaching q_f is simulated between n_2 and n_1.

This completes the proof of Theorem 7.

7 Undecidability

The encoding of *perfect* channel systems in the previous section was only possible because we were considering models with eventually constant domains. In models with expanding domains we can only simulate *lossy* computations of channel systems. Actually, a very simple modification of the formula φ_{S,q_0,q_f} above is enough to prove that $\mathsf{Log}_{\bigcirc}\mathcal{L}_{\mathsf{fin}}$ is undecidable. We begin by showing how to do this, and after that explain how to remove $\bigcirc$ in order to prove undecidability of $\mathsf{Log}\mathcal{L}_{\mathsf{fin}}$.

Proposition 2. *For any channel system S and states q_0 and q_f, one can construct a $\mathcal{TL}_{\bigcirc}$-formula φ_{S,q_0,q_f} which is satisfiable in a model from $\mathcal{L}_{\mathsf{fin}}$ iff, for every $n \in \mathbb{N}$, there exists a lossy computation of S starting with $\langle q_0, \epsilon\rangle$ and reaching q_f at least n times.*

Proof. As above we use markers m that are true in infinitely many states and simulate a *lossy* computation between any two marked states. However, instead of forcing these computations to reach q_f, now we ensure that, for every n, there exist two marked states such that a computation between them reaches q_f at least n times. This will be enforced by the formula $\psi_{\omega\text{-rec}}$ which replaces the conjunct (15) in φ_{S,q_0,q_f}. The formula $\psi_{\omega\text{-rec}}$ is defined as the conjunction of (16)–(19) below.

First we introduce an auxiliary variable s that cannot be true on two different elements of W_x

$$\Box_F^+ \boxdot^+ (\mathsf{s} \to \boxdot\neg\mathsf{s}), \tag{16}$$

and if s is true on some $u \in W_x$, then q_f is also true there

$$\Box_F^+ \boxdot^+ (\mathsf{s} \to q_f) \tag{17}$$

The variable s is used for 'counting.' Whenever marker m is true, we can guarantee that at the next moment of time there exists a *new* domain point where s is true:

$$\Box_F^+ \boxdot^+ \big(\mathsf{m} \to \boxdot(\boxdot\bot \to \bigcirc\Diamond\mathsf{s})\big) \tag{18}$$

(Here we use the fact that the domains can expand.) The next formula together with (18) ensure that if s is true n times in some interval between two markers, then in the next interval it must be true at least $n+1$ times:

$$\Box_F^+ \boxdot^+ \big(\mathsf{s} \to \Box_F(\mathsf{m} \to \neg\mathsf{m}\,\mathcal{U}\,\mathsf{s})\big) \tag{19}$$

Using the standard technique (see, e.g., [5]) formula (19), containing the 'until' operator $\mathcal{U}$, can be replaced with the following $\mathcal{TL}_{\bigcirc}$-formula which is satisfiable iff (19) is satisfiable:

$$\Box_F^+ \boxdot^+ \big(\mathsf{s} \to \Box_F(\mathsf{m} \to \bigcirc p \wedge \Diamond_F \mathsf{s})\big) \wedge \Box_F^+ \boxdot^+ \big((p \to \neg\mathsf{m} \vee \mathsf{s}) \wedge \big(p \to \bigcirc(p \vee \mathsf{s})\big)\big)$$

where p is a fresh variable.

We are now in a position to prove the following:

Theorem 8. $\mathsf{Log}\mathcal{L}_{\mathsf{fin}}$ *is undecidable.*

Proof. Given a channel system S and states q_0, q_f, we construct, by modifying the formula φ_{S,q_0,q_f} above, a $\mathcal{TL}$-formula ψ_{S,q_0,q_f} which is satisfiable in a model from $\mathcal{L}_{\mathsf{fin}}$ iff, for every $n \in \mathbb{N}$, there exists a lossy computation of S starting with $\langle q_0, \epsilon \rangle$ and reaching q_f at least n times.

Although the language $\mathcal{TL}$ does not contain the next-time operator, we can simulate 'locally' some of its properties. Let v_i be a fresh propositional variable. Then we have $(x, u) \models \Diamond_F \mathrm{v}_i \wedge \Box_F \Box_F \neg \mathrm{v}_i$ iff

- $(x + 1, u) \models \mathrm{v}_i$,
- $(y, u) \not\models \mathrm{v}_i$ for all $y > x + 1$.

Thus, at point (x, u) we can refer to the next point $(x+1, u)$ along the time axis. However, this can be done only *once* for given u and v_i. We denote the resulting 'one-off' next-time operator by $\bigcirc_i$ and use it as an extra temporal operator of our language $\mathcal{TL}$ bearing in mind that every occurrence of $\bigcirc_i \varphi$ should be replaced with $\Diamond_F \mathrm{v}_i \wedge \Box_F \Box_F \neg \mathrm{v}_i$, and that $\Box_F^+ \Box^+ (\mathrm{v}_i \to \varphi)$ should be added as a conjunct to the whole formula.

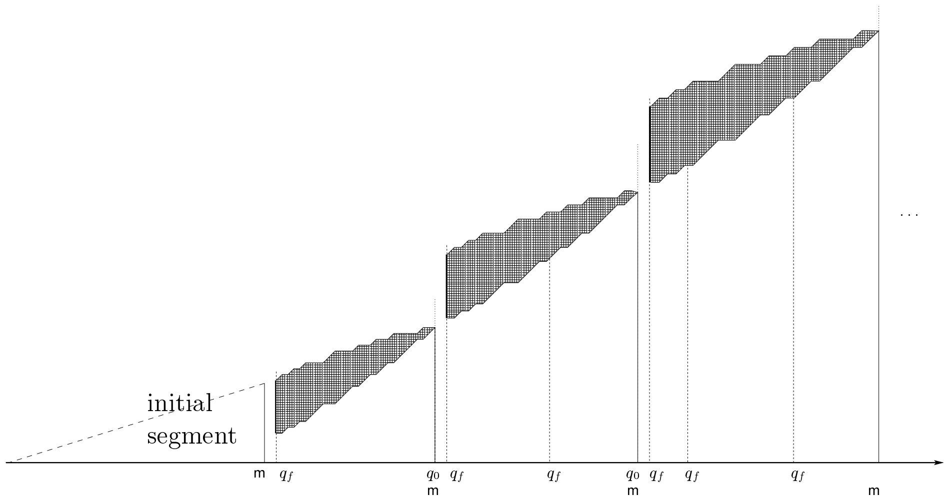

Fig. 4. Encoding of the lossy channel ω-reachability problem.

The $\mathcal{TL}$-encoding of the ω-reachability problem for lossy channels is done in almost the same way as in the proofs of Theorem 7 and Proposition 2. In every next interval between two occurrences of the marker m, we model a computation of the channel system S visiting the state q_f at least one time more than in the previous interval, and the contents of the channel is written on the linear order as a word without gaps. Note, however, that if some point $u \in W_x$ is used for writing a word at time point x, it will never be used again for encoding words in other intervals—simply because our 'surrogate' next-time operators cannot be reused. Fortunately, this is not a real problem: by expanding the domain we can always find the required 'fresh' points; see Fig. 4.

The modification ψ_{S,q_0,q_f} of φ_{S,q_0,q_f} we need keeps conjuncts (2)–(9) intact. We add the conjunct

$$\Box_F^+ \boxdot^+ \neg\big(\mathsf{w} \wedge \Diamond_F(\neg\mathsf{w} \wedge \Diamond_F \mathsf{w})\big) \tag{20}$$

saying that for any given domain point the set of time points with a symbol from Σ written on it is a (possibly empty) interval. In particular, symbols from Σ cannot be written on the same domain points in different intervals between markers. Further, replace (10)–(13) with the following formulas (21)–(25):

$$\Box_F^+ \boxdot^+ \Big(\bigwedge_{a \in \Sigma} \big(a \to \Box_F(\mathsf{w} \to a)\big) \wedge \bigwedge_{a \neq a'} (a \to \neg a') \Big) \tag{21}$$

For every instruction $\delta = (q, !, u, q')$,

$$\Box_F^+ (\delta \to \Diamonddot^+ \bigcirc_1 q) \tag{22}$$

$$\Box_F^+ \Big(\delta \to q' \wedge \Diamonddot^+ (u \wedge \Box_F \neg\mathsf{w}) \wedge \boxdot^+ \big(\mathsf{w} \to \boxdot(\mathsf{w} \leftrightarrow \Diamond_F \mathsf{w})\big)\Big) \tag{23}$$

and for every instruction $\delta = (q, ?, u, q')$,

$$\Box_F^+ \Big(\delta \to \Diamonddot^+ \bigcirc_2 \big(q \wedge \Diamonddot(u \wedge \mathsf{end})\big)\Big) \tag{24}$$

$$\Box_F^+ \Big(\delta \to q' \wedge \Diamonddot^+ (\mathsf{end} \wedge \Box_F \Diamonddot \mathsf{end}) \wedge \boxdot^+ (\mathsf{w} \to \Diamond_F \mathsf{w}) \Big) \tag{25}$$

Note that formulas (22) and (24) may force introduction of new domain points.

We also have to replace formulas (16)–(19) with some other formulas expressing the same property of m and q_f: the number of occurrences of q_f between adjacent markers m is growing in time. In formulas (26)–(27) below, $\mathsf{p} \wedge \Box_F \neg\mathsf{p}$ plays the same role as the variable s in (16)–(17):

$$\Box_F^+ \boxdot^+ \big((\mathsf{p} \wedge \Box_F \neg\mathsf{p}) \to \boxdot\neg(\mathsf{p} \wedge \Box_F \neg\mathsf{p})\big) \tag{26}$$

$$\Box_F^+ \boxdot^+ \big((\mathsf{p} \wedge \Box_F \neg p) \to q_f\big) \tag{27}$$

The following formulas (28)–(32) guarantee that for every $N \in \mathbb{N}$, there are adjacent marked $t_1 < t_2$ such that the number of time points $t \in (t_1, t_2)$ for which $(t, u) \models \mathsf{p} \wedge \Box_F \neg\mathsf{p}$, for some $u \in W_t$, is $\geq N$:

$$\Box_F^+ \boxdot^+ \big(\mathsf{m} \wedge \mathsf{p} \to \Box_F(\mathsf{m} \to \neg\mathsf{p})\big) \tag{28}$$

$$\Box_F^+ \boxdot^+ (\neg\mathsf{p} \to \Box_F \neg\mathsf{p}) \tag{29}$$

$$\Box_F^+ \boxdot^+ \big(\mathsf{p} \wedge \Box_F \neg\mathsf{p} \to \boxdot^+ (\boxdot \bot \to \bigcirc_3 \Diamonddot \mathsf{p})\big) \tag{30}$$

$$\Box_F^+ \boxdot^+ \big(\mathsf{m} \to \boxdot^+ (\boxdot \bot \to \bigcirc_4 \Diamonddot \mathsf{p})\big) \tag{31}$$

$$\Box_F^+ \boxdot^+ (\mathsf{p} \wedge \Box_F \neg\mathsf{p} \to \neg\mathsf{m}) \tag{32}$$

This claim is proved by induction on N. We only show the basis of induction $N = 1$ and indicate how to extend it to the inductive step.

Let t_0 be the first marked time point. By (31), there is $u \in W_{t_0+1}$ such that $(t_0 + 1, u) \models \mathsf{p}$. Two cases are possible now. First, if $(t, u) \models \mathsf{p} \wedge \Box_F \neg \mathsf{p}$ holds for some $t > t_0$ before the next marked point, then we are done. Otherwise, $(t_1, u) \models \mathsf{p}$ for the next marked time point t_1. Let t_2 be the first marked point after t_1. Then, by (28) and (29), $(t, u) \models \neg \mathsf{p}$ for all $t \geq t_2$. It follows that for some t with $t_1 \leq t < t_2$ we must have $(t, u) \models \mathsf{p} \wedge \Box_F \neg \mathsf{p}$. In view of (32), $t \neq t_1$.

For the inductive step we use (30) to ensure that the number of points with $\mathsf{p} \wedge \Box_F \neg \mathsf{p}$ in the next interval between two marked points is at least the same as in the previous one, while (31) adds one more point of this kind.

8 An application to dynamic topological logic

Dynamic topological logic was introduced in 1997 (see, e.g., [20, 1, 21]) as a logical formalism for describing the behaviour of dynamical systems, e.g., in order to specify liveness and safety properties of hybrid systems [3]. Roughly, (some aspects of) the behaviour of such systems are modelled by means of a *topology* $\mathfrak{T}$ on a *space* Δ and a *continuous function* f acting on Δ. What we are interested in is the asymptotic behaviour of iterations of f, in particular, the orbits $w, f(w), f^2(w), \dots$ of states $w \in \Delta$. The language $\mathcal{TL}_{\bigcirc}$ provides a natural formalism for speaking about such iterations, with propositional variables interpreted as subsets of Δ, the modal operator $\Diamond$ interpreted as the topological *closure* operator $\mathbb{C}$ on $\mathfrak{T}$, and the temporal operators $\Diamond_F$ and $\bigcirc$ as iterations of the function f.

More formally, by a *dynamic topological model* we understand a structure

$$\mathfrak{M} = (\Delta, \mathfrak{T}, f, P^1, P^2, \dots),$$

where Δ is a space with topology $\mathfrak{T}$, $f : \Delta \to \Delta$ is a continuous function with respect to this topology, and $P^i \subseteq \Delta$ for all i. For a $\mathcal{TL}_{\bigcirc}$-formula φ and $w \in \Delta$, the *truth relation* $\mathfrak{M}, w \models \varphi$ is defined as follows:

$$\begin{array}{ll}
\mathfrak{M}, w \models p_i & \text{iff } \ w \in P^i, \\
\mathfrak{M}, w \models \Diamond\varphi & \text{iff } \ w \in \mathbb{C}\{v \in \Delta \mid \mathfrak{M}, v \models \varphi\}, \\
\mathfrak{M}, w \models \bigcirc\varphi & \text{iff } \ \mathfrak{M}, f(w) \models \varphi, \\
\mathfrak{M}, w \models \Diamond_F\varphi & \text{iff } \ \mathfrak{M}, f^n(w) \models \varphi \text{ for some } n \in \mathbb{N} \ .
\end{array}$$

A formula φ is *valid in* $\mathfrak{M}$ if $\mathfrak{M}, w \models \varphi$ for every $w \in \Delta$.

Every quasi-order (Δ, R) gives rise to a topological space with the interior operator $\mathbb{I}$ defined by $\mathbb{I}(X) = \{x \in X \mid \forall y \in \Delta\ (xRy \to y \in X)\}$ (as usual, $\mathbb{C}(X) = X \setminus \mathbb{I}(\Delta \setminus X)$). Such spaces are known as *Aleksandrov spaces*. For Aleksandrov spaces, the truth relation for $\Diamond$ can be defined in a more familiar Kripke-style way:

$$\mathfrak{M}, w \models \Diamond\varphi \quad \text{iff} \quad \mathfrak{M}, v \models \varphi \text{ for some } v \in \Delta \text{ such that } wRv.$$

Moreover, it is easy to see that a function f is continuous with respect to this topology iff $\forall w, v \in \Delta\ (wRv \to f(w)Rf(v))$.

By the *dynamic topological logic of Aleksandrov spaces* we understand the set of $\mathcal{TL}_{\bigcirc}$-formulas that are valid in all dynamic topological model based on Aleksandrov spaces.

Theorem 9. *The dynamic topological logic of Aleksandrov spaces is undecidable.*

Proof. Using the techniques developed in [10], one can show that every $\mathcal{TL}_{\bigcirc}$-formula φ is satisfiable in an e-model from $\mathcal{QO}$ iff φ has a dynamic topological model based on an Aleksandrov space.

A lot of problems related to dynamic topological logics remain open. For example, is the dynamic topological logic of Aleksandrov spaces recursively enumerable? Is it finitely axiomatisable? Is the dynamic topological logic of arbitrary topological spaces decidable and/or axiomatisable?

9 Conclusion

Being a very attractive and powerful formalism for representation of and reasoning about systems with changing states, first-order temporal logic is notorious for its bad computational behaviour. This applies, in particular, to first-order temporal logics which can represent non-local constraints on binary relations such as transitivity. The present paper makes one more step in the search for fundamental reasons that could explain this phenomenon and thereby help in finding maximal 'well-behaved' fragments. Here we investigate the potential computational impact of relaxing the standard constant domain assumption by allowing states to expand over time. We consider the standard propositional temporal logic LTL equipped with an additional 'modal' operator for speaking about transitive relations over states. This fragment of first-order temporal logic comes from temporal description logic, specification & verification of hybrid systems, and some other areas. The main results of our research, given by Theorems 1–3 above, show that by allowing expanding domains we can indeed end up with logics having better computational properties. The logics still remain extremely complex, but sometimes they become recursively enumerable or even decidable, which makes them a subject for various theorem proving techniques.

It is worth noting that the same results can be proved for the language containing additionally a modal operator interpreted by the converse R_x^{-1} of R_x in each state $\mathfrak{S}(x)$. Also, as this language interpreted over strict linear orders is expressively complete for the two-variable fragment of first-order logic [23], we can reformulate our results as decidability/undecidability results for the monodic fragment of the two-variable first-order temporal logic over e-models based on finite or arbitrary strict linear orders.

Acknowledgements We are grateful to A. Bovykin, Ph. Schnoebelen, and the members of the London Logic Forum for stimulating discussions, comments and suggestions.

The work on this paper was partially supported by U.K. EPSRC grants no. GR/R42474/01, GR/S63175/01, GR/S61973/01/01, and GR/S63182/01.

References

1. S. Artemov, J. Davoren, and A. Nerode. Modal logics and topological semantics for hybrid systems. Technical Report MSI 97-05, Cornell University, 1997.
2. D. Brand and P. Zafiropulo. On communicating finite-state machines. *Journal of the ACM*, 30:323–342, 1983.
3. J. Davoren and A. Nerode. Logics for hybrid systems. *Proceedings of the IEEE*, 88:985–1010, 2000.
4. A. Finkel. Decidability of the termination problem for completely specified protocols. *Distributed Computing*, 7:129–135, 1994.
5. M. Fisher. A resolution method for temporal logic. In J. Myopoulos and R. Reiter, editors, *Proceedings of IJCAI'91*, pages 99–104. Morgan Kaufman, 1991.
6. M. Fisher, C. Dixon, and M. Peim. Clausal temporal resolution. *ACM Transactions on Computational Logic (TOCL)*, 2(1):12–56, 2001.
7. D. Gabbay, I. Hodkinson, and M. Reynolds. *Temporal Logic*, volume 1. Oxford University Press, 1994.
8. D. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-Dimensional Modal Logics: Theory and Applications*, volume 148 of *Studies in Logic*. Elsevier, 2003.
9. D. Gabelaia, R. Kontchakov, A. Kurucz, F. Wolter, and M. Zakharyaschev. Combining spatial and temporal logics: expressiveness vs. complexity. *Journal of Artificial Intelligence Research*, 23:167–243, 2005.
10. D. Gabelaia, A. Kurucz, F. Wolter, and M. Zakharyaschev. Non-primitive recursive decidability of products of modal logics with expanding domains. Manuscript. Available at `http://www.dcs.kcl.ac.uk/staff/mz`, 2004.
11. D. Gabelaia, A. Kurucz, F. Wolter, and M. Zakharyaschev. Products of 'transitive' modal logics. *Journal of Symbolic Logic*, 2005. In print. Draft available at `http://www.dcs.kcl.ac.uk/staff/mz`.
12. J. Halpern and M. Vardi. The complexity of reasoning about knowledge and time I: lower bounds. *Journal of Computer and System Sciences*, 38:195–237, 1989.
13. G. Higman. Ordering by divisibility in abstract algebras. *Proceedings of the London Mathematical Society*, 2:326–336, 1952.
14. I. Hodkinson, F. Wolter, and M. Zakharyaschev. Decidable fragments of first-order temporal logics. *Annals of Pure and Applied Logic*, 106:85–134, 2000.
15. U. Hustadt and B. Konev. TRP++ 2.0: A temporal resolution prover. In F. Baader, editor, *Automated Deduction. Proceedings of the 19th International Conference on Automated Deduction (CADE-19)*, volume 2741 of *Lecture Notes in Computer Science*, pages 274–278. Springer, 2003.
16. U. Hustadt, B. Konev, A. Riazanov, and A. Voronkov. **TeMP**: A temporal monodic prover. In *Proceedings IJCAR 2004*, volume 3097 of *LNAI*, pages 326–330. Springer, 2004.

17. B. Konev, A. Degtyarev, C. Dixon, M. Fisher, and U. Hustadt. Towards the implementation of first-order temporal resolution: the expanding domain case. *Information and Computation*, 2005. In print. Available as Technical Report ULCS-03-005, The University of Liverpool, Department of Computer Science.
18. B. Konev, R.. Kontchakov, F. Wolter, and M. Zakharyaschev. On dynamic topological and metric logics. Manuscript. Available at `http://www.dcs.kcl.ac.uk/staff/mz`, 2005.
19. R. Kontchakov, C. Lutz, F. Wolter, and M. Zakharyaschev. Temporalising tableaux. *Studia Logica*, 76:91–134, 2004.
20. P. Kremer and Mints. Dynamic topological logic. *Bulletin of Symbolic Logic*, 3:371–372, 1997.
21. P. Kremer and G Mints. Dynamic topological logic. *Annals of Pure and Applied Logic*, 131:133–158, 2005.
22. J.B. Kruskal. Well-quasi-orderings, the tree theorem, and Vázsonyi's conjecture. *Transactions of the American Mathematical Society*, 95:210–225, 1960.
23. C. Lutz, U. Sattler, and F. Wolter. Modal logic and the two-variable fragment. In *Proceedings of Computer Science Logic (CSL 2001)*, pages 262–276. Lecture Notes in Computer Science 2141, Springer, 2001.
24. R. Mayr. Undecidable problems in unreliable computations. *Theoretical Computer Science*, 297:337–354, 2003.
25. K. Schild. Combining terminological logics with tense logic. In *Proceedings of the 6th Portuguese Conference on Artificial Intelligence*, pages 105–120, Porto, 1993.
26. Ph. Schnoebelen. Verifying lossy channel systems has nonprimitive recursive complexity. *Information Processing Letters*, 83:251–261, 2002.
27. S. Schwendimann. *Aspects of Computational Logic*. PhD thesis, Universität Bern, Switzerland, 1998.
28. A. Sistla and E. Clarke. The complexity of propositional linear temporal logics. *Journal of the Association for Computing Machinery*, 32:733–749, 1985.
29. A. Sistla and S. German. Reasoning with many processes. In *Proceedings of the Second IEEE Symposium on Logic in Computer Science*, pages 138–153, 1987.
30. P. Wolper. The tableau method for temporal logic: An overview. *Logique et Analyse*, 28:119–152, 1985.
31. 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.
32. F. Wolter and M. Zakharyaschev. Axiomatizing the monodic fragment of first-order temporal logic. *Annals of Pure and Applied Logic*, 118:133–145, 2002.