Real-time Logics:
Complexity and Expressiveness*†

Rajeev Alur
AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974

Thomas A. Henzinger
Computer Science Department
Cornell University
Ithaca, NY 14853

Abstract. The theory of the natural numbers with linear order and monadic predicates underlies propositional linear temporal logic. To study temporal logics that are suitable for reasoning about real-time systems, we combine this classical theory of infinite state sequences with a theory of discrete time, via a monotonic function that maps every state to its time. The resulting *theory of timed state sequences* is shown to be decidable, albeit nonelementary, and its expressive power is characterized by ω-regular sets. Several more expressive variants are proved to be highly undecidable.

This framework allows us to classify a wide variety of real-time logics according to their complexity and expressiveness. Indeed, it follows that most formalisms proposed in the literature cannot be decided. We are, however, able to identify two elementary real-time temporal logics as expressively complete fragments of the theory of timed state sequences, and we present tableau-based decision procedures for checking validity. Consequently, these two formalisms are well-suited for the specification and verification of real-time systems.

1 Introduction

Linear propositional temporal logic (PTL) is a widely used tool for the specification and verification of reactive and concurrent systems [Pnu77, OL82, LP84, MP92]. One of the appeals of PTL, which is interpreted over infinite sequences of system states, is its strong theoretical connection with the classical first-order theory of the natural numbers with linear order and monadic predicates: PTL captures an elementary, yet expressively complete, fragment of this nonelementary theory [Sto74, GPSS80, SC85]; that is, while any property of state sequences expressible in the monadic first-order theory of $(\mathsf{N}, \leq)$ can also be specified in PTL, checking the validity of PTL-formulas is much simpler than checking the validity of formulas in the underlying classical theory.

*This research was supported in part by an IBM graduate fellowship to the second author, by the National Science Foundation under grants CCR-8812595 and CCR-9200794, by the Defense Advanced Research Projects Agency under contract N00039-84-C-0211, and by the United States Air Force Office of Scientific Research under contracts AFOSR-88-0281 and F49620-93-1-0056.

†A preliminary version of this paper appeared in the *Proceedings of the Fifth Annual IEEE Symposium on Logic in Computer Science* (LICS 1990), pp. 390–401, and an extended version appeared in *Information and Computation* **104**, 1993, pp. 35–77.

PTL admits the specification of qualitative time requirements only, such as the "eventual" occurrence of an event. To enable quantitative reasoning about the timing delays in real-time applications, *real-time logics* include explicit time references and are interpreted over *timed state sequences*, which associate a time stamp with every system state [JM86, AH89, HLP90, Koy90, Ost90]. Although the suitability as specification language has often been demonstrated, most of these previous attempts at formalizing timing considerations remain ad hoc, with little regard to complexity and expressiveness questions. The prime objective of the present paper is to develop a unifying framework for the study of real-time logics, and to identify logics with an elementary validity problem. While elementary decidability is neither necessary nor sufficient for the practicality of a specification formalism, we believe that it plays a major role, and particularly so, as we are interested in real-time properties that can be verified algorithmically. In analogy to the untimed case, we identify the underlying classical theory of timed state sequences, show it to be nonelementarily decidable, and use its complexity and expressiveness as point of reference. We are able to define two orthogonal extensions of PTL that inherit its appeal: they capture elementary, yet expressively complete, fragments of the theory of timed state sequences, and thus are excellent candidates for practical real-time specification languages.

Outline

In Section 2, we define the theory of timed state sequences by combining a theory of state sequences with a theory of time, via a monotonic function that maps every state to its time. As for PTL, the monadic first-order theory of the natural numbers with linear order serves as the theory of states. To model time, we choose the theory of the natural numbers with linear order and congruence relations. We show that the resulting combined theory of timed state sequences is still decidable, and we characterize its expressiveness by ω-regular sets.

We justify our choice of theory of timed state sequences by proving that many conceivable extensions and variations, like additional primitives over time (such as addition), or a dense time domain, result in highly undecidable (Π^1_1-hard) theories. It follows from our results that none of the real-time logics proposed in [JM86], [Har88], [Koy90], and [Ost90] can be decided, which vividly demonstrates that it has not been understood, so far, how expressive a theory of time may be added to reasoning about state sequences without sacrificing decidability.

In [AH89], we proposed *timed* PTL (TPTL) as a natural specification language, and we developed a tableau-based decision procedure for TPTL. It turns out that TPTL captures precisely the fragment of the theory of timed state sequences obtained by combining PTL (the *temporal* fragment of the states component) with the *quantifier-free* fragment of the time component. We argued in [AH89] that it is the restriction of disallowing quantification over time, what yields readable specifications as well as finite-state-based verification methods. In Section 3 we show this restriction to be both harmless, by proving the expressive completeness of TPTL with respect to the underlying classical theory, and essential, by proving the nonelementary nature of TPTL extended with quantification over time variables.

There are also *second-order* versions of our results: the second-order theory of timed state sequences is still decidable, and just as PTL is generalizable to ETL [Wol83], TPTL can be extended to be as expressive as the second-order theory of timed state sequences, at no cost in complexity.

Surprisingly, the addition of *past* temporal operators renders TPTL nonelementary. This induces us to introduce, in Section 4, another expressively complete fragment of the theory of timed state sequences, MTL_P, which includes past temporal operators, but restricts the states that may be related by timing constraints. We present a tableau-based decision procedure for MTL_P, thus demonstrating its possible application for the algorithmic verification of real-time systems.

Both TPTL and MTL_P are, while being elementary, still quite expensive; the corresponding decision procedures require doubly exponential time. In Section 5 we show that this cost is intrinsic to reasoning about real-time constraints: any reasonably succinct and reasonably expressive extension of PTL with timing information is necessarily EXPSPACE-hard. Even the special case of identifying *next-time* with *next-state*, which restricts us to reasoning about synchronous systems, is not cheaper.

2 The Theory of Timed State Sequences

Real-time logics are interpreted over timed state sequences. Given a finite set P of propositions and a linearly ordered time domain $(\mathit{TIME}, \prec)$, a *timed state sequence* (σ, τ) is a pair consisting of an infinite sequence σ of states $\sigma_i \subseteq P$, $i \geq 0$, and a (weakly) monotonic function $\tau : \mathsf{N} \rightarrow \mathit{TIME}$ that maps every state to a time (i.e., $\tau(i) \preceq \tau(i+1)$ for all $i \geq 0$).

A timed state sequence models the state changes along a possible behavior of a real-time system, and the time values associated with states can be viewed as the readings of a fictitious global and discrete clock, which is incremented at fixed time intervals. For a detailed discussion of this so-called *digital-clock* (or *fictitious-clock*) *model* of time, and its applications, we refer the reader to [AH92b] or [Hen91]. Here, we introduce the classical theory of timed state sequences, show its decidability, and characterize its expressiveness by ω-regular sets.

2.1 The classical theory of state sequences

First, we recapitulate briefly why the theory of the natural numbers with linear order and monadic predicates underlies linear-time propositional temporal logics, which are interpreted over infinite sequences of states. We also survey some important results about the complexity and expressive power of PTL.

Syntax and semantics

Let $\mathcal{L}^2$ be the second-order language with (uninterpreted) unary predicate symbols, the binary predicate symbol $\leq$, and quantification over individual variables and unary predicate symbols. Let $\mathcal{L}$ be the first-order fragment of $\mathcal{L}^2$; that is, without quantification over the unary predicates. We interpret $\mathcal{L}^2$ over the natural numbers N, with $\leq$ being interpreted as the usual linear order. Note that over the natural numbers, the constant 0 and the successor function +1 can be defined from the order predicate $\leq$ using first-order quantification: $x = 0$ iff $\forall y.\,(x \leq y)$, and $y = x+1$ iff $x < y \land \forall z.\,(x < z \rightarrow y \leq z)$. Throughout we consider only formulas that contain no free individual variables. Thus, given a formula ϕ of $\mathcal{L}^2$ with the free predicate symbols $p_1, \ldots p_n$, an interpretation I for ϕ specifies the sets $p_1^I, \ldots p_n^I \subseteq \mathsf{N}$. Such an interpretation can be viewed as an infinite sequence σ of states $\sigma_i \subseteq \{p_1, \ldots p_n\}$, $i \geq 0$: let $p_k \in \sigma_i$ iff $i \in p_k^I$. We denote the set of models of ϕ by $\mathcal{M}(\phi)$; that is, $\mathcal{M}(\phi)$ contains the state sequences that satisfy ϕ.

$\mathcal{L}^2$ is essentially the language that underlies the theory S1S, the second-order theory of the natural numbers with successor and monadic predicates. This is because in S1S, the order predicate $\leq$ can be defined from the successor function using second-order quantification: $x \leq y$ iff $\forall p.\,(p(x) \land \forall z.\,(p(z) \rightarrow p(z+1)) \rightarrow p(y))$. Büchi established a close connection between the theory S1S and finite automata over infinite sequences [Büc60] and used this relationship to show that S1S is decidable [Büc62].

Complexity and expressiveness

Formulas of the propositional linear temporal logic PTL can be translated into $\mathcal{L}$ without changing the set of models, by replacing propositions with monadic predicates. For example, the typical response property ϕ_R that "Every p-state is followed by a q-state," is expressed in PTL as

$$\Box(p \rightarrow \Diamond q).$$

This condition can be written in $\mathcal{L}$ as

$$\forall i.\,(p(i) \rightarrow \exists j \geq i.\, q(j)). \qquad (\phi_R)$$

Although PTL corresponds to a proper subset of $\mathcal{L}$, it has the full expressive power of $\mathcal{L}$ [Kam68, GPSS80]; that is, for every $\mathcal{L}$-formula there is a PTL-formula that defines the same property of state sequences. Furthermore, while the validity problem for $\mathcal{L}$ is nonelementary [Sto74], the validity problem for PTL is PSPACE-complete [SC85], and PTL has a singly exponential decision procedure [BMP81].

To attain the greater expressive power of $\mathcal{L}^2$, PTL can be strengthened by adding operators that correspond to right-linear grammars [Wol83]. The resulting logic, extended temporal logic (ETL), has the expressive power of $\mathcal{L}^2$ and, like PTL, still a singly exponential decision procedure.

The expressiveness of $\mathcal{L}^2$ can also be characterized by ω-regular expressions [McN66]: for any formula ϕ of $\mathcal{L}^2$, the set $\mathcal{M}(\phi)$ can be defined by an ω-regular expression over the alphabet $2^{\{p_1,\ldots p_n\}}$. For example, the set $\mathcal{M}(\phi_R)$ of models of the response formula ϕ_R is described by the ω-regular expression

$$(\{p,q\} + \{q\} + \{\} + (\{p\}; true^*; (\{p,q\} + \{q\})))^\omega.$$

The restricted expressive power of $\mathcal{L}$ corresponds to the star-free fragment of ω-regular expressions, in which the Kleene star may be applied only to the expression *true* [MP71, Tho81].[1]

2.2 Adding time to state sequences

To obtain a theory of timed state sequences, we need to identify a suitable time domain $(TIME, \prec)$, with appropriate primitive operations, and couple the theory of state sequences with this theory of time through a "time" function f that associates a time with every state. We choose, as the theory of time, the theory of the natural numbers (i.e., $(TIME, \prec) = (\mathsf{N}, \leq)$) with linear-order and congruence primitives. Since the time cannot decrease from one state to the next, we require that f be monotonic. We will have an opportunity to justify these decisions later, in Subsection 2.4.

Let $\mathcal{L}^2_T$ be a second-order language with two sorts, a *state sort* and a *time sort*. The vocabulary of $\mathcal{L}^2_T$ consists of

- (uninterpreted) unary predicate symbols and the binary predicate symbol $\leq$ over the state sort;
- the constant symbol 0, the unary function symbol $+1$, and the binary predicate symbols $\leq$, $\equiv_2$, $\equiv_3$, …over the time sort;
- the unary function symbol f from the state sort into the time sort;
- quantification over individual variables of both sorts and over unary predicate symbols. We call variables of the state sort *state variables*, and variables of the time sort *time variables* (this usage departs from the standard temporal-logic terminology).

[1]For more details on the results about PTL and ω-regular sets, consult [Eme90] and [Tho90].

By $\mathcal{L}_T$ we denote the first-order fragment of $\mathcal{L}_T^2$ (without quantification over the unary predicate symbols). We restrict our attention to structures that choose the set of natural numbers N as domain for both sorts, and we interpret the primitives in the intended way ($x \equiv_c y$ is interpreted as "x is congruent to y modulo c"). Although the constant 0 and the successor function $+1$ over the time sort can be defined from $\leq$ using first-order quantification (as was demonstrated earlier for the state sort), we include both primitives in the language, because we will also study the fragment of $\mathcal{L}_T^2$ without time variables.

Again, we consider only formulas that contain no free individual variables (of either sort). Given a formula ϕ of $\mathcal{L}_T^2$ with the free predicate symbols $p_1, \ldots p_n$, an interpretation I for ϕ specifies the sets $p_1^I, \ldots p_n^I \subseteq \mathsf{N}$ and the monotonic function $f^I \colon \mathsf{N} \to \mathsf{N}$. The satisfaction relation $\models$ is defined in the standard fashion. Every interpretation I for ϕ can be viewed as a timed state sequence (σ, τ): choose σ as in the untimed case, and let $\tau = f^I$. We denote the set of timed state sequences that satisfy ϕ by $\mathcal{M}_T(\phi)$.

It follows that $\mathcal{L}_T^2$-formulas define properties of timed state sequences. For example, the requirement of bounded response time that "Every p-state is followed by a q-state within time 1," can be defined by a formula of $\mathcal{L}_T$:

$$\forall i.\, (p(i) \rightarrow \exists j \geq i.\, (q(j) \wedge f(j) \leq f(i) + 1)). \qquad (\phi_{BR})$$

An $\mathcal{L}_T^2$-formula ϕ is satisfiable (valid) iff it is satisfied by some (every) timed state sequence. The (second-order) *theory of timed state sequences* is the set of all valid formulas of $\mathcal{L}_T^2$. We prove it to be decidable.

2.3 Decidability and expressiveness

First we show that given an interpretation I for an $\mathcal{L}_T^2$-formula ϕ, the information in f^I essential for determining the truth of ϕ has finite-state character.

Finite-state character of time

Let us consider the bounded-response formula ϕ_{BR} again. A timed state sequence for ϕ_{BR} specifies for every state the truth values of the predicates p and q and the value of the time function f. Since f is interpreted as a monotonic function, it can be viewed as a flexible variable df_δ that records in every state the increase in time from the previous state:

$$df_\delta{}^I(i) \;=\; f^I(i) - f^I(i-1)$$

(throughout we use the convention that $f^I(-1) = 0$ for every interpretation I; thus, df_δ records the initial time in the first state of a timed state sequence). Although the variable df_δ ranges over the infinite domain N, if the time increases by more than 1 from a state to its successor, then the actual value of the increase is of no relevance to the truth of the bounded-response formula ϕ_{BR}. Consequently, to determine the truth of ϕ_{BR} in an interpretation, df_δ can be modeled using a finite number of unary *time-difference predicates*. We employ three new predicates $Tdiff_0$, $Tdiff_1$, and $Tdiff_{\geq 2}$ in the following way: $Tdiff_0$ is true in a state iff the time increase from the previous state is 0, $Tdiff_1$ is true iff $df_\delta = 1$, and $Tdiff_{\geq 2}$ is true iff $df_\delta \geq 2$. Accordingly, we define the notion of an extended state sequence for ϕ_{BR} as a state sequence over the propositions p, q, $Tdiff_0$, $Tdiff_1$, and $Tdiff_{\geq 2}$ such that precisely one of the time-difference propositions $Tdiff_0$, $Tdiff_1$, and $Tdiff_{\geq 2}$ is true in any state.

Given an extended state sequence, we can recover a corresponding timed state sequence: the value of the time function in any $Tdiff_\delta$-state and $Tdiff_{\geq delta}$-state is obtained by adding δ to its value in the previous state (if $Tdiff_\delta$ or $Tdiff_{\geq\delta}$ holds in the first state of the sequence, let δ be the time of the first state). This connection establishes a many-to-one correspondence between the timed state sequences and the extended state sequences for ϕ_{BR}; it induces an equivalence relation on the set of all interpretations for ϕ_{BR} such that the truth of ϕ_{BR} is invariant within each equivalence class. Moreover, every equivalence class is definable by a finite number of predicates.

For formulas with congruence primitives, we need to introduce, apart from time-difference predicates, also unary *time-congruence predicates*, to keep track of the congruence class of the time value of every state. For example, consider the following formula ψ, which states that "p is true in every state with an even time value":

$$\forall i.\, (f(i) \equiv_2 0 \rightarrow p(i)).$$

Given an interpretation I for ψ, the information in f^I can be captured by the two time-congruence predicates $Tcong_{2,0}$ and $Tcong_{2,1}$: the predicate $Tcong_{2,0}$ is true in states with even time, and $Tcong_{2,1}$ is true in states with odd time.

Now we formalize this idea. From ϕ, first obtain an equivalent ϕ' that does not contain the primitives 0 and +1 over the time sort (recall that both these primitives can be defined using $\leq$ and first-order variables). Let $d(\phi)$ be the least common multiple of the set $\{c \mid \equiv_c \text{ occurs in } \phi'\}$, and let $c(\phi)$ be the product of $d(\phi)$ and $3^{Q(\phi)}$, where $Q(\phi)$ is the number of time quantifiers (i.e., quantifiers over variables of the time sort) occurring in ϕ'. Given a formula ϕ of $\mathcal{L}_T^2$ with the free predicate symbols $p_1, \ldots p_n$, an *extended state sequence* J for ϕ specifies the sets $p_1^J, \ldots p_n^J \subseteq \mathsf{N}$, a partition of N into the sets $Tdiff_0^J, \ldots Tdiff_{c(\phi)-1}^J, Tdiff_{\geq c(\phi)}^J$, and another partition of N into the sets $Tcong_{d(\phi),0}^J, \ldots Tcong_{d(\phi),d(\phi)-1}^J$. For any interpretation I for ϕ, the extended state sequence J underlying I is defined as follows:

- J agrees with I on $p_1, \ldots p_n$.
- For all $i \geq 0$ and $0 \leq \delta < c(\phi)$, $i \in Tdiff_\delta^J$ iff $f^I(i) = f^I(i-1) + \delta$.
- For all $i \geq 0$, $i \in Tdiff_{\geq c(\phi)}^J$ iff $f^I(i) \geq f^I(i-1) + c(\phi)$.
- For all $i \geq 0$ and $0 \leq t < d(\phi)$, $i \in Tcong_{d(\phi),t}^J$ iff $f^I(i) \equiv_{d(\phi)} t$.

Lemma 1 (Finite-state character of time) *Given a formula ϕ of $\mathcal{L}_T^2$ and two interpretations I and J for ϕ with the same underlying extended state sequence, $I \in \mathcal{M}_T(\phi)$ iff $J \in \mathcal{M}_T(\phi)$.*

Proof. Consider two interpretations I and J for the $\mathcal{L}_T^2$-formula ϕ that have the same underlying extended state sequence; that is, I and J agree on the free predicate symbols of ϕ, and for each $i \geq 0$, $f^I(i)$ and $f^J(i)$ belong to the same congruence class modulo $d(\phi)$, and either $f^I(i) - f^I(i-1)$ is the same as $f^J(i) - f^J(i-1)$, or both are at least $c(\phi)$.

We use induction on the structure of ϕ to prove our claim. Recall that $c(\phi)$ and $d(\phi)$ are defined by counting the constants and quantifiers appearing in ϕ' obtained from ϕ by eliminating the primitives 0 and +1 over the time sort. Hence, without loss of generality, we assume that ϕ does not contain these primitives.

To handle subformulas with free individual variables properly, we need to strengthen our assumptions about the equivalence of interpretations with respect to a formula. Let ψ be a subformula of ϕ, possibly with free individual variables. Let $c(\psi)$ be the product of $d(\phi)$ and $3^{Q(\psi)}$, where $Q(\psi)$

is the number of time variables bound in ψ. For ease of presentation, we replace the function symbol f by the countable set $\{f_i \mid i \geq 0\}$ of new time variables: for any interpretation I, let $f_i^I = f^I(i)$. By $Tvar(\psi)$ we denote the union of the set $\{f_i \mid i \geq 0\}$ with the set of free time variables of ψ. Two interpretations I' and J' for ψ are equivalent with respect to ψ iff they satisfy the following conditions:

- For every predicate symbol p free in ψ, $p^{I'} = p^{J'}$.
- For every state variable i free in ψ, $i^{I'} = i^{J'}$.
- For all $x, y \in Tvar(\psi)$, $x^{I'} \leq y^{I'}$ iff $x^{J'} \leq y^{J'}$.
- For all $x, y \in Tvar(\psi)$, if either $0 \leq x^{I'} - y^{I'} < c(\psi)$ or $0 \leq x^{J'} - y^{J'} < c(\psi)$, then $x^{J'} - y^{J'} = x^{I'} - y^{I'}$.
- For every $x \in Tvar(\psi)$, $x^{I'} \equiv_{d(\phi)} x^{J'}$.

Clearly, the given two interpretations I and J are equivalent with respect to the given formula ϕ. Thus, it suffices to show that for any subformula ψ of ϕ and equivalent interpretations I' and J' for ψ, $I' \models \psi$ implies $J' \models \psi$. We do so by induction on the structure of ψ.

The interpretations I' and J' agree on the assignment to predicate symbols and state variables of ψ. They may assign different values to the elements in $Tvar(\psi)$, but they agree on their ordering and modulo-$d(\phi)$ congruence classes. Clearly, if ψ is an atomic formula, then $I' \models \psi$ iff $J' \models \psi$.

The case of boolean connectives is straightforward.

Suppose that ψ is of the form $\exists p.\, \psi'$, for a predicate symbol p, and that $I' \models \psi$. Let I'' be an extension of I' such that $I'' \models \psi'$. From the inductive hypothesis, the extension of J' that assigns the set $p^{I''}$ to p is a model of ψ'. Hence $J' \models \psi$. The case that ψ is of the form $\forall p.\, \psi'$ is similar.

If the outermost operator of ψ is a quantifier for a state variable, then we can proceed as in the previous case.

Now consider the case that ψ is of the form $\exists x.\, \psi'$, for a time variable x. Suppose that $I' \models \psi$. Let I'' be an extension of I' such that $I'' \models \psi'$. First note that $c(\psi') = d(\phi) \cdot 3^{Q(\psi)-1}$, and therefore $c(\psi) = 3c(\psi')$. We extend J' to an interpretation J'' for ψ' in the following way: if for some $y \in Tvar(\psi)$, $|y^{I'} - x^{I''}| < c(\psi')$, then choose $x^{J''}$ to be $y^{J'} + x^{I''} - y^{I'}$. Otherwise, let $y_1, y_2 \in Tvar(\psi)$ be such that $y_1^{I'} < x^{I''} < y_2^{I'}$. Note that $y_2^{I'} - y_1^{I'}$ is at least $c(\psi)$, and hence, so is $y_2^{J'} - y_1^{J'}$. We choose $x^{J''}$ between $y_1^{J'}$ and $y_2^{J'}$ at a distance at least $c(\psi')$ from either of them. Furthermore, since $c(\psi') \geq d(\phi)$, the difference between $c(\psi)$ and $2c(\psi')$ is at least $d(\phi)$, and we can require the modulo-$d(\phi)$ congruence class of $x^{J''}$ to be the same as that of $x^{I''}$. Now I'' and J'' satisfy the requirements listed above. Using the inductive hypothesis, $J'' \models \psi'$, and thus $J' \models \psi$. The case of universal quantification is similar. ∎

It follows that the extended state sequence that underlies a given interpretation for a $\mathcal{L}_T^2$-formula ϕ has enough information for deciding the truth of ϕ. Consequently, every formula ϕ can be viewed as characterizing a set $\mathcal{M}_T^*(\phi)$ of extended state sequences, instead of defining the set $\mathcal{M}_T(\phi)$ of timed state sequences. We say that the set

$$\mathcal{M}_T^*(\phi) \;=\; \{J \mid J \text{ underlies some } I \in \mathcal{M}_T(\phi)\}$$

contains the *untimed models* of ϕ.

Regular nature of the time primitives

Our next task is to show that the set $\mathcal{M}^*_T(\phi)$ of untimed models is ω-regular for every $\mathcal{L}^2_T$-formula ϕ. This is achieved by constructing a formula in the language $\mathcal{L}^2$ that defines the same set of extended state sequences. For instance, the untimed models of the bounded-response formula ϕ_{BR} are exactly the models of the $\mathcal{L}$-formula

$$\forall i.\,(p(i) \rightarrow \exists j \geq i.\,(q(j) \wedge (\psi_0(i,j) \vee \psi_1(i,j)))),$$

where

$$\begin{array}{rcl} \psi_0(i,j) & = & \forall k.\,(i < k \leq j \rightarrow \mathit{Tdiff}_0(k)), \\ \psi_1(i,j) & = & \exists k.\,(i < k \leq j \wedge \mathit{Tdiff}_1(k) \wedge \forall k' \neq k.\,(i < k' \leq j \rightarrow \mathit{Tdiff}_0(k'))). \end{array}$$

A generalization of this construction leads to the following theorem.

Theorem 1 (Regular nature of the time primitives) *For every formula ϕ of $\mathcal{L}^2_T$, there exists a formula ψ of $\mathcal{L}^2$ that contains the additional time-difference predicates $\mathit{Tdiff}_0, \mathit{Tdiff}_1, \ldots \mathit{Tdiff}_{\geq c(\phi)}$ and the time-congruence predicates $\mathit{Tcong}_{d(\phi),0}, \ldots \mathit{Tcong}_{d(\phi),d(\phi)-1}$, such that $\mathcal{M}^*_T(\phi) = \mathcal{M}(\psi)$. Furthermore, if $\phi \in \mathcal{L}_T$ then $\psi \in \mathcal{L}$.*

Proof. Given an $\mathcal{L}^2_T$-formula ϕ, we construct an equivalent (with respect to extended state sequences) $\mathcal{L}^2$-formula ψ in four steps.

First, we eliminate all time quantifiers. Let I be an interpretation for ϕ, and let $\Delta = c(\phi) + d(\phi)$. It is easy to find an interpretation J with the same underlying extended state sequence such that $f^J(i) \leq f^J(i-1) + \Delta$ for all $i \geq 0$. By Lemma 1, we know that $J \models \phi$ iff $I \models \phi$. Based on this observation we perform the following transformation: a subformula $\exists x.\,\psi(x)$, where x is a time variable, is replaced by the disjunction

$$\bigvee_{\delta=0}^{\Delta} \psi(\delta) \vee \exists i_x.\, \bigvee_{\delta=0}^{\Delta} \psi(f(i_x) + \delta),$$

for a new state variable i_x (the addition of a constant δ stands for δ applications of the successor function $+1$). Let ϕ' be the formula obtained from ϕ by applying the above transformation repeatedly until there are no time variables left; clearly $\mathcal{M}^*_T(\phi) = \mathcal{M}^*_T(\phi')$.

The second step, resulting in ϕ'', models the primitive time arithmetic of comparisons and addition by constants by the time-difference predicates. For instance, consider the subformula $f(i) + 1 \leq f(j)$, for state variables i and j. Intuitively, for $f(i)$ to be less than $f(j)$ in any interpretation, state i has to precede state j, and the time increase from the previous state has to be positive for some intermediate state. Hence we replace the subformula by

$$i < j \wedge \exists k.\,(i < k \leq j \wedge \neg \mathit{Tdiff}_0(k)).$$

Similarly, the subformulas $f(i) \leq f(j)$ and $f(i) \leq f(j) + 1$ can be replaced by $\psi_0(j,i)$ and $\psi_0(j,i) \vee \psi_1(j,i)$, respectively. The generalization of this technique to subformulas of the form $f(i) + c \leq f(j)$, $f(i) \leq f(j) + c$, $f(i) \leq c$, and $c \leq f(j)$, for arbitrary constants $c > 1$, is straightforward.

In a third step, we model the congruence primitives of ϕ'' with the help of the time-congruence predicates. Consider a subformula of the form $f(i) + c \equiv_d f(j)$. Since there are only a finite number of modulo-$d(\phi)$ congruence classes to which $f(i)$ and $f(j)$ can belong, we can use a case analysis to express this relationship. We replace the subformula by

$$\bigwedge_{k=1}^{d} \left(\bigvee_{k'=1}^{d(\phi)/d} Tcong_{d(\phi),(k+dk')\ mod\ d(\phi)}(i) \ \leftrightarrow \ \bigvee_{k'=1}^{d(\phi)/d} Tcong_{d(\phi),(k+c+dk')\ mod\ d(\phi)}(j) \right).$$

Subformulas of the form $f(i) \equiv_d c$ can be handled similarly.

Let ϕ''' be the formula that results from eliminating all time primitives in the described way. The desired $\mathcal{L}^2$-formula ψ is obtained by adding to ϕ''' the following conjuncts:

- For every state $i \geq 0$, precisely one of the time-difference predicates $Tdiff_0, \ldots Tdiff_{c(\phi)-1}$, $Tdiff_{\geq c(\phi)}$ is true.
- For every state $i \geq 0$, exactly one of the time-congruence predicates $Tcong_{d(\phi),0}, \ldots Tcong_{d(\phi),d(\phi)-1}$ is true.
- For all $i \geq 0$, the congruence classes of i and $i+1$, and the time jump $f(i+1) - f(i)$ are related in a consistent fashion:

$$\forall i. \bigwedge_{k=0}^{c(\phi)-1} \bigwedge_{k'=0}^{d(\phi)-1} \left(\begin{array}{l} Tdiff_k(i+1) \wedge Tcong_{d(\phi),k'}(i) \ \rightarrow \\ Tcong_{d(\phi),(k'+k)\ mod\ d(\phi)}(i+1) \end{array} \right). \blacksquare$$

Theorem 1, combined with the earlier stated facts about $\mathcal{L}^2$, gives the following important results regarding the decidability and expressiveness of the theory of timed state sequences.

Corollary 1 (Decidability) *The validity problem for the language $\mathcal{L}_T^2$ is decidable.*

Clearly, the validity problem is nonelementary already for the first-order language $\mathcal{L}_T$, as $\mathcal{L}$ is a fragment of $\mathcal{L}_T$ (recall that $\mathcal{L}$ is shown to be nonelementary in [Sto74]).

Corollary 2 (Expressiveness) *Given a formula ϕ of $\mathcal{L}_T^2$ with the free predicate symbols $p_1, \ldots p_n$, the set $\mathcal{M}_T^*(\phi)$ can be characterized by an ω-regular expression over the alphabet*

$$2^{\{p_1, \ldots p_n\}} \times \{Tdiff_0, \ldots Tdiff_{c(\phi)-1}, Tdiff_{\geq c(\phi)}\} \times \{Tcong_{d(\phi),0}, \ldots Tcong_{d(\phi),d(\phi)-1}\}.$$

Furthermore, if $\phi \in \mathcal{L}_T$, then $\mathcal{M}_T^*(\phi)$ can be defined by a star-free ω-regular expression.

2.4 Undecidable extensions and variants

We now justify our choice of $(\mathsf{N}, \leq, \equiv)$ as the theory of time, by showing that several formalisms for real-time reasoning with an expressive power greater than that of $\mathcal{L}_T^2$ are undecidable. In [AH89], we proved the Π_1^1-completeness of certain syntactic and semantic variants of the real-time temporal logic TPTL. Here, these results are refined, extended, and presented in the framework of the theory of timed state sequences.

Theorem 2 (Undecidable theories of real time) *The validity problems for the following two-sorted first-order theories are Π_1^1-complete:*[2]

[2]The class Π_1^1 consists of highly undecidable problems, including some nonarithmetical sets. For an exposition of the analytical hierarchy, consult [Rog67].

	state theory	*time theory*	*time function (from states to time)*
1	$(\mathsf{N}, \leq)$	$(\mathsf{N}, =, 0, +1)$	*(arbitrary)* f
2	$(\mathsf{N}, \leq)$ *with monadic predicates*	$(\mathsf{N}, =, 0, +1, \cdot 2)$	*identity* f
3	$(\mathsf{N}, \leq)$ *with monadic predicates*	$(\mathsf{D}, =, 0, S)$	*strictly monotonic* f
4	$(\mathsf{N}, \leq)$ *with monadic predicates*	$(\mathsf{N}, =, 0, +1)$	*identity* f *and strictly monotonic* f'

where $(\mathsf{D}, \prec)$ *is a dense linear order and the "successor" function* S *satisfies the following two properties (for all* $x, y \in \mathsf{D}$*):*

$$x \prec S(x),$$

$$x \prec y \rightarrow S(x) \prec S(y).$$

Proof. (1) First, we observe that the satisfiability of a formula ϕ can, in all cases, be phrased as a Σ_1^1-sentence that asserts the existence of a model for ϕ. For instance, in Case 2, an interpretation I for ϕ may be encoded in first-order arithmetic by finitely many sets of natural numbers; say, one for each unary predicate p in ϕ, characterizing the states in which p holds, and one to encode pairs of state numbers and associated times. It is a routine exercise to express, as a first-order formula, that ϕ holds in I. In Case 3, the Löwenheim-Skolem theorem ensures the existence of countable models, and again, elementary arithmetic can be used to encode such models. Thus the satisfiability problem is in Σ_1^1 in each case.

(2) Now let us show Σ_1^1-hardness. The problem of deciding if a nondeterministic Turing machine has, over the empty tape, a computation in which the start state is visited infinitely often, is Σ_1^1-complete [HPS83]. For ease of encoding, we prove our results using 2-counter machines instead of Turing machines.

A *nondeterministic 2-counter machine* M consists of two counters C and D, and a sequence of n instructions, each of which may increment or decrement one of the counters, or jump, conditionally upon one of the counters being zero. After the execution of a non-jump instruction, M proceeds nondeterministically to one of two specified instructions. We represent the configurations of M by triples $\langle l, c, d\rangle$, where $0 \leq l < n$, $c \geq 0$, and $d \geq 0$ are the current values of the location counter and the two counters C and D, respectively. The consecution relation on configurations is defined in the standard way. A *computation* of M is an infinite sequence of related configurations, starting with the initial configuration $\langle 0, 0, 0\rangle$. A computation is *recurring* iff it contains infinitely many configurations with the value of the location counter being 0.

The problem of deciding if a nondeterministic 2-counter machine has a recurring computation, is Σ_1^1-hard [AH89]. Thus, to show that the satisfiability problem for a language is Σ_1^1-hard, it suffices, given a nondeterministic 2-counter machine M, to construct a formula ϕ_M such that ϕ_M is satisfiable iff M has a recurring computation.

Σ_1^1-hardness of Case 1. We show that the monotonicity constraint on time is necessary for the decidability of $\mathcal{L}_T$; otherwise, the time function can be used to encode computations of M. We write a formula ϕ_M all of whose models correspond to recurring computations of M. An interpretation I encodes a computation of M iff for all $i \geq 0$, $f^I(3i) = l$, $f^I(3i+1) = n + c$, and $f^I(3i+2) = n + d$ for the i-th configuration $\langle l, c, d\rangle$ of the computation. First we specify the initial configuration by the formula

$$f(0) = 0 \wedge f(1) = n \wedge f(2) = n$$

(recall that the constant 0 and the successor function +1 over the state sort are definable from $\leq$ using first-order quantification). Then we ensure the proper consecution of configurations by adding a conjunct ϕ_l for every instruction $0 \leq l < n$ of M. For instance, the instruction 1 that increments the counter C and proceeds nondeterministically either to instruction 2 or to instruction 3, contributes the conjunct

$$\forall i.\,(f(i) = 1 \rightarrow ((f(i+3) = 2 \vee f(i+3) = 3) \wedge f(i+4) = f(i+1) + 1 \wedge f(i+5) = f(i+2))).$$

The recurrence condition can be expressed by the formula

$$\forall i.\,\exists j \geq i.\, f(j) = 0.$$

Let ϕ_M be the conjunction of these $n+2$ formulas. Then the formula ϕ_M is satisfiable iff M has a recurring computation. Note that ϕ_M contains only equality, the constant 0, and the successor function over the time sort, and no unary predicates. Case 1 follows.

Σ_1^1-hardness of Case 2. We show that a certain extremely modest relaxation of the timing constraints admitted in $\mathcal{L}_T$, namely allowing the multiplication by 2 over the time domain, leads to Σ_1^1-hardness. This result holds even under the restriction that the time function f is the identity function; that is, "time" acts as a state counter. If f is an arbitrary monotonic function, it suffices to require that

$$f(0) = 0 \;\wedge\; \forall i.\,(f(i+1) = f(i) + 1).$$

To encode computations of M, we use the unary predicates $p_1, \ldots p_n$, r_1, and r_2. We require that at most one of these predicates is true in any state; hence we may identify states with predicate symbols. The configuration $\langle l, c, d\rangle$ of M is represented by the finite sequence of states that starts with a p_l-state and contains c number of r_1-states and d number of r_2-states.

We write f_i for $f(i)$, and $i \in (x, y)$ for the condition $\exists j, k.\,(j < i < k \wedge f_j = x \wedge f_k = y)$, which asserts that the time of state i lies in the open interval (x, y). The initial configuration is defined by the formula

$$p_0(1) \wedge \forall i \in (1, 2).\, \neg(r_1(i) \vee r_2(i)).$$

The crucial property that allows a language to specify the consecution relation of configurations, and thus the set of computations of M, is the ability to copy an arbitrary number of r_j-states ($j = 1, 2$). With the availability of multiplication by 2, we can enforce that the i-th configuration of a computation corresponds, for all $i \geq 0$, to the finite sequence of states that is mapped to the half-open time interval $[2^i, 2^{i+1})$. We can then copy groups of r_j-states by establishing a one-to-one correspondence between r_j-states at time t and r_j-states at time $2t$. Clearly there are enough gaps to accommodate an additional r_j-state when required by an increment instruction.

For instance, the instruction 1 that increments the counter C and proceeds nondeterministically either to instruction 2 or to instruction 3, can be expressed as follows:

$$\forall i.\,(p_1(i) \rightarrow (\psi_1 \wedge \psi_2(r_1) \wedge \psi_3 \wedge \psi_2(r_2) \wedge \psi_4)),$$

where

$$\begin{array}{rcl}
\psi_1 & = & \exists j.\,(f_j = 2f_i \wedge (p_2(j) \vee p_3(j))), \\
\psi_2(r) & = & \forall j \in (f_i, 2f_i).\,(r(j) \rightarrow \exists k.\,(f_k = 2f_j \wedge r(k))), \\
\psi_3 & = & \exists j \in (2f_i, 4f_i).\left(\begin{array}{l} r_1(j) \wedge \forall k.\,(2f_k = f_j \rightarrow \neg r_1(k)) \wedge \\ \forall j' \in (2f_i, 4f_i).\,((j' \neq j \wedge r_1(j')) \rightarrow \exists k.\,(2f_k = f_{j'} \wedge r_1(k))) \end{array}\right), \\
\psi_4 & = & \forall j \in (2f_i, 4f_i).\,(r_2(j) \rightarrow \exists k.\,(2f_k = f_j \wedge r_2(k))).
\end{array}$$

The consequent of the implication ensures that given the configuration of M that is encoded by the states with times in the interval I_1: $[f_i, 2f_i)$, the states with times in the interval I_2: $[2f_i, 4f_i)$ encode the successor configuration that results from executing instruction 1. The first conjunct ψ_1 updates the location counter. The second conjunct $\psi_2(r_1)$ requires I_2 to contain at least as many r_1-states as I_1; together with the third conjunct ψ_3 it assures that I_2 has precisely one r_1-state more than I_1. The last two conjuncts $\psi_2(r_2)$ and ψ_4 assert that the number of r_2-states in I_2 is the same as in I_1.

The recurrence condition can be expressed by the formula

$$\forall i.\, \exists j \geq i.\, p_0(j).$$

Σ_1^1-hardness of Case 3. Now we attempt to model time as a dense linear order $(\mathit{TIME}, \prec) = (\mathsf{D}, \prec)$; that is, between any two given time points there is another time point. We show that even the simple arithmetic of equality and addition by a constant (S) leads to a highly undecidable theory. Common examples of a dense linear order are the rational numbers, and the real numbers. The result holds already for strictly monotonic time functions. If f is an arbitrary monotonic function, it suffices to require that

$$\forall i.\, (f(i) \neq f(i+1)).$$

As in the previous case, we employ the predicates $p_1, \ldots p_n$, r_1, and r_2, and encode the configuration $\langle l, c, d \rangle$ of M by a state sequence that starts with a p_l-state and contains c number of r_1-states and d number of r_2-states. The proof depends again on the ability to copy groups of r_j-states ($j = 1, 2$). This time, we are able to have the i-th configuration of a computation of M correspond, for all $i \geq 0$, to the finite sequence of states that is mapped to the time interval $[S^i(0), S^{i+1}(0))$, for some arbitrary element $0 \in \mathsf{D}$. Since f is strictly monotonic, every state has a unique time and we can establish a one-to-one correspondence between r_j-states at time t and time $S(t)$; the formula defining the recurring computations of M can be obtained from the formula constructed in Case 2 simply by replacing the operation $\cdot 2$ with S. Note that the denseness of the domain allows us to squeeze arbitrarily many states into any nonempty interval.

Σ_1^1-hardness of Case 4. This case corresponds to having two time bases (or "clocks") f and f' that are updated, from one state to the next, independently of each other. The result holds already for the special case in which f is the identity function and f' is strictly monotonic. The only primitives over the time sort used in our proof are equality and the successor function.

The encoding of M-computations is very similar to the one used in Case 2. We let the i-th configuration of a computation correspond to the sequence of 2^i states with times in the interval $[2^i, 2^{i+1})$. Even though the assertion language does not include the primitive of multiplication by 2, it can be simulated with the help of the second time function f'. We restrict ourselves to interpretations in which $f'(i) = 2i$ for all $i \geq 0$. This condition is enforced by the formula

$$f'(0) = 0 \wedge \forall i.\, (f'(i+1) = f'(i) + 2).$$

By replacing, in the formula constructed for Case 2, every term of the form $2f_i$ with $f'(i)$, we obtain again a formula that encodes the recurring computations of M. ■

Let us consider the ramifications of Theorem 2 for developing logics for reasoning about real-time systems. We look at the implications of all four cases of the theorem. The fact that the monotonicity constraint on the time function is required for decidability (Case 1) has little consequences in the context of real-time logics, as we are interested only in nondecreasing time functions anyway.

Choosing the domain of time

When designing a real-time logic we need to select an appropriate mathematical domain to represent time. Ideally, to model asynchronous systems, whose global state changes can be arbitrarily close in time, we would like to choose a dense linear order. Since the ordering predicate and addition by constant time values are the basic primitives needed to express the simplest of timing constraints, the undecidability of the resulting theory (Case 3) is a stumbling block in the design of useful logics over dense time. For example, the real-time branching-time logics considered in [ACD90] and [Lew90] use the set of real numbers to model time, and hence the corresponding validity problems are undecidable.[3]

Choosing the operations on time

Having constrained ourselves to a discrete time domain, we need to choose the operations on time that are admitted in a logic. We have proved (Case 2) that the addition of time variables causes undecidability. Indeed, using our results and techniques, we can show the Π_1^1-hardness of various real-time logics that have been proposed in the literature, including the logics of [JM86], [Har88], [Koy90], and [Ost90], all of which contain addition as a primitive operation on time. In [HLP90], decidability is proved for a real-time logic with addition; this logic puts, however, such substantial restrictions on the use of time quantifiers that it is not closed under negation.

The real-time logic RTL [JM86] can be viewed as a two-sorted logic with multiple monotonic functions from the state sort to the time sort. Our result (Case 4) implies that RTL is undecidable, even if we restrict its syntax to allow only the successor primitive over time (RTL allows addition over time).

On the other hand, we have shown that the congruence primitives over time can be added to the language without sacrificing decidability. Furthermore, we have proved decidability also for the second-order case. This is why we choose the first-order theory of $(\mathsf{N}, \leq)$ with monadic predicates (for state sequences) combined with the theory of $(\mathsf{N}, \leq, \equiv)$ (for time) as the theory of timed state sequences.

3 Timed Temporal Logic: TPTL

In [AH89], we introduced an extension of PTL that is interpreted over timed state sequences. We developed a tableau-based decision procedure and a model-checking algorithm for this *Timed Propositional Temporal Logic* (TPTL).

In this section, we study the expressiveness of TPTL. We compare the properties of timed state sequences that are expressible in the modal language TPTL with those that are expressible in the classical language $\mathcal{L}_T$. TPTL is shown to correspond to an expressively complete fragment of $\mathcal{L}_T$; that is, the set of models of any $\mathcal{L}_T$-formula can be defined by a TPTL-formula. This result is important as it establishes TPTL as a sufficiently expressive specification language; it shows that the improvement in complexity in moving from the full first-order theory of timed state sequences (nonelementary) to TPTL (EXPSPACE) is not achieved at the cost of expressive power.

We also look at two natural extensions of TPTL that correspond to larger fragments of $\mathcal{L}_T$ and, therefore, are still decidable. However, both generalizations turn out to be nonelementary.

[3] However, unlike linear-time logics, the cited branching-time logics over dense time have a decidable model-checking problem: given a description of a finite-state system A with timing constraints, and a dense branching-time specification ϕ, there are PSPACE algorithms to check if A satisfies ϕ.

TPTL can, on the other hand, be generalized to attain the full expressiveness of the second-order language $\mathcal{L}_T^2$ at no cost in complexity.

3.1 Syntax and semantics

We briefly recall the definition of TPTL. The real-time logic PTL is obtained from the temporal logic PTL by adding variables that refer to time. A time variable x is bound by a *freeze quantifier* "$x.$" that "freezes" the variable x to the time of the local temporal context. Let $\phi(x)$ be a formula in which the variable x occurs free. Then $x.\,\phi(x)$ is satisfied at the i-th position of the timed state sequence $\rho = (\sigma, \tau)$ iff $\phi(\tau(i))$ is satisfied at the i-th position of ρ (the formula $\phi(\tau(i))$ is obtained from $\phi(x)$ by replacing all free occurrences of the variable x with the constant $\tau(i)$). For example, in the formula $\Diamond x.\,\phi(x)$, the time reference x is bound to the time of the state at which ϕ is "eventually" true; the formula asserts that $\phi(\tau(i))$ is true at some position of a timed state sequence $\rho = (\sigma, \tau)$. Dually, the formula $\Box x.\,\phi(x)$ asserts that $\phi(\tau(i))$ is true at every position of ρ.

This extension of PTL with explicit references to the times of states admits the expression of timing constraints by atomic formulas that relate the times of different states. The formulas of TPTL are built from propositions and timing constraints by boolean connectives, temporal operators, and freeze quantifiers. For instance, the bounded-response requirement that "Every p-state is followed by a q-state within time 1" can be defined by the formula

$$\Box x.\,(p \to \Diamond y.\,(q \wedge y \leq x + 1)). \qquad (\phi'_{BR})$$

The application of TPTL as a specification language for real-time systems is discussed in [AH89, Hen91].

Syntax of TPTL

Given a finite set P of proposition symbols and a set V of (time) variables, the terms π and the formulas ϕ of TPTL are inductively defined as follows:

$$\pi := x + c \mid c$$

$$\phi := p \mid \pi_1 \leq \pi_2 \mid \pi_1 \equiv_d \pi_2 \mid \mathit{false} \mid \phi_1 \to \phi_2 \mid \bigcirc\phi \mid \phi_1 \mathcal{U} \phi_2 \mid x.\,\phi$$

for $x \in V$, $p \in P$, and integer constants $c \geq 0$ and $d \geq 2$.[4] Standard abbreviations include x (for $x + 0$), $=$, and $\wedge$; additional temporal operators are defined in terms of the *next* operator $\bigcirc$ and the *until* operator $\mathcal{U}$ as in PTL: $\Diamond\phi$ stands for $\mathit{true}\,\mathcal{U}\phi$, and $\Box\phi$ stands for $\neg\Diamond\neg\phi$.

Note that the timing constraints allow only the addition of integer constants to time variables, not the addition of variables. From a logical point of view, this restriction limits us to the successor operation on time. However, unlike $\mathcal{L}_T^2$, TPTL is not defined using the successor operator $+1$, because for determining the length of a TPTL-formula ϕ we assume that all integer constants in ϕ are given in a reasonably succinct (e.g., binary) encoding. The size of a formula is important for locating the computational complexity of problems whose input includes formulas of TPTL [AH89].

[4] TPTL as originally defined in [AH89] differs syntactically in that the freeze quantifiers are coupled with the temporal operators. This coupling does not restrict us in any essential way: by separating the freeze quantifier "$x.$" from the temporal operators, we admit more formulas (such as $\Box(x.\,\phi \to x.\,\psi)$), for each of which there is an equivalent formula in which every quantifier is either part of a prefix or follows a temporal operator ($\Box x.\,(\phi \to \psi)$).

Semantics of TPTL

The formulas of TPTL are interpreted over timed state sequences with $\mathit{TIME} = \mathsf{N}$.[5] Let $\rho = (\sigma, \tau)$ be a timed state sequence over P, let $i \in \mathsf{N}$, and let $\mathcal{E}: V \to \mathsf{N}$ be an interpretation (environment) for the variables. The triple $(\rho, i, \mathcal{E})$ satisfies the TPTL-formula ϕ iff $(\rho, i) \models_{\mathcal{E}} \phi$, where the satisfaction relation $\models$ is inductively defined as follows:

$(\rho, i) \models_{\mathcal{E}} p$ iff $p \in \sigma_i$;

$(\rho, i) \models_{\mathcal{E}} \pi_1 \leq \pi_2$ iff $\mathcal{E}(\pi_1) \leq \mathcal{E}(\pi_2)$;

$(\rho, i) \models_{\mathcal{E}} \pi_1 \equiv_d \pi_2$ iff $\mathcal{E}(\pi_1) \equiv_d \mathcal{E}(\pi_2)$;

$(\rho, i) \not\models_{\mathcal{E}} \mathit{false}$;

$(\rho, i) \models_{\mathcal{E}} \phi_1 \to \phi_2$ iff $(\rho, i) \models_{\mathcal{E}} \phi_1$ implies $(\rho, i) \models_{\mathcal{E}} \phi_2$;

$(\rho, i) \models_{\mathcal{E}} \bigcirc\phi$ iff $(\rho, i+1) \models_{\mathcal{E}} \phi$;

$(\rho, i) \models_{\mathcal{E}} \phi_1 \mathcal{U} \phi_2$ iff $(\rho, j) \models_{\mathcal{E}} \phi_2$ for some $j \geq i$, and $(\rho, k) \models_{\mathcal{E}} \phi_1$ for all $i \leq k < j$;

$(\rho, i) \models_{\mathcal{E}} x.\phi$ iff $(\rho, i) \models_{\mathcal{E}[x:=\tau(i)]} \phi$.

Here $\mathcal{E}(x + c) = \mathcal{E}(x) + c$ and $\mathcal{E}(c) = c$, and $\mathcal{E}[x := t]$ denotes the environment that agrees with $\mathcal{E}: V \to \mathsf{N}$ on all variables except x, which is mapped to $t \in \mathit{TIME}$.

A TPTL-formula ϕ is *satisfiable* [*valid*] iff $(\rho, 0) \models_{\mathcal{E}} \phi$ for some [every] timed state sequence ρ and some [every] environment $\mathcal{E}$. The truth value of a closed formula, which contains no free variables, is completely determined by a timed state sequence alone. Henceforth, we shall consider only closed formulas of TPTL. A timed state sequence ρ is a *model* of the (closed) formula ϕ iff $(\rho, 0) \models_{\mathcal{E}} \phi$, for any environment $\mathcal{E}$. By $\mathcal{M}_T(\phi)$ we denote the set of models of ϕ.

TPTL **as a fragment of the theory of timed state sequences**

Every TPTL-formula ϕ can be translated into the classical language $\mathcal{L}_T$, while preserving the set of models $\mathcal{M}_T(\phi)$. For every proposition p of TPTL, we use a corresponding unary state predicate $p(i)$ of $\mathcal{L}_T$. We translate a TPTL-formula ϕ to the $\mathcal{L}_T$-formula $F_0(\phi)$, where the mappings F_i, for $i \geq 0$, are defined by induction on the structure of TPTL-formulas:

$F_i(p) = p(i)$;

$F_i(\pi_1 \leq \pi_2) = \pi_1 \leq \pi_2$;

$F_i(\pi_1 \equiv_d \pi_2) = \pi_1 \equiv_d \pi_2$;

$F_i(\mathit{false}) = \mathit{false}$;

$F_i(\phi_1 \to \phi_2) = F_i(\phi_1) \to F_i(\phi_2)$;

$F_i(\bigcirc\phi) = F_{i+1}(\phi)$;

$F_i(\phi_1 \mathcal{U} \phi_2) = \exists j \geq i.\, (F_j(\phi_2) \wedge \forall i \leq k < j.\, F_k(\phi_1))$;

[5] In [AH89], timed state sequences are required to satisfy the progress condition that time diverges (i.e., the time function is unbounded). While the progress requirement make sense for any real-time semantics, it is definable within TPTL itself: $\Box x.\, \Diamond y.\, (y > x)$.

$F_i(x.\,\phi) = F_i(\phi)[x := f(i)]$; that is, the formula obtained from $F_i(\phi)$ by replacing all free occurrences of x by $f(i)$.

It is not hard to see that a TPTL-formula ϕ is true over a timed state sequence ρ iff the $\mathcal{L}_T$-formula $F_0(\phi)$ is true over ρ:

$$\mathcal{M}_T(\phi) \;=\; \mathcal{M}_T(F_0(\phi)).$$

For example, the bounded-response formula ϕ'_{BR} is equivalent to its translation $\phi_{BR} = F_0(\phi'_{BR})$:

$$\forall i \geq 0.\,(p(i) \rightarrow \exists j \geq i.\,(q(j) \wedge f(j) \leq f(i)+1)).$$

Note that the mapping F_0 embeds TPTL into $\mathcal{L}_T$; its range constitutes a proper subset of all well-formed $\mathcal{L}_T$-formulas. In particular, the translation does not introduce any time variables. Thus, just as PTL corresponds to a subset of $\mathcal{L}$, we may view TPTL as a fragment of $\mathcal{L}_T$: quantification over the state sort is restricted to the "temporal" way of PTL, and quantification over the time sort is prohibited entirely.

3.2 Expressive completeness

In [AH89] we showed that in a pleasing analogy to PTL versus $\mathcal{L}$, TPTL constitutes an elementary fragment of $\mathcal{L}_T$: the validity of a TPTL-formula with N logical and temporal operators, and K as the product of its constants, can be decided in time $2^{O(N \cdot K)}$. To complete this analogy, we show here that the restrictions imposed by TPTL on the quantification in $\mathcal{L}_T$-formulas do not diminish its expressive power. In other words, any property of timed state sequences that is definable in $\mathcal{L}_T$ can be defined in TPTL.

Theorem 3 (Expressive completeness of TPTL**)** *For every formula ϕ of $\mathcal{L}_T$, there is a formula ψ of* TPTL *such that $\mathcal{M}_T(\phi) = \mathcal{M}_T(\psi)$.*

Proof. Given an $\mathcal{L}_T$-formula ϕ, we construct an equivalent TPTL-formula ψ in four steps. By Theorem 1 we obtain an $\mathcal{L}$-formula ϕ', with additional time-difference predicates Tdiff_δ and $\mathit{Tdiff}_{\geq\delta}$ and time-congruence predicates $\mathit{Tcong}_{d,t}$, such that $\mathcal{M}^*_T(\phi) = \mathcal{M}(\phi')$. By the expressive completeness of PTL, there is a PTL-formula ϕ'' such that $\mathcal{M}(\phi') = \mathcal{M}(\phi'')$ [GPSS80].

We transform ϕ'' into an equivalent PTL-formula ϕ''' such that every time-difference proposition Tdiff_δ and $\mathit{Tdiff}_{\geq\delta}$ is either not within the scope of any temporal operator, or immediately preceded by a *next* operator. This can be done by repeatedly rewriting subformulas of the form $\bigcirc(\phi_1 \rightarrow \phi_2)$ and $\phi_1 \mathcal{U} \phi_2$ to $\bigcirc\phi_1 \rightarrow \bigcirc\phi_2$ and $\phi_2 \vee (\phi_1 \wedge (\bigcirc\phi_1)\mathcal{U}(\bigcirc\phi_2))$, respectively. From ϕ''' we arrive at ψ by replacing every time-difference proposition Tdiff_δ and $\mathit{Tdiff}_{\geq\delta}$ that is not within the scope of a temporal operator with $x.\,(x = \delta)$ and $x.\,(x \geq \delta)$, respectively; by replacing every subformula $\bigcirc\mathit{Tdiff}_\delta$ and $\bigcirc\mathit{Tdiff}_{\geq\delta}$ with $x.\bigcirc y.\,(y = x+\delta)$ and $x.\bigcirc y.\,(y \geq x+\delta)$, respectively; and by replacing every time-congruence proposition $\mathit{Tcong}_{d,t}$ with $x.\,(x \equiv_d t)$. ■

We conclude the discussion of properties expressible in TPTL by interpreting the logic over pure ("timeless") state sequences, and by investigating the expressive power of the congruence relations.

Timeless expressiveness

With every TPTL-formula ϕ we can associate a set of state sequences by collecting the state components of all models of ϕ. Given an infinite sequence σ of states and a TPTL-formula ϕ, let $\sigma \in \mathcal{M}_\exists(\phi)$ iff there is a time function τ such that $(\sigma, \tau) \in \mathcal{M}_T(\phi)$. When interpreted in this fashion, TPTL can define strictly more properties of state sequences than PTL. For example, the property $even(p)$, that "p holds in every even state," is not expressible in PTL [Wol83]. In TPTL, we may (ab)use time to identify the even states as precisely those in which the time does not increase:

$$x.\bigcirc y.\,(x = y) \wedge \Box x.\bigcirc y.\,(x = y \rightarrow (p \wedge \bigcirc z.\,(z > y \wedge \bigcirc u.\,(u = z)))).$$

The following theorem shows that the expressive power of TPTL with respect to state sequences is that of the second-order language $\mathcal{L}^2$ or, equivalently, ω-regular expressions.

Theorem 4 (Timeless expressiveness of TPTL**)** *For every formula ϕ of* TPTL*, there is a formula ψ of $\mathcal{L}^2$ such that $\mathcal{M}_\exists(\phi) = \mathcal{M}(\psi)$; and conversely, for every formula ψ of $\mathcal{L}^2$, there is a formula ϕ of* TPTL *such that $\mathcal{M}(\psi) = \mathcal{M}_\exists(\phi)$.*

Proof. (1) Given a TPTL-formula ϕ, we know how to construct an equivalent $\mathcal{L}_T$-formula ϕ'. By Theorem 1 we obtain an $\mathcal{L}$-formula ϕ'', with additional time-difference predicates $Tdiff_\delta$ and $Tdiff_{\geq\delta}$ and time-congruence predicates $Tcong_{d,t}$, such that $\mathcal{M}^*_T(\phi') = \mathcal{M}(\phi'')$. The $\mathcal{L}^2$-formula ψ that binds all of the new time predicates in ϕ'' by an existential prefix is easily seen to have the desired models.

(2) In order to show the second implication, we use a normal-form theorem for $\mathcal{L}^2$. Given an $\mathcal{L}^2$-formula ψ, there is an equivalent $\mathcal{L}^2$-formula ψ' of the form $\exists p_1 \ldots \exists p_n.\, \psi'_M$ whose matrix ψ'_M contains no second-order quantifiers [Büc62]. We construct a TPTL-formula ϕ that characterizes the models of ψ' by using the (existentially quantified) time function to encode the interpretation of the unary predicates p_j, $1 \leq j \leq n$, that are bound in ψ'.

Assign to every subset $J_\delta \subseteq \{1, \ldots n\}$ a unique code $\delta \in \mathsf{N}$. By the expressive completeness of PTL, there is a PTL-formula ψ''_M such that $\mathcal{M}(\psi'_M) = \mathcal{M}(\psi''_M)$ [GPSS80]. From ψ''_M, we obtain ϕ by replacing every proposition p_j, $1 \leq j \leq n$, with $x.\bigcirc y.\, \bigvee_{j\in J_\delta}(y = x+\delta)$. Now it is straightforward to establish a one-to-many correspondence between the models $I = (p_1^I, \ldots p_n^I)$ of ψ'_M and the timed state sequences (σ, τ) that satisfy ϕ: given I, let $\tau(i+1) = \tau(i) + \delta$ such that $J_\delta = \{j \mid i \in p_j^I\}$; and given τ, let $i \in p_j^I$ iff $j \in J_{\tau(i+1)-\tau(i)}$ (assume that $j \notin J_\delta$ if δ is no proper code). ■

It follows that $\mathcal{L}_T$, with the time function existentially quantified, has the full expressive power of the second-order language $\mathcal{L}^2$. In fact, the proof given above shows that equality and successor over the time sort are sufficient to achieve this timeless expressiveness.

Expressive power of congruences

If we disallow the use of congruence relations in TPTL, the resulting logic is strictly less expressive. Consider the following formula ϕ:

$$\Box x.\,(x \equiv_2 0 \rightarrow p).$$

This formula characterizes the timed state sequences in which "p is true at all even times." We show that this property is not expressible without congruence relations. Suppose that the TPTL-formula ψ, which does not contain any congruence relations, were equivalent to ϕ. Let $c-1$ be the largest constant that occurs in ψ. It is not difficult to see that ψ cannot distinguish between the timed state sequences $\rho_1 = (\sigma, \lambda i.\,(c+1))$ and $\rho_2 = (\sigma, \lambda i.\,(c+2))$, for any state sequence σ. Yet if p is not continuously true in σ, only one of ρ_1 and ρ_2 satisfies ϕ.

Note that TPTL without congruence relations has the same expressive power as the first-order language $\mathcal{L}_T$ without congruences. However, as has been pointed out above, the congruence primitives do not affect the "timeless" expressiveness of these formalisms. For example, we have demonstrated that the property that "p holds in every even *state*" (as opposed to every state with an even *time*) can be defined without congruences.

3.3 Nonelementary extensions

We have seen that TPTL restricts $\mathcal{L}_T$ to "temporal" quantification over the state sort and no quantification over the time sort. Can we relax these restrictions without sacrificing elementary decidability? Arbitrary quantification over the state sort encompasses full $\mathcal{L}$ and is, therefore, nonelementary. In this subsection, we first study the generalization of TPTL that admits quantification over the time sort, and show it to be nonelementary as well. Then we try to add past temporal operators to TPTL, an extension that does not affect the complexity of PTL. Therefore it is quite surprising that the past operators render TPTL nonelementary.

TPTL **with quantification over time**

Several authors, including [PH88], [Ost90], and [LA92], have proposed to use first-order temporal logic with a flexible variable *now*, which represents the time in every state, for the specification of real-time properties. For instance, they write the bounded-response requirement ϕ_{BR} as

$$\forall x.\, \Box((p \wedge now = x) \rightarrow \Diamond(q \wedge now \leq x+1)),$$

using rigid (global) time variables like x to refer to the time (i.e., the value of *now*) of different temporal contexts. Eliminating the flexible variable *now*, we see that this notation corresponds to TPTL extended with classical universal and existential quantification over time variables:

$$\forall x.\, \Box y.\, ((p \wedge y = x) \rightarrow \Diamond z.\, (q \wedge z \leq x+1)).$$

We call this generalization of TPTL, whose syntax definition is supplemented by the new clause

if ϕ is a formula and $x \in V$, then $\exists x.\, \phi$ is also a formula,

quantified TPTL or TPTL$_\exists$. Given a timed state sequence ρ, a position $i \geq 0$, and an environment $\mathcal{E}$, the classical quantifiers are interpreted as usual:

$(\rho, i) \models_{\mathcal{E}} \exists x.\, \phi$ iff $(\rho, i) \models_{\mathcal{E}[x:=t]} \phi$ for some $t \in \mathsf{N}$.

TPTL$_\exists$ seems, on the surface, more expressive than TPTL, because it can state properties of times that are not associated with any state. But it is easy to see that TPTL$_\exists$ can still be embedded into $\mathcal{L}_T$; let

$F_i(\exists x.\, \phi) \;=\; \exists x.\, F_i(\phi).$

From Corollary 1, it follows that the validity problem for TPTL$_\exists$ is decidable; and from Theorem 3, it follows that the expressive power of TPTL$_\exists$, measured as the sets of timed state sequences definable in the logic, is the same as that of TPTL. We show that TPTL$_\exists$ is, however, not elementarily decidable. This provides additional justification for our preference for TPTL over the existing notation with first-order quantifiers over time: prohibiting quantification over time not only leads, as argued in [AH89], to a more natural specification language, but is necessary for the existence of verification algorithms, such as the tableau techniques for TPTL.

Theorem 5 (Complexity of TPTL$_\exists$**)** *The validity problem for* TPTL$_\exists$ *is nonelementary.*

Proof. We translate the nonelementary monadic first-order theory of $(\mathsf{N}, \leq)$ [Sto74] into TPTL$_\exists$. With the help of the formula

$$\Box x.\, \bigcirc y.(y = x+1) \qquad (\phi_{+1})$$

we force time to act as a state counter, which allows us to simulate quantifiers over the state sort by the time quantifiers of $\text{TPTL}_\exists$. Given a formula ϕ of $\mathcal{L}$, we construct a formula ψ of $\text{TPTL}_\exists$ such that ϕ is valid iff the $\text{TPTL}_\exists$-formula $\phi_{+1} \rightarrow \psi$ is valid. The formula ψ is obtained from ϕ by replacing every atomic subformula of the form $p(i)$ with $\Diamond x. (p \wedge x = i)$ (read the state quantifiers of ϕ as quantifiers over the time sort). ■

TPTL **with past**

In [LPZ85], PTL is extended with the past temporal operators $\ominus$ (*previous*) and $\mathcal{S}$ (*since*), the past analogues of $\bigcirc$ and $\mathcal{U}$. These operators can be added at no extra cost, and although they do not increase the expressive power of PTL, they allow a more direct and convenient expression of many properties. Let TPTL_P be the logic that results from TPTL by adding the following clause to the inductive definition of formulas:

if ϕ_1 and ϕ_2 are formulas, then so are $\ominus\phi_1$ and $\phi_1 \mathcal{S} \phi_2$.

The meaning of the two past operators is given by

$(\rho, i) \models_{\mathcal{E}} \ominus\phi$ iff $i > 0$ and $(\rho, i-1) \models_{\mathcal{E}} \phi$;

$(\rho, i) \models_{\mathcal{E}} \phi_1 \mathcal{S} \phi_2$ iff $(\rho, j) \models_{\mathcal{E}} \phi_2$ for some $j \leq i$ and $(\rho, k) \models_{\mathcal{E}} \phi_1$ for all $j < k \leq i$.

TPTL_P can still be embedded into $\mathcal{L}_T$:

$$F_0(\ominus\phi) = \mathit{false};$$

$$F_{i+1}(\ominus\phi) = F_i(\phi);$$

$$F_i(\phi_1 \mathcal{S} \phi_2) = \exists j \leq i. (F_j(\phi_2) \wedge \forall j < k \leq i. F_k(\phi_1)).$$

Hence the validity problem for this logic is, again, decidable, and its expressive power is no greater than that of TPTL. However, unlike in the case of PTL, there is a heavy price to be paid for adding the past operators.

Theorem 6 (Complexity of TPTL_P) *The validity problem for* TPTL_P *is nonelementary.*

Proof. Again, we are able to use the nonelementary nature of the monadic first-order theory of $(\mathsf{N}, \leq)$. By adopting time as a state counter, we can simulate true existential quantification over time by $\Diamond$, because $\diamondminus$ ($\diamondminus\phi$ is defined as $\mathit{true}\mathcal{S}\phi$) allows us to restore the correct temporal context. Given a formula ϕ of $\mathcal{L}$, we construct a formula ψ of TPTL_P such that ϕ is valid iff the TPTL_P-formula $\phi_{+1} \rightarrow \psi$ is valid. The first step in translating ϕ is the same as in the proof of Theorem 5. In a second step we replace every subformula of the form $\exists x. \phi'$ by $y. \Diamond x. \diamondminus z. (z = y \wedge \phi')$. ■

3.4 Timed extended temporal logic

PTL does not have the full expressive power of the second-order language $\mathcal{L}^2$; recall that the property $\mathit{even}(p)$, that "p is true in every even state,"

$$\exists q. (q(0) \wedge \forall i. (q(i) \rightarrow (p(i) \wedge \neg q(i+1) \wedge q(i+2)))),$$

is not definable in PTL [Wol83]. This observation prompted the definition of *Extended Temporal Logic* (ETL), which includes a temporal operator for every right-linear grammar. ETL has the same expressiveness as $\mathcal{L}^2$ or, equivalently, ω-regular expressions, and yet a singly exponential decision procedure.

The situation for TPTL is similar: there is no TPTL-formula whose models are precisely the timed state sequences in which, independent of the time function, p holds at every even state. For suppose there were such a formula ϕ; we show that this would imply the expressibility of $even(p)$ in $\mathcal{L}$. First construct an $\mathcal{L}$-formula ϕ' that is equivalent to ϕ and contains the additional time-difference and time-congruence predicates $Tdiff_t$, $Tdiff_{\geq t}$, and $Tcong_{d,t}$, as usual. Then replace in ϕ' all occurrences of $Tdiff_t$, $Tdiff_{\geq t}$, and $Tcong_{d,t}$ by *true* or *false* depending on whether or not $t = 0$. This simplification does not affect the truth of the formula over interpretations all of whose times are permanently 0. Thus, the resulting formula ψ is satisfied by a state sequence σ iff $(\sigma, \lambda i.\, 0) \in \mathcal{M}_T(\phi)$; that is, iff p is true in every even state of σ.

Analogously to PTL, we are able to generalize TPTL to *Timed Extended Temporal Logic* (TETL) by introducing temporal grammar operators. We show that TETL has the full expressive power of $\mathcal{L}_T^2$, while being no more expensive than TPTL.

Syntax and semantics of TETL

Given a set P of propositions symbols and a set V of variables, the terms of TETL are those of TPTL. The formulas ϕ of TETL are inductively defined as follows:

$$\phi := p \mid \pi_1 \leq \pi_2 \mid \pi_1 \equiv_d \pi_2 \mid \mathit{false} \mid \phi_1 \rightarrow \phi_2 \mid \mathcal{G}(\phi_1, \ldots \phi_m) \mid x.\, \phi$$

where $x \in V$, $p \in P$, $d \geq 2$, and $\mathcal{G}(a_1, \ldots a_m)$ is a right-linear grammar with the m terminal symbols $a_1, \ldots a_m$.[6]

As with TPTL, TETL-formulas are interpreted over timed state sequences. Given a timed state sequence ρ, a position $i \geq 0$, and an environment $\mathcal{E}$, the semantics of the grammar operators is defined by the following clause:

$(\rho, i) \models_{\mathcal{E}} \mathcal{G}(\phi_1, \ldots \phi_m)$ iff there is a (possibly infinite) word $w = a_{w_0} a_{w_1} a_{w_2} \ldots$ generated by $\mathcal{G}(a_1, \ldots a_m)$ such that $(\rho, i+j) \models_{\mathcal{E}} \phi_{w_j}$ for all $j \geq 0$.

We restrict ourselves to closed formulas of TETL, and the timed state sequences that satisfy a (closed) formula ϕ are collected in the set $\mathcal{M}_T(\phi)$.

Note that all temporal operators of TPTL are definable by the grammar operators of TETL. For example, the *always* operator $\Box$ corresponds to the grammar $\mathcal{G}_{\Box}(a)$ with the only production

$$\mathcal{G}_{\Box}(a) \rightarrow a\, \mathcal{G}_{\Box}(a)$$

(we identify grammars with their starting nonterminal symbols). The property $even(p)$, which is not expressible in TPTL, can be stated as $\mathcal{G}_{even}(true, p)$, for the production

$$\mathcal{G}_{even}(a_1, a_2) \rightarrow a_1 a_2\, \mathcal{G}_{even}(a_1, a_2).$$

Complexity of TETL

By putting together the tableau methods for ETL [Wol83] and TPTL [AH89], we develop a doubly-exponential-time decision procedure for TETL. This procedure is near-optimal; we go on to show the validity problem for TETL to be EXPSPACE-complete.

[6] Like ETL, TETL can alternatively be defined using ω-automata connectives, instead of grammar operators [WVS83].

Our presentation follows [AH89] closely, but is kept terse; the interested reader should consult this reference for details. For the sake of keeping the presentation simple, we assume that all grammar operators correspond to productions of the form

$$\mathcal{G}(a_1, \ldots a_m) \rightarrow a_{i_1} \mid a_{i_2}\mathcal{G}'(a_{j_1}, \ldots a_{j_n}).$$

We are given a TETL-formula ϕ and wish to determine if ϕ is valid; that is, if the negated formula $\neg\phi$ is unsatisfiable. Let z be a variable that does not occur freely in ϕ, and obtain ϕ' from ϕ by replacing all variable-free terms c with $z + c$. It is not difficult to see that $\neg\phi$ is satisfiable iff TETL-formula $\phi'' = z.\bigcirc\neg\phi'$ in satisfiable. Moreover, ϕ'' contains no absolute time references; all timing constraints in ϕ'' can be simplified to be of the forms $x \leq y + c$, $x + c \leq y$, and $x \equiv_d y + c$, for integer constants $c \geq 0$ and $d \geq 2$.

As with TPTL, for checking the satisfiability of the TETL-formula ϕ'', we may restrict ourselves to timed state sequences $\rho = (\sigma, \tau)$ all of whose time steps $\tau(i+1) - \tau(i)$, $i \geq 0$, are bounded by the product K of all constants that occur in ϕ'' (a constant $c > 0$ occurs in ϕ'' iff ϕ'' contains a subformula of the form $x \leq y + (c-1)$ or $x + (c-1) \leq y$, or the predicate symbol $\equiv_c$). The time information in ρ has, therefore, finite-state character; it can be modeled by finitely many new propositions Tdiff_δ, $0 \leq \delta \leq K$, which represent the time differences δ between successive states. This observation allows us to modify the tableau-based decision procedure for ETL to handle formulas with time references. The decision procedure for ETL is, in fact, included in our procedure as the special case in which ϕ'' contains no timing constraints.

The key idea that underlies all tableau methods for temporal logics is that any formula can be split into two conditions: a present requirement on the initial state and a future requirement on the rest of the model. For example, the invariant $z.\Box\psi$ can be satisfied by both $z.\psi$ and $z.\bigcirc\Box\psi$ being true at the initial position of a timed state sequence. In order to propagate a real-time requirement on the successor state properly, all timing constraints need to be updated to account for the time increase δ from the initial state to its successor. Consider the formula $z.\bigcirc\Box\psi$. This condition is true at the initial position of a timed state sequence iff the next position satisfies the updated formula $z.\Box\psi[z := z - \delta]$ (where all free occurrences of z in ψ are replaced by $z - \delta$). If the number of conditions generated in this way is finite, checking for satisfiability of a formula is reducible to checking for satisfiability in a finite structure, the initial tableau. For $\delta > 0$, a naive replacement of z by $z - \delta$ would, however, successively generate infinitely many new formulas. Fortunately, the monotonicity of time can be exploited to keep the tableau finite. The observation that any variable x is always instantiated, in the "future," to a value greater than or equal to the "current" time z, allows us to simplify timing constraints of the form $z \leq x + c$ and $x + c \leq z$ to *true* and *false*, respectively.

Given a TETL-formula $z.\psi$ and $\delta \in \mathsf{N}$, we define the TETL-formula $z.\psi^\delta$ that results from updating all references z in ψ according to the time difference δ by induction on δ:

- $z.\psi^0$ equals $z.\psi$;
- $\phi^{\delta+1}$ is obtained from ϕ^δ by replacing every term of the form $z + (c+1)$ with $z + c$, and every subformula of the form $z \leq x + c$, $x + c \leq z$, and $z \equiv_d x + c$ with *true*, *false*, and $z \equiv_d x + ((c+1) \ mod \ d)$, respectively.

We collect all conditions that may arise by recursively splitting the formula ϕ'' into its present and future parts in the closure of ϕ''. The *closure* set $\mathit{Closure}(\phi'')$ of ϕ'' is the smallest set of formulas containing ϕ'' that is closed under the following operation Sub:

$$\mathit{Sub}(z.(\psi_1 \rightarrow \psi_2)) = \{z.\psi_1, z.\psi_2\};$$

$$Sub(z.\bigcirc\psi) \;=\; \{z.\,\psi^{\delta} \mid \delta \geq 0\};$$

$$Sub(z.\,\mathcal{G}(\psi_1,\ldots\psi_m)) \;=\; \{z.\,\psi_{i_1}, z.\,\psi_{i_2}, z.\bigcirc\mathcal{G}'(\psi_{j_1},\ldots\psi_{j_n})\};$$

$$Sub(z.\,x.\,\psi) = \{z.\,\psi[x := z]\}.$$

Note that all formulas in $Closure(\phi'')$ are closed. Let N be the number of connectives, quantifiers, and grammar operators in ϕ'', where every grammar operator is counted as the number of non-terminal symbols in the corresponding grammar, and recall that K is the product of all constants that occur in ϕ''. By induction on the structure of ϕ'', it can be shown that

$$|Closure(\phi'')| \;\leq\; 2N\cdot K.$$

Tableaux for TETL are finite, directed state graphs (Kripke structures) with local consistency constraints on all vertices. The vertices are labeled by consistent sets of formulas that are closed under "subformulas" and express conditions on the current state and its successor states. Every vertex contains, in addition, a single proposition $Tdiff_{\delta}$ that denotes the time difference δ from the predecessor states.

Formally, we define the vertices of a tableau for ϕ'' as the maximally consistent subsets of the finite universe

$$Closure^*(\phi'') \;=\; Closure(\phi'') \cup \{Tdiff_{\delta} \mid 0 \leq \delta \leq K\}$$

of TETL-formulas. A subset $\Phi \subseteq Closure^*(\phi'')$ is (maximally) *consistent* iff it satisfies the following conditions, where all formulas range only over the finite set $Closure^*(\phi'')$:

- $Tdiff_{\delta} \in \Phi$ for precisely one $0 \leq \delta \leq K$; this $\delta \in \mathsf{N}$ is referred to as δ_{Φ}^{-}.
- $z.\,(z \sim z + c) \in \Phi$ iff $0 \sim c$ holds in the natural numbers (for $\sim$ one of $\leq$, $\geq$, and $\equiv_d$).
- $z.\,false \notin \Phi$.
- $z.\,(\psi_1 \rightarrow \psi_2) \in \Phi$ iff either $z.\,\psi_1 \notin \Phi$ or $z.\,\psi_2 \in \Phi$.
- $z.\,\mathcal{G}(\psi_1,\ldots\psi_m) \in \Phi$ iff either $z.\,\psi_{i_1} \in \Phi$, or both $z.\,\psi_{i_2} \in \Phi$ and $z.\bigcirc\mathcal{G}'(\psi_{j_1},\ldots\psi_{j_n}) \in \Phi$.
- $z.\,x.\,\psi \in \Phi$ iff $z.\,\psi[x := z] \in \Phi$.

The *initial tableau* $\mathcal{T}(\phi'')$ for the formula ϕ'' is a directed graph whose vertices are the consistent subsets of $Closure^*(\phi'')$, and which contains an edge from Φ to Ψ iff for all formulas $z.\bigcirc\psi$ in $Closure(\phi'')$,

$$z.\bigcirc\psi \in \Phi \;\text{ iff }\; z.\,\psi^{\delta_{\Psi}^{-}} \in \Psi.$$

The significance of the (finite) initial tableau $\mathcal{T}(\phi'')$ for ϕ'' is that the models of ϕ'' correspond precisely to a certain class of infinite paths through $\mathcal{T}(\phi'')$. From an infinite path $\Phi_0\Phi_1\Phi_2\ldots$ through $\mathcal{T}(\phi'')$ we wish to construct a timed state sequence ρ such that for all formulas $z.\,\psi \in Closure(\phi'')$ and positions $i \geq 0$, the sequence ρ satisfies $z.\,\psi$ at position i iff $z.\,\psi \in \Phi_i$. This property is almost ensured by the local consistency conditions used in the construction of $\mathcal{T}(\phi'')$. But suppose that $z.\,\mathcal{G}_{\Box}(\psi) \in Closure(\phi'')$ and $z.\,\mathcal{G}_{\Box}(\psi) \notin \Phi_0$. Then for the "eventuality" $\neg z.\,\mathcal{G}_{\Box}(\psi)$ to be true at position 0, it is necessary that $z.\,\psi^{\Delta}$ is false at some later position $k \geq 0$, for a suitable accumulated time difference Δ, which is not guaranteed by the local consistency conditions (the scenario that $z.\,\psi^{\Delta} \in \Phi_k$ and $z.\,\mathcal{G}_{\Box}(\psi) \notin \Phi_k$ for all $k \geq 0$ is consistent with the definition of the initial tableau). Hence we need to consider only those infinite paths through $\mathcal{T}(\phi'')$ along which all eventualities are satisfied in time.

An eventuality $\neg z.\,\mathcal{G}(\psi_1, \ldots \psi_m)$ is *fulfillable* along the finite path $\Phi_0\Phi_1 \ldots \Phi_k$ iff either $z.\,\psi_{i_2} \notin \Phi_0$, or $k \geq 1$ and $\neg z.\,\mathcal{G}'(\psi_{j_1}, \ldots \psi_{j_n})^{\delta_{\overline{\Phi}_1}}$ is fulfillable along the path $\Phi_1\Phi_2 \ldots \Phi_k$. By combining the corresponding arguments for ETL and TPTL, it can be shown that the TETL-formula ϕ'' is satisfiable iff the initial tableau $\mathcal{T}(\phi'')$ contains an infinite path $\Phi_0\Phi_1\Phi_2 \ldots$ such that

- $\phi'' \in \Phi_0$;
- for all formulas $z.\,\mathcal{G}(\psi_1, \ldots \psi_m) \in \mathit{Closure}(\phi'')$ and all $i \geq 0$, if $z.\,\mathcal{G}(\psi_1, \ldots \psi_m) \notin \Phi_i$, then the eventuality $\neg z.\,\mathcal{G}(\psi_1, \ldots \psi_m)$ is fulfillable along some finite prefix $\Phi_i\Phi_{i+1} \ldots \Phi_k$, $k \geq i$.

This result suggests a decision procedure for TETL: to determine if the formula ϕ is valid, construct the initial tableau for ϕ'' and employ the standard polynomial-time techniques for checking if the tableau contains an infinite path along which all eventualities are satisfied [Wol83]. Since the initial tableau contains $O(K \cdot 2^{NK})$ vertices, each of size $O(N \cdot K)$, the graph $\mathcal{T}(\phi'')$ can be constructed and checked for infinite paths in deterministic time exponential in $O(N \cdot K)$.

Theorem 7 (Deciding TETL**)** *The validity problem for a formula ϕ of* TETL *can be decided in deterministic time exponential in $O(N \cdot K)$, where $N - 1$ is the number of connectives, quantifiers, and grammar operators in ϕ, and K is the product of all constants that occur in ϕ (every grammar operator in ϕ is counted as the number of nonterminal symbols in the corresponding grammar).*

Note that the length of a TETL-formula ϕ whose constants are represented in binary, is $O(N + \log K)$. It follows that the tableau-based decision procedure for TETL is, as in the case of TPTL, doubly exponential in the length of the input formula (although only singly exponential in N, the "untimed" part of ϕ, and thus, singly exponential for ETL). The algorithm outlined here may be improved along the lines of [Wol83] to avoid the construction of the entire initial tableau. Such improvements, however, cannot lower the complexity significantly; indeed, TETL is EXPSPACE-hard.

Theorem 8 (Complexity of TETL**)** *The validity problem for* TETL *is EXPSPACE-complete.*

Proof. To show that TETL is in EXPSPACE, we follow the argument that ETL is in PSPACE, which develops a nondeterministic version of the tableau decision procedure and then applies Savitch's theorem [Wol83]. EXPSPACE-hardness follows immediately from the corresponding result for TPTL [AH89]. ∎

Expressiveness of TETL

Although TETL is no harder to decide than TPTL, we have demonstrated that its expressive power is strictly greater, by defining the property $\mathit{even}(p)$. The following theorem characterizes the expressiveness of TETL as equivalent to the second-order language $\mathcal{L}^2_T$.

Theorem 9 (Expressiveness of TETL**)** *For every formula ϕ of* TETL*, there is a formula ψ of $\mathcal{L}^2_T$ such that $\mathcal{M}_T(\phi) = \mathcal{M}_T(\psi)$; and conversely, for every formula ψ of $\mathcal{L}^2_T$, there is a formula ϕ of* TETL *such that $\mathcal{M}_T(\psi) = \mathcal{M}_T(\phi)$.*

Proof. (1) We extend the translation F_0 that embeds TPTL into $\mathcal{L}_T$ to accommodate the grammar operators of TETL. The target formulas will contain second-order quantifiers over unary predicates, and thus belong to $\mathcal{L}_T^2$. Again, assume that all grammar operators correspond to productions of the form

$$\mathcal{G}(a_1, \ldots a_m) \;\rightarrow\; a_{i_1} \mid a_{i_2}\mathcal{G}'(a_{j_1}, \ldots a_{j_n}).$$

We add the following clause to the definition of F_k, $k \geq 0$:

$$F_k(\mathcal{G}_0(\phi_1, \ldots \phi_m)) \;=\; \exists p_{\mathcal{G}_0} \ldots \exists p_{\mathcal{G}_M}.\, (p_{\mathcal{G}_0}(k) \,\wedge\, \forall k' \geq k.\, \bigwedge_{0 \leq l \leq M} \phi_{\mathcal{G}_l}(k'))$$

for some new unary predicate symbols $p_{\mathcal{G}_0}, \ldots p_{\mathcal{G}_M}$, where $\mathcal{G}_0, \ldots \mathcal{G}_M$ are all nonterminal symbols that occur in the grammar $\mathcal{G}_0(a_1, \ldots a_m)$, and $\phi_{\mathcal{G}}(k)$ stands for the $\mathcal{L}_T^2$-formula

$$p_{\mathcal{G}}(k) \;\rightarrow\; (F_k(\phi_{i_1}) \,\vee\, (F_k(\phi_{i_2}) \,\wedge\, p_{\mathcal{G}'}(k+1))).$$

Consider an arbitrary timed state sequence ρ. We show, by induction on the structure of ϕ, that $(\rho, k) \models_{\mathcal{E}} \phi$ iff $(\rho, k) \models_{\mathcal{E}} F_k(\phi)$ for all positions $k \geq 0$ and all environments $\mathcal{E}$.

The crucial case that ϕ has the form $\mathcal{G}_0(\phi_1, \ldots \phi_m)$ is derived as follows. To establish the existence of appropriate predicates $p_{\mathcal{G}_l}$, $0 \leq l \leq M$, let $p_{\mathcal{G}_l}$ be true at position $k' \geq k$ iff $(\rho, k') \models_{\mathcal{E}} \mathcal{G}_l(\phi_1, \ldots \phi_m)$. On the other hand, given the predicates $p_{\mathcal{G}_l}$ satisfying $\phi_{\mathcal{G}_l}(k')$ for all $k' \geq k$, we can construct a word $w = a_{w_0}a_{w_1}a_{w_2}\ldots$ generated by $\mathcal{G}_0(a_1, \ldots a_m)$ such that $(\rho, k') \models_{\mathcal{E}} \phi_{w_{k'-k}}$. It follows that for any TETL-formula ϕ, the $\mathcal{L}_T^2$-formula $F_0(\phi)$ is equivalent to ϕ.

(2) The argument for the expressive completeness of TETL with respect to $\mathcal{L}_T^2$ is analogous to the corresponding proof for TPTL and $\mathcal{L}_T$ (use the expressive completeness of ETL with respect to $\mathcal{L}^2$). ■

Let us complete the expressibility picture by a few remarks. The *timeless* expressiveness of TETL is clearly again that of the second-order language $\mathcal{L}^2$, and thus no more than that of TPTL. It is also immediate that the congruence relations contribute even to the expressive power of TETL (and $\mathcal{L}_T^2$) in a nontrivial way; the property that "p is true at all even times" is still not definable without congruence relations.

TPTL **with quantification over propositions**

There are several alternatives to the grammar operators of ETL. PTL can be extended by fixed-point operators (thus obtaining a variant of the propositional μ-calculus [Koz83]) or by second-order quantification over propositions (QPTL of [Wol82, Sis83]) in order to achieve the full expressive power of $\mathcal{L}^2$. While fixed points can be viewed as generalized grammar operators and yield to tableau methods, QPTL is nonelementary. It is straightforward to show that both extensions have, indeed, the expected, analogous effect in the TPTL-framework; they give decidable real-time specification languages with the expressiveness of $\mathcal{L}_T^2$. However, *timed* QPTL is, as a superset of QPTL, nonelementary, and thus unsuitable for algorithmic verification.

4 Metric Temporal Logic: MTL

Several authors have tried to adapt temporal logic to reason about real-time properties by interpreting modalities as real-time operators. For example, [Koy90] suggests the notation $\Diamond_{\leq c}$ to express the notion "eventually within time c." Similar temporal operators that are parameterized with constant bounds have been used in [PH88] and in [EMSS89] (in the context of branching-time logics). In this section, we extend PTL by time-bounded temporal operators and interpret the resulting logic over timed state sequences. For example, the bounded-response property that "Every p-state is followed by a q-state within time 1" will be written as

$$\Box(p \rightarrow \Diamond_{\leq 1} q).$$

It is easy to see that we have, in fact, only obtained a notational variant of a subset of TPTL. For instance, the formula $\Diamond_{\leq c}\phi$ can be rewritten as $x.\Diamond y.(y \leq x + c \wedge \phi)$.

We show that PTL with bounded temporal operators is interesting, and worth studying in its own right, for two reasons. First, and surprisingly, it is already as expressive as full TPTL. And secondly, it may, unlike full TPTL, be enriched by past temporal operators without sacrificing its elementary decidability. Following [Koy90], we refer to PTL with bounded temporal operators as *Metric Temporal Logic* (MTL); the addition of past temporal operators yields MTL_P. We conclude that MTL_P, too, represents a suitable formalism for the specification and algorithmic verification of real-time systems: just like TPTL, MTL_P corresponds to an expressively complete and yet elementary fragment of $\mathcal{L}_T$. But the two subsets of $\mathcal{L}_T$ that correspond to TPTL and MTL_P, respectively, are not identical. Either logic can state certain properties more directly and succinctly than the other one, and may therefore be preferred for some specifications.

4.1 Syntax and semantics

Given a set P of propositions, the formulas ϕ of MTL_P are defined inductively as follows:

$$\phi := p \mid \mathit{false} \mid \phi_1 \rightarrow \phi_2 \mid \bigcirc_I \phi \mid \ominus_I \phi \mid \phi_1 \mathcal{U}_I \phi_2 \mid \phi_1 \mathcal{S}_I \phi_2$$

for $p \in P$.[7] The subscript I is one of the following:

1. An *interval* of N, whose end-points are integer constants. Intervals may be open, half-open, or closed; empty, bounded, or unbounded. We freely denote intervals by pseudo-arithmetic expressions. For example, the expressions $\leq c_1$ and $> c_2$ stand for the closed interval $[0, c_1]$ and the open interval (c_2, ∞), respectively. The expression $\pm\delta \pm I$, where I is an interval and $\delta \in \mathsf{N}$, denotes the shifted interval $\{\pm\delta \pm t \mid t \in I\}$.

2. A *congruence expression* of the form $\equiv_d c$, for integer constants $c \geq 0$ and $d \geq 2$. In this case, the expression $\pm\delta \pm I$ denotes the set $\{t \in \mathsf{N} \mid t \equiv_d c\}$.

As in the case of TPTL, all integer constants in an MTL_P-formula are given in a binary encoding. The defined operators $\Diamond_I \phi$ and $\Box_I \phi$ stand for $\mathit{true}\mathcal{U}_I \phi$ and $\neg\Diamond_I \neg\phi$, respectively. The logic MTL is the future fragment of MTL_P (i.e., without the *bounded-previous* and *bounded-since* operators $\ominus_I$ and $\mathcal{S}_I$).

The formulas of MTL_P are interpreted over timed state sequences with $\mathit{TIME} = \mathsf{N}$. Instead of giving MTL_P its own semantics, we translate every MTL_P-formula ϕ into a TPTL_P-formula $G(\phi)$:

$$G(p) = p;$$

$$G(\mathit{false}) = \mathit{false};$$

[7] We choose not to constrain the first arguments of the *until* and *since* operators by subscripts, because we feel that doing so would impair the readability of formulas and, as we shall see, would not increase the expressive power of the logic.

$$G(\phi_1 \to \phi_2) \;=\; G(\phi_1) \to G(\phi_2);$$

$$G(\bigcirc_I \phi) \;=\; x.\bigcirc y.\, (y \in x + I \,\wedge\, \phi);$$

$$G(\ominus_I \phi) \;=\; x.\ominus y.\, (y \in x - I \,\wedge\, \phi);$$

$$G(\phi_1 \mathcal{U}_I \phi_2) \;=\; x.\, (\phi_1 \mathcal{U}\, y.\, (y \in x + I \,\wedge\, \phi_2));$$

$$G(\phi_1 \mathcal{S}_I \phi_2) \;=\; x.\, (\phi_1 \mathcal{S}\, y.\, (y \in x - I \,\wedge\, \phi_2)).$$

Observe that for every possible subscript I, the expression $y \in x + I$ (or $y \in x - I$) can be written as an atomic formula of TPTL. For instance, if I is an interval of the form $[c_1, c_2)$, then the expressions $y \in x + I$ and $y \in x - I$ stand for the timing constraints $x + c_1 \leq y < x + c_2$ and $y + c_1 \leq x < y + c_2$, respectively, and if I is a congruence expression the form $\equiv_d c$, then the expressions $y \in x + I$ and $y \in x - I$ both stand for the timing constraint $y \equiv_d c$.

We take an MTL$_P$-formula ϕ to define the same property as the TPTL$_P$-formula $G(\phi)$:

$$\mathcal{M}_T(\phi) \;=\; \mathcal{M}_T(G(\phi))$$

is the set of models of ϕ.

Note that the *bounded-next* formula $\bigcirc_{=2}\, p$ is satisfied at position 0 of a timed state sequence iff at position 1, there is a p-state and its time is 2 greater than the time at position 0. A *bounded weak-next* operator is definable by duality: the formula $\neg\bigcirc_{=2} \neg p$ requires that the second state is a p-state only if the time increase between the first and the second state is 2. Similarly, for any time interval I, the formula $\Diamond_I\, p$ asserts that there is a p-state with a time that is within the interval I of the "current" time, and the formula $\Box_I\, p$ stipulates that all states in that interval are p-states (although there may be none). In particular, for all timed state sequences ρ and all positions $i \geq 0$:

$(\rho, i) \models \Diamond_I\, \phi$ iff $(\rho, j) \models \phi$ for some $j \geq i$ with $\tau(j) \in \tau(i) + I$;

$(\rho, i) \models \Box_I\, \phi$ iff $(\rho, j) \models \phi$ for all $j \geq i$ with $\tau(j) \in \tau(i) + I$.

On the other hand, the formula $\Diamond_{\equiv_2 1}\, p$ is true at position i of a timed state sequence iff p is true at some later position $j \geq i$ whose time $\tau(j)$ is odd (independent of the current time $\tau(i)$). While time intervals constrain the time differences between states, congruence expressions refer to the absolute times of states.

We usually suppress the universal interval $[0, \infty)$ as a subscript. The so constrained MTL$_P$-operators $\bigcirc, \ominus, \mathcal{U}$, and $\mathcal{S}$, coincide with the corresponding unconstrained future and past operators of PTL. Thus MTL$_P$ is, like TPTL, a conservative extension of PTL. Moreover, from our embedding of MTL$_P$ into TPTL$_P$, it follows that both TPTL and MTL$_P$ are orthogonal fragments of TPTL$_P$ and, hence, $\mathcal{L}_T$. While TPTL prohibits past operators, MTL$_P$ corresponds to a subset of TPTL$_P$ wherein all timing constraints relate only variables that refer to "adjacent" temporal contexts.

4.2 Complexity

We present a doubly-exponential-time decision procedure for MTL$_P$ and show that the validity problems for both MTL and MTL$_P$ are EXPSPACE-complete. This result establishes that MTL$_P$, like TPTL, corresponds also to an elementary fragment of the nonelementary first-order language $\mathcal{L}_T$.

We generalize the standard tableau-based decision procedure for PTL [BMP81, Wol82] to MTL_P. To deal with timing requirements, the tableau algorithm for MTL_P modifies the techniques developed for TPTL [AH89] for handling past operators. The crucial property that guarantees the finiteness of the tableau being constructed is that in both cases, the temporal precedence between any two temporal contexts that are related by a timing constraint is uniquely determined. For MTL_P-formulas, which may contain past operators, it is due to the monotonicity of time, and the fact that MTL_P can relate only adjacent temporal contexts.

Before giving a formal definition of the tableau method for MTL_P, we indicate first how the algorithm proceeds for a sample input. Suppose that the time increases by 1 from a state to its successor (in general, the time increase between states can be bounded for any given formula, and thus reduced to a finite number of different cases). In order to satisfy, say, the formula $\Diamond_{<c}\,\psi$ in the current state, we have to satisfy either ψ now, or $\Diamond_{<c-1}\,\psi$ in the succeeding state. Continuing this splitting of requirements into a present and a future part, we will eventually arrive at the condition $\Diamond_{<1}\,\psi$, which forces ψ to be satisfied in the current state. Since every input formula ϕ generates only a finite number of requirements on states in the described fashion, ϕ is satisfiable iff it is satisfiable in a finite tableau. By bounding the maximal size of this tableau, we obtain the following result.

Theorem 10 (Deciding MTL_P) *The validity problem for a formula ϕ of MTL_P can be decided in deterministic time exponential in $O(C\cdot N)$, where $N-1$ is the number of boolean and temporal connectives in ϕ, and $C-1$ is the largest constant that occurs in ϕ as an interval end-point.*

Proof. Suppose we are given an MTL_P-formula ϕ and wish to determine iff ϕ is valid; that is, iff its negation $\phi' = \neg\phi$ is satisfiable.

We define the *closure* set $\mathit{Closure}(\phi')$ of the formula ϕ' to be the smallest set containing ϕ' that is closed under the following operation Sub:

$$\begin{aligned}
\mathit{Sub}(\psi_1 \rightarrow \psi_2) &= \{\psi_1, \psi_2\};\\
\mathit{Sub}(\bigcirc_I\,\psi) &= \{\psi\};\\
\mathit{Sub}(\ominus_I\,\psi) &= \{\psi\};\\
\mathit{Sub}(\psi_1 \mathcal{U}_I\,\psi_2) &= \{\psi_1, \psi_2\} \cup \{\bigcirc(\psi_1 \mathcal{U}_{I-\delta}\,\psi_2) \mid \delta \geq 0\};\\
\mathit{Sub}(\psi_1 \mathcal{S}_I\,\psi_2) &= \{\psi_1, \psi_2\} \cup \{\ominus(\psi_1 \mathcal{S}_{I-\delta}\,\psi_2) \mid \delta \geq 0\}.
\end{aligned}$$

Note that if I is an interval bounded by the right end-point c_2, then $I-\delta = \emptyset$ for all $\delta > c_2$, and if I is an unbounded interval and has the left end-point c_1, then $I - \delta = \mathsf{N}$ for all $\delta > c_1$. On the other hand, if I is a congruence expression of the form $\equiv_d c$, then $I-\delta$ stands for the unchanged congruence expression $\equiv_d c$. It follows that the closure set of ϕ' is finite.

Let $N-1$ be the number of boolean and temporal operators in ϕ', and let $C-1$ be the largest constant that occurs in ϕ' as an interval end-point. It is not difficult to check that

$$|\mathit{Closure}(\phi')| \;\leq\; 2C\cdot N.$$

As with TPTL, for checking the satisfiability of ϕ', we may restrict ourselves to timed state sequences $\rho = (\sigma, \tau)$ all of whose time steps $\tau(i+1) - \tau(i)$, $i \geq 0$, are bounded by the product K of C and all constants d that occur within subscripts of the form $\equiv_d c$ in ϕ'. The time information in ρ has, therefore, finite-state character and can be modeled by the new time-difference propositions Tdiff_δ

and time-congruence propositions $Tcong_{K,t}$, for $0 \leq \delta \leq K$ and $0 \leq t < K$. As usual, the proposition $Tdiff_\delta$ represents in the initial state of a timed state sequence, the initial time δ, and in all other states, the time difference δ from the predecessor state; the proposition $Tcong_{K,t}$ represents in every state the remainder t modulo K of the current time. For ease of presentation we use, in addition, the time-difference propositions $Tdiff'_\delta$, for $0 \leq \delta \leq K$, to represent in every state the time difference δ to the successor state.

Let $Closure^*(\phi')$ denote the set that is obtained from $Closure(\phi')$ by adding all of the new propositions $Tdiff_\delta$, $Tdiff'_\delta$, and $Tcong_{K,t}$. A subset $\Phi \subseteq Closure^*(\phi')$ is (maximally) *consistent* iff it satisfies the following conditions, where all formulas range only over the finite set $Closure^*(\phi')$ (let I be an interval):

- $Tdiff_\delta \in \Phi$ for precisely one $0 \leq \delta \leq K$; this $\delta \in \mathsf{N}$ is referred to as δ^-_Φ.
- $Tdiff'_\delta \in \Phi$ for precisely one $0 \leq \delta \leq K$; this $\delta \in \mathsf{N}$ is referred to as δ^+_Φ.
- $Tcong_{K,t} \in \Phi$ for precisely one $0 \leq \delta < K$; this $\delta \in \mathsf{N}$ is referred to as γ_Φ.
- $false \notin \Phi$.
- $\psi_1 \rightarrow \psi_2 \in \Phi$ iff either $\psi_1 \notin \Phi$ or $\psi_2 \in \Phi$.
- $\psi_1 \mathcal{U}_I \psi_2 \in \Phi$ iff

  (1) $I \neq \emptyset$ and
  (2) either $0 \in I$ and $\psi_2 \in \Phi$, or $\psi_1 \in \Phi$ and $\bigcirc(\psi_1 \mathcal{U}_{I-\delta^+_\Phi} \psi_2) \in \Phi$.

- $\psi_1 \mathcal{U}_{\equiv_d c} \psi_2 \in \Phi$ iff either $\gamma_\Phi \equiv_d c$ and $\psi_2 \in \Phi$, or $\psi_1 \in \Phi$ and $\bigcirc(\psi_1 \mathcal{U}_{\equiv_d c} \psi_2) \in \Phi$.
- $\psi_1 \mathcal{S}_I \psi_2 \in \Phi$ iff

  (1) $I \neq \emptyset$ and
  (2) either $0 \in I$ and $\psi_2 \in \Phi$, or $\psi_1 \in \Phi$ and $\ominus(\psi_1 \mathcal{S}_{I-\delta^+_\Phi} \psi_2) \in \Phi$.

- $\psi_1 \mathcal{S}_{\equiv_d c} \psi_2 \in \Phi$ iff either $\gamma_\Phi \equiv_d c$ and $\psi_2 \in \Phi$, or $\psi_1 \in \Phi$ and $\ominus(\psi_1 \mathcal{U}_{\equiv_d c} \psi_2) \in \Phi$.

The *initial tableau* $\mathcal{T}(\phi')$ for the formula ϕ' is a directed graph whose vertices are the consistent subsets of $Closure^*(\phi')$, and which contains an edge from Φ to Ψ iff all of the following conditions are met

- $\delta^+_\Phi = \delta^-_\Psi$.
- $\gamma_\Psi \equiv_K \gamma_\Phi + \delta^+_\Phi$.
- For all $\bigcirc_I \psi \in Closure(\phi')$, $\bigcirc_I \psi \in \Phi$ iff $\delta^+_\Phi \in I$ and $\psi \in \Psi$.
- For all $\bigcirc_{\equiv_d c} \psi \in Closure(\phi')$, $\bigcirc_{\equiv_d c} \psi \in \Phi$ iff $\gamma(\Psi) \equiv_d c$ and $\psi \in \Psi$.
- For all $\ominus_I \psi \in Closure(\phi')$, $\ominus_I \psi \in \Psi$ iff $\delta^-_\Psi \in I$ and $\psi \in \Phi$.
- For all $\ominus_{\equiv_d c} \psi \in Closure(\phi')$, $\ominus_{\equiv_d c} \psi \in \Psi$ iff $\gamma(\Phi) \equiv_d c$ and $\psi \in \Phi$.

All models of ϕ' correspond to certain infinite paths through the initial tableau $\mathcal{T}(\phi')$ for ϕ', and vice versa. Namely, the formula ϕ' is satisfiable iff $\mathcal{T}(\phi')$ contains an infinite path $\Phi_0\Phi_1\Phi_2\ldots$ such that

- $\phi' \in \Phi_0$;
- Φ_0 contains no $\ominus$-formula;
- for all $i \geq 0$ and intervals I, $\psi_1 \mathcal{U}_I\, \psi_2 \in \Phi_i$ implies $\psi_2 \in \Phi_j$ for some $j \geq i$ with $\Sigma_{i \leq k < j} \delta^+_{\Phi_k} \in I$;
- for all $i \geq 0$, $\psi_1 \mathcal{U}_{\equiv_d c}\, \psi_2 \in \Phi_i$ implies $\psi_2 \in \Phi_j$ for some $j \geq i$ with $\gamma_{\Phi_j} \equiv_d c$.

The proof is similar to the corresponding argument for TPTL.

Since the initial tableau $\mathcal{T}(\phi')$ contains $O(K \cdot 2^{CN})$ vertices, each of size $O(C{\cdot}N)$, the graph $\mathcal{T}(\phi')$ can be constructed and checked for infinite paths in deterministic time exponential in $O(C \cdot N)$. ■

Note that although the worst-case running time of the tableau algorithm is slightly faster for MTL$_P$ than for TPTL (for which the product of all constants appears in the exponent), it is still doubly exponential in the length of the input formula. Still, both formalisms are EXPSPACE-complete.

Theorem 11 (Complexity of MTL$_P$**)** *The validity problems for* MTL *and* MTL$_P$ *are EXPSPACE-complete.*

Proof. From a nondeterministic version of the tableau algorithm, it follows that MTL$_P$ is in EXPSPACE. The corresponding lower bound for MTL can be shown similarly to the analogous result for TPTL, by simulating EXPSPACE-bounded Turing machines [AH89]. ■

4.3 Expressive completeness

Because of the past operators, MTL$_P$ can express certain properties more succinctly than TPTL. On the other hand, consider the following TPTL-formula, which asserts that "Every p-state is followed by a q-state and, later, an r-state within time 5:

$$\Box x.\, (p \rightarrow \Diamond(q \wedge \Diamond y.\, (r \wedge y \leq x + 5))).$$

This property has no natural expression in MTL$_P$. However, because of the discrete nature of the underlying time domain[8], the property can be translated into MTL as follows:

$$\Box(p \rightarrow \bigvee_{\delta=0}^{5} \Diamond_{=\delta}\, (q \wedge \Diamond_{\leq 5-\delta}\, r)).$$

Indeed, we show that the expressiveness of MTL is no less than that of TPTL in any crucial way. Only properties that put constraints on the time of the initial state of a timed state sequence, such as "The time of the initial state is 2" ($x.\,(x = 2)$ in TPTL), are not definable in our version of MTL$_P$. It can be argued that for the purpose of analyzing real-time systems, the absolute time of the initial state is of no importance.

Formally, a timed state sequence (σ, τ) *initial* iff the time of its initial state is 0; that is, $\tau(0) = 0$. The following theorem states that if the expressiveness of a real-time logic is measured by the sets of initial timed state sequences that are definable, then MTL (and MTL$_P$) has the same expressive power as $\mathcal{L}_T$ or, equivalently, TPTL.

Theorem 12 (Expressive completeness of MTL**)** *For every formula ϕ of $\mathcal{L}_T$, there is a formula ψ of* MTL *such that for every initial timed state sequence ρ, $\rho \in \mathcal{M}_T(\phi)$ iff $\rho \in \mathcal{M}_T(\psi)$.*

[8]If the underlying time domain is a dense linear order, then the given TPTL-property cannot be expressed in MTL$_P$(this result will be reported in the journal version of [AFH91]).

Proof. As in the proof of the expressive completeness of TPTL, given a formula ϕ of $\mathcal{L}_T$, we construct a PTL-formula ϕ' with additional time-difference propositions Tdiff_t and $\mathit{Tdiff}_{\geq t}$ and time-congruence propositions $\mathit{Tcong}_{d,t}$, such that $\mathcal{M}^*_T(\phi) = \mathcal{M}(\phi')$. Furthermore, in ϕ' all of the new propositions are either not within the scope of any temporal operator, or immediately preceded by a *next* operator.

From ϕ' we obtain the desired MTL-formula ψ by eliminating the time-difference and time-congruence propositions as follows. Since we consider only initial models, replace each proposition Tdiff_t, $\mathit{Tdiff}_{\geq t}$, and $\mathit{Tcong}_{d,t}$ that is not within the scope of any temporal operator by *true* or *false*, depending on whether or not $t = 0$. Then we replace each subformula $\bigcirc \mathit{Tdiff}_t$ by $\bigcirc_{=t}\, \mathit{true}$, each subformula $\bigcirc \mathit{Tdiff}_{\geq t}$ by $\bigcirc_{\geq t}\, \mathit{true}$, and each subformula $\bigcirc \mathit{Tcong}_{d,t}$ by $\bigcirc_{\equiv_d t}\, \mathit{true}$. (Observe that only the *next* operator needs to be subscripted with time constraints.) ■

5 Discussion

We showed that only a very weak arithmetic over a discrete domain of time can be combined with PTL to obtain decidable real-time logics. We then identified two ways of constraining the syntax further to find elementary real-time extensions of PTL with the full expressive power of the underlying classical theory of timed state sequences. We conclude that TPTL and MTL_P occupy a position among real-time logics that is as appealing as the standing of PTL for qualitative finite-state reasoning. However, both TPTL and MTL_P have EXPSPACE-complete decision problems. Our decision procedures are of a time complexity doubly exponential in the length of the timing constraints (although only singly exponential in the number of logical and temporal operators). PTL, on the other hand, is PSPACE-complete, and has a singly exponential decision procedure. We claim that this discrepancy is necessary because reasoning in $\mathcal{L}_T$ is intrinsically expensive.

A closer look at our proof of the EXPSPACE-hardness of TPTL [AH89] suggests that any extension of PTL that allows the expression of timing constraints of the form "The time of one state is within a certain constant distance c from the time of another state" ($\Box(p \rightarrow \Diamond_{=c}\, q)$ in MTL) is EXPSPACE-hard, provided that all time constants are encoded in binary. Even the identification of *next-time* with *next-state* (time as a state counter) is of no help in complexity: introducing the abbreviation $\bigcirc^k$ for a sequence of k consecutive *next* operators makes PTL EXPSPACE-hard. Thus the price of an extra exponential for real-time reasoning is caused by the succinctness of the binary encoding of integer constants.

In [NV92], it is claimed that already PTL is sufficiently expressive to specify real-time properties; indeed, it is claimed that PTL has the same expressive power as TPTL. At first glance, this may seem puzzling, and hence some clarification is in order. Our proof of Theorem 1 shows that for every $\mathcal{L}_T$-formula ϕ, the set $\mathcal{M}^*_T(\phi)$ of extended state sequences that satisfy ϕ can be defined by a $\mathcal{L}$-formula and, therefore, by a formula of PTL. This means that every TPTL-property can be characterized by a PTL-formula that explicitly refers to time-difference (and time-congruence) propositions. We believe that a specification language should separate the timing constraints clearly from the state propositions, and the role of time-difference propositions should be limited to verification algorithms and expressiveness proofs.

Since an preliminary version of this paper appeared [AH90], research on real-time logics has progressed. We will briefly review some of the recent results that build on the work presented here. The reviewed results fall into two areas: reasoning about dense time, and reasoning about multiple time lines. For a detailed survey of recent developments, we refer the reader to [AH92b].

Dense time. Our undecidability result for dense time domains (Subsection 2.4) highlights the obstacles in finding a decidable formalism for reasoning about dense time. For instance, if the logic MTL is interpreted over dense timed state sequences (e.g., timed state sequences in which the time values are chosen from the real numbers), the validity problem becomes undecidable. Indeed, a close inspection of the proof of Theorem 2 reveals that already a very simple class of real-time properties causes undecidability under a dense semantics: the only timing constraints required are conditions of the form (using MTL-notation)

$$\Box \bigcirc_{>0} true,$$

so that any time identifies a unique state, and of the form

$$\Box(p \rightarrow \Diamond_{=1} q),$$

to assert that "Every p-state is followed by a q-state precisely after time 1."

In [AFH91], we isolated a decidable fragment of MTL, which is called *Metric Interval Temporal Logic* (MITL). The syntax of MITL allows only *nonsingular* intervals as constraints on temporal operators. This syntactic restriction ensures that the time difference between two state changes can be enforced only with finite precision. In particular, punctuality requirements of the form $\Diamond_{=1} q$ cannot be defined in MITL. The logic MITL is interpreted over dense timed state sequences and has an EXPSPACE-complete decision problem. Recently, decidability has been shown also for the extension of MITL with past operators [AH92a]. Thus, there are two orthogonal options to escape the predicament that is caused by the trade-off between the realistic modeling of time and the ability to verify timing properties.[9] While MTL adopts the semantic abstraction of approximating real-time by the natural numbers, MITL pursues the syntactic approach of approximating punctuality requirements.

Multiple time lines. We showed that two independent time functions lead also to undecidability (Subsection 2.4). Independent time functions can represent the readings of different clocks in a distributed system. For instance, in a system with two independent clocks, the requirement that "Every p-state is followed by a q-state within 2 ticks of the first clock, and within 1 tick of the second" can be written in a TPTL-like notation as

$$\Box(x_1, x_2).\, (p \rightarrow \Diamond(y_1, y_2).\, (q \wedge y_1 \leq x_1 + 2 \wedge y_2 \leq x_2 + 1))). \qquad (\dagger)$$

Theorem 2 implies the undecidability of such a logic. A decidable extension of TPTL with multiple time lines has been introduced in [WME92]. There the syntax is constrained by the sort requirement that all variables within any atomic formula must refer to the same time line (note that this condition is satisfied by the formula (†)).

Acknowledgements. We thank David Dill, Zohar Manna, and Amir Pnueli for helpful discussions.

References

[ACD90] R. Alur, C. Courcoubetis, and D.L. Dill. Model checking for real-time systems. In *Proceedings of the Fifth Annual Symposium on Logic in Computer Science*, pages 414–425. IEEE Computer Society Press, 1990.

[AD90] R. Alur and D.L. Dill. Automata for modeling real-time systems. In M.S. Paterson, editor, *ICALP 90: Automata, Languages, and Programming*, Lecture Notes in Computer Science 443, pages 322–335. Springer-Verlag, 1990.

[9] Alternatively, an automata-theoretic approach has also yielded useful results towards obtaining a theory of dense timed state sequences [AD90, AH92a].

[AFH91] R. Alur, T. Feder, and T.A. Henzinger. The benefits of relaxing punctuality. In *Proceedings of the Tenth Annual Symposium on Principles of Distributed Computing*, pages 139–152. ACM Press, 1991.

[AH89] R. Alur and T.A. Henzinger. A really temporal logic. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 164–169. IEEE Computer Society Press, 1989.

[AH90] R. Alur and T.A. Henzinger. Real-time logics: complexity and expressiveness. In *Proceedings of the Fifth Annual Symposium on Logic in Computer Science*, pages 390–401. IEEE Computer Society Press, 1990.

[AH92a] R. Alur and T.A. Henzinger. Back to the future: towards a theory of timed regular languages. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*. IEEE Computer Society Press, 1992.

[AH92b] R. Alur and T.A. Henzinger. Logics and models of real time: a survey. In J.W. de Bakker, K. Huizing, W.-P. de Roever, and G. Rozenberg, editors, *Real Time: Theory in Practice*, Lecture Notes in Computer Science 600, pages 74–106. Springer-Verlag, 1992.

[BMP81] M. Ben-Ari, Z. Manna, and A. Pnueli. The temporal logic of branching time. In *Proceedings of the Eighth Annual Symposium on Principles of Programming Languages*, pages 164–176. ACM Press, 1981.

[Büc60] J.R. Büchi. Weak second-order arithmetic and finite automata. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 6:66–92, 1960.

[Büc62] J.R. Büchi. On a decision method in restricted second-order arithmetic. In E. Nagel, P. Suppes, and A. Tarski, editors, *Proceedings of the First International Congress on Logic, Methodology, and Philosophy of Science 1960*, pages 1–11. Stanford University Press, 1962.

[Eme90] E.A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 995–1072. Elsevier Science Publishers (North-Holland), 1990.

[EMSS89] E.A. Emerson, A.K. Mok, A.P. Sistla, and J. Srinivasan. Quantitative temporal reasoning. Presented at the First Annual Workshop on Computer-aided Verification, Grenoble, France, 1989.

[GPSS80] D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the temporal analysis of fairness. In *Proceedings of the Seventh Annual Symposium on Principles of Programming Languages*, pages 163–173. ACM Press, 1980.

[Har88] E. Harel. Temporal analysis of real-time systems. Master's thesis, The Weizmann Institute of Science, Rehovot, Israel, 1988.

[Hen91] T.A. Henzinger. *The Temporal Specification and Verification of Real-time Systems*. PhD thesis, Stanford University, 1991.

[HLP90] E. Harel, O. Lichtenstein, and A. Pnueli. Explicit-clock temporal logic. In *Proceedings of the Fifth Annual Symposium on Logic in Computer Science*, pages 402–413. IEEE Computer Society Press, 1990.

[HPS83] D. Harel, A. Pnueli, and J. Stavi. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 26(2):222–243, 1983.

[JM86] F. Jahanian and A.K. Mok. Safety analysis of timing properties in real-time systems. *IEEE Transactions on Software Engineering*, SE-12(9):890–904, 1986.

[Kam68] J.A.W. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California at Los Angeles, 1968.

[Koy90] R. Koymans. Specifying real-time properties with metric temporal logic. *Real-time Systems*, 2(4):255–299, 1990.

[Koz83] D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27(3):333–354, 1983.

[LA92] L. Lamport and M. Abadi. An old-fashioned recipe for real time. In J.W. de Bakker, K. Huizing, W.-P. de Roever, and G. Rozenberg, editors, *Real Time: Theory in Practice*, Lecture Notes in Computer Science 600, pages 1–27. Springer-Verlag, 1992.

[Lew90] H.R. Lewis. A logic of concrete time intervals. In *Proceedings of the Fifth Annual Symposium on Logic in Computer Science*, pages 380–389. IEEE Computer Society Press, 1990.

[LP84] O. Lichtenstein and A. Pnueli. Checking that finite-state concurrent programs satisfy their linear specification. In *Proceedings of the 11th Annual Symposium on Principles of Programming Languages*, pages 97–107. ACM Press, 1984.

[LPZ85] O. Lichtenstein, A. Pnueli, and L.D. Zuck. The glory of the past. In R. Parikh, editor, *Logics of Programs*, Lecture Notes in Computer Science 193, pages 196–218. Springer-Verlag, 1985.

[McN66] R. McNaughton. Testing and generating infinite sequences by a finite automaton. *Information and Control*, 9(5):521–530, 1966.

[MP71] R. McNaughton and S. Papert. *Counter-free Automata*. The MIT Press, 1971.

[MP92] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer-Verlag, 1992.

[NV92] E. Nassor and G. Vidal-Naquet. Suitability of the propositional temporal logic to express properties of real-time systems. In *Proceedings of the Ninth Symposium on Theoretical Aspects of Computer Science*, 1992.

[OL82] S. Owicki and L. Lamport. Proving liveness properties of concurrent programs. *ACM Transactions on Programming Languages and Systems*, 4(3):455–495, 1982.

[Ost90] J.S. Ostroff. *Temporal Logic of Real-time Systems*. Research Studies Press, 1990.

[PH88] A. Pnueli and E. Harel. Applications of temporal logic to the specification of real-time systems. In M. Joseph, editor, *Formal Techniques in Real-time and Fault-tolerant Systems*, Lecture Notes in Computer Science 331, pages 84–98. Springer-Verlag, 1988.

[Pnu77] A. Pnueli. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 46–57. IEEE Computer Society Press, 1977.

[Rog67] H. Rogers, Jr. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill Book Company, 1967.

[SC85] A.P. Sistla and E.M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, 1985.

[Sis83] A.P. Sistla. *Theoretical Issues in the Design and Verification of Distributed Systems.* PhD thesis, Harvard University, 1983.

[Sto74] L.J. Stockmeyer. *The Complexity of Decision Problems in Automata Theory and Logic.* PhD thesis, Massachusetts Institute of Technology, 1974.

[Tho81] Wolfgang Thomas. A combinatorial approach to the theory of ω-automata. *Information and Control*, 48(3):261–283, 1981.

[Tho90] W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 133–191. Elsevier Science Publishers (North-Holland), 1990.

[WME92] F. Wang, A.K. Mok, and E.A. Emerson. Asynchronous propositional temporal logic. In *Proceedings of the 12th ICSE*, 1992.

[Wol82] P. Wolper. *Synthesis of Communicating Processes from Temporal-Logic Specifications.* PhD thesis, Stanford University, 1982.

[Wol83] P. Wolper. Temporal logic can be more expressive. *Information and Control*, 56(1/2):72–99, 1983.

[WVS83] P. Wolper, M.Y. Vardi, and A.P. Sistla. Reasoning about infinite computation paths. In *Proceedings of the 24th Annual Symposium on Foundations of Computer Science*, pages 185–194. IEEE Computer Society Press, 1983.