Decidability and Incompleteness Results for First-Order Temporal Logics of Linear Time

Stephan Merz, Institut für Informatik
Ludwig-Maximilians-Universität München
Theresienstr. 39, W-8000 München 2, FRG
merz@sunhegering1.informatik.uni-muenchen.de

August 1, 1991

Abstract

The question of axiomatizability of first-order temporal logics is studied w.r.t. different semantics and several restrictions on the language. The validity problem for logics admitting flexible interpretations of the predicate symbols or allowing at least binary predicate symbols is shown to be Π^1_1-complete. In contrast, it is decidable for temporal logics with rigid monadic predicate symbols but without function symbols and identity.

1 Introduction

Temporal logic of linear time is widely used in the field of specification and verification of parallel programs (for an overview, see [Krö87] and [Pnu86]). Its propositional version, PTL, is adequate for the description of finite-state systems; the decidability of its validity (and hence also its satisfiability) problem yields, in principle, a procedure for the synthesis of such systems from PTL specifications (cf. [MW84]). For more complex systems, however, it is either necessary or convenient to employ first-order temporal logic (FTL for short). FTL has been known to be strongly incomplete, i.e., it does not admit a (finitistic) sound and complete formal system (see, e.g., [Sza86, SzH88, Aba89, Krö90]). However, the known proofs left some questions open concerning minimum requirements on the language of FTL necessary for obtaining incompleteness. In particular, to our knowledge all proofs relied on first-order languages with identity.

In the following we will consider both variants of FTL semantics that appear in the literature: One approach, in the following referred to as TLV, is based on a rigid interpretation of the predicate symbols but a state-dependent valuation of a designated subset of the free variables called *local variables* or *flexible constants* (cf. [Krö87, Sza86]). The alternative approach, called TLP in the sequel, is based on flexible interpretations of the predicate symbols (see [Bau89]). Some formalisms combine TLP and TLV, e.g., see [Aba89, AM90]. In both cases, we make a distinction between logics with or without identity. Furthermore, restrictions on the arities of the predicate symbols and, in particular, monadic logics allowing only unary predicate symbols, will be considered.

It will be shown that the validity problem for — even monadic — non-propositional TLP is Π^1_1-complete, hence these logics are strongly incomplete. On the other hand, it is decidable for pure monadic TLV without identity. The incompleteness results are obtained as consequences of or similarly to the strong incompleteness theorem of [SzH88] for TLV with identity whereas the decidability of monadic TLV is proved in the same way as the standard decidability result for classical monadic first-order logic.

2 Syntax and Semantics

In the following, we consider first-order languages without constant and function symbols.[1] The *alphabet* of a TLV-language $\mathcal{L}$ consists of a denumerable set X^{glob} of global variables (typical elements of which are $x, y, x_0, \ldots$), a denumerable set X^{loc} of local variables $z, z_0, \ldots$, for every $n \geq 0$, at most denumerably many n-ary predicate symbols $P, Q, \ldots$, the symbols $\neg, \rightarrow, \circ, \Box, \mathsf{atnext}, \forall$, and parentheses. In case of TLV with identity (denoted as $TLV^=$), $\mathcal{L}$ moreover contains the binary predicate symbol $=$ which will be used in the familiar infix notation. The set of predicate symbols (except possibly $=$) is called the *signature* of $\mathcal{L}$; TLV-languages containing at most predicates from a given set $\mathcal{S}$ will be denoted by $TLV(\mathcal{S})$. In particular, $TLV(P^n)$ will denote TLV-languages containing at most n-ary predicate symbols.

In the absence of function symbols, the only terms of $\mathcal{L}$ are global and local variables. Formulas are defined inductively as usual, where, however, only global variables may be bound by the quantifier $\forall$. Further classical and temporal connectives like $\wedge$, $\vee$, $\leftrightarrow$, $\exists$, $\Diamond$, unless are assumed to be defined as syntactic abbreviations (see [Krö87]) and will be used freely. Notions like free and bound (global) variables, universal closure etc. are defined as in classical first-order logic.

An *interpretation* for a TLV-language $\mathcal{L}$ is a triple $\mathcal{K} = (\mathcal{I}, \mathcal{W}, \xi)$ consisting of a classical first-order structure $\mathcal{I}$ with domain $|\mathcal{I}| \neq \emptyset$ which assigns a predicate $\mathcal{I}(P) : |\mathcal{I}|^n \rightarrow \{\mathsf{t}, \mathsf{f}\}$ to every n-ary predicate symbol P, an ω-sequence of states $\mathcal{W} = (\eta_0, \eta_1, \ldots)$, where each state η_i is a valuation $\eta_i : X^{loc} \rightarrow |\mathcal{I}|$ of the local variables, and a global valuation $\xi : X^{glob} \rightarrow |\mathcal{I}|$. In the case of $TLV^=$, the interpretation $\mathcal{I}(=)$ of $=$ is fixed as the identity on $|\mathcal{I}|$. Let $\mathcal{K} = (\mathcal{I}, \mathcal{W}, \xi)$ be an interpretation and

[1] Let us note that constant symbols are excluded only for technical simplicity. This will be evident in the proof of theorem 4.

$d \in |\mathcal{I}|$. Then $\mathcal{K}[x/d]$ denotes the interpretation $\mathcal{K}'$ defined as $\mathcal{K}' = (\mathcal{I}, \mathcal{W}, \xi')$ where $\xi'(y) = \xi(y)$ for all global variables $y \in X^{glob}$ different from x and $\xi'(x) = d$.

The interpretation $\mathcal{K}_i(t) \in |\mathcal{I}|$ of a term t is given by $\xi(t)$ or by $\eta_i(t)$ according to $t \in X^{glob}$ or $t \in X^{loc}$ being a global or local variable, respectively. The satisfaction relation $\mathcal{K}_i \models A$ for a TLV-formula A at the state η_i is inductively defined as follows (cf. [Krö87]):

$$
\begin{array}{lcl}
\mathcal{K}_i \models P(t_1, \ldots, t_n) & \iff & \mathcal{I}(P)(\mathcal{K}_i(t_1), \ldots, \mathcal{K}_i(t_n)) = \mathsf{t}, \\
\mathcal{K}_i \models \neg A & \iff & \mathcal{K}_i \not\models A, \\
\mathcal{K}_i \models A \to B & \iff & \mathcal{K}_i \not\models A \text{ or } \mathcal{K}_i \models B, \\
\mathcal{K}_i \models \forall x A & \iff & \mathcal{K}_i[x/d] \models A \text{ for all } d \in |\mathcal{I}|, \\
\mathcal{K}_i \models \circ A & \iff & \mathcal{K}_{i+1} \models A, \\
\mathcal{K}_i \models \Box A & \iff & \mathcal{K}_j \models A \text{ for all } j \geq i, \\
\mathcal{K}_i \models A \;\mathsf{atnext}\; B & \iff & \mathcal{K}_j \not\models B \text{ for all } j > i \text{ or} \\
 & & \mathcal{K}_k \models A \text{ for the smallest } k > i \text{ such that } \mathcal{K}_k \models B
\end{array}
$$

The alphabet of a TLP-language $\mathcal{L}$ similarly contains global variables and predicate symbols, logical connectives, parentheses and, for $TLP^=$, the binary predicate symbol $=$. It does not, however, contain local variables. Hence, only global variables are terms. Again, the signature may be restricted to some set of predicate symbols; in particular, the language of *propositional temporal logic* (PTL) only contains 0-ary predicate symbols. Since 0-ary predicate symbols cannot be applied to terms, the language of PTL will moreover be assumed not to contain global variables.

A TLP-interpretation is also defined as a triple $\mathcal{K} = (\mathcal{I}, \mathcal{W}, \xi)$ where, however, $\mathcal{I}$ does not interpret the predicate symbols of the language, but only defines the universe $|\mathcal{I}|$. $\mathcal{W} = (\eta_0, \eta_1, \ldots)$ is an ω-sequence of *states*, every state η_i mapping each n-ary predicate symbol P to a predicate $\eta_i(P) : |\mathcal{I}|^n \to \{\mathsf{t}, \mathsf{f}\}$, and $\xi : X^{glob} \to |\mathcal{I}|$ is a (global) variable valuation as above. In $TLP^=$-interpretations, $\eta_i(=)$ is the identity on $|\mathcal{I}|$ for every state η_i. The satisfaction relation $\mathcal{K}_i \models A$ for a TLP-formula A at a state η_i is similarly defined as above for TLV. In the case of PTL, the notions of universe and variable valuation are irrelevant, thus $\mathcal{K}$ only consists of the state sequence $\mathcal{W}$.

Let $\mathcal{K}$ be a TLV- or TLP-interpretation. We say that $\mathcal{K}$ *satisfies* a formula A ($\mathcal{K} \models A$) iff $\mathcal{K}_0 \models A$ holds. A is called TLV*-satisfiable* (resp., TLP*-satisfiable*) if there is a TLV-interpretation (resp., a TLP-interpretation) $\mathcal{K}$ such that $\mathcal{K} \models A$.[2] A is called (TLV- or TLP-) *valid* ($\models A$) if every (TLV- or TLP-) interpretation $\mathcal{K}$ satisfies A.

[2] The definition given here is known as *initial semantics*. Frequently, another definition is given where A is called satisfiable if there exist an interpretation $\mathcal{K}$ and $i \in \mathbb{N}$ such that $\mathcal{K}_i \models A$. However, both definitions are equivalent as far as general satisfiability in arbitrary temporal interpretations is concerned, cf. [Pnu86].

3 Interpreting TLP in TLV

In this section, we prove that for every TLP-formula A — not containing the equality predicate — there is a TLV-formula $\tilde{A}$ such that A is TLP-satisfiable (or TLP-valid) if and only if $\tilde{A}$ is TLV-satisfiable (TLV-valid, respectively). As a consequence, incompleteness results can be transferred from TLP to TLV.

Let in the following $\mathcal{L}$ be a TLP-language and define a TLV-language $\tilde{\mathcal{L}}$:

- The set X^{glob} of global variables of $\tilde{\mathcal{L}}$ is the same as in $\mathcal{L}$.
- For every $n \geq 0$ such that $\mathcal{L}$ contains an n-ary predicate smbol, $\tilde{\mathcal{L}}$ contains an $(n+1)$-ary predicate symbol $Elem^n$. $\tilde{\mathcal{L}}$ contains no other predicate symbols.
- For every predicate symbol P of $\mathcal{L}$ there is a local variable z_P in $\tilde{\mathcal{L}}$.

Now, the formula $\tilde{A}$ is inductively defined as follows:

$$\widetilde{P(x_1,\ldots,x_n)} := Elem^n(x_1,\ldots,x_n,z_P) \text{ for all atomic formulas } P(x_1,\ldots,x_n),$$
$$\widetilde{\neg A} := \neg\tilde{A}, \quad \widetilde{A \to B} := \tilde{A} \to \tilde{B}, \quad \widetilde{\forall x A} := \forall x \tilde{A},$$
$$\widetilde{\circ A} := \circ\tilde{A}, \quad \widetilde{\Box A} := \Box\tilde{A}, \quad \widetilde{A \,\mathsf{atnext}\, B} := \tilde{A} \,\mathsf{atnext}\, \tilde{B}$$

Lemma 1 *Let A be a TLP-formula. Then A is TLP-satisfiable (TLP-valid) iff $\tilde{A}$ is TLV-satisfiable (TLV-valid, respectively).*

Proof: We only prove the claim for satisfiability, validity then follows by duality. Suppose there is a TLP-interpretation $\mathcal{K} = (\mathcal{I}, \mathcal{W}, \xi)$ such that $\mathcal{K}_0 \models A$. We construct a TLV-interpretation $\tilde{\mathcal{K}} = (\tilde{\mathcal{I}}, \tilde{\mathcal{W}}, \tilde{\xi})$ which satisfies $\tilde{A}$.

We define the universe $|\tilde{\mathcal{I}}|$ as $|\mathcal{I}| \times \mathcal{P}(|\mathcal{I}|^*)$, i.e., the set of pairs consisting of an element of the original universe $|\mathcal{I}|$ and a set of sequences (or tuples) of elements of $|\mathcal{I}|$. The global variable valuation $\tilde{\xi}(x)$ is defined by $\tilde{\xi}(x) := (\xi(x), \emptyset)$ for $x \in X^{glob}$. The interpretation of the $(n+1)$-place predicate symbol $Elem^n$ (for $n \geq 0$) is given by

$$\tilde{\mathcal{I}}(Elem^n)(\underbrace{(d_1, M_1)}_{\in|\tilde{\mathcal{I}}|}, \ldots, \underbrace{(d_n, M_n)}_{\in|\tilde{\mathcal{I}}|}, \underbrace{(d_{n+1}, M_{n+1})}_{\in|\tilde{\mathcal{I}}|}) = \mathsf{t} \quad \Longleftrightarrow \quad (d_1 \ldots d_n) \in M_{n+1}.$$

The state sequence $\tilde{\mathcal{W}}$ is defined by giving the valuation of the local variables z_P of $\tilde{\mathcal{L}}$ in every state $\tilde{\eta}_i$:

$$\tilde{\eta}_i(z_P) := (d_0, \{(d_1 \ldots d_n) \mid \eta_i(P)(d_1, \ldots, d_n) = \mathsf{t}\}),$$

where $d_0 \in |\mathcal{I}| \neq \emptyset$ is an arbitrary but fixed element.

For $\{x_1,\ldots,x_n\} \subseteq X^{glob}$, two global variable valuations $\tilde{\xi}$ and $\tilde{\xi}'$ are called $\{x_1,\ldots,x_n\}$-equivalent ($\tilde{\xi} \approx_{\{x_1,\ldots,x_n\}} \tilde{\xi}'$) if, for $1 \leq i \leq n$, $\tilde{\xi}(x_i)$ and $\tilde{\xi}'(x_i)$ agree in their first component. A straightforward induction shows that $\tilde{\mathcal{K}}_i(\tilde{B}) = \tilde{\mathcal{K}}'_i(\tilde{B})$ for any formula B if $\tilde{\mathcal{I}} = \tilde{\mathcal{I}}'$, $\tilde{\mathcal{W}} = \tilde{\mathcal{W}}'$ and $\tilde{\xi} \approx_{FV(B)} \tilde{\xi}'$ for the set $FV(B)$ of free variables of B. Notice that this claim fails for formulas containing equality, $x = y$ being a simple counter-example.

We now prove that for all TLP-formulas B of the language $\mathcal{L}$, all TLP-interpretations $\mathcal{K}' = (\mathcal{I}, \mathcal{W}, \xi')$ whose global variable valuation ξ' agrees with the valuation ξ of $\mathcal{K}$ on the free variables of B and all $i \in I\!N$ we have $\mathcal{K}'_i \models B$ iff $\tilde{\mathcal{K}}'_i \models \tilde{B}$. In particular, this proves the satisfiability of $\tilde{A}$.

- $\mathcal{K}'_i \models P(x_1,\ldots,x_n)$
  $\iff \quad \eta_i(P)(\xi'(x_1),\ldots,\xi'(x_n)) = \mathsf{t}$
  $\iff \quad \tilde{\eta}_i(z_P) = (d_0, M)$ where $(\xi'(x_1)\ldots\xi'(x_n)) \in M$
  $\iff \quad \tilde{\mathcal{I}}(Elem^n)((\xi'(x_1),\emptyset),\ldots,(\xi'(x_n),\emptyset),\tilde{\eta}_i(z_P)) = \mathsf{t}$
  $\iff \quad \tilde{\mathcal{K}}'_i \models Elem^n(x_1,\ldots,x_n,z_P)$
- $\mathcal{K}'_i \models \neg B \quad \iff \quad \mathcal{K}'_i \not\models B \quad \overset{\text{I.H.}}{\iff} \quad \tilde{\mathcal{K}}'_i \not\models \tilde{B} \quad \iff \quad \tilde{\mathcal{K}}'_i \models \neg\tilde{B}$
- Similar arguments hold for $B \to C$, $\circ B$, $\Box B$ and B atnext C.
- $\mathcal{K}'_i \models \forall x B$
  $\iff \quad \mathcal{K}'_i[x/d] \models B$ for every $d \in |\mathcal{I}|$
  $\overset{\text{I.H.}}{\iff} \quad \tilde{\mathcal{K}}'_i[x/(d,\emptyset)] \models \tilde{B}$ for every $d \in |\mathcal{I}|$
  $\iff \quad \tilde{\mathcal{K}}'_i[x/(d,M)] \models \tilde{B}$ for every $d \in |\mathcal{I}|, M \subseteq |\mathcal{I}|^*$ (see above remark)
  $\iff \quad \tilde{\mathcal{K}}'_i \models \forall x \tilde{B}$

Conversely, assume $\tilde{\mathcal{K}} = (\tilde{\mathcal{I}}, \tilde{\mathcal{W}}, \tilde{\xi})$ to be a TLV-interpretation such that $\tilde{\mathcal{K}}_0 \models \tilde{A}$. Define a TLP-interpretation $\mathcal{K} = (\mathcal{I}, \mathcal{W}, \xi)$ by $|\mathcal{I}| := |\tilde{\mathcal{I}}|$, $\xi := \tilde{\xi}$ and $\eta_i(P)(d_1,\ldots,d_n) = \mathsf{t}$ iff $\tilde{\mathcal{I}}(Elem^n)(d_1,\ldots,d_n,\tilde{\eta}_i(z_P)) = \mathsf{t}$.

Again, by a simple inductive argument similar to the above we conclude that for all formulas B of the language $\mathcal{L}$ and all $i \in I\!N$ we have $\mathcal{K}_i \models B$ iff $\tilde{\mathcal{K}}_i \models \tilde{B}$. Thus, in particular, $\mathcal{K}_0 \models A$, which means that A is satisfiable. ♠

Observe that the lemma also holds for TLP-languages admitting (rigid) constant and function symbols, but not for $TLP^=$.

As a consequence of this lemma, strong incompleteness results for TLV can be obtained as consequences of similar results for TLP. More precisely, suppose that strong incompleteness has been established for TLP-languages allowing at most n-ary predicate symbols. By the above lemma, every TLP-valid formula A of such a language can be "translated" to a TLV-valid formula $\tilde{A}$ using at most $(n+1)$-ary predicate symbols. Thus, TLV-languages admitting $(n+1)$-ary predicate symbols also have to be strongly incomplete. Moreover, it can easily be shown that every TLP-formula containing *at most* n-ary predicate symbols can be rewritten into an equivalent formula containing *only* n-ary predicate symbols. The TLV-formula resulting from the above translation then contains *exactly one* $(n+1)$-ary predicate symbol. Hence, we obtain the following

Corollary 2 *If for some n, $TLP(P^n)$ does not admit a sound and complete finitistic axiomatization, then neither does $TLV(P^{n+1})$. In fact, one $(n+1)$-ary predicate symbol suffices to obtain strong incompleteness.*

4 Incompleteness of monadic TLP

In order to apply the results of the preceding section, we first have to establish the incompleteness of $TLP(P^n)$, for some n. Which is the weakest class of TLP-languages for which incompleteness may be obtained? Without equality symbol, the empty signature does not yield a logical language, $TLP(P^0)$ (propositional temporal logic) is decidable and TLP with rigid predicate symbols only (in particular, $TLP^=$ with empty signature) reduces to classical first-order logic. We will now establish a lemma which allows to translate the incompleteness proof for $TLV^=(\emptyset)$ given by Szalas and Holenderski in [SzH88] into one for $TLP(P^1)$, i.e., pure monadic TLP.

Notation: Let $\mathcal{L}$ be the language $TLP(\{P_1, \ldots, P_m\})$ whose signature contains exactly m unary predicate symbols. The formula $Unique(P_k)$ (for $1 \leq k \leq m$) is defined as

$$Unique(P_k) \;\equiv\; \forall x \forall y (\Diamond(P_k(x) \wedge P_k(y)) \rightarrow \Box \bigwedge_{j=1}^{m} (P_j(x) \leftrightarrow P_j(y)))$$

Intuitively, $Unique(P_k)$ states that, in any state, P_k can only be true for two elements if they are "undistinguishable" from each other by every predicate symbol of $\mathcal{L}$.

The following lemma shows how to interpret $TLV^=(\emptyset)$ in $TLP(P^1)$. It is assumed that $TLV^=(\emptyset)$-formulas do not contain any subformulas of the form $x = y$ for global variables x, y. However, this is only a technical restriction since any subformula $x = y$ appearing negatively (resp., positively) may be equivalently replaced by $(x = z \wedge z = y)$ (resp., $x = z \rightarrow z = y$) where z is a new local variable.

Lemma 3 *For every $TLV^=(\emptyset)$-formula A not containing atomic subformulas of the form $x = y$ for global variables $x, y \in X^{glob}$ there is a formula A' of monadic TLP such that A' is TLP-satisfiable (TLP-valid) if and only if A is $TLV^=$-satisfiable ($TLV^=$-valid, respectively).*

Proof: Again, we only prove the claim for satisfiability, since validity follows by duality.

Without loss of generality, we may assume that all atomic subformulas of A are of the form $x = z$ where $x \in X^{glob}$ is a global variable and $z \in X^{loc}$ is a local variable: Any equation of the form $z = z'$ where $z, z' \in X^{loc}$ may be equivalently rewritten as either $\forall x(x = z \rightarrow x = z')$ or $\exists x(x = z \wedge x = z')$. Let there be m local variables $z_1, \ldots, z_m$ appearing in A. For every z_i we introduce a monadic predicate symbol P_i and replace every atomic subformula $x = z_i$ of A by $P_i(x)$. A' is then obtained from the resulting TLP-formula by adding the conjunct

$$F \;\equiv\; \bigwedge_{i=1}^{m} Unique(P_i) \;\wedge\; \Box \bigwedge_{i=1}^{m} \exists x P_i(x)$$

Let $\mathcal{K} = (\mathcal{I}, \mathcal{W}, \xi)$ be a $TLV^{=}(\emptyset)$-interpretation such that $\mathcal{K} \models A$. We construct a $TLP(P_1, \ldots, P_n)$-interpretation $\mathcal{K}' = (\mathcal{I}', \mathcal{W}', \xi')$ satisfying A' by defining

$$|\mathcal{I}'| := |\mathcal{I}|, \quad \xi'(x) = \xi(x), \quad \eta_i'(P_j)(d) = \mathsf{t} \iff \eta_i(z_j) = d \quad (1 \le j \le m).$$

Obviously, by the definition of η_i', we have $\mathcal{K}' \models F$. We show that for all $i \in I\!N$ and all $TLV^{=}(\emptyset)$-formulas B containing at most the local variables $z_1, \ldots, z_m$, $\mathcal{K}_i \models B$ iff $\mathcal{K}_i' \models B'$, where B' is obtained from B by replacing all subformulas $x = z_j$ by $P_j(x)$; this ensures $\mathcal{K}' \models A'$:

- $B \equiv x = z_j$:
  $\mathcal{K}_i \models x = z_j \iff \xi(x) = \eta_i(z_j) \iff \eta_i'(P_j)(\xi'(x)) = \mathsf{t} \iff \mathcal{K}_i' \models P_j(x)$
- The remaining cases are proved by structural induction. (Note that here also the quantifier case is routine since domains and valuations of global variables agree in $\mathcal{K}$ and $\mathcal{K}'$.)

Conversely, assume there is a TLP-interpretation $\mathcal{K}' = (\mathcal{I}', \mathcal{W}', \xi')$ such that $\mathcal{K}_0' \models A'$. We define an equivalence relation $\sim$ on $|\mathcal{I}'|$ by

$$a \sim b \iff a = b \text{ or for some } i \in I\!N \text{ and } 1 \le j \le m,\ \eta_i'(P_j)(a) = \eta_i'(P_j)(b) = \mathsf{t}$$

In fact, $\sim$ is a congruence on $|\mathcal{I}'|$ with respect to $P_1, \ldots, P_m$:

- $\sim$ is obviously reflexive and symmetrical.
- $a \sim b, b \sim c \;\Rightarrow\; a \sim c$:
  The only non-trivial case occurs if $a \neq b$ and $b \neq c$. But then, we have

  $$(*) \quad \eta_i(P_j)(a) = \eta_i(P_j)(b) = \mathsf{t} \quad \text{and} \quad (**) \quad \eta_k(P_l)(b) = \eta_k(P_l)(c) = \mathsf{t}$$

  for some $i, k \in I\!N$ and $1 \le j, l \le m$. From $\mathcal{K}_0' \models F$ and $(*)$ we may deduce $\eta_k(P_l)(a) = \eta_k(P_l)(b)$ which, together with $(**)$, yields $\eta_k(P_l)(a) = \eta_k(P_l)(c) = \mathsf{t}$, hence $a \sim c$.

- $a \sim b \Rightarrow \eta_i(P_j)(a) = \eta_i(P_j)(b)$ for all $i \in I\!N, 1 \le j \le m$:
  proof similar to the previous case

Next, define a $TLV^=(\emptyset)$-interpretation $\mathcal{K} = (\mathcal{I}, \mathcal{W}, \xi)$ as follows (where the congruence class of $a \in |\mathcal{I}|$ modulo $\sim$ is denoted by $[a]$):

$$|\mathcal{I}| = |\mathcal{I}'|/\sim, \quad \eta_i(z_j) = [a] \iff \eta_i'(P_j)(a) = \mathsf{t}, \quad \xi(x) = [\xi'(x)]$$

($\eta_i(z_j)$ is well-defined because $\sim$ is a congruence on $|\mathcal{I}'|$ with respect to $P_1, \ldots, P_m$.)

We convince ourselves that for all $i \in I\!N$ and all TLP-formulas B' containing only monadic predicate symbols from among $P_1, \ldots, P_m$, we have $\mathcal{K}_i' \models B'$ iff $\mathcal{K}_i \models B$ where B is the $TLV^=$-formula obtained from B' by replacing every subformula $P_j(x)$ by $x = z_j$:

- $\mathcal{K}_i' \models P_j(x) \iff \eta_i'(P_j)(\xi'(x)) = \mathsf{t} \iff \eta_i(z_j) = [\xi'(x)] \iff \mathcal{K}_i \models x = z_j$
- $\mathcal{K}_i' \models \forall x B'$
  $\iff$ For all $d' \in |\mathcal{I}'|, \mathcal{K}_i'[x/d'] \models B'$
  $\overset{\text{I.H.}}{\iff}$ For all $d' \in |\mathcal{I}'|, \mathcal{K}_i[x/[d']] \models B$
  $\iff$ For all $d \in |\mathcal{I}|, \mathcal{K}_i[x/d] \models B$
  $\iff$ $\mathcal{K}_i \models \forall x B$
- The remaining cases are proved by simple application of the induction hypothesis.

In particular, this means $\mathcal{K} \models A$, thus A is satisfiable. ♠

From this lemma and the incompleteness of $TLV^=(\emptyset)$ we obtain the following

Theorem 4 *There is no sound and complete finitistic axiomatization for monadic TLP.*

Proof: [SzH88] proved that $TLV^=(\emptyset)$ augmented by flexible boolean variables and individual constants is strongly incomplete. They did so by constructing, for any given Turing machine $\mathcal{T}$, a formula f_{loop} such that

$\models f_{loop}$ iff $\mathcal{T}$ does not halt with empty input.

Individual constants c can be replaced, preserving validity, by new local variables z_c which are forced to constant values by adding the conjunct $\forall x(x = z_c \to \Box(x = z_c))$. Flexible boolean variables b can either be treated as conventional local variables (introducing a new individual constant $true$ and replacing every occurence of b by $b = true$), or they can simply be considered as (flexible) propositional variables which, by definition, are contained in $TLP(P^1)$.

After these changes, all non-propositional atomic subformulas of f_{loop} as given in [SzH88] are of the form $x = z$ for a global variable $x \in X^{glob}$ and a local variable $z \in$

X^{loc}, hence we can apply lemma 3 to obtain a monadic TLP-formula f'_{loop} which is TLP-valid iff f_{loop} is $TLV^=$-valid, i.e. iff $\mathcal{T}$ loops. The set of looping Turing machines, being the complement of the halting problem, is not recursively enumerable whereas any finitistic axiomatization would yield a recursively enumerable set of tautologies. Thus, $TLP(P^1)$ is strongly incomplete. ♠

The incompleteness proof of [SzH88] is based on the temporal operators $\circ, \Box$ as well as binary operators like atnext or unless. However, binary operators are not essential for this proof; it can be rephrased using only $\circ$ and $\Box$ at the expense of introducing additional local variables in the case of $TLV^=(\emptyset)$ or monadic predicate symbols for $TLP(P^1)$. Moreover, their proof can be strengthened to yield Π^1_1-hardness of the validity problem. As a consequence of Szalas' completeness theorem for his infinitary axiomatization of temporal logic [Sza87], the validity problem is contained in Π^1_1, thus establishing Π^1_1-completeness.

Theorem 5 *The set of tautologies of the monadic first-order temporal logic $TLP(P^1)$ is Π^1_1-complete if the language contains $\circ$ and $\Box$ as only temporal connectives.*

Proof: given in the appendix ♠

With this theorem and the aid of corollary 2 the incompleteness results of [SzH88] and also of [Krö90] for TLV can now be strengthened: It is not necessary to assume languages with identity for TLV to obtain strong incompleteness; allowing just one binary predicate symbol suffices.

Corollary 6 *The validity problem for TLV is Π^1_1-complete if the language contains one (uninterpreted) binary predicate symbol and at least the temporal operators $\circ$ and $\Box$.*

In the next section the validity problem for $TLV(P^1)$ will be shown to be decidable, rendering the above condition regarding the signature of TLV-languages not only sufficient, but also necessary for strong incompleteness.

5 Decidability of pure monadic TLV

In contrast to the above incompleteness results we will show here that the validity of monadic TLV-formulas (without function symbols or equality) is decidable. The proof idea employed in classical first-order logic (see, e.g., [Ack54]) carries over to $TLV(P^1)$. Notice, however, that in classical first-order logic, adding identity to an otherwise monadic signature preserves decidability in contrast to the incompleteness result for $TLV^=(\emptyset)$ given by [SzH88]. The proof will be divided into two parts: First, we show that a formula A is valid iff A holds in all interpretations with universes of a certain finite cardinality depending on the number of distinct predicate symbols appearing in A. Then, we convince ourselves that the validity of a formula in all structures of a given finite cardinality can be reduced to the validity of a formula of propositional temporal logic and is therefore decidable.

Lemma 7 *Let A be a formula of pure monadic TLV containing p unary predicate symbols $P_1, \ldots, P_p$. A is valid iff $\mathcal{K} \models A$ for all TLV-interpretations $\mathcal{K} = (\mathcal{I}, \mathcal{W}, \xi)$ whose universe $|\mathcal{I}|$ contains at most 2^p elements.*

Proof: We only have to prove the "if"-part. Assume A to be valid in all interpretations whose universes contain at most 2^p elements and let $\mathcal{K} = (\mathcal{I}, \mathcal{W}, \xi)$ be *any* $TLV(\{P_1, \ldots, P_p\})$-interpretation. Define an equivalence relation $\sim$ on $|\mathcal{I}|$ by

$$a \sim b \quad \Longleftrightarrow \quad \mathcal{I}(P_i)(a) = \mathcal{I}(P_i)(b) \quad \text{for all } 1 \leq i \leq p,$$

i.e., $a \sim b$ iff a and b are undistinguishable by all predicates assigned to $P_1, \ldots, P_p$ in $\mathcal{K}$. The relation $\sim$ induces at most 2^p equivalence classes on $|\mathcal{I}|$, each class being characterized by the subset of predicates holding of its members. Now, define another TLV-interpretation $\tilde{\mathcal{K}} = (\tilde{\mathcal{I}}, \tilde{\mathcal{W}}, \tilde{\xi})$ by

$$\begin{array}{l} |\tilde{\mathcal{I}}| := |\mathcal{I}|/\sim\,, \quad \tilde{\mathcal{I}}(P_i)([d]) := \mathcal{I}(P_i)(d) \text{ for } [d] \in |\tilde{\mathcal{I}}| \text{ and } 1 \leq i \leq p,^{3} \\ \tilde{\eta}_i(z) = [\eta_i(z)] \text{ for } i \in I\!N \text{ and } z \in X^{loc}, \quad \tilde{\xi}(x) = [\xi(x)] \text{ for } x \in X^{glob} \end{array}$$

Since $||\tilde{\mathcal{I}}|| \leq 2^p$, our assumption yields $\tilde{\mathcal{K}} \models A$. A straightforward structural induction shows that for all $TLV(P_1, \ldots, P_p)$-formulas B and all $i \in I\!N$, $\tilde{\mathcal{K}}_i \models B$ iff $\mathcal{K}_i \models B$. In particular, we obtain $\mathcal{K} \models A$ and, since $\mathcal{K}$ was arbitrary, $\models A$. Notice that the bound for the size of the universe does not depend on the number of 0-ary predicate symbols possibly contained in A because the universe $|\mathcal{I}|$ is irrelevant for their interpretation. ♠

Of course, if a formula A is valid in all interpretations $\mathcal{K}$ with universes of cardinality $||\mathcal{I}|| = n$, it is also valid in all interpretations $\mathcal{K}'$ with cardinality m where $0 < m \leq n$: We may add $n - m$ elements which are undistinguishable from some arbitrary element of $|\mathcal{I}'|$ by all predicates of the language without affecting the truth value of A in any state of $\mathcal{K}$.

Therefore, the validity problem of $TLV(\{P_1, \ldots, P_p\})$ reduces to checking validity in all interpretations whose universe has cardinality 2^p. This problem can be encoded in propositional temporal logic.

Lemma 8 *For every formula A of monadic TLV and every natural number n, there is a formula A' of propositional temporal logic which is satisfiable iff A is satisfiable in a TLV-interpretation whose universe has cardinality n.*

[3] $\tilde{\mathcal{I}}(P_i)$ is well-defined because of $\mathcal{I}(P_i)(d) = \mathcal{I}(P_i)(d')$ for $d \sim d'$.

Proof: Without loss of generality, A may be assumed to be a closed formula where, for technical convenience, all bound variables are distinct. Furthermore, w.l.o.g., we may assume the universe of the TLV-interpretation to be $|\mathcal{I}| = \{1, \ldots, n\}$. Suppose A contains p unary predicate symbols $P_1, \ldots, P_p$, q propositional variables $Q_1, \ldots, Q_q$ and m local variables $z_1, \ldots, z_m$. Let the language of propositional logic contain, in addition to $Q_1, \ldots, Q_q$, $p \cdot n$ propositional variables $P_{11}, \ldots, P_{1n}, \ldots, P_{p1}, \ldots, P_{pn}$ and $m \cdot n$ propositional variables $Z_{11}, \ldots, Z_{1n}, \ldots, Z_{m1}, \ldots, Z_{mn}$. The formula A' is defined as the conjunction $A' \equiv P \wedge Z \wedge \mathcal{S}(A)$ of the following PTL-formulas:

$$
\begin{aligned}
P &\equiv \bigwedge_{j=1}^{p} \bigwedge_{k=1}^{n} (\Diamond P_{jk} \rightarrow \Box P_{jk}) \;\wedge\; \bigwedge_{j=1}^{q} (\Diamond Q_j \rightarrow \Box Q_j), \\
Z &\equiv \Box \bigwedge_{j=1}^{m} (\bigvee_{k=1}^{n} Z_{jk} \;\wedge\; \bigwedge_{k=1}^{n} (Z_{jk} \rightarrow \bigwedge_{l \neq k} \neg Z_{jl}))
\end{aligned}
$$

Intuitively, P_{jk} will encode the value of $P_j(k)$ in $\mathcal{I}$. The formula P models rigid interpretation of all predicate symbols[4] in TLV. Z_{jk} will stand for "z_j is assigned the element $k \in |\mathcal{I}|$". The formula Z states that at every state, each variable z_j is assigned exactly one element.

The remaining conjunct $\mathcal{S}(A)$ is inductively defined according to the outermost operator of A, where $B(x/k)$ denotes the result of replacing all occurences of the variable x in B by the number k:

$$
\begin{array}{l}
\mathcal{S}(\forall x B(x)) := \mathcal{S}(B(x/1)) \wedge \ldots \wedge \mathcal{S}(B(x/n)) \\
\mathcal{S}(\neg B) := \neg \mathcal{S}(B), \quad \mathcal{S}(B \rightarrow C) := \mathcal{S}(B) \rightarrow \mathcal{S}(C), \\
\mathcal{S}(\bigcirc B) := \bigcirc \mathcal{S}(B), \quad \mathcal{S}(\Box B) := \Box \mathcal{S}(B), \quad \mathcal{S}(B \;\mathsf{atnext}\; C) := \mathcal{S}(B) \;\mathsf{atnext}\; \mathcal{S}(C) \\
\mathcal{S}(P_j(k)) := P_{jk}, \quad \mathcal{S}(P_j(z_k)) := \bigvee_{l=1}^{n} (Z_{kl} \wedge P_{jl}), \quad \mathcal{S}(Q_j) := Q_j
\end{array}
$$

Observe that $\mathcal{S}(A)$ is indeed a formula of propositional temporal logic since A was assumed to be closed.

Let us call a PTL-interpretation $\mathcal{K}'$ and a TLV-interpretation $\mathcal{K}$ (with domain $|\mathcal{I}| = \{1, \ldots, n\}$) *adjoined* if they satisfy the following conditions:

$$
\begin{array}{l}
\mathcal{K}' \models P \wedge Z, \quad \mathcal{I}(Q_j) = \eta'_0(Q_j) \quad (\text{for } 1 \leq j \leq q), \\
\mathcal{I}(P_j)(k) = \eta'_0(P_{jk}) \quad (\text{for } 1 \leq j \leq p \text{ and } 1 \leq k \leq n) \text{ and} \\
\eta_i(z_j) = k \iff \eta'_i(Z_{jk}) = \mathsf{t} \quad (\text{for } i \in I\!N, 1 \leq j \leq m \text{ and } 1 \leq k \leq n)
\end{array}
$$

It is easily checked that for every TLV-interpretation $\mathcal{K}$ there is an adjoined PTL-interpretation $\mathcal{K}'$ and, conversely, that for every PTL-interpretation satisfying $P \wedge Z$, there is an adjoined TLV-interpretation. Moreover, for all TLV-formulas B and all $i \in I\!N$, if $\mathcal{K}$ and $\mathcal{K}'$ are adjoined TLV- and PTL-structures, then $\mathcal{K}_i \models B$ iff $\mathcal{K}'_i \models \mathcal{S}(B')$, where B' results from B by replacing every free global variable x by $\xi(x) \in |\mathcal{I}|$:

[4] For logics admitting flexible interpretations of 0-ary predicate symbols, the only change in this proof would be to drop the conjunct enforcing strict interpretation of the Q_j's. In particular, flexible propositional variables may be added to monadic TLV without affecting its decidability.

First we note that from $\mathcal{K}' \models P$ we may deduce that for all $i \in I\!N$, $\eta'_i(P_{kl}) = \eta'_0(P_{kl})$ and $\eta'_i(Q_j) = \eta'_0(Q_j)$.

- $B \equiv Q_j$:
  $\mathcal{K}_i \models Q_j \iff \mathcal{I}(Q_j) = \mathsf{t} \iff \eta'_0(Q_j) = \mathsf{t} \iff \eta'_i(Q_j) = \mathsf{t} \iff \mathcal{K}'_i \models Q_j$
- $B \equiv P_j(x)$: Let $\xi(x) = k$.
  $\mathcal{K}_i \models P_j(x) \iff \mathsf{t} = \mathcal{I}(P_j)(k) = \eta'_0(P_{jk}) = \eta'_i(P_{jk}) \iff \mathcal{K}'_i \models \mathcal{S}(P_j(k))$
- $B \equiv P_j(z_k)$:
  $\mathcal{K}_i \models P_j(z_k) \iff \mathcal{I}(P_j)(\eta_i(z_k)) = \mathsf{t}$
  $\iff$ for some $1 \leq l \leq n$, $\eta_i(z_k) = l$ and $\mathcal{I}(P_j)(l) = \mathsf{t}$
  $\iff$ for some $1 \leq l \leq n$, $\eta'_i(Z_{kl}) = \mathsf{t}$ and $\eta'_i(P_{jl}) = \eta'_0(P_{jl}) = \mathsf{t}$
  $\iff \mathcal{K}'_i \models \bigvee_{l=1}^{n}(Z_{kl} \wedge P_{jl})$
- $B \equiv \forall x C(x)$:
  $\mathcal{K}_i \models \forall x C(x)$
  $\iff$ for all $d \in |\mathcal{I}|, \mathcal{K}_i[x/d] \models C(x)$
  $\iff \mathcal{K}_i[x/1] \models C(x)$ and $\ldots$ and $\mathcal{K}_i[x/n] \models C(x)$
  $\stackrel{\text{I.H.}}{\iff} \mathcal{K}'_i \models \mathcal{S}(C'(x/1))$ and $\ldots$ and $\mathcal{K}'_i \models \mathcal{S}(C'(x/n))$
  $\iff \mathcal{K}'_i \models \mathcal{S}(\forall x C(x))$
- The remaining cases are proved by a simple application of the induction hypothesis.

Therefore, if A is TLV-satisfiable (over the n-element universe $|\mathcal{I}|$), then there is a PTL-structure satisfying A' and vice versa. ♠

The two preceding lemmas, together with the decidability of propositional temporal logic, and applying the duality between satisfiability and validity, immediately yield the decidability result for monadic TLV.

Theorem 9 *The validity of a formula A of pure monadic first-order temporal logic based on rigid predicates and local variables (or flexible constant symbols) is decidable.*

6 Conclusion

In this paper, the problem of axiomatizability was studied for several variants of first-order temporal logics. Most of the logics considered were found to be strongly incomplete. In the case of monadic logics the result depends on the underlying semantics: Validity is decidable for monadic logics based on rigid predicates and local variables (or flexible constants), but it is Π^1_1-complete for those logics that admit flexible interpretations of the predicate symbols.

Our results hold uniformly for logics containing only the temporal connectives $\circ$ and $\Box$ as well as for logics containing also binary operators like atnext or unless or even operators definable by regular languages as introduced in [Wol83]. The decidability proof for $TLV(P^1)$ can readily be generalized to allow past-time operators as well (see, e.g., [LPZ85]).

Let us note that the incompleteness proofs could similarly be obtained if the constant-domain assumption used in both semantics were given up in favor of nested-domain semantics where the universe associated with later states contains that of earlier states.

Instead of giving new incompleteness proofs directly, we established reductions of the validity problems between $TLV(P^{n+1})$ and $TLP(P^n)$ on one hand (Lemma 1) and $TLP(P^1)$ and $TLV^{=}(\emptyset)$ on the other (Lemma 3). Besides shortening the proofs of the incompleteness results, these reductions shed some light on the relative expressiveness of different FTL variants and may thus be of some interest in their own right.

Finally, our results can be summarized in the following table:

	signature		
logic	empty	monadic	≥ 1 bin. pred. symbol
TLV	(no language)	decidable	non-axiomatizable
$TLV^{=}$	non-axiomatizable	non-axiomatizable	non-axiomatizable
TLP	(no language)	non-axiomatizable	non-axiomatizable
$TLP^{=}$	decidable[5]	non-axiomatizable	non-axiomatizable

Table 1: Validity problem of different first-order temporal logics

Appendix: Incompleteness of $TLP(P^1)$

Below we give a direct proof of incompleteness for the logic $TLP(P^1)$ using only the temporal operators $\circ$ and $\Box$. The proof follows the incompleteness proof of [SzH88] for $TLV^{=}$ but avoids the use of binary temporal operators. It can be reformulated as an incompleteness proof for TLV-logics containing at least one binary predicate symbol using lemma 1. The general idea to eliminate, e.g., atnext-formulas is to replace formulas of the form $A(\vec{x}) \rightarrow (B(\vec{y})$ atnext $C(\vec{z}))$ by the two formulas $A(\vec{x}) \rightarrow BAC(\vec{y}, \vec{z})$ and $\Box\forall\vec{y}\forall\vec{z}(BAC(\vec{y}, \vec{z}) \rightarrow \circ(C(\vec{z}) \rightarrow B(\vec{y})) \land \circ(\neg C(\vec{z}) \rightarrow BAC(\vec{y}, \vec{z})))$ where BAC is a new predicate symbol of suitable arity. However, some care has to be taken to introduce unary predicate symbols only and to handle atnext-formulas in the antecedent of implications.

By definition, $TLP(P^1)$ contains unary predicate symbols as well as propositional variables. Moreover, for readability, we will use individual constants in our formulas with the following convention: A "constant" c is a shorthand notation for introducing a new monadic predicate symbol C together with the formulas $\exists x \Box C(x)$ and

[5]The language of $TLP^{=}(\emptyset)$ contains only rigid symbols, hence all temporal operators can be deleted without affecting the truth value of any formula. (A atnext B has to be replaced by $B \rightarrow A$.) Thus it corresponds to classical equational logic which is decidable.

$Unique(C)$ (for the definition of $Unique$, see section 4). A "prime formula" $P(c)$ is an abbreviation for $\forall x(C(x) \rightarrow P(x))$.

As in [SzH88], the computation of a deterministic Turing machine $\mathcal{T}$ with designated final (accepting) state q_f will be encoded in the states of interpretations satisfying a formula built according to the transition table of $\mathcal{T}$. (One could also attempt to employ Harel's domino technique [Har86] for obtaining Π_1^1-hardness; however, it seems technically easier to encode a linear sequence of finite sequences representing the currently used part of a Turing machine tape instead of encoding two-dimensional domino tilings in the linear state sequence of FTL.) The tape of $\mathcal{T}$ will be assumed to extend infinitely to the right side only such that there is a leftmost tape cell. Any attempt to move the head to the left of that cell will be considered illegal and cause $\mathcal{T}$ to block, i.e., stop without accepting. The logical language employed contains the individual constants ℓ representing the leftmost tape cell, an individual constant a for each tape symbol a of the tape alphabet Σ of $\mathcal{T}$ (in particular, $\#$ will stand for the blank symbol), and a constant q for each (machine) state $q \in Q$ of $\mathcal{T}$ (in particular, q_0 denotes the starting state of $\mathcal{T}$). Besides the predicate symbols representing the constants, we will use the monadic predicate symbols $L, T, Q, LUE, TAL_a, ML_a, MR_a$ (for $a \in \Sigma$), EAL, Ext as well as the propositional variables H, HBE, EBH, QAH_q (for $q \in Q$) and $NoExt$ whose intuitive meaning will be introduced below.

The computation of $\mathcal{T}$ is encoded as a succession of subsequences of states where each subsequence represents a configuration and each state represents a tape cell. The tape symbol a carried by a cell in a configuration is encoded by the truth of the prime formula $T(a)$ in the corresponding state. The propositional variable H will be true at states representing the tape cell scanned by the head of $\mathcal{T}$ in the respective configuration. At such a state, $Q(q)$ is true for the (machine) state q of the configuration. Moreover, each cell of $\mathcal{T}$'s tape is given a unique label so that it can be identified in subsequent configurations. This label will be represented by the unary predicate L. We assume successive configurations to be of non-decreasing length, thus all labels contained in a configuration will recur in all subsequent configurations. The leftmost tape cell is labelled by ℓ, hence the formula $L(\ell)$ can serve as a delimiter between configurations. If $\mathcal{T}$ stops at some configuration (either blocking or accepting), the corresponding subsequence of states will be followed by an infinite state sequence without H becoming true.

The conjunction of the following formulas constitutes the formula f_{comp}:

- Each cell carries exactly one label (modulo the congruence defined in the proof of lemma 3):

(1) $Unique(L), \quad \Box\exists x L(x)$

- Each cell encodes exactly one tape symbol and machine state:

(2) $Unique(T), \quad \Box \operatorname*{exor}_{a\in\Sigma} T(a), \quad Unique(Q), \quad \Box \operatorname*{exor}_{q\in Q} Q(q)$

- The labels of all cells constituting one configuration are distinct:

  (3) $\Box\forall x(L(x) \to \circ LUE(x)),\ \ \Box\forall x(LUE(x) \to L(\ell) \vee (\neg L(x) \wedge \circ LUE(x))$

  These formulas are obtained by eliminating the unless-operator in $\Box\forall x(L(x) \to (\neg L(x))$ unless $L(\ell))$.

- The succession of cell labels is preserved at all subsequent configurations:

  (4) $\Box\forall x\forall y(L(x) \wedge \circ(\neg L(\ell) \wedge L(y)) \to \Box(L(x) \leftrightarrow \circ(\neg L(\ell) \wedge L(y))$

  Compare this with the formula $(TAn3)$ of [Krö90]. Observe also that the formulas simulating the transitions (given below) force $L(\ell)$ at the beginning of the subsequent configuration if $\mathcal{T}$ can make a move.

- The head scans at most one tape cell of every configuration; if no cell is marked as being scanned by the head, the configuration is infinite:

  (5) $\Box(L(\ell) \to HBE),\quad \Box(HBE \to H \vee \circ(\neg L(\ell) \wedge HBE))$

  (6) $\Box(H \to \circ EBH),\quad \Box(EBH \to L(\ell) \vee \circ(\neg H \wedge EBH))$

  These formulas are obtained by eliminating the binary operators in $\Box(L(\ell) \to H\ \vee\ H$ before $L(\ell))$ and $\Box(H \to \neg H$ unless $L(\ell))$.

- The initial configuration is of length 1 and encodes the empty tape and (machine) state q_0:

  (7) $L(\ell) \wedge H \wedge Q(q_0) \wedge T(\#) \wedge \circ L(\ell)$

- Tape symbols of cells not currently scanned by the head are preserved by the transition to the next configuration. (The following formula schemes have to be instantiated for all — finitely many — tape symbol constants $a \in \Sigma$):

  (8) $\Box\forall x(L(x) \wedge T(a) \wedge \neg H \to TAL_a(x))$

  (9) $\Box\forall x(TAL_a(x) \to \circ(L(x) \to T(a)) \wedge \circ(\neg L(x) \to TAL_a(x))$

  The formula $TAL_a(x)$ represents $T(a)$ atnext $L(x)$.

- $\mathcal{T}$ blocks if it had to move its head to the left when the head scans the leftmost tape position. For every transition $(q, a) \mapsto (q', a', L)$ of the transition table M of $\mathcal{T}$ we add the following formula:

  (10) $\Box(L(\ell) \wedge H \wedge Q(q) \wedge T(a) \to \circ\Box\neg H)$

- Transitions $(q,a) \mapsto (q',a',L)$ of $\mathcal{T}$ are simulated by the following set of formulas whenever the head is not at the leftmost tape cell. The length of the next configuration stays the same as that of the current one:

$$
\begin{array}{ll}
(11) & \forall x \Box (L(x) \wedge \circ(\neg L(\ell) \wedge H \wedge Q(q) \wedge T(a)) \rightarrow \\
& \qquad \circ(ML_{a'}(x) \wedge QAH_{q'} \wedge NoExt) \\
(12) & \forall x \Box (ML_{a'}(x) \rightarrow (L(x) \rightarrow H \wedge \circ T(a')) \wedge (\neg L(x) \rightarrow \circ ML_{a'}(x))) \\
(13) & \Box (QAH_{q'} \rightarrow \circ(H \rightarrow Q(q')) \wedge \circ(\neg H \rightarrow QAH_{q'})) \\
(14) & \Box (NoExt \rightarrow (\circ\neg L(\ell) \rightarrow \circ NoExt) \wedge \forall x (L(x) \wedge \circ L(\ell) \rightarrow EAL(x))) \\
(15) & \forall x \Box (EAL(x) \rightarrow \circ(L(x) \rightarrow \circ L(\ell)) \ \wedge \circ(\neg L(x) \rightarrow EAL(x)))
\end{array}
$$

  The formula $ML_{a'}(x)$ stands for "move the head to the tape cell labelled by x, and replace the tape symbol of this cell's neighbor cell by a'". $QAH_{q'}$ represents the formula $Q(q')$ atnext H, $NoExt$ stands for "do not extend the length of the tape portion from the current configuration to the next one", and $EAL(x)$ for $(\circ L(\ell))$ atnext $L(x)$.

- Similarly, for every transition $(q,a) \mapsto (q',a',R)$ moving the head to the right we add the following formulas. If the head scans the last cell represented in the current configuration, the number of cells representing the subsequent configuration is incremented by one:

$$
\begin{array}{ll}
(16) & \forall x \Box (H \wedge Q(q) \wedge T(a) \wedge L(x) \rightarrow \\
& \qquad (MR_{a'}(x) \wedge QAH_{q'} \wedge (\circ L(\ell) \rightarrow Ext(x)) \wedge (\circ\neg L(\ell) \rightarrow NoExt))) \\
(17) & \forall x \Box (MR_{a'}(x) \rightarrow \circ(L(x) \rightarrow T(a') \wedge \circ H) \wedge \circ(\neg L(x) \rightarrow MR_{a'}(x))) \\
(18) & \forall x \Box (Ext(x) \rightarrow \circ(L(x) \rightarrow \circ(\neg L(\ell) \wedge T(\#)) \wedge \circ\circ L(\ell)) \wedge \\
& \qquad\qquad \circ(\neg L(x) \rightarrow Ext(x)))
\end{array}
$$

  The formula $MR_{a'}(x)$ plays an analogous role as $ML_{a'}(x)$ above. $Ext(x)$ stands for "extend the tape portion from the current configuration to the next one by one blank cell where the last cell of the current configuration is labelled by x".

- $\mathcal{T}$ stops when it is in its final state q_f (in this case, no transition can be applicable):

$$(19) \quad \Box (H \wedge Q(q_f) \rightarrow \circ\Box\neg H)$$

  (For our purposes we do not have to distinguish between accepting and blocking runs of $\mathcal{T}$ but only between finite and infinite runs.)

Lemma 10 *If $\mathcal{K} \models f_{comp}$ for some TLP-interpretation $\mathcal{K}$, then the computation of $\mathcal{T}$ starting with the empty tape is encoded in the state sequence $\mathcal{W}$ of $\mathcal{K}$.*

Proof: From the formulas (1) and (2), each state η_i encodes exactly one machine state, tape symbol and cell label (modulo the congruence relation defined in the proof of lemma 3). We will divide $\mathcal{W}$ into subsequences $sq_0, sq_1, \ldots$ of states such that $L(\ell)$ is true at the first state of each sq_i but at no other state. Each finite subsequence sq_i consisting of the states $\eta_0^{(i)}, \ldots, \eta_{p^{(i)}}^{(i)}$ corresponds to a configuration $c_i = (c_0^{(i)}, \ldots, c_{p^{(i)}}^{(i)}, q^{(i)}, k^{(i)})$ of $\mathcal{T}$, where $c_0^{(i)} \ldots c_{p^{(i)}}^{(i)}$ is a string of tape symbols representing the part of the tape that has been used during the computation up to c_i, $q^{(i)} \in Q$ is the machine state of c_i and $k^{(i)}$ is the index of the cell currently scanned by the head (therefore, $0 \leq k^{(i)} \leq p^{(i)}$).

- $c_j^{(i)} \in \Sigma$, for $0 \leq j \leq p^{(i)}$, is given by the (unique) tape symbol constant a, for which $\eta_j^{(i)}(T(a)) = \mathsf{t}$.
- By (5) and (6), if sq_i is finite, it contains exactly one state $\eta_{k^{(i)}}^{(i)}$ such that $\eta_{k^{(i)}}^{(i)}(H) = \mathsf{t}$. The index $k^{(i)}$ gives the position of the head, and the current state of c_i is determined by the (unique) machine state constant $q^{(i)}$ for which $\eta_{k^{(i)}}^{(i)}(Q(q^{(i)}) = \mathsf{t}$.

By (7), the first subsequence sq_0 consists only of the state η_0 and obviously encodes the start configuration of $\mathcal{T}$. Within any configuration, each cell is determined by its unique cell label, cf. (3). Formula (4) forces the ordering of labels to be maintained in all subsequent configurations, but allows new cells (which by (3) have to carry new labels) to be added at the right. From (8) and (9) it follows that all cells except possibly the one scanned by the head will keep their tape symbols at the next configuration. If the head scans the leftmost tape cell ℓ at configuration c_i and $\mathcal{T}$ had to move to the left, or if $\mathcal{T}$ signals acceptance by entering its final state, (10) or (19), respectively, force sq_{i+1} to be infinite (otherwise, it would have to contain a state marked by H). Since $\mathcal{T}$ was assumed to be deterministic, at each non-final configuration there is exactly one applicable transition. If $\mathcal{T}$ does not stop at configuration c_i, the formulas (11) – (18) force the subsequence sq_{i+1} to encode the configuration obtained from c_i by application of the appropriate transition. In particular, the length $p^{(i+1)}$ of sq_{i+1} is forced to be $p^{(i)} + 1$ iff $\mathcal{T}$ makes a move to the right when the head scans the rightmost cell represented in sq_i, otherwise, $p^{(i+1)} = p^{(i)}$. In case $p^{(i+1)} = p^{(i)} + 1$, (18) forces the newly added cell to contain a blank symbol. Thus the sequence of configurations $(c_0, c_1, \ldots)$ reconstructed from the state sequence $\mathcal{W}$ of $\mathcal{K}$ indeed encodes the computation of $\mathcal{T}$ starting with the empty tape. ♠

Lemma 11 *f_{comp} has a model $\mathcal{K}$.*

Proof: Any denumerable set $|\mathcal{I}|$ can be taken to be the universe of $\mathcal{K}$. The construction of the state sequence $\mathcal{W}$ from the computation of $\mathcal{T}$ has been described above if the computation is infinite. If $\mathcal{T}$ stops (either accepting or blocking), the state sequence can easily be extended to yield a model of f_{comp}. ♠

From the above lemmas we immediately obtain the Π^1_1-hardness of the validity problem for monadic $TLP(\bigcirc, \Box)$:

Consider the formula $f_{loop} \equiv f_{comp} \wedge \Box\Diamond(H \wedge Q(q_0))$. By the above lemmas, f_{loop} is satisfiable iff $\mathcal{T}$, starting on the empty tape, enters its initial state infinitely often. This set of Turing machines is known to be Σ^1_1-complete (cf. [HPS83]). By duality, the validity problem of monadic $TLP(\bigcirc, \Box)$ is Π^1_1-hard. As a consequence of [Sza87] (applying lemma 1), the validity problem for full TLP is contained in Π^1_1, which completes the proof of theorem 5.

Acknowledgements

I would like to thank Prof. Dr. F. Kröger for his encouragement and support as well as for many helpful discussions. Hans Gaßner, Frank Leßke, Barbara Paech and Holger Schlingloff read preliminary versions of this paper and suggested valuable expository improvements. The suggestions offered by two anonymous referees led to the clarification of some weak spots.

References

[Aba89] M. Abadi: *The Power of Temporal Proofs*; TCS **65**, pp. 35–84 (1989)

[AM90] M. Abadi, Z. Manna: *Nonclausal Deduction in First-Order Temporal Logic*; Journal of the ACM **37**, No. 2, pp. 279–317 (1990)

[Ack54] W. Ackermann: *Solvable Cases of the Decision Problem* (North Holland, Amsterdam, 1954)

[Bau89] M. Baudinet: *Temporal Logic Programming is Complete and Expressive*; Proc. ACM Conf. on Princ. of Prog. Lang. (1989), pp. 267–280

[Har86] D. Harel: *Effective Transformations on Infinite Trees, with Applications to High Undecidability*; Journal of the ACM **33**, No. 1, pp. 224–248 (1986)

[HPS83] D. Harel, A. Pnueli, J. Stavi: *Propositional Dynamic Logic of Nonregular Programs*; J. of Computer and System Sciences **26** (1983), pp. 222–243

[Krö87] F. Kröger: *Temporal Logic of Programs*; Springer EATCS Monographs on Theoretical Computer Science **8** (Springer, Berlin, 1987)

[Krö90] F. Kröger: *On the Interpretability of Arithmetic in Temporal Logic*; TCS **73**, pp. 47–60 (1990)

[LPZ85] O. Lichtenstein, A. Pnueli, L. Zuck: *The Glory of the Past*; in: Logics of Programs, pp. 196–218, Springer LNCS **193** (1985)

[MW84] Z. Manna, P. Wolper: *Synthesis of Communicating Processes from Temporal Logic Specifications*; ACM Transactions on Programming Languages **6**, pp. 68–93 (1984)

[Pnu86] A. Pnueli: *Applications of Temporal Logic to the Specification and Verification of Reactive Systems: A Survey of Current Trends*; in: Lecture Notes in Computer Science **224** (Springer, Berlin, 1986)

[Sza86] A. Szalas: *Concerning the Semantic Consequence Relation in First-Order Temporal Logic*; TCS **47**, pp. 329–334 (1986)

[Sza87] A. Szalas: *A Complete Axiomatic Characterization of First-Order Temporal Logic of Linear Time*; TCS **54**, pp. 199–214 (1987)

[SzH88] A. Szalas, L. Holenderski: *Incompleteness of First-Order Temporal Logic with Until*; TCS **57**, pp. 317–325 (1988)

[Wol83] P. Wolper: *Temporal Logic Can Be More Expressive*; Information and Control **56**, pp. 72–99 (1983)