March 1990

Report No. STAN-CS-90-1307

Real-Time Logics: Complexity and Expressiveness

by

Rajeev Alur and Thomas A. Henzinger

Departments of Computer Science and Medicine

Stanford University
Stanford, California 94305

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1a REPORT SECURITY CLASSIFICATION

1b RESTRICTIVE MARKINGS

2a SECURITY CLASSIFICATION AUTHORITY

3 DISTRIBUTION / AVAILABILITY OF REPORT

2b DECLASSIFICATION, DOWNGRADING SCHEDULE

4 PERFORMING ORGANIZATION REPORT NUMBER(S)
STAN - CS - 90 - 1307

5 MONITORING ORGANIZATION REPORT NUMBER(S)

6a NAME OF PERFORMING ORGANIZATION
DEPT. OF COMPUTER SCIENCE

6b OFFICE SYMBOL
(If applicable)

7a NAME OF MONITORING ORGANIZATION

6c. ADDRESS (City, State, and ZIP Code)
STANFORD UNIVERSITY
STANFORD, CA 94305

7b ADDRESS (City, State, and ZIP Code)
N00039-84-C-0211

8a NAME OF FUNDING / SPONSORING ORGANIZATION
DARPA

8b OFFICE SYMBOL
(If applicable)

9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c ADDRESS (City, State, and ZIP Code)
ARLINGTON, VA 22209

10 SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
REAL-TIME LOGICS: COMPLEXITY AND EXPRESSIVENESS

12 PERSONAL AUTHOR(S)
RAJEEV ALUR, THOMAS A. HENZINGER

13a TYPE OF REPORT

13b TIME COVERED
FROM __________ TO ____

14 DATE OF REPORT (Year, Month, Day)

15 PAGE COUNT

16 SUPPLEMENTARY NOTATION

17 COSATI CODES

FIELD	GROUP	SUB-GROUP

18 SUBJECT TERMS (Continue on reverse if necessary and rdentrfy by block number)

19 ABSTRACT

Abstract. The theory of the natural numbers with linear order and monadic predicates underlies propositional linear temporal logic. To study temporal logics for real-time systems, we combine this classical theory of infinite state sequences with a theory of 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 w-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. In fact, 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 give tableau-based decision procedures. Consequently, these two formalisms are well-suited for the specification and verification of real-time systems.

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT
□ UNCLASSIFIED/UNLIMITED □ SAME AS RPT □ DTIC USERS

21 ABSTRACT SECURITY CLASSIFICATION

22a NAME OF RESPONSIBLE INDIVIDUAL
ZOHAR MANNA

22b TELEPHONE (Include Area Code)
(415) 723-2273

22c OFFICE SYMBOL

Previous editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

Real-time Logics: Complexity and Expressiveness[1,2]

Rajeev Alur Thomas A. Henzinger
Department of Computer Science
Stanford University

March 15, 1990

Abstract. The theory of the natural numbers with linear order and monadic predicates underlies propositional linear temporal logic. To study temporal logics for real-time systems, we combine this classical theory of infinite state sequences with a theory of 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 w-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. In fact, 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 give tableau-based decision procedures. Consequently, these two formalisms are well-suited for the specification and verification of real-time systems.

1 Introduction

Linear propositional temporal logic (PTL) has been demonstrated to be **a** working tool for the specification and verification of reactive systems ([Pn77], [OL82], [LP84], [MP89]). Its practical appeal stems from the strong theoretical connections that PTL, which is interpreted over infinite sequences of states, enjoys with the underlying classical first-order theory of the natural numbers with linear order and monadic predicates: PTL captures an elementary, yet expressively

[1] This research was supported in part by an IBM graduate fellowship to the second author, by the National Science Foundation under grant CCR-8812595, 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 880281 and 90-0057.

[2] An abbreviated version of this paper appears in the proceedings of the 5th Annual IEEE Symposium on *Logic in Computer Science* (1990).

complete, fragment of this nonelementary theory ([SC85], [GPSS80], [St74]); that is, any property of state sequences expressible in the monadic first-order theory of $(\mathbb{N}, \leq)$ can also be specified in PTL, which has a much simpler decision problem.

PTL admits, however, only the specification of qualitative time requirements, such as an event occurring "eventually." 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 with every state ([JM86], [Os87], [Ha88], [AH89], [Ko89], [HLP90]). Even though the suitability as specification language has often been demonstrated, most of these previous attempts remain ad hoc, with little regard to complexity and expressiveness questions.

The prime objective of this paper is to develop a unifying framework for the study of real-time logics. 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 unary monotonic function that maps every state to its time. As for PTL, the monadic first-order theory of $(\mathbb{N}, \leq)$ serves as the theory of states. To model time, we choose the theory of $(\mathbb{N}, \leq, \equiv)$. We show that the resulting combined theory is still decidable, and characterize its expressiveness by w-regular sets.

We claim that this theory of timed state sequences is indeed *the* theory for reasoning about finite-state real-time systems. All 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 by [JM86], [Os87], [Ha88], and [Ko89] can be decided, which vividly demonstrates that it has not been understood, so far, how expressive a theory of time may be added, without sacrificing decidability, to reasoning about state sequences.

In [AH89], we proposed *timed* PTL (TPTL) as a natural specification language, and developed a tableau-based decision procedure. 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 this restriction of disallowing quantification over time, what yields readable specifications as well as finite-state-based verification methods. In Section

3 we show it 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 by quantification over time variables.

There are, in fact, *second-order* versions of all our theorems: the second-order theory of timed state sequences is still decidable, and just as PTL is generalizable to ETL ([Wo83]), TPTL can be extended to be as expressive as this second-order theory, at no cost in complexity.

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

Both TPTL and MTL are, while being elementary, still quite expensive; the respective decision procedures work in doubly exponential time. In Section 5 we show that this cost is, however, intrinsic to real-time reasoning: any reasonably succinct and reasonably expressive extension of PTL is necessarily EXPSPACE-hard. Even the special case of identifying *nezt-time* with *nezt-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 of propositions P and a time domain *TIME,* a *timed state sequence* $\rho = (\sigma, \tau)$ is a pair consisting of an infinite sequence CT of states $\sigma_i \subseteq P$, i $\geq$ 0, and a map $\tau : \mathbb{N} \rightarrow TIME$ that associates a time with every state. We introduce the classical theory of timed state sequences, show its decidability, and characterize its expressiveness by o-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.

Let $\mathcal{L}^2$ be the second-order language with unary predicate symbols and the binary predicate symbol $\leq$, and let $\mathcal{L}$ be its first-order fragment. We interpret $\mathcal{L}^2$ over the natural numbers, with $\leq$ being interpreted as the usual linear order. 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 $p_1^I, \ldots p_n^I \subseteq \mathbb{N}$. 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 $i \in p_k^I$). By $\mathcal{M}(\phi)$ we denote the set of state sequences that satisfy ϕ.

Observe that $\mathcal{L}^2$ is essentially the language underlying 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 (and vice versa). It was first shown by Biichi that the theory S1S is decidable ([Bü62]).

Formulas of the propositional linear temporal logic PTL can be faithfully translated into $\mathcal{L}$, by replacing propositions with monadic predicates. For example, the typical response property that "Every pstate is followed by a q-state," is expressed in PTL as

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

It can be written in $\mathcal{L}$ as

$$\text{Vi. (p(i)} \rightarrow \exists j \geq i . q(j)), \qquad (\phi_R)$$

without changing the set of models.

Although PTL corresponds to a proper subset of $\mathcal{L}$, it has the full expressive power of $\mathcal{L}$ ([Ka68], [GPSS80]); that is, for every L-formula there is a PTL-formula specifying the same property of state sequences. Furthermore, the validity problem for $\mathcal{L}$ is nonelementary ([St74]), whereas PTL is only PSPACE-complete ([SC85]), and has a singly exponential decision procedure ([BMP81]).

To attain the greater expressive power of $\mathcal{L}^2$, PTL may be strengthened by adding operators that correspond to right-linear grammars ([Wo83]). 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 w-regular expressions ([Mc66], [Th81]): for any formula ϕ of $\mathcal{L}^2$, the set $\mathcal{M}(\phi)$ can be defined by an w-regular expression over the alphabet $\mathcal{P}(\{p_1, \ldots p_n\})$. For example, $\mathcal{M}(\phi_R)$ is described by the expression

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

The restricted expressive power of $\mathcal{L}$ corresponds to the star-free fragment of w-regular expressions (in which the Kleene star may be applied only to the expression true).

2.2 Adding time to state sequences

To obtain a theory of *timed* state sequences, we need to identify a suitable time domain *TIME*, with appropriate primitives, and couple the theory of state sequences with this theory of time through a unary ("time") function f, which associates a time with every state. We choose, as the theory of time, the theory of the natural numbers (i.e., $TIME = $ N) 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.

Let $\mathcal{L}_T^2$ be a second-order language with two sorts, namely a *state* sort and a *time* sort. The vocabulary of $\mathcal{L}_T^2$ consists of unary predicate symbols and the binary predicate symbol $\leq$ over the state sort, the unary function symbol f from the state sort into the time sort, and the binary predicate symbols $\leq$, $\equiv_2, \equiv_3, \ldots$ over the time sort. By $\mathcal{L}_T$ we denote the first-order fragment of $\mathcal{L}_T^2$.

We restrict our attention to structures that choose the set of natural numbers N as domain for both sorts, and interpret the primitives in the intended way. Thus, 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 a monotonic function $f^I : \mathsf{N} \rightarrow TIME$. The satisfaction relation is defined as usual. Every interpretation I for ϕ can be viewed as a timed state sequence (a, τ) (choose σ as in the untimed case, and let $\tau = f^I$); by $\mathcal{M}_T(\phi)$ we denote the set of timed state sequences that satisfy ϕ.

It follows that $\mathcal{L}_T^2$-formulas specify 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 written as a formula of C_T:

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

(note that the successor functions, over either sort, are definable in $\mathcal{L}_T$).

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 sentences of $\mathcal{L}_T^2$. We prove it to be decidable.

2.3 Decidability and expressibility

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.

Let us consider the sample 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. Since f is interpreted as a monotonic function, it can be viewed as a state variable f_δ recording, in every state, the increase in time from the previous state. Although f_δ ranges over the infinite domain N, observe that 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 ϕ_{BR}.

Consequently, to determine the truth of ϕ_{BR}, the state variable f_δ can be modeled using a *finite* number of unary *time-difference* predicates. We employ the three new predicates $Tdiff_0$, $Tdiff_1$, and $Tdiff_2$ in the following way: $Tdiff_0$ is true of a state iff the time increase from the previous state is 0, $Tdiff_1$ is true iff it is 1, and $Tdiff_2$ is true iff it is greater than 1. 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 $Tdiff_2$ such that precisely one of the propositions $Tdiff_0$, $Tdiff_1$, and $Tdiff_2$ 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 a $Tdiff_t$-state is obtained by adding t to its value in the previous state (if $Tdiff_t$ holds in the first state, let t be its time). This establishes a many-to-one correspondence between the timed 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 any equivalence class. Every equivalence class is, furthermore, definable by a finite number of propositions.

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" :

$$\text{Vi. } (f(i) \equiv_2 0 \rightarrow p(i)).$$

Given an interpretation I for ψ, the information in f^I can be captured by the two predicates $Tcong_0$ and $Tcong_1$: $Tcong_0$ is true for states with even time, and $Tcong_1$ is true for states with odd time.

Now we formalize this idea. Let $c(\phi)$ be the least common multiple of the set $\{c \mid \equiv_c \text{ occurs in } \phi\}$, and $d(\ \phi)$ the product of $c(\phi)$ and 4^Q, where Q is the number of time quantifiers (i.e., quantifiers over variables of the time sort) occurring in ϕ.

Given a formula ϕ of $\mathcal{L}^2_T$ 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 $Tdiff^J_0, \ldots Tdiff^J_{d(\phi)}$, and another partition of N into the sets $Tcong^J_0,.\ \ Tcong^J_{c(\phi)-1}$. 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 i $\geq$ 0 and $0 \leq t < d(\phi)$, i $\in Tdiff^J_t$ iff $f(i) = f^I(i - 1) + t$.
- For $i \geq 0$, $i \in Tdiff^J_{d(\phi)}$ iff $f(i) \geq f^I(i - 1) +$ d(4).
- For i $\geq$ 0 and $0 \leq t < c(\phi)$, i $\in Tcong^J_t$ iff $f(i) \equiv_{c(\phi)} t$.

(Throughout we use the convention that, for any interpretation I, $f^I(-1) = 0$.)

Lemma [Finite-state character of time]. *Given a formula ϕ of $\mathcal{L}^2_T$ 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 $\mathcal{L}^2_T$-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, $f(i)$ and $f^J(i)$ belong to the same congruence class modulo $c(\phi)$, and either $f'(i) - f^{I}(i-1)$ is the same as $f^{J}(i) - f^{J}(i-1)$, or both are at least $d(\phi)$.

We use induction on the structure of ϕ to prove our claim. To handle subformulas with free 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 variables. Let $d(\psi)$ be the product of $c(\phi)$ and 4^{Q}, where Q is the number of time variables bound in ψ. For ease of presentation, we represent the function f by the countable set of variables $\{f_i \mid i \geq 0\}$: for any interpretation I, let $f_i^I = f'(i)$. By $Tvar(\psi)$ *we* denote the union of the set of free time variables of ψ with $\{f_i : i \geq 0\}$. We say that two interpretations I' and *J ’* for ψ are equivalent with respect to ψ iff they satisfy the following conditions:

- For every predicate symbol q free in ψ, $q^{I'} = q^{J'}$.
- For every state variable ϰ free in ψ, $i^{I'} = i^{J'}$.
- For all x, y $\in Tvar(\psi)$, $x^{I'} \leq$ *y”* iff $x^{J'} \leq y^{J'}$.
- For every x, y $\in Tvar(\psi)$, if $0 \leq x^{I'} -$ y” $< d(\psi)$, then $x^{J'} - y^{J} = x^{I'} - y^{I'}$, and vice versa.
- For every x $\in Tvar(\psi)$, $x^{I'} \equiv_{c(\phi)} x^{J'}$.

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 ϕ.

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-c(ϕ) 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 3x. ψ', 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 $d(\psi') = c(\phi)\ .\ 4^{Q-1}$. We extend *J ’* to an interpretation J'' for ψ' in the following way: if for ‘some y $\in Tvar(\psi)$, $|y^{I'} - x^{I''}| < d(\psi')$, 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 $d(\psi)$, 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 $d(\ \psi')$ from either of them. Furthermore, since the difference between $d(\psi)$ and $2d(\psi')$ is at least $c(\phi)$, we can require the modulo-$c(\phi)$ 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 hence, *J’* $\models \psi$. The case of universal quantification is similar. ■

It follows that the extended state sequence underlying a given interpretation for a $\mathcal{L}^2_T$-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 satisfying extended state sequences, instead of a set of satisfying timed state sequences. Our next task is to show that this set is w-regular. This is achieved by constructing a formula in the language $\mathcal{L}^2$ that is satisfied by the same extended state sequences.

For instance, the extended state sequences that satisfy ϕ_{BR} are the same as the models of the following formula:

$$\forall i. \left[p(i) \rightarrow \exists j \geq i. \left(q(j) \wedge \left(\begin{array}{c} \forall k. (i < k \leq j \rightarrow Tdiff_0(k)) \vee \\ \exists k. \left(\begin{array}{c} i < k \leq j \wedge Tdiff_1(k) \wedge \\ \forall k' \neq k. \left(\begin{array}{c} i < k' \leq j \rightarrow \\ Tdiff_0(k') \end{array} \right) \end{array} \right) \end{array} \right) \right) \right].$$

Theorem [Regular nature of the time primitives]. *Given a formula ϕ of $\mathcal{L}^2_T$, there exists a formula ψ of $\mathcal{L}^2$, with* additional *time-difference* predicates $Tdiff_0, \ldots Tdiff_{d(\phi)}$ *and* time-congruence *predicates* $Tcong_0, \ldots Tcong_{c(\phi)-1}$, *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 $t = d(\phi)+c(\phi)$. We can easily find an interpretation J with the same underlying extended state sequence, such that $f^J(i) \leq f^J(i-1) + t$ for all i $\geq$ 0. By the previous lemma, we know furthermore that $J \models \phi$ **iff** $I \models \phi$. Based on this observation we perform the following transformation: a subformula $\exists y. \psi(y)$, where y is a time variable, is replaced by the disjunction

$$\underset{k=0}{v} \psi(k) \vee \exists i_y. \bigvee_{k=0}^{t} \psi(f(i_y) + k),$$

for a new state variable i_y. Let ϕ' be the formula obtained from ϕ by applying the above transformation repeatedly until there are no time quantifiers 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

$$(i < j) \land \exists k.\ (i < k \leq j \land \neg Tdiff_0(k)).$$

Similarly, $f(i) \leq f(j)$ and $f(i) \leq f(j) + 1$ can be replaced by

$$\forall k.\ (j < k \leq i \rightarrow Tdiff_0(k)) \qquad (\phi_0)$$

and

$$\phi_0 \lor \exists k.\ [j < k \leq i \land Tdiff_1(k) \land \forall k' \neq k.\ (j < k' \leq i \rightarrow Tdiff_0(k'))],$$

respectively. The generalization to subformulas of the form $f(i) + c \leq f(j)$ and $f(i) \leq f(j) + c$, for arbitrary $c > 1$, 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 is only a finite number of modulo-$c(\phi)$ 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

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

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

Let ϕ''' be the formula resulting 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 $Tdiff_0, \ldots Tdiff_{d(\phi)}$ is true.

- For every state $i \geq 0$, exactly one of the time-congruence predicates $Tcong_0, \ldots Tcong_{c(\phi)-1}$ 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:

$$\forall i.\ \bigwedge_{k=0}^{d(\phi)-1} \bigwedge_{k'=0}^{c(\phi)-1} \begin{pmatrix} Tdiff_k(i + 1) \land Tcong_{k'}(i) \rightarrow \\ Tcong_{(k'+k) \bmod c(\phi)}(i + 1) \end{pmatrix}.$$

■

The above theorem, 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 [Decidability]. *The validity problem for the language $\mathcal{L}_T^2$ is decidable.* ∎

Clearly, the validity problem is nonelementary even for the first-order language $\mathcal{L}_T$, as $\mathcal{L}$ is a fragment of $\mathcal{L}_T$ (recall that $\mathcal{L}$ was shown to be nonelementary in [St74]).

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

$$\mathcal{P}(\{p_1, \ldots p_n\}) \times \{\, Tdiff_0, \ldots Tdiff_{d(\phi)}\} \times \{\, Tcong_0, \ldots Tcong_{c(\phi)-1}\}$$

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

2.4 Undecidable extensions and variants

Now we 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 highly 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 [Undecidable theories of real time]. *The following two-sorted first-order theories are Π_1^1 -complete:*

	state theory	time theory	time function (from states to time)
1	$(\mathsf{N}, \leq)$	$(\mathsf{N}, +1)$	f
2	$(\mathsf{N}, \leq)$ with monadic predicates	$(\mathsf{N}, \cdot 2)$	identity f
3	$(\mathsf{N}, \leq)$ with monadic predicates	dense linear order (D, $\preceq$) with "successor" S: $x \prec S(x)$ $x \prec Y \rightarrow S(x) \prec S(y)$	strictly monotonic f
4	$(\mathsf{N}, \leq)$ with monadic predicates	$(\mathsf{N}, +1)$	identity f and st rictlv monotonic f'

∎

Proof: First, we observe that the satisfiability of a formula ϕ can, in all cases, be phrased as a Xi-sentence, asserting 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 for which p holds. It is routine 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 (and decode) such models. Thus satisfiability problem is in Σ^1_1 in each case.

Now let us prove Et-hardness. The problem of deciding whether a nondeterministic Turing machine has, over the empty tape, a computation in which the start state is visited infinitely often, is known to be 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), 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 obvious way. A *computation* of *M* is an infinite sequence of related configurations, starting with the initial configuration $\langle 0,0,0 \rangle$. It is called recurring iff it contains infinitely many configurations with the value of the location counter being 0.

The problem of deciding whether a given nondeterministic 2-counter machine has a recurring computation, is Σ^1_1-hard ([AH89]). Thus, to show that the satisfiability problem of a language is Σ^1_1-hard, it suffices, given a nondeterministic a-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 map can be used to encode (and decode) computations of *M*. We write a formula ϕ_M all of whose models correspond to recurring computations of *M*. A computation Γ of *M* is encoded by the interpretation *I* 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 I’.

First, specify the initial configuration, by

$$f(0) = 0 \text{ A } f(1) = n \text{ A } f(2) = n. \qquad (\phi_{INIT})$$

Then ensure proper consecution 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, to either instruction 2 or 3, contributes the conjunct

$$\text{Vi.} \left[f(i) = 1 \rightarrow \begin{pmatrix} (f(i+3) = 2 \vee f(i+3) = 3) \text{ A} \\ f(i+4) = f(i+1) + 1 \wedge \\ f(i+5) = f(i+2) \end{pmatrix} \right]. \qquad (\phi_1)$$

The recurrence condition can be expressed by the formula

$$\forall i.\ \exists j \geq \text{i. f(j)} = 0. \qquad (\phi_{RECUR})$$

Clearly, the conjunction ϕ_M of these n + 2 formulas is satisfiable iff M has a recurring computation.

Note that ϕ_M uses only the successor primitive over time, and no unary predicates. Case 1 follows.

Xi-hardness of Case *2:* We show that a certain extremely modest relaxation of the timing constraints admitted in $\mathcal{L}_T$, namely allowing the primitive of multiplication by *2* over the time domain, leads to Xi-hardness. This result holds even under the restriction that the time function f is the identity function; that is, "time" acts merely as a state counter.

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 of any state; hence we may identify states with predicate symbols. The configuration (I, c, d) of M is represented by the finite sequence of states that starts with a p_l**-state**, and contains precisely c r_1**-states** and a! r_2**-states**.

The initial configuration as well as the recurrence condition can be expressed easily. 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-states. With the availability of multiplication by 2, we are able to have the i-th configuration of a computation correspond, for **all** $i \geq 0$, to the finite sequence of states that is mapped to the time interval $[2^i, 2^{i+1})$. Then we can copy groups of r-states by establishing a one-to-one correspondence of r-states at time *t* and time *2t;* clearly there are enough gaps to accommodate an additional r-state when required by an increment instruction.

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

$$\forall i. \left[p_1(i) \rightarrow \left[\begin{array}{ll} \exists j. & [f(j) = 2f(i) \wedge (p_2(j) \vee p_3(j))] \wedge \\ \forall j. & \left[\begin{array}{l} f(i) < \mathbf{f(j)} < 2f(i) \wedge r_1(j) \rightarrow \\ \exists k. (f(k) = 2f(j) \wedge r_1(k)) \end{array} \right] \wedge \\ \exists j. & \left[\begin{array}{l} 2f(i) < j < 4f(i) \wedge r_1(j) \wedge \\ \forall k. (2f(k) = f(j) \rightarrow \neg r_1(k)) \wedge \\ \forall j' \neq j. \left(\begin{array}{l} 2f(i) < j' < 4f(i)\ A\ r_1(j) \rightarrow \\ \exists k. (2f(k) = f(j') \wedge r_1(k)) \end{array} \right) \end{array} \right] \wedge \\ \forall j. & \left[\begin{array}{l} f(i) < f(j) < 2f(i) \wedge r_2(j) \rightarrow \\ \exists k. (f(k) = 2f(j) \wedge r_2(k)) \end{array} \right] \wedge \\ \forall j. & \left[\begin{array}{l} 2f(i) < f(j) < 4f(i) \wedge r_2(j) \rightarrow \\ \exists k. (2f(k) = f(j) \wedge r_2(k)) \end{array} \right] \end{array} \right] \right].$$

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 $I_2 : [2f(i), 4f(i))$ encode the configuration that results from executing instruction 1. The first conjunct updates the location counter. The second conjunct requires I_2 to contain at least as many r_1-states as I_1; together with the third conjunct it assures that I_2 has precisely one r_1-state more than I_1. The last two conjuncts together state that the number of r_2-states in I_2 is the same as in I_1.

Et-hardness of Case 3: Now we attempt to model time over a dense domain *TIME* = D; that is, between any two given time points there is another time point. We show that even the simple arithmetic of linear order $(\preceq)$ and addition by a constant (S) leads to a highly undecidable theory. Examples for (D, $\preceq$, S) are the rational numbers (Q, $\leq$, + $1)$, and the reals.

As in the previous case, we employ the predicates p_1, . . . p_n, r_1, and r_2: a configuration $\langle l$, c, $d \rangle$ of *M* is encoded by the state sequence $p_i r_1^c r_2^d$. The proof depends, once more, on the ability to copy groups of r-states. 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 $\in$ D, because the denseness of the domain allows us to squeeze arbitrarily many states into any non-empty interval.

Since every state has a unique time, and we can establish a one-to-one correspondence of r_j-states (j $= 1, 2)$ 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 $\cdot 2$ by S.

Xi-hardness of Case 4: This case corresponds to having *two* time bases, 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 increasing.

The encoding of M-computations is very similar to the one used in Case 2; the i-th configuration of *M* corresponds to the sequence of 2' states in the interval $[2^i, 2^{i+1})$. The assertion language does not include the primitive of multiplication by 2, which can, however, 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 conjunct

$$f'(0) = 0 \text{ A Vi. } (f'(i + 1) = f'(i) + 2).$$

By replacing, in the formula constructed in Case 2, every term of the form $2f(i)$ by f' (i) , we obtain again a formula encoding the recurring computations of *M*. ∎

Let us consider the implications of these results on developing logics for real-time systems, which justify our decisions in the choice of $\mathcal{L}_T^2$.

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, since we are interested only in monotonic time functions anyway.

When designing a real-time logic we need to select an appropriate domain for modeling time. Ideally, for asynchronous systems, where changes in the global state of the system 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 major stumbling block in the design of useful logics over dense time. For example, the real-time (branching-time) logics considered in [AD90] and [Le90] use the set of real numbers to model time, and hence are undecidable.

Having constrained ourselves to a discrete time domain, we need to choose the operations on time admitted in the logic. While previous works have used addition as one of the primitives, the above theorem (Case 2) shows that it introduces undecidability. Using our results and techniques, we can show the undecidability (in fact, Π_1^1-**hardness**) of various real-time logics proposed earlier, such as [JM86], [Os87], [Ha88], and [Ko89], all of which include addition. In [HLP90], decidability is proved for a real-time logic with addition; this logic puts, however, substantial restrictions on the use of time quantifiers.

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 decidabilty for the second-order case as well. Thus we claim that 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)is *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 model-checking algorithm for this *timed* propositional *temporal* logic (TPTL), thus demonstrating its suitability for the verification and synthesis of real-time systems.

In this section, we study the expressiveness of TPTL. We compare the properties of timed state sequences expressible in TPTL with those expressible in the underlying 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 characterized by a TPTL-formula. This result is important as it establishes TPTL as a sufficiently expressive specification language; it shows that the gains in complexity in moving from the full first-order theory of timed state sequences (nonelementary) to TPTL (doubly exponential) are 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, thus affirming our choice of TPTL as verification formalism. 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. This real-time temporal logic is obtained from PTL by adding a time quantifier "$x.$" that binds the associated variable x to the "current" time: $x.\phi(x)$ holds at state σ_i of the timed state sequence (σ, τ) iff $\phi(\ \tau(i))$ does. For example, in the formula Ox. ϕ, the time reference x is bound to the time of the state at which ϕ is "eventually" true.

This extension of PTL with references to the times of states admits the addition of timing constraints; that is, atomic formulas that relate the times of different states. The formulas of TPTL are built from propositions and timing constraints by connectives, temporal operators, and time quantifiers. For instance, the typical bounded response property that "Every pstate is followed by a q-state within time 1" can be stated as

$$\Box x.\ (p \rightarrow oy.\ (q\ A\ y \leq x + 1)). \qquad (\phi_{BR})$$

Let us be more precise. Given a set P of proposition symbols and a set V of variables, the terms π and formulas ϕ of TPTL are inductively defined as follows:

- $\pi := x \mid c \mid x + c$
- $\phi := p \mid \pi_1 \leq \pi_2 \mid \pi_1 \equiv_d \pi_2 \mid \mathbf{false} \mid \phi_1 \rightarrow \phi_2 \mid \bigcirc\phi \mid \phi_1 \mathcal{U}\, \phi_2 \mid x.\phi$

for x $\in$ V, $p \in P$, c $\geq$ 0, and $d \geq 2$.[3] Additional temporal operators such as 0 *(eventually)* and • I (always) are defined in terms of $\bigcirc$ (next) and $\mathcal{U}$ (until) as usual.

The formulas of TPTL are interpreted over timed state **sequences.**[4] The timed state sequence $\rho =$ (a, τ) satisfies ϕ iff (p, 0) $\models_{\mathcal{E}_0} \phi$ for the initial environment $\mathcal{E}_0$: $V \rightarrow \{\tau(0)\}$, where the truth predicate $\models$ **is** inductively defined as follows:

[3] TPTL as originally defined in [AH89] differs syntactically in that the time quantifiers are coupled with the temporal **opearators**. Observe that this coupling does not restrict us in any essential way: by separating the time quantifier "$x.$" from the temporal operators, we admit more formulas (such as ❑ (z.4 $\rightarrow x.\psi$)), for each of which there is, however, an equivalent formula in which every quantifier follows a temporal operator $(\Box x.\ (\phi \rightarrow \psi))$.

‘In [AH89], timed state sequences are required to satisfy the two additional conditions of *initiality* $(x = 0)$ and progress $(\Box x.\Diamond y.y > x)$. These requirements make sense for any real-time specification language, but we have just demonstrated that they are expressible within TPTL itself.

- $(\rho, i) \models_{\mathcal{E}} p$ iff $p \in \sigma_i$
- $(\rho, i) \models_{\mathcal{E}} \pi_1 \leq (\equiv_c)\, \pi_2$ iff $\mathcal{E}(\pi_1) \leq (\equiv_d)\ \mathcal{E}(\pi_2)$,
  for $\mathcal{E}(x +$ **c) =** $\mathcal{E}(x)$ **+ c and E(c) = c**
- $(\rho, i) \not\models_{\mathcal{E}}$ **false**
- $(\rho, i) \models_{\mathcal{E}} \phi_1 \rightarrow \phi_2$ 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 (p, $j) \models_{\mathcal{E}} \phi_2$ for some $j \geq$ i, and
  (p, $k) \models_{\mathcal{E}} \phi_1$ for all $i \leq k < j$
- $(\rho, i) \models_{\mathcal{E}} x. \phi$ iff $(\rho, i) \models_{\mathcal{E}[\tau(i)/x]} \phi$.

(Here $\mathcal{E}[t/x]$ denotes the environment that agrees with $\mathcal{E} : V \rightarrow \mathit{TIME}$ **on all** variables except x, which is mapped to $t \in \mathit{TIME}$.) Note that every TPTL-formula is equivalent to its closure, in which all free variables are bound by a prefix of time quantifiers.

Every TPTL-formula ϕ can be translated into $\mathcal{L}_T$, while preserving the set of models $\mathcal{M}_T(\phi)$. For every proposition p of TPTL, we have a corresponding unary state predicate p(i) of $\mathcal{L}_T$. A closed TPTL-formula ϕ is true over a timed state sequence ρ iff the &-formula $F_0(\phi)$ is true over ρ, where F_i (for $i \geq$ 0) is inductively defined as follows:

- $F_i(p) = p(i)$
- $F_i(\pi_1 \leq \pi_2) = \pi_1 \leq \pi_2, \quad F_i(\pi_1 \equiv_d \pi_2) = \pi_1 \equiv_d \pi_2$
- $F_i($**false**$) =$ **false,** $\quad F_i(\phi_1 \rightarrow \phi_2) = F_i(\phi_1) \rightarrow F_i(\phi_2)$
- $F_i(\bigcirc\phi) = F_{i+1}(\phi)$
- $F_i(\phi_1 \mathcal{U}\, \phi_2) =$ 3j ≥ i. $(F_j(\phi_2)$ ✌ $\forall i \leq k <$ er✍ $F_k(\phi_1))$
- $F_i(x.\ \phi) = F_i(\phi)[f(i)/x]$.

(We write $\phi[f(i)/x]$ for the formula that is obtained from ϕ by replacing all free occurrences of x by $f(i)$.)

For example, the bounded response property ϕ_{BR} is equivalent to its translation $F_0\,(\phi_{BR})$:

$$\text{Vi} \geq 0.\ (\text{p(i)} \rightarrow \exists j \geq i.\ (q(j) \wedge f(j) \leq f(i) + 1)).$$

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. 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, while quantification over the time sort is prohibited entirely.

3.2 Expressiveness

In [AH89] we have shown that, in a pleasing analogy to PTL versus C, TPTL constitutes in fact an elementary fragment of CT: the satisfiability of a given TPTL-formula with N logical and temporal connectives, 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 CT-formulas do not diminish its expressive power. In other words, any property of timed state sequences that can be specified in $\mathcal{L}_T$ can already be specified in TPTL.

The natural embedding F_0 gives, for any TPTL-formula ϕ, an equivalent CT-formula $F_0(\phi)$, thus demonstrating that $\mathcal{L}_T$ is as expressive as TPTL. By the following theorem, the converse is also true.

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

Proof: Given an CT-formula ϕ, we construct an equivalent TPTL-formula ψ in four steps. By the theorem on the regular nature of the time primitives we obtain an C-formula ϕ', with additional time-difference predicates $Tdiff_t$ and time-congruence predicates $Tcong_t$, such that $\mathcal{M}_T^{*}(\phi) = \mathcal{M}(\phi')$. By the expressive completeness of PTL, there is a PTL-formula ϕ'' such that $\mathcal{M}(\phi')$ equals $\mathcal{M}(\phi'')$ ([GPSS80]).

We transform ϕ'' into an equivalent PTL-formula ϕ''' such that every time-difference proposition $Tdiff_t$ 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.

Define the constants $d(\phi)$ and $c(\phi)$ as in Section 2.3. From ϕ''' we arrive at ψ by replacing every time-difference proposition $Tdiff_t$ that is not within the scope of a temporal operator by x. x = t (and x. x $\geq t$, if $t = d(\phi)$), every subformula $\bigcirc Tdiff_t$ by $x.\bigcirc y.\ y = x + t$ (and $x.\bigcirc y.\ y \geq x + t$, if $t = d(\phi)$), and every time-congruence proposition $Tcong_t$ by x.x $\equiv_{c(\phi)} t$. ■

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

3.2.1 Timeless expressiveness

With every TPTL-formula ϕ we can associate a set of ***state*** sequences by projecting the timed state sequences in $\mathcal{M}_T(\phi)$. Given a state sequence σ and a TPTL-formula ϕ, let $\sigma \in \mathcal{M}_{\exists}(\phi)$ iff there is a time map τ such that (a, τ) $\models \phi$.

Interpreted in this fashion, TPTL can specify 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 pure PTL ([Wo83]). In TPTL, we may (ab)use time to identify the even states as precisely those in which the time does not increase:

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

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 [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 vice versa.* ■

Proof: Given a TPTL-formula ϕ, we know how to construct an equivalent &-formula ϕ'. By the theorem on the regular nature of the time primitives we obtain an L-formula ϕ'', with additional time-difference predicates $Tdiff_t$ and time-congruence predicates $Tcong_t$, 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.

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 ([B ü62]). We construct a TPTL-formula ϕ that characterizes the models of ψ', by using the (existentially quantified) time map to encode the interpretation of the unary predicates p_j ($1 \leq$ j $\leq n$), which are bound in ψ'.

Assign to every subset $J_t \subseteq$ *(1, . . . n}* a unique code $t \in TIME$. By the expressive completeness of PTL, $\mathcal{M}(\psi'_M) = \mathcal{M}(\psi''_M)$ for some PTL-formula ψ''_M ([GPSS80]). From ψ''_M, we obtain ϕ by replacing every proposition p_j, $1 \leq j \leq n$, by $x.\bigcirc y.\ \bigvee_{j \in J_t} y =$ x + t. It is straightforward to establish a one-to-many correspondence between the models $I = (\sigma, p_1^I, \ldots p_n^I)$ of ψ'_M and the timed state sequences (a, τ) satisfying ϕ: given I, let $\tau(i + 1) = \tau(i) + t$ such that $J_t = \{j \mid p!(i)\}$, and given τ, let *p, (i)* iff j $\in J_{\tau(i+1)-\tau(i)}$ (assume that j $\notin J_t$ if t 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.

3.2.2 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:

$$\text{ax. } (x \equiv_2 0 \rightarrow p).$$

It 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 be the largest constant occurring in ψ. It is easy to convince yourself that ψ cannot distinguish between the timed state sequences ρ_1 = (a, λi. (c + 1)) and ρ_2 = $(\sigma$, Xi. (c + $2))$, for any σ. Yet if p is continuously false 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 in the previous subsection, 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 specified 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, clearly nonelementary. In the following subsection, we 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 pure PTL. Therefore it is quite surprising that the past operators render TPTL nonelementary.

3.3.1 TPTL **with quantification over time**

Several authors, such as [Os87] and [Ha88], have proposed to use first-order temporal logic with a single dynamic (state) variable, T, that represents the time in every state, for the specification of real-time properties. For instance, they write our typical bounded response property ϕ_{BR} from above essentially as

$$\Box \quad \mathrm{V}x.\ (p\ A\ T = x \rightarrow \Diamond(q \wedge T \leq x + 1)),$$

using auxiliary rigid (global) variables like x to refer to the time (i.e., the value of *T)* of different temporal contexts.

Eliminating the state variable *T,* we see that this notation corresponds to TPTL extended by classical universal and existential first-order quantification over time:

$$\Box \quad y.\ vx.\ (p\ A\ y = x \rightarrow oz.\ (q\ A\ z \leq x + 1)).$$

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, an index i $\geq$ 0, and an environment $\mathcal{E}$, the classical quantifiers are interpreted as usual:

$$(\rho, i) \models_{\mathcal{E}} 3x.\ \phi \text{ iff } (p,\ i) \models_{\mathcal{E}[t/x]} \phi \text{ for some } t \in TIME.$$

$\mathrm{TPTL}_\exists$ 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 $\mathrm{TPTL}_\exists$ can still be embedded into $\mathcal{L}_T$ (let $F_i(\exists x.\ \phi) = 3x.\ F_i(\phi)$). The satisfiability of $\mathrm{TPTL}_\exists$ is, therefore, decidable, and its expressive power, measured as the sets of timed state sequences specifiable in the logic, is the same as that of TPTL.

We show that $\mathrm{TPTL}_\exists$ 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 feasible verification methods, such as the tableau techniques for TPTL.

Theorem [Complexity of $\mathrm{TPTL}_\exists$]. *The satisfiability problem **of** $\mathrm{TPTL}_\exists$ is nonelementay.* ■

Proof: We translate the nonelementary monadic first-order theory of (N, $\leq$) ([St74]) into $\mathrm{TPTL}_\exists$: by forcing the time to act as a state counter (using [illegible] x + 1), state quantifiers can be simulated by the time quantifiers of $\mathrm{TPTL}_\exists$.

Given a formula ϕ of $\mathcal{L}$, we construct a formula ψ of $\mathrm{TPTL}_\exists$ such that ϕ is satisfiable iff the conjunction of ψ and ❑ z.Oy. y = x + 1 is satisfiable. The formula ψ is obtained from ϕ by replacing every atomic subformula of the form p(i) by Ox. *(p A x* $=$ i) (read the quantifiers of ϕ as quantifiers over the time sort). ■

3.3.2 TPTL with **past**

In [LPZ85], PTL is extended with the past temporal operators $\ominus$ (previous) and S *(since),* the duals 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 certain 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 ϕ_1 S ϕ_2." The meaning of the past operators is given by

- $(\rho, i) \models_\mathcal{E} \ominus\phi$ **iff** $i = 0$ or (p, $i - 1) \models_\mathcal{E} \phi$, and
- $(\rho, i) \models_\mathcal{E} \phi_1\ \mathcal{S}\ \phi_2$ **iff** $(\rho, j) \models_\mathcal{E} \phi_2$ for some $j \leq i$ and $(\rho, k) \models_\mathcal{E} \phi_1$ *for all $j < k \leq i$.*

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

- $F_0(\ominus\phi) = \mathbf{true},\quad F_{i+1}(\ominus\phi) = F_i(\phi)$
- $F_i(\phi_1\,\mathcal{S}\,\phi_2) = \exists j \leq i.\ (F_j(\phi_2) \wedge \forall j < k \leq i.\ F_k(\phi_1))$.

Hence the satisfiability of 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 surprisingly heavy price to be paid for adding the past operators.

Theorem [Complexity of TPTL$_P$]. *The satisfiability problem **of*** TPTLp *is nonelementay.* ■

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 0, **because**$\diamondminus$ allows us to restore the correct temporal context.

Given a formula ϕ of $\mathcal{L}$, we construct a formula ψ of TPTLp such that ϕ is satisfiable iff the conjunction of ψ and $\Box x.\bigcirc y.$ y = x + **1** is satisfiable. The first step in translating ϕ is the same as in the proof of the nonelementary complexity of TPTL$_\exists$. In a second step we replace every subformula of the form 3x. φ by y. (0X.02. $(z$ = **y A** $\varphi)$ **VOX. oz. (z = y A** φ)). ■

3.4 Timed ETL

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

$$\exists q. \left[q(0) \wedge \forall i. \left(q(i) \rightarrow \boldsymbol{p(i)} \text{ A } \neg q(i+1) \text{ A } q(i+2)\right)\right],$$

is not expressible in PTL ([Wo83]). That is why Wolper has defined *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, w-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 map, p holds at every even **state.**

Suppose there were such a formula ϕ; we show that this would imply the expressibility of *even(p)* in $\mathcal{L}$. First construct an &formula ϕ' that is equivalent to ϕ and contains the additional time-difference and time-congruence predicates Tdiff_t and *Tcong,,* as usual. Then replace, in ϕ', all occurrences of Tdiff_t and *Tcong,* **by true or false depending on whether** 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.

However, **analogously** to PTL, we are able to generalize TPTL to *timed extended temporal* logic, TETL, by introducing temporal grammar operators. TETL is shown to have the full expressive power of $\mathcal{L}^2_T$, while being no more expensive than TPTL.

3.4.1 Syntax and semantics

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

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

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, .[5]

As with TPTL, TETL-formulas are interpreted over timed state sequences. Given a timed state sequence ρ, an index $i \geq 0$, 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 $j \geq 0$.

All temporal operators of TPTL are expressible by the grammar operators of TETL; for example, the TPTL-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 formula even(p), which is not expressible in TPTL, can be stated as $\mathcal{G}_{even}(\textbf{true}, p)$, for the production

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

3.4.2 Complexity

By putting together the tableau methods for ETL ([Wo83]) and TPTL ([AH89]), we develop a doubly-exponential-time decision procedure for TETL. This procedure is near-optimal; we go on to show the satisfiability problem for TETL to be EXPSPACE-complete.

Our presentation follows [AH89] closely.[6] For the sake of keeping the presentation simple, we assume that all grammar operators correspond to productions of the form

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

Furthermore, all TETL-formulas contain a single free variable, T (which refers to the initial time), and only timing assertions of the forms $x \leq y + c$, $x + c \leq y$, and $x \equiv_d y + c$, for $d > c \geq 0$. This can be achieved by renaming of variables, and easy simplifications.

[5] Like ETL, TETL can alternatively be defined using automata connectives for all Büchi-automata, instead of grammar operators ([WVS83]).

[6] The careful reader may have noticed that we use, throughout, time-difference propositions $Tdiff_t$ that indicate the time increase t from the predecessor states, as opposed to [AH89], where these propositions represent the time difference to the *successor* states. This is necessary, because we have relaxed the *initiality* condition $\tau(0) = 0$ on timed state sequences (σ, τ).

As with TPTL, for checking the satisfiability of a given 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 occurring 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 the new propositions $Tdiff_t$, $0 \leq t \leq K$, representing the time differences t between successive states.

This allows us to modify the tableau-based decision procedure for ETL ([Wo83]), to handle formulas with time references. It is, in fact, included in our procedure as the special case in which ϕ contains no timing constraints.

The key observation underlying 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 eventuality $\Diamond\phi$ can be satisfied by either ϕ or $\bigcirc\Diamond\phi$ being true in the initial state.

In order to propagate the requirement on the successor state properly, all timing constraints need to be updated to account for the time increase t from the initial state to its successor. Consider the formula $\bigcirc\phi(T)$, and recall that the free occurrences of T are references to the initial time. This condition is true in the initial state iff the next state satisfies the updated formula $\phi(T-t)$.

If the number of conditions generated in this way is finite, checking for satisfiability is reducible to checking for satisfiability in a finite structure, the initial tableau. For $t > 0$, a naive replacement of T by $T - t$ would, however, successively generate infinitely many new formulas. Fortunately, the *monotonicity* of time can be exploited to keep the tableau finite; the observation that x is always instantiated, in the "future," to a value greater than or equal to T, allows us to simplify timing assertions of the form $T \leq x + c$ and $x + c \leq T$ to true and false, respectively.

We define, therefore, the formula ϕ^t that results from updating all time references T in ϕ, inductively as follows: $\phi^0 = \phi$; and ϕ^{t+1} is obtained from ϕ^t by replacing all terms of the form $T + c$ (for $c > 0$) by $T + (c-1)$, and all subformulas of the form $T \leq x + c$, $x + c \leq T$, and $T \equiv_d x + c$ (for $c \geq 0$) by true, **false**, and $T \equiv_d x + ((c+1) \bmod d)$, respectively.

Now let us collect all conditions that may arise by recursively splitting a formula into its present and future parts. The closure $Cl(\phi)$ of a TETL-formula ϕ is the smallest set containing ϕ that is closed under the following operation *Sub:*

- $Sub(\psi_1 \rightarrow \psi_2) = \{\psi_1, \psi_2\}$
- $Sub(\bigcirc\psi) = \{\psi^t \mid 0 \leq t \leq K\}$
- $Sub(\mathcal{G}(\psi_1, \ldots \psi_m)) = \{\psi_{i_1}, \psi_{i_2}, \bigcirc\mathcal{G}'(\psi_{j_1}, \ldots \psi_{j_n})\}$

- $Sub(x.\ \psi(x)) = \{\psi(T)\}$.

Let N be the number of connectives, quantifiers, and grammar operators in ϕ, where every grammar operator is counted as the number of nonterminal symbols in the corresponding grammar. By induction on the structure of ϕ, it can be shown that $|Cl(\phi)| \leq 2N \cdot K$.

Tableaux for TETL are finite, directed state graphs (Kripke structures) with local and global consistency constraints on all states. The states are represented by consistent sets of formulas that are closed under "subformulas," expressing conditions on the current state and the successor states. Every state contains, in addition, a proposition $Tdiff_t$, $0 \leq t \leq K$, which denotes the time difference to the predecessor states.

Formally, we define the states as the maximally consistent subsets of the finite universe

$$Cl^* (\phi) = Cl(\phi) \cup \{ Tdiff_t \mid 0 \leq t \leq K\}$$

of TETL-formulas. The set $\Phi \subseteq Cl^*(\phi)$ is (maximally) *consistent* iff it satisfies the following conditions (where all formulas range only over $Cl^*(\phi)$):

- $Tdiff_t \in \Phi$ for precisely one t with $0 \leq t \leq K$; this $t \in TIME$ is referred to as $Lastdiff(\Phi)$.

- $\textbf{false} \not\in \Phi$.

- $\psi_1 \rightarrow \psi_2 \in \Phi$ iff either $\psi_1 \not\in \Phi$ or $\psi_2 \in \Phi$.

- $\mathcal{G}(\psi_1, \ldots \psi_m) \in \Phi$ iff either $\psi_{i_1} \in \Phi$, or both $\psi_{i_2} \in \Phi$ and $\bigcirc\mathcal{G}'(\psi_{j_1}, \ldots \psi_{j_n}) \in \Phi$.

- $x.\psi(x) \in \Phi$ iff $\psi(T) \in \Phi$.

- $T \sim T + c \in \Phi$ iff $0 \sim c$ holds in N (for $\sim$ one of $\leq$, $\geq$, $\equiv_d$, or its negation).

Now we are ready to define the initial tableau in a way that ensures the *global* consistency of both temporal and real-time constraints as well. The initial *tableau* $\mathcal{T}(\phi)$ for the TETL-formula ϕ is a directed graph whose vertices are the consistent subsets of $Cl^*(\phi)$, and which contains an edge from Φ to Ψ iff, for all $\bigcirc\psi \in Cl(\phi)$,

$$\bigcirc\psi \in \Phi \ \text{ iff } \ \psi^{Lastdiff(\Psi)} \in \Psi.$$

The significance of the (finite) initial tableau $\mathcal{T}(\phi)$ for the formula ϕ is that every model of ϕ corresponds to an infinite path through $\mathcal{T}(\phi)$ along which all eventualities are satisfied ("fulfillable") in time, and vice versa. An eventuality $\neg\mathcal{G}(\psi_1, \ldots \psi_m)$ is called ***fulfillable*** along the finite path $\Phi_0\Phi_1 \ldots \Phi_k$ iff either $\psi_{i_2} \not\in \Phi_0$, or $k \geq 1$ and $\neg\mathcal{G}'(\psi_{j_1}, \ldots \psi_{j_n})^{Lastdiff(\Phi_1)}$ is fulfillable along $\Phi_1\Phi_2 \ldots \Phi_k$. By combining the corresponding arguments for ETL and TPTL, it can be shown that a TETL-formula ϕ is satisfiable iff $\mathcal{T}(\phi)$ contains an infinite path $\Phi_0\Phi_1\Phi_2 \ldots$ such that $\phi \in \Phi_0$ and, for every $i \geq 0$, $\neg\mathcal{G}(\psi_1, \ldots \psi_m) \in \Phi_i$ implies that $\neg\mathcal{G}(\psi_1, \ldots \psi_m)$ is fulfillable along $\Phi_i\, \Phi_{i+1} \ldots \Phi_k$ for some $k \geq$ i.

This result suggests a decision procedure for TETL: construct the initial tableau, and employ the usual, polynomial techniques for checking whether the tableau contains an infinite path along which all eventualities are satisfied. Since the initial tableau contains $O(K \cdot 2^{N \cdot K})$ states, each of size $O(N \cdot K)$, $\mathcal{T}(\phi)$ can be constructed and checked for infinite paths in deterministic time exponential in $O(N \cdot K)$.

Theorem [Deciding TETL]. *The satisfiability* of a TETL-formula ϕ *is decidable in determinzstic time exponential in* $O(N \cdot K)$, *where N is the number of connectives, quantifiers, and grammar operators in* ϕ, *and K is the product of all constants occurring in* ϕ *(recall that every grammar operator is counted us the number of nonterminal symbols in the corresponding grammar).* ■

Note that the length L of a formula whose constants are represented in binary, is 0(N + log K). *So* we have a decision procedure for TETL that is doubly exponential in L (although only singly exponential in N, the "untimed" part, and thus, singly exponential for ETL).

The algorithm outlined here may be improved along the lines of [Wo83] to avoid the construction of the entire initial tableau. This does not, however, lower the doubly exponential deterministic-time bound; in fact, TETL is EXPSPACE-hard.

Theorem [Complexity of TETL]. *The satisfiability problem* of 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 ([Wo83]). EXPSPACE-hardness follows immediately from the corresponding result for TPTL ([AH89]). ■

3.4.3 Expressiveness

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

Theorem [Expressiveness of TETL]. *For every formula* ϕ *of* TETL, *there exists a formula* ψ *of* $\mathcal{L}_T^2$ *such that* $\mathcal{M}_T(\phi) = \mathcal{M}_T(\psi)$, *and vice versa.* ■

Proof: 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 the nonterminal symbols occurring in the grammar $\mathcal{G}_0(a_1, \ldots$ a,), 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 (p, k) $\models_{\mathcal{E}} \phi$ iff $(\rho, k) \models_{\mathcal{E}} F_k(\phi)$ for all $k \geq 0$ and 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 in state $k' \geq k$ 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. 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 expressible without congruence relations.

3.4.4 TPTL **with quantification over propositions**

There are several alternatives to the grammar operators of ETL. PTL can be extended by fixed-point operators (obtaining a variant of the propositional μ-calculus of [Ko82]) or second-order quantification over propositions (QPTL of [Si83]) 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 as a verification formalism.

4 Metric Temporal Logic: MTL

Several authors have tried to adapt temporal logic to reason about real-time properties by interpreting its modalities as bounded operators. For example, [Ko89] suggests the notation $\Diamond_{<c}$ to express "eventually within time c." Similar temporal operators that are subscripted with constant bounds are used in [Ha88] and [EMSS89].

In this section, we extend PTL by such bounded temporal operators and interpret the resulting logic over timed state sequences. For example, the typical bounded response property that "Every p-state is followed by a q-state within time 1" will be written as ❑ (p $\rightarrow \Diamond_{<1}$ q).

It is easy to see that we have, in-fact, only obtained a notational variant of a subset of TPTL (rewrite every subformula $\Diamond_{<c}\phi$ as x. Oy. (y $\leq$ x + c A ϕ)).

We show that this formalism 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 operators, thus resulting in what we call (following [Ko89]) *metric temporal logic* (MTL), without sacrificing its elementary decidability.

Hence we are able to conclude that MTL represents, again, a suitable specification and verification formalism: just like TPTL, MTL corresponds to an expressively complete and yet elementary fragment of $\mathcal{L}_T$ with a tableau-based decision procedure. But the two subsets of $\mathcal{L}_T$ corresponding to TPTL and MTL, respectively, are not identical; either one of them 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 of propositions P, the formulas ϕ of MTL are defined inductively as follows :

$$\phi := p \mid \mathbf{false} \mid \phi_1 \rightarrow \phi_2 \mid \bigcirc_{\sim c}\phi \mid \ominus_{\sim c}\phi \mid \phi_1 \,\mathcal{U}_{\sim c}\, \phi_2 \mid \phi_1 \,\mathcal{S}_{\sim c}\, \phi_2$$

for $p \in P$, $\sim$ being one of $<, =, >$, or $\equiv_d$, and c $\geq$ 0, $d \geq 2$. The defined operators $\Diamond_{\sim c}\phi$ and $\Box_{\sim c}\phi$ stand for $\mathbf{true}\,\mathcal{U}_{\sim c}\,\phi$ and $\neg\Diamond_{\sim c}\neg\phi$, respectively; other abbreviations include $\Diamond_{>c}\phi$ (for $\Diamond_{=c}\phi \vee \Diamond_{>c}\phi$) and unbounded 0 (for $\Diamond_{\geq 0}$).

The formulas of MTL are interpreted over timed state sequences. Instead of giving MTL its own semantics, we translate any MTL-formula ϕ into a TPTL$_P$-formula $G(\phi)$ (let $\sim$ stand for $<, >$, or $=$):

. $G(p)$ = P

- $G(\mathbf{false})$ = **false,** $G(\phi_1 \rightarrow \phi_2) = G(\phi_1) \rightarrow G(\phi_2)$

. $G(\bigcirc_{\sim c}\phi) = x.\bigcirc y.$ ☎✡ $\sim x + c \wedge \phi)$, $G(\bigcirc_{\equiv_d c}\phi) = \bigcirc y.(y \equiv_d$ **c A** $\phi)$

- $G(\ominus_{\sim c}\,\phi) = x.\ominus y.\,(x \sim y + c \,\wedge\, \phi)$, $\quad G(\ominus_{\equiv_d c}\,\phi) = \ominus y.\,(y \equiv_d c \,\wedge\, \phi)$
- $G(\phi_1\,\mathcal{U}_{\sim c}\,\phi_2) = x.\,(\phi_1\,\mathcal{U}\,y.\,(y \sim x + c \,\wedge\, \phi_2))$
- $G(\phi_1\,\mathcal{U}_{\equiv_d c}\,\phi_2) = \phi_1\,\mathcal{U}\,y.\,(y \equiv_d c \,\wedge\, \phi_2)$

. $G(\phi_1\,\mathcal{S}_{\sim c}\,\phi_2) = x.\,(\phi_1\,\mathcal{S}\,y.\,(x \sim y + c$ A 4 2))

- $G(\,\phi_1\,\mathcal{S}_{\equiv_d c}\,\phi_2)$ = $\phi_1\,\mathcal{S}\,y.$ ☎✡ $\equiv_d c \wedge \phi_2).$

Note that $\Diamond_{=3}$ p holds in a state if *p* is true in some future state whose time is 3 greater than the current time. However, $\Diamond_{\equiv_2 1}$ *p* holds in a state if *p* is true in some future state whose time is odd; the congruence subscripts refer to the *absolute* times of states.

It follows that both TPTL and MTL are orthogonal fragments of TPTL_P and, hence, $\mathcal{L}_T$: while TPTL prohibits past operators, MTL corresponds to a subset of TPTL_P wherein all timing constraints relate only variables that refer to "adjacent" temporal contexts.

4.2 Complexity

We show that the satisfiability problem of MTL is much simpler than the corresponding nonelementary problem of full TPTLp, by generalizing the standard tableau-decision procedure for PTL ([BMP81]) to MTL.

The tableau algorithm for MTL uses the techniques developed for TPTL in [AH89]. 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 related by a timing constraint is uniquely determined. Before giving a formal definition, we indicate first how the algorithm proceeds for a sample input.

Suppose that the time increases by one unit 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, $\Diamond_{<c}\ \phi$ in the current state, we have to satisfy either ϕ now, or $\Diamond_{<c-1}\ \phi$ in the succeeding state. Continuing this splitting of requirements into a present and a future part, we will eventually arrive at $0_{<1}\ \phi$, forcing ϕ to be satisfied in the current state.

Since every input formula ψ generates only a finite number of requirements on states in the described fashion, ψ is satisfiable iff it is satisfiable in a finite tableau. By bounding the maximal size of this tableau, we obtain the following result.

Theorem [Deciding MTL]. *The satisfiability **of*** an MTL-formula ϕ *can be decided in **deterministic** time ezponential in* $O(C$ *. N), where N is the number **of** propositional and tempoml* connectives **in** ϕ, *and* $C - 1$ *is the largest constant occurring, as a subscript, in* ϕ*.* ∎

Proof: Throughout, let $\sim$ stand for <, >, or =. Define the *closure* $Cl(\phi)$ of the MTL-formula ϕ to be the smallest set containing ϕ that is closed under the following operation *Sub:*

- $Sub(\psi_1 \rightarrow \psi_2) = \{\psi_1, \psi_2\}$
- $Sub(\bigcirc_{\sim c}\psi) = \{\psi\}$
- $Sub(\ominus_{\sim c}\psi) = \{\psi\}$
- $Sub(\psi_1\,\mathcal{U}_{\sim c}\,\psi_2) = \{\psi_1, \psi_2, \bigcirc(\psi_1\,\mathcal{U}\,\psi_2)\} \cup \{\bigcirc(\psi_1\,\mathcal{U}_{\sim c'}\,\psi_2) \mid 0 \leq c' \leq c\}$
- $Sub(\psi_1\,\mathcal{U}_{\equiv_d c}\,\psi_2) = \{\psi_1, \psi_2, \bigcirc(\psi_1\,\mathcal{U}_{\equiv_d c}\,\psi_2)\}$
- $Sub(\psi_1\,\mathcal{S}_{\sim c}\,\psi_2) = \{\psi_1, \psi_2, \ominus(\psi_1\,\mathcal{S}\,\psi_2)\} \cup \{\ominus(\psi_1\,\mathcal{S}_{\sim c'}\,\psi_2) \mid 0 \leq c' \leq c\}$
- $Sub(\psi_1\,\mathcal{S}_{\equiv_d c}\,\psi_2) = \{\psi_1, \psi_2, \ominus(\psi_1\,\mathcal{S}_{\equiv_d c}\,\psi_2)\}$.

If C- 1 is the largest constant occurring in ϕ, and N is the number of connectives (propositional and temporal) in ϕ, then $|Cl(\phi)| \leq 2C\cdot$ *N*.

As in TPTL, for checking the satisfiability of ϕ, we may restrict ourselves to timed state sequences $\rho = (\sigma, \tau)$ all of whose time steps $\tau(i+1) -$ T(i), i $\geq$ 0, are bounded by the product K of all constants occurring, as subscripts, in ϕ (count a subscript of the form $\equiv_d$ c as d). The time information in ρ has, therefore, finite-state character; it can be modeled by the new propositions $Tdiff_t$ and *Tcong,,* , 0 $\leq$ ♦ $\leq K$ and 0 $\leq t'$ < K, representing, in any state, the time difference ♦ from the predecessor state and the remainder t' modulo K of the current time. For ease of presentation we use, in addition, the propositions $Tdiff'_t$, 0 $\leq$ ♦ $\leq$ K, to represent the time difference ♦ to the successor state.

Let $Cl^*(\phi)$ denote the set obtained from *Cl(\$)* by adding the new propositions $Tdiff_t$, $Tdiff'_t$, and *Tcong,.* A subset Φ of *Cl ’* (ϕ) is called (maximally) *consistent* iff it satisfies the following conditions (where all formulas range only over the finite set *Cl (#)):*

- $Tdiff_t \in \Phi$ for exactly one ♦ with 0 $\leq t \leq K$; this ♦ $\in$ *TIME* is referred to as $Lastdiff(\Phi)$.
- $Tdiff'_t \in \Phi$ for exactly one ♦ with 0 $\leq$ ♦ $\leq K$; this ♦ $\in$ *TIME* is referred to *as* $Nextdiff(\Phi)$.
- *Tcong,* $\in \Phi$ for exactly one ♦ with 1 $\leq$ ♦ $\leq K$; this ♦ $\in TIME$ is referred to *as* $Congclass(\Phi)$.
- **false** $\notin \Phi$.
- \$1 $\rightarrow \psi_2 \in \Phi$ **iff** either $\psi_1 \notin \Phi$ **or** $\psi_2 \in \Phi$.
- $\psi_1\,\mathcal{U}_{=c}\,\psi_2 \in \Phi$ iff either c = 0 and $\psi_2 \in \Phi$, or $\psi_1 \in \Phi$, $Nextdiff(\Phi) \leq$ *c*, and $\bigcirc(\psi_1\,\mathcal{U}_{=c-Nextdiff(\Phi)}\,\psi_2) \in \Phi$.

- $\psi_1 \mathcal{U}_{<c} \psi_2 \in \Phi$ ♓⤢⤢ ♍ ✇ 🗀🖃 and either $\psi_2 \in \Phi$, or $\psi_1 \in \Phi$, $Nextdiff(\Phi) < c$, and $\bigcirc(\psi_1 \mathcal{U}_{<c-Nextdiff(\Phi)}$ $ 2) $\in \Phi$.

- $\psi_1 \mathcal{U}_{>c}$ $2 $\in \Phi$ iff $\psi_1 \in \Phi$, and either $Nextdiff(\Phi) \leq c$ and $\bigcirc(\psi_1 \mathcal{U}_{>c-Nextdiff(\Phi)}$ $2) $\in \Phi$, or $Nextdiff(\Phi) > c$ and $\bigcirc(\psi_1 \mathcal{U} \psi_2) \in \Phi$.

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

Similar conditions are put on the S-formulas in Φ, to ensure their consistency with $Lastdiff(\Phi)$.

The *initial tableau* $\mathcal{T}(\phi)$ for the MTL-formula ϕ is a directed graph whose vertices are the consistent subsets of $Cl^*(\phi)$, and which contains an edge from Φ to Ψ iff all of the following conditions are met:

- $Nextdiff(\Phi) = Lastdiff(\Psi)$.

- $Congclass(\Psi) \equiv_K Congclass(\Phi) + Nextdiff(\Phi)$.

- For all $\bigcirc_{\sim c} \psi \in Cl(\phi)$, $\bigcirc_{\sim c} \psi \in \Phi$ iff $\psi \in \Psi$ and $Nextdiff(\Phi) \sim c$.

- For all $\bigcirc_{\equiv_d c} \psi \in Cl(\phi)$, $\bigcirc_{\equiv_d c} \psi \in \Phi$ iff $\psi \in \Psi$ and $Congclass(\Psi) \equiv_d c$.

- For all $\ominus_{\sim c} \psi \in Cl(\phi)$, $\ominus_{\sim c} \psi \in \Psi$ iff $\psi \in \Phi$ and $Nextdiff(\Phi) \sim c$.

- For all $\ominus_{\equiv_d c} \psi \in Cl(\phi)$, $\ominus_{\equiv_d c} \psi \in \Psi$ iff $\psi \in \Phi$ and $Congclass(\Phi) \equiv_d c$.

It follows that an MTL-formula ϕ is satisfiable iff the initial tableau $\mathcal{T}(\phi)$ contains an infinite path $\Phi = \Phi_0\Phi_1\Phi_2 \ldots$ such that

- $\phi \in \Phi_0$,

- Φ_0 contains no Q-formula,

- for all ⲏ ≥ 0, $\psi_1 \mathcal{U}_{\sim c} \psi_2 \in \Phi_i$ implies $\psi_2 \in \Phi_j$ for some $j \geq i$ with $\Sigma_{i \leq k < j} Nextdiff(\Phi_k) \sim c$, and

- 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 $Congclass(\Phi_j) \equiv_d c$.

The proof is similar to the corresponding argument for TPTL ([AH89]).

Since the initial tableau contains $O(K \, . \, 2^{C \cdot N})$ states, each of size $O(C \cdot N)$, $\mathcal{T}(\phi)$ 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 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. In fact, both formalisms are EXPSPACE-complete.

Theorem [Complexity of MTL]. *The satisfiability problem for* MTL is *EXFSPACE-complete.* ■

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

4.3 Expressiveness

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

$\Box$ *x*-$[p \longrightarrow \Diamond(q$ *A* ⚐⍓📫 $(r \wedge$ ⍓ $\leq x+5))]$.

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

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

In fact, 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, such as "The time of the initial state is 2" (x = 2 in TPTL), are are not expressible in our version of MTL. It can be argued that for the purpose of the analysis of real-time systems, the absolute time of the initial state is of no importance.

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

Theorem [Expressive completeness of MTL]. *For every formula ϕ of $\mathcal{L}_T$, there* ***exists*** *a formula ψ of* MTL *(without past operators) such that $\rho \models \phi$ iff $\rho \models \psi$* for every initial timed state *sequence p.* ■

Proof: As in the proof of the expressive completeness of TPTL, given a formula ϕ of $\mathcal{L}_T$, construct a PTL-formula ϕ' with additional time-difference propositions $Tdiff_t$, $0 \leq t \leq d(\phi)$, and time-congruence propositions *Tcong,,* $0 \leq t < c(\phi)$, such that $\mathcal{M}^{\text{-}}_T(\phi) = \mathcal{M}(\phi')$. Furthermore, in ϕ' all propositions $Tdiff_t$ and *Tcong,* are either not within the scope of any temporal operator, or immediately preceded by a ***next*** operator.

From ϕ' we obtain the desired formula ψ by eliminating the time-difference and time-congruence propositions as follows. Since we consider only initial models, replace each $Tdiff_t$ and *Tcong,* that is not within the scope of any temporal operator by **true** or **false,** depending on whether t = 0. Then replace

$\bigcirc\, Tdiff_t$ (for $0 \leq t < d(\phi)$) by $\bigcirc_{=t}$ true, $\bigcirc Tdiff_{d(\phi)}$ by $\bigcirc_{\geq d(\phi)}$ true, and $\bigcirc\, Tcong_t$ by $\bigcirc_{\equiv_{c(\phi)} t}$ true. (Observe that only the *next* operator needs to be subscripted.) ■

5 Discussion

We have shown that only a very weak arithmetic over a discrete domain of time can be combined with PTL to obtain decidable real-time logics. We have 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.

Thus, TPTL and MTL occupy a position among real-time logics that is as appealing as the standing of PTL for qualitative reasoning. However, both TPTL and MTL have EXPSPACE-complete satisfiability problems. Our decision algorithms are of a time complexity doubly exponential in the length of the timing constraints (though only singly exponential in the number of temporal and logical operators). On the other hand, PTL is PSPACE-complete, and has a singly exponential decision procedure. We claim that this is 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 from the time of another state," using binary encoding for the time constants, is EXPSPACE-hard. 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 successive *next* operators makes PTL EXPSPACE-hard! Thus the price of **an** extra exponential is caused by the succinctness of the notation introduced by the binary encoding of the constants.

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

References

[AD90] R. Alur, D.L. Dill, "Model-checking for real-time systems," 5th IEEE LICS, 1990.

[AH89] R. Alur, T.A. Henzinger, "A really temporal logic," 30th IEEE FOCS, 1989.

[BMP81] M. Ben-Ari, Z. Manna, A. Pnueli, "The temporal logic of branching time," 8th ACM POPL, 1981.

[Bü62] J. R. Büchi, "On a decision method in restricted second-order arithmetic," *Proc. Internat. Congr. Logic, Methodology, and* ***Philosophy of*** Science 1960, Stanford Univ. Press, 1962.

[EMSS89] E.A. Emerson, A.K. Mok, A.P. Sistla, J. Srinivasan, "Quantitative temporal reasoning," presented at the Workshop on Finite-State Concurrency, Grenoble, France, 1989.

[GPSS80] D. Gabbay, A. Pnueli, S. Shelah, J. Stavi, "On the temporal analysis of fairness," 7th ACM POPL, 1980.

[Ha88] E. Harel, *Temporal Analysis of Real-time Systems,* M.S. Thesis, Weizmann Institute, 1988.

[HLP90] E. Harel, 0. Lichtenstein, A. Pnueli, "Explicit-clock temporal logic," 5th IEEE LICS, 1990.

[HPS83] D. Harel, A. Pnueli, J. Stavi, "Propositional dynamic logic of regular programs," J. Computer and System Sciences 26, 1983.

[JM86] F. Jahanian, A.K. Mok, "Safety analysis of timing properties in real-time systems," IEEE Trans. on Software Engineering SE-12, 1986.

[Ka68] H.W. Kamp, *Tense Logic and the Theory of Linear Order,* Ph.D. Thesis, UCLA, 1968.

[Ko82] D. Kozen, "Results on the propositional μ-calculus," 9th EATCS ICALP, 1982.

[Ko89] R. Koymans, *Specifying Message Passing and Time-critical Systems with Temporal Logic,* Ph.D. Thesis, Eindhoven Univ. of Tech., 1989.

[Le90] H. Lewis, "A logic of concrete time intervals," 5th IEEE LICS, 1990.

[LP84] 0. Lichtenstein, A. Pnueli, "Checking that finite-state concurrent programs satisfy their linear specification," 11th ACM POPL, 1984.

[LPZ85] 0. Lichtenstein, A. Pnueli, L. Zuck, "The glory of the past," Conf. on Logics of Programs, Springer *LNCS* **193**, 1985.

[Mc66] R. McNaughton, "Testing and generating infinite sequences by a finite automaton," Information *and* Control **9**, 1966.

[MP89] Z. Manna, A. Pnueli, "The anchored version of the temporal framework," *Linear* Time, *Branching Time, and Partial Order in Logics and Models for Concurrency* (J.W. deBakker, W.P. deRoever, and G. Rosenberg, eds.), Springer *LNCS 354, 1989.*

[OL82] S. Owicki, L. Lamport, "Proving liveness properties of concurrent programs," ACM *TOPLAS 4,* 1982.

[Os87] J.S. Ostroff, *Temporal Logic of Real-time Systems,* Ph.D. Thesis, Univ. of Toronto, 1987. (Also Research Studies Press, 1990.)

[Pn77] A. Pnueli, "The temporal logic of programs," 18th IEEE FOCS, 1977.

[Si83] A.P. Sistla, *Theoretical Issues in the Design and Verification of Distributed Systems,* Ph.D. Thesis, Harvard Univ., 1983.

[SC85] A.P. Sistla, E.M. Clarke, "The complexity of propositional linear temporal logics ," *JACM 32, 1985.*

[St74] L.J. Stockmeyer, *The Complezity of Decision Problems in Automata Theory and Logic,* Ph.D. Thesis, MIT, 1974.

[Th81] W. Thomas, "A combinatorial approach to the theory of w-automata," Information and Control 48, 1981.

[Wo83] P. Wolper, "Temporal logic can be more expressive," Information and *Control 56, 1983.*

[WVS83] P. Wolper, M.Y. Vardi, A.P. Sistla, "Reasoning about infinite computation paths," 24th IEEE FOCS, 1983.