# On the Expressivity and Complexity of Quantitative Branching-Time Temporal Logics

F. Laroussinie [a], Ph. Schnoebelen [a] and M. Turuani [b]

[a] *Lab. Specification and Verification, ENS de Cachan & CNRS UMR 8643, 61 av. Pdt Wilson, F-94235 Cachan Cedex, France*

[b] *LORIA, Campus Scientifique, BP 239, F-54506 Vandoeuvre-lès-Nancy Cedex, France*

---

**Abstract**

We investigate extensions of *CTL* allowing to express quantitative requirements about an abstract notion of time in a simple discrete-time framework, and study the expressive power of several relevant logics.

When only subscripted modalities are used, polynomial-time model checking is possible even for the largest logic we consider, while the introduction of freeze quantifiers leads to a complexity blow-up.

*Key words:* CTL, Branching-time temporal logic, discrete time, expressivity, model checking, complexity.

---

## 1 Introduction

**Temporal logic** is widely used as a formal language for specifying the behavior of reactive systems (see [Eme90]). This approach allows *model checking*, i.e. the automatic verification that a finite state system satisfies its expected behavioral specifications [CGP99]. The main limitation to model checking is the state-explosion problem but, in practice, *symbolic model checking* techniques [BCM$^+$92] have been impressively successful, and model checking is now commonly used in the design of critical reactive systems.

**Real-time.** While temporal logics only deal with "before and after" properties, *real-time temporal logics* and more generally *quantitative temporal logics* aim at expressing quantitative properties of the time elapsed during computations. Popular real-time logics are based on timed transition systems and appear in several tools (e.g., HyTech, Uppaal, Kronos). The main drawback is that model checking is expensive [ACD93,AL99].

**Efficient model checking.** By contrast, some real-time temporal logics retain usual discrete Kripke structures as models and allow to refer to quantitative information with "bounded" modalities such as "$\mathsf{AF}_{\leq 10} A$" meaning that $A$ will inevitably occur *in at most 10 steps* [Koy90]. Emerson et al. showed that *RTCTL* (i.e. *CTL* plus bounded modalities "$\mathsf{A\_U}_{\leq k}\,\_$" and "$\mathsf{E\_U}_{\leq k}\,\_$" in the Kripke structure framework) still enjoys the bilinear model checking time complexity of *CTL* [EMSS91]. A specific aspect of this framework is that the underlying Kripke structures have no inherent concept of time. It is the designer of the Kripke structure who decides to encode the flow of elapsing time by this or that event, so that the temporal logics in use are more properly called *quantitative temporal logics* than real-time logics.

**Our contribution.** One important question is how far can one go along the lines of *RTCTL*-like logics while still allowing efficient model checking? In this article, we study two quantitative extensions of *CTL*, investigate their expressive power and evaluate the complexity of model checking.

The first extension, called $TCTL_{\mathrm{s}}$, s for "subscripts", is basically the most general logic along the lines of the *RTCTL* proposal : modalities "$\mathsf{A\_U\_}$" and "$\mathsf{E\_U\_}$" are extended with union of integer intervals, this allows combining "$\leq k$", "$\geq k$" and "$= k$". We show this brings real improvements in expressive power, and model checking is still in polynomial time. This extends earlier results for *RTCTL* and *CRTCTL* [EMSS92]. Moreover we use a more general control of the flow of time since we do not require that one transition uses one unit of time.

The second extension, called $TCTL_{\mathrm{c}}$, c for "clocks", uses formula clocks, a.k.a. freeze quantifiers [AH94], and is a more general way of counting events. $TCTL_{\mathrm{c}}$ can still be translated directly into *CTL* but model checking is expensive.

The results on the expressive power formalize natural intuitions which (as far as we know) have never been proven formally (they are only mentioned in [EMSS92]), even in the dense time framework [1]. Furthermore, in our discrete time framework our results on expressive power must be stated in terms

[1] See e.g. the conjecture at the end of [ACD90] which becomes an unproved statement in [ACD93].

of how succinctly a logic can express this or that property. Such proofs are scarce in the temporal logic literature (one example is [Wil99]).

We also investigate the expressive power of the next-free fragments of these logics ($TCTL_{\rm s}\backslash\mathsf{X}$ and $TCTL_{\rm c}\backslash\mathsf{X}$) and we show that they cannot be translated into $CTL\backslash\mathsf{X}$ any more. Moreover we show that $TCTL_{\rm c}\backslash\mathsf{X}$ can be translated into $TCTL_{\rm s}\backslash\mathsf{X}$ but $TCTL_{\rm c}\backslash\mathsf{X}$ remains more succinct than $TCTL_{\rm s}\backslash\mathsf{X}$.

**Related work.** Many timed logics exist in the linear-time framework [AH92]. In the branching-time framework, $TCTL_{\rm s}$ and $TCTL_{\rm c}$ are similar to (and inspired from) logics used in dense real-time frameworks [ACD93]. However, in the discrete framework we use here, their behavior is quite different. Our results on complexity of model checking build on ideas from [DS98,KVW00,AL99,ET99].

Other branching-time extensions of $RTCTL$ have been considered. In [EMSS92], temporal logics allowing modalities $\mathsf{E\_U\_}$ and $\mathsf{A\_U\_}$ with subscripts of the form "$> k$" and "$= k$" are considered: first it is shown that model checking remains polynomial for these logics, and the complexity of satisfiability is characterized. Expressivity is not discussed. [Lew90] considers a more general type of timed structures but does not discuss expressivity, nor complexity of model checking. Counting with regular patterns makes model checking intractable [ET97]. Merging different time scales makes model checking NP-complete [ET99]. Allowing parameters makes model checking use time exponential in the number of parameters [ET99].

Another extension with freeze variables can be found in [YMW97] where richer constraints on the number of occurrences of events can be stated (making satisfiability undecidable) but where the "until" modality is not included and the expressive power of different kinds of constraints is not investigated.

**Plan of the paper.** We introduce the basic notions and definitions in § 2. We discuss expressive power in § 3 and model checking in § 4. We assume the reader is familiar with standard notions of branching-time temporal logic (see [Eme90]) and structural complexity (see [Pap94]).

## 2 $CTL$ + discrete time

We write $\mathbb{N}$ for the set of natural numbers, and $AP = \{A, B, \ldots\}$ for a finite set of *atomic propositions*. Temporal formulae are interpreted over states in Kripke structures. Formally,

**Definition 2.1.** *A Kripke structure (a "KS") is a tuple* $S = \langle Q_S, R_S, l_S \rangle$ *where* $Q_S = \{q_1, \dots\}$ *is a non-empty set of* states, $R_S \subseteq Q_S \times Q_S$ *is a total* transition relation, *and* $l_S : Q_S \to 2^{AP}$ *labels every state with the propositions it satisfies.*

Below, we drop the "$S$" subscript in our notation whenever no ambiguity will arise. A *computation* in a KS is an infinite sequence $\pi$ of the form $q_0 q_1 \dots$ s.t. $(q_i, q_{i+1}) \in R$ for all $i \in \mathbb{N}$. For $i \in \mathbb{N}$, $\pi(i)$ (resp. $\pi_{|i}$) denotes the $i$-th state, $q_i$ (resp. $i$-th prefix: $q_0 q_1, \dots, q_i$). We write $\Pi(q)$ for the set of all computations starting from $q$. Since $R$ is total, $\Pi(q)$ is never empty.

**The flow of time.** We assume a special atomic proposition $tick \in AP$ that describes the elapsing of time in the model. The intuition is that states labeled by *tick* are states where we observe that time has just elapsed, that is the *clock just ticked*. Equivalently, we can see all transitions as taking 1 time unit if they reach a state labeled by *tick*, and as being instantaneous otherwise. Thus KS's with *tick* can be seen as *discrete timed structures*, i.e. KS's where transitions $(q, q') \in R$ are labeled by a natural number: the time it takes to fire the transition [2].

Given a computation $\pi = q_0 q_1 \dots$ and $i \geq 0$ , $\mathsf{Time}(\pi_{|i})$ denotes $|\{j \mid 0 < j \leq i \wedge tick \in l(q_j)\}|$, the time it took to reach $q_i$ from $q_0$ along $\pi$. In pictures, we use different grey levels to distinguish *tick* states from non-*tick* ones.

### *2.1 TCTL*$_{\mathrm{s}}$*, or CTL with bounded modalities*

**Definition 2.2 (Syntax of** $TCTL_{\mathrm{s}}$**).** $TCTL_{\mathrm{s}}$ *formulae are given by the following grammar:*

$$\varphi, \psi \ ::= \ \neg\varphi \mid \varphi \wedge \psi \mid \mathsf{EX}\varphi \mid \mathsf{E}\varphi\mathsf{U}_I\,\psi \mid \mathsf{A}\varphi\mathsf{U}_I\,\psi \mid A \mid B \mid \dots$$

*where* $I$ *can be any finite union* $[a_1, b_1[ \cup \dots \cup [a_n, b_n[$ *of disjoint integer intervals with* $0 \leq a_1 < b_1 < a_2 < b_2 < \cdots a_n < b_n \leq \omega$*.*

Standard abbreviations include $\top, \bot, \varphi \vee \psi, \varphi \Rightarrow \psi, \dots$ as well as $\mathsf{EF}_I\,\varphi$ (for $\mathsf{E}\top\mathsf{U}_I\,\varphi$), $\mathsf{AF}_I\,\varphi$ (for $\mathsf{A}\top\mathsf{U}_I\,\varphi$), $\mathsf{EG}_I\,\varphi$ (for $\neg\mathsf{AF}_I\,\neg\varphi$), and $\mathsf{AG}_I\,\varphi$ (for $\neg\mathsf{EF}_I\,\neg\varphi$).

Moreover we let $\mathsf{U}_{<k}$ stand for $\mathsf{U}_{[0,k[}$ , $\mathsf{U}_{>k}$ for $\mathsf{U}_{[k+1,\omega[}$ , and $\mathsf{U}_{=k}$ for $\mathsf{U}_{[k,k+1[}$ . The usual *CTL* operators are included since the usual $\mathsf{U}$ corresponds to $\mathsf{U}_{<\omega}$ .

[2] While discrete timed structures can be considered as more natural for modeling real-time systems, KS's with *tick* are a technically simpler framework since they do not need numerical labels on transitions. Section 5 has more on the pros and cons of the *tick* model.

**Definition 2.3 (Semantics).** *Figure 1 defines when a state $q$ in some KS $S$, satisfies a TCTL$_\mathrm{s}$ formula $\varphi$, written $q \models \varphi$, by induction over the structure of $\varphi$.*

$$
\begin{array}{ll}
q \models A & \text{iff } A \in l(q), \\
q \models \neg\varphi & \text{iff } q \not\models \varphi, \\
q \models \varphi \wedge \psi & \text{iff } q \models \varphi \text{ and } q \models \psi, \\
q \models \mathsf{EX}\varphi & \text{iff there exists } \pi \in \Pi(q) \text{ s.t. } \pi \models \mathsf{X}\varphi, \\
q \models \mathsf{E}\varphi\mathsf{U}_I\,\psi & \text{iff there exists } \pi \in \Pi(q) \text{ s.t. } \pi \models \varphi\mathsf{U}_I\,\psi \\
q \models \mathsf{A}\varphi\mathsf{U}_I\,\psi & \text{iff for all } \pi \in \Pi(q), \text{ we have } \pi \models \varphi\mathsf{U}_I\,\psi \\
\pi \models \mathsf{X}\varphi & \text{iff } \pi(1) \models \varphi, \\
\pi \models \varphi\mathsf{U}_I\,\psi & \text{iff there exists } i \geq 0 \text{ s.t. } \mathsf{Time}(\pi_{|i}) \in I, \\
& \quad \pi(i) \models \psi, \text{ and } \pi(j) \models \varphi \text{ for all } 0 \leq j < i,
\end{array}
$$

Fig. 1. Semantics of *TCTL*$_\mathrm{s}$

We let *TCTL*$_\mathrm{s}[<]$, *TCTL*$_\mathrm{s}[<,=]$, etc. denote the fragments of *TCTL*$_\mathrm{s}$ where only simple constraints using only $<$ (resp. $<$ or $=$, etc.) are allowed. E.g., *RTCTL* is *TCTL*$_\mathrm{s}[<]$ and *CRTCTL* (see [EMSS92]) is *TCTL*$_\mathrm{s}[<,>,=]$ (with the proviso that our KS's have *tick*'s).

### *2.2 TCTL*$_\mathrm{c}$*, or CTL with formula clocks*

*TCTL*$_\mathrm{c}$ uses freeze quantifiers [AH94]. Here "clocks" are introduced in the formula, set to zero when they are bound, and can be referenced "later" in arbitrary ways. This standard construct gives more flexibility than subscripts.

**Definition 2.4 (Syntax).** *For a set $Cl = \{x, y, \dots\}$ of* clocks, *TCTL*$_\mathrm{c}$ *formulae are given by the following grammar:*

$$\varphi, \psi \ ::= \ \neg\varphi \mid \varphi \wedge \psi \mid \mathsf{EX}\varphi \mid \mathsf{E}\varphi\mathsf{U}\psi \mid \mathsf{A}\varphi\mathsf{U}\psi \mid x \ \underline{\mathtt{in}}\ \varphi \mid x \sim k \mid A \mid B \mid \dots$$

*where $\sim$ can be any comparison operator in $\{=, \leq, <, \geq, >\}$, and $k \in \mathbb{N}$ can be any number.*

An occurrence of a formula clock $x$ in some $x \sim k$ is *bound* if it is in the scope of a "$x$ $\underline{\mathtt{in}}$ " freeze quantifier, otherwise it is *free*. A formula is *closed* if it has no free variables. Only closed formulae express properties of states in KS's.

**Definition 2.5 (Semantics).** *TCTL*$_\mathrm{c}$ *formulae are interpreted over a state of a KS $S$ together with a* valuation $v : Cl \to \mathbb{N}$ *of the clocks in $\varphi$. Figure 2* *defines when $q, v \models \varphi$ in some KS $S$ by induction over the structure of $\varphi$. For $m \in \mathbb{N}$, $v+m$ denotes the valuation which maps each clock $x \in Cl$ to the value $v(x)+m$, and $v[x \leftarrow 0]$ is $v$ where now $x$ evaluates to 0.*

$$
\begin{array}{ll}
q, v \models A & \text{iff } A \in l(q), \\
q, v \models \neg\varphi & \text{iff } q, v \not\models \varphi, \\
q, v \models \varphi \wedge \psi & \text{iff } q, v \models \varphi \text{ and } q, v \models \psi, \\
q, v \models \mathsf{EX}\varphi & \text{iff there exists } \pi \in \Pi(q) \text{ s.t. } \pi, v \models \mathsf{X}\varphi \\
q, v \models \mathsf{E}\varphi\mathsf{U}\psi & \text{iff there exists } \pi \in \Pi(q) \text{ s.t. } \pi, v \models \varphi\mathsf{U}\psi \\
q, v \models \mathsf{A}\varphi\mathsf{U}\psi & \text{iff for all } \pi \in \Pi(q) \text{ we have } \pi, v \models \varphi\mathsf{U}\psi \\
q, v \models x \ \underline{\mathtt{in}}\ \varphi & \text{iff } q, v[x \leftarrow 0] \models \varphi \\
q, v \models x \sim k & \text{iff } v(x) \sim k \\
\pi, v \models \mathsf{X}\varphi & \text{iff } \pi(1), v+d \models \varphi \text{ with } d = \mathsf{Time}(\pi_{|1}) \\
\pi, v \models \varphi\mathsf{U}\psi & \text{iff there exists } i \geq 0 \text{ s.t. } \pi(i), v+d_i \models \psi \text{ and for } 0 \leq j < i \\
& \qquad \pi(j), v+d_j \models \varphi, \text{ where } d_l \stackrel{\text{def}}{=} \mathsf{Time}(\pi_{|l})
\end{array}
$$

Fig. 2. Semantics of $TCTL_{\mathrm{c}}$

Clearly the $TCTL_{\mathrm{s}}$ operators could be defined with the $TCTL_{\mathrm{c}}$ operators:

$$
\mathsf{E}\varphi\mathsf{U}_I\,\psi \equiv x\ \underline{\mathtt{in}}\ \Big(\mathsf{E}\varphi\mathsf{U}(I(x) \wedge \psi)\Big), \qquad \mathsf{A}\varphi\mathsf{U}_I\,\psi \equiv x\ \underline{\mathtt{in}}\ \Big(\mathsf{A}\varphi\mathsf{U}(I(x) \wedge \psi)\Big)
$$

where, for $I$ of the form $[a_1, b_1[\cup\cdots\cup[a_n, b_n[$, $I(x)$ denotes the *clocks constraint*

$$
I(x) \stackrel{\text{def}}{=} \bigvee_{i=1}^{n} \Big((a_i \leq x) \wedge (x < b_i)\Big).
$$

Hence $TCTL_{\mathrm{s}}$ can be seen as a fragment of $TCTL_{\mathrm{c}}$ where only one formula clock is allowed (and used in restricted ways).

For $n \in \mathbb{N}$, we let $TCTL_{\mathrm{c}}[n]$ denote the fragment of $TCTL_{\mathrm{c}}$ where at most $n$ distinct formula clocks occur: the previous remark states that $TCTL_{\mathrm{s}}$ can be seen as a fragment of $TCTL_{\mathrm{c}}[1]$.

Observe that, in the spirit of $TCTL_{\mathrm{s}}$, our definition for $TCTL_{\mathrm{c}}$ only allows simple clock constraints of form $x \sim k$. This is no real loss of expressive power since the other kind of constraints one usually allows, that is, constraints of the form $x \sim y + k$, can be expressed in $TCTL_c$, along the scheme

$$x \ \underline{\texttt{in}} \ \ldots y \ \underline{\texttt{in}} \ [\ldots x \sim y + k \ldots] \ \equiv \ x \ \underline{\texttt{in}} \ \ldots y \ \underline{\texttt{in}} \left( \begin{array}{r} x \sim k \ \Rightarrow \ [\ldots \top \ldots] \\ \wedge \ x \not\sim k \ \Rightarrow \ [\ldots \bot \ldots] \end{array} \right).$$

A standard observation for logics such as $TCTL_c$ is that the actual values recorded in $v$ are only relevant up to a certain point depending on the formula at hand. Let $M_\varphi$ denote the largest constant appearing in $\varphi$ (largest $k$ in the "$x \sim k$"'s) and, for $m \in \mathbb{N}$, let $v \equiv_m v'$ when for any $x \in Cl$, either $v(x) = v'(x)$ or $v(x) > m < v'(x)$ (i.e. $v$ and $v'$ agree, or are beyond $m$).

**Lemma 2.6.** *Let $\varphi$ be a $TCTL_c$ formula. If $v \equiv_m v'$ and $m \geq M_\varphi$, then we have $q, v \models \varphi$ iff $q, v' \models \varphi$.*

*Proof.* Easy induction over the structure of $\varphi$, using the fact that $v \equiv_m v'$ entails $v + k \equiv_m v' + k$ and $v[x \leftarrow 0] \equiv_m v'[x \leftarrow 0]$. □

**Remark 2.7.** *A related property is used by Emerson et al. in their study of RTCTL: when checking $q \models \varphi$ inside some KS with $\mid Q \mid = m$ states, it is possible to replace by $m$ any constant $k$ larger than $m$ in the subscripts of $\varphi$. We emphasize that this property does not hold for $TCTL_s[=]$ (it does hold for $TCTL_s[<,>]$).* □

**Size of formulae.** The size of our formulae is the length of the string [3] used to write them down in a sufficiently succinct way, e.g., $\mid \mathsf{A}\alpha\mathsf{U}_I \beta \mid$ is $1+ \mid \alpha \mid + \mid \beta \mid + \mid I \mid$. For $I$ of the form $[a_1, b_1[ \cup \cdots \cup [a_n, b_n[$, we have $\mid I \mid \stackrel{\text{def}}{=} \lceil \log a_1 \rceil + \cdots + \lceil \log b_n \rceil$ (considering abusively that $\lceil \log(0) \rceil = \lceil \log(\omega) \rceil = 1$). $ht(\varphi)$ denotes the temporal height of formula $\varphi$. As usual, it is the maximal number of nested modalities in $\varphi$. Obviously, $ht(\varphi)$ is smaller than the size of $\varphi$ (even when viewed as a dag).

### *2.3 Basic properties*

Two $TCTL_s$ (resp. $TCTL_c$) formulae $\varphi$ and $\varphi'$ are said to be *equivalent* (denoted by $\varphi \equiv \varphi'$) whenever for any state $q$ in any KS $S$ (resp. for any state $q$ in any KS $S$ and for any valuation $v$ of $\varphi$ clocks), we have $q \models \varphi \ \Leftrightarrow \ q \models \varphi'$ (resp. $q, v \models \varphi \ \Leftrightarrow \ q, v \models \varphi'$).

[3] We sometimes see a formula as a dag, where identical subformulae are only counted once. Such cases are stated explicitly.

We have the following basic properties for $TCTL_{\mathrm{s}}$ formulae:

$$\begin{aligned}
\mathsf{E}\,\varphi\,\mathsf{U}_{<i+1}\ \psi &\equiv \mathsf{E}\,\varphi\,\mathsf{U}_{\leq i}\ \psi \\
\mathsf{A}\,\varphi\,\mathsf{U}_{<i+1}\ \psi &\equiv \mathsf{A}\,\varphi\,\mathsf{U}_{\leq i}\ \psi \\
\mathsf{E}\,\varphi\,\mathsf{U}_{\leq i+1}\ \psi &\equiv \mathsf{E}\,\varphi\,\mathsf{U}_{\leq 1}\Big(\mathsf{E}\,\varphi\,\mathsf{U}_{\leq i}\ \psi\Big) \\
\mathsf{A}\,\varphi\,\mathsf{U}_{\leq i+1}\ \psi &\equiv \mathsf{A}\,\varphi\,\mathsf{U}_{\leq 1}\Big(\mathsf{A}\,\varphi\,\mathsf{U}_{\leq i}\ \psi\Big)
\end{aligned}$$

The following properties, which will be used in later proofs, hold for any (open or closed) $TCTL_{\mathrm{c}}$ formula $\varphi$ and $\psi$:

$$\begin{aligned}
\mathsf{E}\,\varphi\,\mathsf{U}\,\psi &\equiv \psi \vee \Big(\varphi \wedge \mathsf{EX}\,\mathsf{E}\,\varphi\,\mathsf{U}\,\psi\Big) \\
\mathsf{E}\,\varphi\,\mathsf{U}\,\psi &\equiv \mathsf{E}\,(\varphi \wedge \neg tick)\,\mathsf{U}\Big((\neg tick \wedge \psi) \vee (tick \wedge \mathsf{E}\,\varphi\,\mathsf{U}\,\psi)\Big) \\
\mathsf{AF}\,\varphi &\equiv \varphi \vee \mathsf{AX}\,\mathsf{AF}\,\varphi \\
\mathsf{AF}\,\varphi &\equiv \mathsf{A}\,(\neg tick)\,\mathsf{U}\Big((\neg tick \wedge \varphi) \vee (tick \wedge \mathsf{AF}\,\varphi)\Big)
\end{aligned}$$

## 3 Expressivity

### *3.1 Expressivity of $TCTL_{\mathrm{s}}$ and $TCTL_{\mathrm{c}}$*

Formally, $TCTL_{\mathrm{s}}$ or $TCTL_{\mathrm{c}}$ do not add expressive power to $CTL$:
**Theorem 3.1.** *Any closed $TCTL_{\mathrm{c}}$ (or $TCTL_{\mathrm{s}}$) formula is equivalent to a CTL formula.*

*Proof.* With any $TCTL_{\mathrm{c}}$ formula $\varphi$, and valuation $v$, we associate a $CTL$ formula $(\varphi)^v$ s.t. $q, v \models \varphi$ iff $q \models (\varphi)^v$ for any state $q$ *of any Kripke structure.* If $\varphi$ has no free clock variables, we have $q, v \models \varphi \quad\Leftrightarrow\quad q, v' \models \varphi$ for any valuations $v, v'$ and then $(\varphi)^v$ and $(\varphi)^{v'}$ are $CTL$ formulae equivalent to $\varphi$. The definition of $(\varphi)^v$ is given by the following rewrite rules:

$$\begin{aligned}
(\varphi \wedge \psi)^v &\stackrel{\text{def}}{=} \varphi^v \wedge \psi^v & \qquad (x \sim k)^v &\stackrel{\text{def}}{=} \begin{cases} \top & \text{if } v(x) \sim k, \\ \bot & \text{otherwise} \end{cases} \\
(\neg\varphi)^v &\stackrel{\text{def}}{=} \neg\varphi^v & & \\
(A)^v &\stackrel{\text{def}}{=} A & (x\ \underline{\mathtt{in}}\ \varphi)^v &\stackrel{\text{def}}{=} \varphi^{v[x\leftarrow 0]}
\end{aligned}$$

$$(\mathsf{EX}\varphi)^v \stackrel{\text{def}}{=} \begin{cases} \mathsf{EX}\ \varphi^v & \text{if } v+1 \equiv_{M_\varphi} v, \\ \mathsf{EX}\Big(\neg tick \wedge (\varphi)^v\Big) \vee \mathsf{EX}\Big(tick \wedge (\varphi)^{v+1}\Big) & \text{otherwise} \end{cases}$$

$$(\mathsf{AF}\varphi)^v \stackrel{\text{def}}{=} \begin{cases} \mathsf{AF}\varphi^v & \text{if } v+1 \equiv_{M_\varphi} v, \\ \varphi^v \vee \mathsf{AX}\Big[\mathsf{A}(\neg tick)\ \mathsf{U}\ \Big((\neg tick \wedge \varphi^v) \vee (tick \wedge (\mathsf{AF}\varphi)^{v+1})\Big)\Big] & \text{otherwise} \end{cases}$$

$$(\mathsf{E}\varphi\ \mathsf{U}\ \psi)^v \stackrel{\text{def}}{=} \begin{cases} \mathsf{E}\varphi^v\ \mathsf{U}\ \psi^v & \text{if } v+1 \equiv_{M_{\varphi,\psi}} v, \\ \psi^v \vee \Big[\varphi^v \wedge \mathsf{EX}\Big(\mathsf{E}(\varphi^v \wedge \neg tick)\ \mathsf{U}\ \Big((\psi^v \wedge \neg tick) \vee (tick \wedge (\mathsf{E}\varphi \mathsf{U} \psi)^{v+1})\Big)\Big)\Big] & \\ \quad\text{otherwise} & \end{cases}$$

This gives a well-founded definition for $(\_)^v$ since in the right-hand sides either $(\_)^v$ is recursively applied to subformulae, or $(\_)^{v+1}$ is applied on the same formula (or both). But moving from $(\_)^v$ to $(\_)^{v+1}$ is only done until $v \equiv_M v+1$, which is bound to eventually happen. It is easy to check that the correctness invariant (i.e., "$q, v \models \varphi$ iff $q \models (\varphi)^v$") is preserved by these rules: The second rule for $\mathsf{AF}$ and the second rule for $\mathsf{E\_U\_}$ are based on the equivalences given in subsection 2.3. □

The translation we just gave is easy to describe but the resulting $(\varphi)^v$ formulae have enormous size. It turns out that this cannot be avoided. Even more, we can say that moving from *CTL* to $TCTL_s[<]$ to $TCTL_s$ to ... allows writing new formulae that have no succinct equivalent at the previous level. This was partially observed in [EMSS92] but no proof was given.

**Theorem 3.2.** *(1) $TCTL_s[<]$ can be exponentially more succinct than CTL, (2) $TCTL_s[<,>]$ can be exponentially more succinct than $TCTL_s[<]$.*

The proof is given by the following lemmas.

**Lemma 3.3.** *Any CTL formula equivalent to $\mathsf{EF}_{<n} A$ (a $\log n$-sized formula) has temporal height at least $n$.*

*Proof.* Consider the KS described in Figure 3. We show by structural induction over $\varphi$ that for any *CTL* formula $\varphi$, $ht(\varphi) \leq i$ implies ($\alpha_i \models \varphi$ iff $\alpha_{i+1} \models \varphi$):

- The cases $A$, $\neg\varphi_1$ and $\varphi_1 \wedge \varphi_2$ are trivial.
- Assume $\varphi$ is some $\mathsf{EX}\varphi_1$. $ht(\varphi) \leq i$ entails $ht(\varphi_1) \leq i-1$. If $\alpha_i \models \varphi$, then we have $\alpha_{i-1} \models \varphi_1$ and then, by ind. hyp., we have $\alpha_i \models \varphi_1$, this entails $\alpha_{i+1} \models \varphi$. Conversely if $\alpha_{i+1} \models \varphi$, then we have $\alpha_i \models \varphi_1$ and then, by ind. hyp., we have $\alpha_{i-1} \models \varphi_1$ which entails $\alpha_i \models \varphi$.
- Assume $\varphi$ is some $\mathsf{E}\ \varphi_1\ \mathsf{U}\ \varphi_2$. If $\alpha_i \models \varphi$, then we have either $\alpha_i \models \varphi_2$ or $\alpha_i \models \varphi_1$ and $\alpha_{i-1} \models \varphi$. In the first (resp. second) case, the induction hypothesis allows us to deduce $\alpha_{i+1} \models \varphi_2$ (resp. $\alpha_{i+1} \models \varphi_1$) and in both case it gives $\alpha_{i+1} \models \varphi$. Conversely if $\alpha_{i+1} \models \varphi$, we have either $\alpha_{i+1} \models \varphi_2$

(and then, by ind. hyp., we have $\alpha_i \models \varphi_2$ and $\alpha_i \models \varphi$) or it entails directly $\alpha_i \models \varphi$.

- The case where $\varphi$ is some $\mathsf{A}\,\varphi_1\,\mathsf{U}\,\varphi_2$ is similar to the previous one.

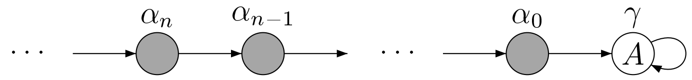


Fig. 3. $\alpha_n \models \mathsf{EF}_{<n+1}\,A$ and $\alpha_{n+1} \not\models \mathsf{EF}_{<n+1}\,A$

On the other hand, $\alpha_j \models \mathsf{EF}_{<n}\,A$ iff $j < n$. Thus any *CTL* formula equivalent to $\mathsf{EF}_{<n}\,A$ must have temporal height larger than or equal to $n$. $\square$

**Lemma 3.4.** *Any $TCTL_{\mathrm{s}}[<]$ formula equivalent to $\mathsf{EF}_{>n}\,A$ (a $\log n$-sized formula) has temporal height at least $n$.*

*Proof.* Consider the KS described in Figure 4.

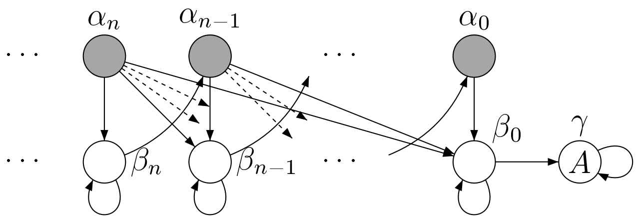


Fig. 4. $\alpha_n \not\models \mathsf{EF}_{>n}\,A$ and $\alpha_{n+1} \models \mathsf{EF}_{>n}\,A$

One shows (by structural induction over $\varphi$) that for any formula $\varphi$ in $TCTL_{\mathrm{s}}[<]$, $ht(\varphi) \leq i$ implies ($\alpha_i \models \varphi$ iff $\alpha_{i+1} \models \varphi$) and ($\beta_i \models \varphi$ iff $\beta_{i+1} \models \varphi$). For example, consider the case where $\varphi$ is some $\mathsf{E}\varphi_1\,\mathsf{U}_{<k}\,\varphi_2$:

- assume $\alpha_{i+1} \models \varphi$, then either $\alpha_{i+1} \models \varphi_2$ (and then $\alpha_i \models \varphi_2$ by i.h.) or $\alpha_{i+1} \models \varphi_1$ (and $\alpha_i \models \varphi_1$) and there is some $j \leq i+1$ s.t. $\beta_j \models \varphi$. Either $j \leq i$ and clearly we have $\alpha \models \varphi$ or $\beta_{i+1} \models \varphi$ and: $\beta_{i+1} \models \varphi_2$ (then $\beta_i \models \varphi_2$, this gives the result) or $\beta_{i+1} \models \varphi_1$ and $\alpha_i \models \varphi$. In every case we have $\alpha_i \models \varphi$. Now assume $\alpha_i \models \varphi$, it is easy to deduce $\alpha_{i+1} \models \varphi$.
- assume $\beta_{i+1} \models \varphi$, we want to show $\beta_i \models \varphi$. Either $\beta_{i+1} \models \varphi_2$ (and then $\beta_i \models \varphi_2$ by i.h. and $\beta_i \models \varphi$) or $\beta_{i+1} \models \varphi_1$ (and then $\beta_i \models \varphi_1$) and $\alpha_i \models \mathsf{E}\varphi_1\mathsf{U}_{<k-1}\,\varphi_2$. Either $\alpha_i \models \varphi_2$ (and this entails $\alpha_{i-1} \models \varphi_2$ by i.h. and this gives the result) or $\alpha_i \models \varphi_1$ (and $\alpha_{i-1} \models \varphi_1$) and: either $\beta_i \models \mathsf{E}\varphi_1\mathsf{U}_{<k-1}\,\varphi_2$ (this entails the result) or $\beta_j \models \mathsf{E}\varphi_1\mathsf{U}_{<k-1}\,\varphi_2$ for some $j < i$, and then we get the result due to the transition $(\alpha_{i-1}, \beta_j)$ and because $\varphi_1$ holds for $\beta_i$ and $\alpha_{i-1}$.
  Assume $\beta_i \models \varphi$. We want to show $\beta_{i+1} \models \varphi$. Either $\beta_i \models \varphi_2$ (and then $\beta_{i+1} \models \varphi$ by i.h.) or $\beta_i \models \varphi_1$ (and then $\beta_{i+1} \models \varphi_1$) and $\alpha_{i-1} \models \mathsf{E}\varphi_1\mathsf{U}_{<k-1}\varphi_2$: either $\alpha_{i-1} \models \varphi_2$ (and then $\alpha_i \models \varphi_2$ by i.h. and $\beta_{i+1} \models \varphi$) or $\alpha_i \models \varphi_1$ (and $\alpha_{i+1} \models \varphi_1$) and $\beta_j \models \mathsf{E}\varphi_1\mathsf{U}_{<k-1}\varphi_2$ for some $j < i$ and we also get the result due to the transition $(\alpha_{i+1}, \beta_j)$.

On the other hand, $\alpha_j \models \mathsf{EF}_{>n} A$ iff $j > n$. Thus any $TCTL_\text{s}[<]$ equivalent to $\mathsf{EF}_{>n} A$ must have temporal height larger than $n$. □

Let us mention three (natural) conjectures that would allow to separate further fragments:

**Conjecture 3.5.** *(1) $TCTL_\text{s}[<,>,=]$ can be exponentially more succinct than $TCTL_\text{s}[<,>]$,*

*(2) $TCTL_\text{c}[1]$ can be exponentially more succinct than $TCTL_\text{s}$,*

*(3) $TCTL_\text{c}[n+1]$ can be exponentially more succinct than $TCTL_\text{s}[n]$.*

Despite repeated attempts, we have been unable to prove these conjectures. The first point is based on the conjecture that any $TCTL_\text{s}[<,>]$ formula equivalent to $\mathsf{EF}_{=k} A$ has temporal height at least $k$. For the second one, we conjecture that any $TCTL_\text{s}$ formula equivalent to $x \ \underline{\mathtt{in}}\ \mathsf{EF}\big(A \wedge \mathsf{EF}(B \wedge x = k)\big)$ has size at least $k$. Finally we conjecture that any $TCTL_\text{c}[n]$ formula equivalent to $\varphi_{n+1} = \mathsf{EF}(A_1 \wedge x_1 \ \underline{\mathtt{in}}\ \mathsf{EF}(A_2 \wedge \ldots x_{n+1} \ \underline{\mathtt{in}}\ (B \wedge \bigwedge_{i=1}^{n+1} x_i = k_i)))$ has size exponential in $|\varphi_{n+1}|$.

We have explained how $TCTL_\text{s}$ fragments become more and more expressive when we allow subscripts with $<$, then also with $>$, then also with $=$. Adding subscripts of the form "$= k$" is an important step: they enhance expressivity, they make model checking more complex (see § 4) and once we have $TCTL_\text{s}[<,>,=]$, subscripts with intervals are just a convenient shorthand:

**Theorem 3.6.** *$TCTL_\text{s}$ is not more succinct than $TCTL_\text{s}[<,>,=]$.*

*Proof.* For $I$ of the form $\bigcup_{i=1\ldots n}[a_i, b_i[$, we denote by $I-k$ the set $\bigcup_{i=1\ldots n}[a_i-k, b_i-k[$ (after the obvious normalization if $k > a_1$).

Let $\varphi$ be a $TCTL_\text{s}$ formula. We build an equivalent $TCTL_\text{s}[<,>,=]$ formula $\tilde{\varphi}$ with the following equivalences:

$$\mathsf{E}\,\alpha\,\mathsf{U}_I\,\beta \equiv \bigvee_{i=1\ldots n} \mathsf{E}\,\alpha\,\mathsf{U}_{=a_i}\,(\mathsf{E}\,\alpha\,\mathsf{U}_{<b_i-a_i}\,\beta)$$

$$\mathsf{A}\,\alpha\,\mathsf{U}_I\,\beta \equiv \begin{cases} \mathsf{A}\,\alpha\,\mathsf{U}_{=a_1}\,(\mathsf{A}\,\alpha\,\mathsf{U}_{I-a_1}\,\beta) & \text{if } a_1 > 0, \\ \\ \quad\neg\mathsf{E}(\neg\beta)\mathsf{U}_{<b_1}\,(\neg\,\alpha\ \wedge \neg\beta) & \\ \wedge\,\neg\mathsf{E}(\neg\beta)\mathsf{U}_{=b_1}\,\Big(\neg\mathsf{A}\,\alpha\,\mathsf{U}_{=a_2-b_1}\,(\mathsf{A}\,\alpha\,\mathsf{U}_{I-a_2}\,\beta)\Big) & \text{otherwise} \end{cases}$$

The correctness of the first equivalences is easy to check. The last one is based on the following equivalences where we use the $CTL^+$ notation [Eme90], and where $I'$ denotes the fragment $[a_2, b_2[\cup[a_3, b_3[\cup\ldots$:

$$\begin{aligned}
\mathsf{A}\,\alpha\,\mathsf{U}_{[0;b_1[\cup I'}\,\beta &\equiv \neg\mathsf{E}\Big(\neg(\alpha\mathsf{U}_{<b_1}\,\beta)\wedge\neg(\alpha\mathsf{U}_{I'}\,\beta)\Big)\\
&\equiv \neg\mathsf{E}\Big(\Big((\neg\beta)\mathsf{U}_{<b1}\,(\neg\alpha\wedge\neg\beta)\vee\mathsf{G}_{<b_1}\,(\neg\beta)\Big)\wedge\neg(\alpha\mathsf{U}_{I'}\,\beta)\Big)\\
&\equiv \neg\mathsf{E}\Big((\neg\beta)\mathsf{U}_{<b1}\,(\neg\alpha\wedge\neg\beta)\vee\Big(\mathsf{G}_{<b_1}\,(\neg\beta)\wedge\neg(\alpha\mathsf{U}_{I'}\,\beta)\Big)\Big)\\
&\equiv \neg\mathsf{E}\Big((\neg\beta)\mathsf{U}_{<b1}\,(\neg\alpha\wedge\neg\beta)\Big)\wedge\neg\mathsf{E}\Big(\mathsf{G}_{<b_1}\,(\neg\beta)\wedge\neg(\alpha\mathsf{U}_{I'}\,\beta)\Big)\\
&\equiv \neg\mathsf{E}\Big((\neg\beta)\mathsf{U}_{<b1}\,(\neg\alpha\wedge\neg\beta)\Big)\wedge\neg\mathsf{E}(\neg\beta)\mathsf{U}_{=b_1}\,\Big(\mathsf{E}\neg(\alpha\mathsf{U}_{I'-b_1}\,\beta)\Big)\\
&\equiv \neg\mathsf{E}\Big((\neg\beta)\mathsf{U}_{<b1}\,(\neg\alpha\wedge\neg\beta)\Big)\wedge\neg\mathsf{E}(\neg\beta)\mathsf{U}_{=b_1}\,\Big(\neg\mathsf{A}\alpha\mathsf{U}_{I'-b_1}\,\beta\Big)
\end{aligned}$$

Finally we have [4] $\mathsf{A}\alpha\mathsf{U}_{I'-b_1}\,\beta\equiv\mathsf{A}\alpha\mathsf{U}_{=a_2-b_1}\,\Big(\mathsf{A}\alpha\mathsf{U}_{I'-a_2}\,\beta\Big)$.

In this translation, the size of the resulting $\tilde{\varphi}$, seen as a dag, is linear in the size of $\varphi$ seen as a dag [5]. □

### *3.2 Expressivity of next-free fragments: $TCTL_s\backslash\mathsf{X}$ and $TCTL_c\backslash\mathsf{X}$*

$TCTL_s\backslash\mathsf{X}$ and $TCTL_c\backslash\mathsf{X}$ are the fragments of $TCTL_s$ and, respectively, $TCTL_c$ where the "next-time", $\mathsf{X}$, modality is not allowed. The importance of such fragments was stressed in [Lam83].

The subscript operators of $TCTL_s\backslash\mathsf{X}$, or the formula clocks of $TCTL_c\backslash\mathsf{X}$, clearly add expressive power to $CTL\backslash\mathsf{X}$. Indeed no $CTL\backslash\mathsf{X}$ formula can distinguish two Kripke structures which are stuttering equivalent [BCG88] while such structures can sometimes be distinguished by $TCTL_s\backslash\mathsf{X}$ formulae and their ability to count the number of *tick*. For example, the two structures of Figure 5 (where only grey states satisfy *tick*) are stuttering equivalent but $q\models\mathsf{EF}_{>0}\,tick$ and $q'\not\models\mathsf{EF}_{>0}\,tick$. Therefore $TCTL_s\backslash\mathsf{X}$ and $TCTL_c\backslash\mathsf{X}$ cannot be translated into $CTL\backslash\mathsf{X}$.

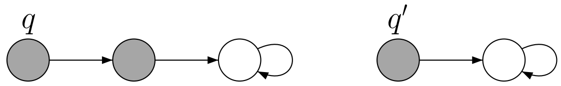


Fig. 5. $q\models\mathsf{EF}_{>0}\,tick$ and $q'\not\models\mathsf{EF}_{>0}\,tick$.

Nevertheless $TCTL_c\backslash\mathsf{X}$ can be translated into $TCTL_s\backslash\mathsf{X}$:

[4] Remember that $b_1 < a_2$.

[5] Viewing formulae as dags is convenient here, and agree with our later use of Theorem 3.6 when we investigate efficient model checking for $TCTL_s$.

**Theorem 3.7.** *Any closed* $TCTL_c \backslash \mathsf{X}$ *formula is equivalent to a* $TCTL_s \backslash \mathsf{X}$ *formula.*

*Proof.* We adapt the translation of Theorem 3.1 and do without $\mathsf{EX}$ modalities in the right-hand sides of the rewrite rules. With any $TCTL_c \backslash \mathsf{X}$ formula $\varphi$, and valuation $v$, we associate a $TCTL_s \backslash \mathsf{X}$ formula $(\varphi)^v$ s.t. for any state $q$ of any Kripke structure, $q, v \models \varphi$ iff $q \models (\varphi)^v$. If $\varphi$ has no free clock variable, we have $q, v \models \varphi \Leftrightarrow q, v' \models \varphi$ for any valuations $v, v'$ and then $(\varphi)^v$ and $(\varphi)^{v'}$ are $TCTL_s \backslash \mathsf{X}$ formulae equivalent to $\varphi$. We use the rules of the proof of Theorem 3.1 for operators $\wedge$, $\neg$ and $(x \ \underline{\mathtt{in}}\ \varphi)$, for atomic constraints $(x \sim k)$, and for atomic propositions. There just remains to give new rules for $(\mathsf{E}\varphi \,\mathsf{U}\, \psi)^v$ and [6] $(\mathsf{EG}\varphi)^v$:

$$(\mathsf{EG}\ \varphi)^v \stackrel{\text{def}}{=} \begin{cases} \mathsf{EG}\ \varphi^v & \text{if } v+1 \equiv_{M_\varphi} v, \\ \mathsf{EG}\Big(\neg \mathit{tick} \wedge \varphi^v\Big) \ \vee\ \mathsf{E}\ \varphi^v\ \mathsf{U}_{=1} \Big[\mathit{tick} \wedge (\mathsf{EG}\ \varphi)^{v+1}\Big] & \text{otherwise} \end{cases}$$

$$(\mathsf{E}\varphi\ \mathsf{U}\ \psi)^v \stackrel{\text{def}}{=} \begin{cases} \mathsf{E}\ \varphi^v\ \mathsf{U}\ \psi^v & \text{if } v+1 \equiv_{M_{\varphi,\psi}} v, \\ \psi^v \vee \Big[\varphi^v \wedge \Big((\mathsf{E}\ \varphi^v\ \mathsf{U}_{\leq 0}\ \psi^v)\ \vee\ \mathsf{E}\ \varphi^v\ \mathsf{U}_{=1}\Big(\mathit{tick} \wedge (\mathsf{E}\varphi \mathsf{U} \psi)^{v+1}\Big)\Big)\Big] \\ \quad \text{otherwise} \end{cases}$$

Again these rules give a well-founded definition for $(\_)^v$. □

Observe that Lemma 3.4 also shows that $TCTL_s[<,>] \backslash \mathsf{X}$ can be exponentially more succinct than $TCTL_s[<] \backslash \mathsf{X}$. Moreover conjecture 3.5 can be extended to the next-free fragments. Finally note that Theorem 3.6 still holds for $TCTL_s \backslash \mathsf{X}$ because its proof does not use the $\mathsf{EX}$ operator: $TCTL_s \backslash \mathsf{X}$ is not more succinct than $TCTL_s[<,>,=] \backslash \mathsf{X}$.

## 4 Model checking

For the logics we investigate, the *model checking problem* is the problem of computing whether $q \models \varphi$ for $q$ a state of a KS $S$ and $\varphi$ a temporal formula. In this section we analyze the complexity of model checking problems for $TCTL_s$ and $TCTL_c$.

Given a KS $S$ and a formula $\varphi$, the complexity of model checking can be evaluated in term of $|S|$ ($\stackrel{\text{def}}{=} |Q_S| + |R_S|$) and $|\varphi|$. But, following [VW86], more discriminating information can be obtained by also looking at the *program*

[6] It is simpler to deal with $\mathsf{EG}$ operator instead of $\mathsf{AF}$.

*complexity* of model checking (i.e., the complexity when $\varphi$ is fixed and $S, q$ is the only input) and its *formula complexity* (i.e., when $S, q$ is fixed and $\varphi$ is the only input).

### *4.1 Model checking $TCTL_{\mathrm{s}}$*

$TCTL_{\mathrm{s}}$ model checking can be done efficiently.

**Theorem 4.1.** • *There exists a model checking algorithm for $TCTL_{\mathrm{s}}$ formulae that runs in time $O\big(|Q_S|^3 \times |\varphi|\big)$.*

- *There exists a model checking algorithm for $TCTL_{\mathrm{s}}[<,>]$ formulae that runs in time $O\big(|S| \times |\varphi|\big)$.*

*Proof.* In [EMSS92] it is proven that *RTCTL* model checking can be done in time $O\big(|S| \times |\varphi|\big)$ and *CRTCTL* model checking can be done in time $O\big(|Q|^3 \times |\varphi|\big)$. These algorithms can be easily adapted to take into account our *tick* proposition [7], yielding model checking algorithms for $TCTL_{\mathrm{s}}[<]$ and $TCTL_{\mathrm{s}}[<,>,=]$. Now the extension to $TCTL_{\mathrm{s}}$ only relies on Theorem 3.6.

If $\varphi$ belongs to $TCTL_{\mathrm{s}}[<,>]$, model checking can be done in time $O\big(|S| \times |\varphi|\big)$. Procedures to deal with the "$< k$" subscripts are simply adapted from algorithms for *RTCTL*, and it only remains to deal with the "$> k$" subscripts. First note that any subscript "$> k$" can be replaced by "$> \min(k, |Q|)$". Moreover we have the two following equivalences:

$$\mathsf{E}\alpha\mathsf{U}_{>k}\,\beta \equiv \mathsf{EG}_{\leq k}\left(\mathsf{E}\alpha\mathsf{U}_{>0}\,\beta\right)$$
$$\mathsf{A}\alpha\mathsf{U}_{>k}\,\beta \equiv \mathsf{AG}_{\leq k}\left(\mathsf{A}\alpha\mathsf{U}_{>0}\,\beta\right)$$

and it only remains to use efficient procedures for $\mathsf{E}\_\mathsf{U}_{>0}\,\_$ and $\mathsf{A}\_\mathsf{U}_{>0}\,\_$ or, equivalently, to observe that they can be encoded succinctly with *CTL* operators :

$$\mathsf{E}\alpha\mathsf{U}_{>0}\,\beta \equiv \alpha \wedge \mathsf{EX}\Big(\mathsf{E}\alpha\mathsf{U}(tick \wedge \mathsf{E}\alpha\mathsf{U}\beta)\Big)$$
$$\mathsf{A}\alpha\mathsf{U}_{>0}\,\beta \equiv \alpha \wedge \mathsf{AX}\Big(\mathsf{A}\alpha\mathsf{U}(tick \wedge \mathsf{A}\alpha\mathsf{U}\beta)\Big)$$

This gives an algorithm running in time $O\big(|S| \times |\varphi|\big)$.

□

[7] Remember that [EMSS92] assumes that any transition takes exactly one time unit.

Unlike $TCTL_s$, model checking for $TCTL_c$ cannot be done efficiently (even when considering a fixed KS).

**Theorem 4.2.** *The model checking problem for $TCTL_c$ is PSPACE-complete. The formula complexity of $TCTL_c$ model checking is PSPACE-complete.*

The proof is given by the following two propositions since it is sufficient to show that $TCTL_c$ model checking is in PSPACE, and that PSPACE-hardness occurs even with a fixed $S$.

**Proposition 4.3.** *There exists a fixed KS on which model checking $TCTL_c$ formulae is PSPACE-hard.*

*Proof.* This relies on ideas from [AL99]: let $P$ be an instance of QBF (Quantified Boolean Formula, a PSPACE-complete problem). Formally, $P$ is some $Q_1 b_1 \dots Q_n b_n.\varphi$, where every $Q_i$ is a quantifier in $\{\exists, \forall\}$, and $\varphi$ is a propositional formula over the boolean variables $b_1, \dots, b_n$. We reduce $P$ to a model checking problem $S, q \models \Phi$. Here $S$ is the simple KS given by $Q_S \stackrel{\text{def}}{=} \{q\}$, $R_S \stackrel{\text{def}}{=} \{(q,q)\}$, and $l_S(q) \stackrel{\text{def}}{=} \{tick\}$. Hence $S$ has only behavior, where the clock ticks at every step.

$\Phi$ is the following $TCTL_c$ formula:

$$t \ \underline{\mathtt{in}}\ \mathsf{EF}\Big[t{=}1 \wedge O_1\Big(x_1 \ \underline{\mathtt{in}}\ \mathsf{EF}\Big[t{=}2 \wedge \dots \Big(t{=}i \wedge O_i(x_i \ \underline{\mathtt{in}}\ \mathsf{EF}(t{=}i{+}1 \wedge \dots \\ \mathsf{EF}(t{=}n{+}1 \wedge \tilde{\varphi}) \dots \Big)\Big]\Big)\Big]$$

where $O_i$ is $\mathsf{EF}_{\leq 1}$ (resp. $\mathsf{EG}_{\leq 1}$) if $Q_i$ is $\exists$ (resp. $\forall$), and where $\tilde{\varphi}$ is $\varphi$ where occurrences of $b_i$ have been replaced by "$x_i = n+1-i$". The intuition behind this is that, in $\Phi$, the clock $x_i$ is reset at (global) time $t = i$ or $t = i+1$ and, depending on this reset time, the atomic propositions $b_i$ will be interpreted as true or false after the $n{+}1$-*th* transition. The operator $\mathsf{EF}_{\leq 1}$ (resp. $\mathsf{EG}_{\leq 1}$) allows to quantify existentially (resp. universally) over these two reset times for $x_i$. Clearly $P$ is true iff $S, q \models \Phi$. □

**Proposition 4.4.** *Model checking for $TCTL_c$ can be done in PSPACE.*

*Proof.* Assume $S = \langle Q, R, l \rangle$ is a KS, $\varphi$ is a $TCTL_c$ formula, $Cl = \{x_1, \dots, x_n\}$ is the set of clocks occurring in $\varphi$, and $M$ is the largest constant appearing in $\varphi$.

A simple way to explain how one can check whether $S, q \models \varphi$ in PSPACE, is to transform this question into a *symbolic model checking* [8] problem for *CTL*, known to be in PSPACE. Following the classical "region graph" approach [ACD93], the state space that has to be explored is $\mathcal{S} = \{(q, v) \mid q \in Q \text{ and } v \in \mathbb{N}^{Cl}\}$. In fact, only valuations in $\{0, 1, \ldots, M, M+1\}^{Cl}$ needs be considered (cf. Lemma 2.6).

Transitions between these *extended states* reflect the semantical clauses of Fig. 2: we have $(q, v) \to (q', v')$ iff there exists a transition $(q, q') \in R$ and $v' = v + 1$ (resp. $v' = v$) if $tick \in l(q')$ and $v + 1 \not\equiv_M v$ (resp. if $tick \notin l(q')$ or $v + 1 \equiv_M v$).

The operator $x_i$ $\underline{\texttt{in}}$ ... can be seen as a special kind of next operator dealing with virtual transitions between any $(q, v)$ to $(q, v[x_i \leftarrow 0])$. The clock constraints $x \sim k$ inside $\varphi$ can be considered as atomic propositions over extended states. Therefore it remains to perform a classical model checking algorithm for *CTL* over the extended states space.

The set of extended states (and the relations among them) can be seen as a synchronous product of $n+1$ finite graphs : $S$ and the $n$ clocks. Each clock is a graph with $M+2$ states, corresponding to the $M+2$ possible values. $M$ is in $O(2^{|\varphi|})$ but we can represent the graphs for one clock as a synchronization of $\lceil \log_2 M + 2 \rceil$ 2-states graphs, so that we end up checking a synchronization of $O(|\varphi|)$ graphs having size 2 or $|S|$. $\square$

**Remark 4.5.** *In practice, and with the help of a CTL symbolic model checker, one can easily use the above approach for model checking $TCTL_c$ formulae, and the resulting algorithm runs in time $O(|S| \times M^{|Cl|} \times |\varphi|)$. For example, with SMV, one just adds one variable for each formula clock and update them in the obvious way. This approach seems to be much more practical than a translation-based one that would rely on Theorem 3.1.*

The proof that $TCTL_c$ model checking is PSPACE-hardness (Prop. 4.3) uses several freeze quantifiers, but in section 3 we proved that any $TCTL_s$ formula can be rewritten as a $TCTL_c[1]$ formula (i.e. a $TCTL_c$ formula with one formula clock). Then an interesting question is the complexity of model checking for $TCTL_c[1]$.

In fact, a complexity blow-up already occurs with one freeze quantifier:
**Proposition 4.6.** *The model checking problem for $TCTL_c[1]$ is NP-hard and coNP-hard.*

*Proof.* We show NP-hardness by a reduction from SAT and we derive coNP-hardness from the fact that $TCTL_c[1]$ is closed under negation.

[8] Called *model checking for concurrent systems* in [KVW00] where the PSPACE algorithm is given.

Let $\Phi = \bigwedge_i \bigvee_j l_{i,j}$ be an instance of SAT, where the $l_{i,j}$ are literals built with the boolean variables $\{b_1, \ldots, b_n\}$. We reduce this to a $TCTL_c[1]$ model checking problem. Let $p_1, \ldots, p_n$ be the $n$ first prime numbers. Note that $p_n$ is bounded by a polynomial function of $n$ (a consequence of the Prime Number Theorem). Moreover let $M$ denote $p_1 \times \cdots \times p_n$: w.l.o.g. we assume $M > 1$, i.e. $n > 0$.

With any integer $m$ we associate a valuation $v_m$ of the $b_k$'s with, for $k = 1, \ldots, n$,

$$v_m(b_k) = \top \stackrel{\text{def}}{\Leftrightarrow} m \bmod p_k = 0.$$

Observe that any valuation $v$ of the $b_k$'s is a $v_m$ for some $0 < m \leq M$: in fact, taking $m = \prod_{v(b_k)=\top} p_k$ suffices.

Consider now $S$, the KS described in Figure 6 where every state is labeled by *tick*.

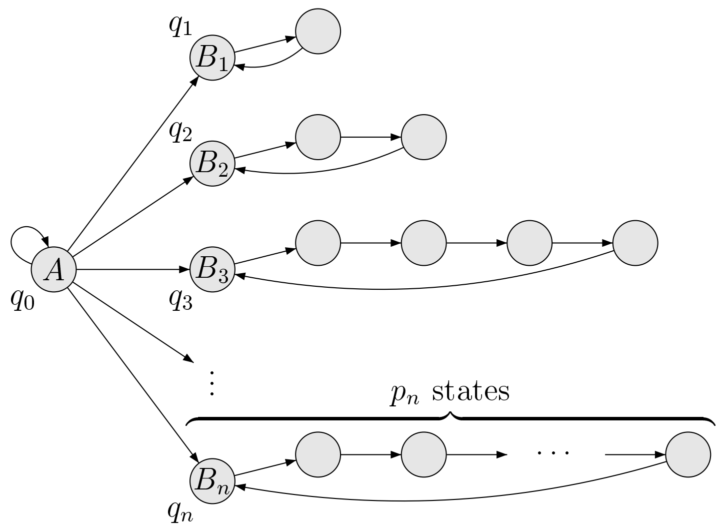


Fig. 6. Reducing SAT to a model checking problem for $TCTL_c[1]$.

The interesting paths in $S$ will loop in $q_0$ for some time and then enter a state $q_k$ (at some time $m$) where it is only possible to follow a loop made of $p_k$ distinct states, and visit $q_k$ at times $m, m+p_k, m+2p_k, \ldots$ Hence if one reaches $q_k$ as time $m \leq M$, one can eventually obtain $B_k \wedge x = M$ iff $m \bmod p_k = 0$, i.e. iff $v_m(b_k) = \top$.

Now consider the formula $\Psi$ defined as follows:

$$\Psi \stackrel{\text{def}}{=} x \ \underline{\texttt{in}}\ \mathsf{EF} \left( A \ \wedge \ x < M \ \wedge \ \bigwedge_i \bigvee_j \mathsf{EX}\, \psi_{i,j} \right),$$

where $\psi_{i,j}$ depends on literal $l_{i,j}$:

$$\psi_{i,j} \stackrel{\text{def}}{=} \begin{cases} B_k \wedge \mathsf{EF}(B_k \wedge x = M) & \text{if } l_{i,j} = b_k, \\ B_k \wedge \mathsf{AG}(B_k \Rightarrow x \neq M) & \text{if } l_{i,j} = \neg b_k. \end{cases}$$

Hence $\psi_{i,j}$ states that we arrive in $q_k$ at a time $m$ s.t. $v_m(l_{i,j}) = \top$. Clearly $q_0 \models \Psi$ iff $\Phi$ is satisfiable. Both $S$ and $\Psi$ have polynomial size (and can be built in logspace) because the subformulae $x \sim M$ have size $O(\log M)$, i.e. $O(n^2 \log n)$. $\square$

### *4.3 Overview of the complexity of model checking*

The following table gives a synthetic summary of complexity measures for model checking *CTL*, $TCTL_{\mathrm{s}}$ and $TCTL_{\mathrm{c}}$, showing that model checking the full $TCTL_{\mathrm{s}}$ is as tractable as model checking *CTL* (in both arguments). On the other hand, model checking $TCTL_{\mathrm{c}}$ requires polynomial space even for a fixed Kripke structure.

<table>
<tr><th></th><th>CTL</th><th>TCTL<sub>s</sub></th><th>TCTL<sub>c</sub></th></tr>
<tr><td>Complexity of model checking</td><td colspan="2">P-complete, (1)</td><td rowspan="2">PSPACE-complete, (4)</td></tr>
<tr><td>Formula complexity</td><td colspan="2">LOGSPACE, (2)</td></tr>
<tr><td>Program complexity</td><td colspan="3">NLOGSPACE-complete, (3)</td></tr>
</table>

**Filling the table.** These results are justified as follows:

(1) Model checking $TCTL_{\mathrm{s}}$ is in P as we just saw. That P-hardness already occurs for the $\mathsf{EX}$ fragment of *CTL* is folklore.

(2) We need to show that, for fixed $S$ and $q$, model checking $TCTL_{\mathrm{s}}$ formulae can be done in LOGSPACE. We first explain the idea for *CTL*: one computes $Mod_S(\varphi) = \{q \in Q_S \mid q \models \varphi\}$ in a compositional way. $\varphi$ is seen as a tree, any subformula evaluates to some subset of $Q$, and $Mod_S(\mathsf{A}\psi_1\mathsf{U}\psi_2)$, or $Mod_S(\mathsf{E}\psi_1\mathsf{U}\psi_2)$, is obtained by combining $Mod_S(\psi_1)$ and $Mod_S(\psi_2)$. For fixed $S$, combining the $Mod(\psi_i)$ can be done in constant time, e.g. by table lookup since, writing $N$ for $|Q|$, there are only $2^N$ (a finite number) possible values. Therefore, evaluating $Mod(\varphi)$ is just evaluating an expression over a finite domain, which can be done smartly using classical algorithms that store at most $\lceil \log_2 |\varphi| \rceil$ intermediary results [Ers58]. Hence the LOGSPACE complexity.

This approach directly extends to $TCTL_{\mathrm{s}}[<,>]$ where all constants in subscripted modalities can be bounded by $N$ (cf. Remark 2.7) so that we only have to deal with a finite number of distinct modalities.

For the full $TCTL_{\mathrm{s}}$ we need one more trick: We extend the labeling of $S$ so that every state is a proposition, with $q \models q'$ iff $q = q'$, and observe that for fixed $S$ and $k > N$, $q \models \mathsf{E}\varphi\mathsf{U}_{=k}\,\psi$ is equivalent to

$$\bigvee_{q' \in Q} \bigvee_{k_1 \leq N} \bigvee_{\substack{1 \leq k_2 \leq N \\ (k-k_1) \bmod k_2 = 0}} \Big( q \models \mathsf{E}\varphi\mathsf{U}_{=k_1}\, q' \;\wedge\; q' \models \psi \wedge \mathsf{E}\varphi\mathsf{U}_{=k_2}\, q' \Big).$$

Hence evaluating $Mod_S(\mathsf{E}\varphi\mathsf{U}_{=k}\psi)$ only needs to compute a bounded number of $(k-k_1) \bmod k_2$ and a bounded number of combinations of $Mod_s(\varphi)$ and $Mod_S(\psi)$. The same approach is used for $A\varphi\mathsf{U}_{=k}\,\psi$.

(3) The program complexity of model checking $TCTL_{\mathrm{s}}$ and $TCTL_{\mathrm{c}}$ is NLOGSPACE-complete since these logics are not more expressive than $CTL$ (Theorem 3.1), for which the NLOGSPACE-complete program complexity is given in [KVW00].

(4) Due to Theorem 4.2

### *4.4 Symbolic model checking.*

When it comes to symbolic model checking (i.e., when $S$ is given under the form of a synchronized product of $k$ structures $S_1, \ldots, S_k$), $CTL$ model checking becomes PSPACE-complete [KVW00]. This is also true for $TCTL_{\mathrm{s}}$ and $TCTL_{\mathrm{c}}$ with the same arguments:

**Theorem 4.7.** *The symbolic model checking problem for $TCTL_{\mathrm{s}}$ and $TCTL_{\mathrm{c}}$ is PSPACE-complete.*

## 5 Remarks about the discrete timed structures

In section 2 we mentioned the *discrete timed structures* (i.e. KS's where transitions $(q, q') \in R$ carry a natural number $d$, the time it takes to fire the transition, written $q \xrightarrow{d} q'$) as a possible model to describe real-time systems. This observation was also done in [EMSS92] where it is explained that model checking algorithms can be easily adapted to this case. Here we want to point out two difficulties related to this model: one in semantics, and one in complexity.

**Semantics.** The natural extension of our semantics for quantitative temporal logics over discrete timed structures (DTS) simply redefines $\mathsf{Time}(\pi_{|i})$ for an execution $\pi$ and a position $i$ as $\sum_{j=0}^{i-1} d_j$, where $d_j$ is the duration of the transition $\pi(j) \xrightarrow{d_j} \pi(j+1)$.

One first problem with this extension is that $q \xrightarrow{d} q'$ and $0 < d' < d$ does not imply the existence of an intermediary state $q''$ s.t. $q \xrightarrow{d'} q''$ and $q'' \xrightarrow{d-d'} q'$. Therefore, some natural equivalences between temporal logics formulae do not hold any more in DTS. For example, $\mathsf{E}\varphi\mathsf{U}_{<k+1}\,\psi$ and $\mathsf{E}\varphi\mathsf{U}_{\leq 1}\,(\mathsf{E}\varphi\mathsf{U}_{<k}\,\psi)$ are not equivalent.

**Complexity.** Using DTS entails a complexity blow-up for model checking $TCTL_{\mathrm{s}}$ (this is not connected to the above-mentioned semantical difficulties). Indeed we have:

**Proposition 5.1.** *Model checking $\mathsf{EF}_{=k}\,A$ over DTS is NP-hard.*

*Proof.* We consider the NP-complete SUBSET-SUM problem [GJ79, p. 223]. An instance is a finite set $A = \{a_1, \dots, a_n\}$ of natural numbers, together with some integer $B$. The problem consists in deciding whether there exists a subset $A'$ of $A$ s.t. $B = \sum_{a \in A'} a$. We reduce this to a model checking problem for $\mathsf{EF}_{=B}\,C$ over a DTS $S$ defined in Figure 7.

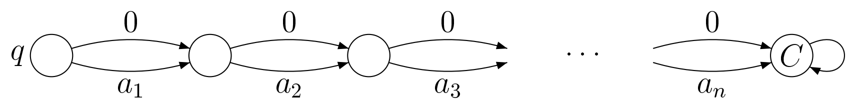


Fig. 7. DTS associated with an instance of SUBSET-SUM.

□

## 6 Conclusion

We investigated the expressive power and the complexity of model checking for $TCTL_{\mathrm{s}}$ and $TCTL_{\mathrm{c}}$, two quantitative extensions of $CTL$ along the lines of $RTCTL$ [EMSS91,ET99].

Our results show that allowing more and more general subscripts in $TCTL_{\mathrm{s}}$ really increase the expressive power. Since this happens in a framework where, strictly speaking, everything can be translated into $CTL$, our results have to be stated in terms of succinctness.

We showed that $TCTL_{\mathrm{s}}$, while more succinct than $RTCTL$, still allows an efficient model checking algorithm. By contrast model checking $TCTL_{\mathrm{c}}$, the extension of $CTL$ with freeze quantifiers, faces a complexity blow-up.

## References


[ACD90] R. Alur, C. Courcoubetis, and D. Dill. Model-checking for real-time systems. In *Proc. 5th IEEE Symp. Logic in Computer Science (LICS'90), Philadelphia, PA, USA, June 1990*, pages 414–425, 1990.

[ACD93] R. Alur, C. Courcoubetis, and D. Dill. Model-checking in dense real-time. *Information and Computation*, 104(1):2–34, 1993.

[AH92] R. Alur and T. A. Henzinger. Logics and models of real time: A survey. In *Real-Time: Theory in Practice, Proc. REX Workshop, Mook, NL, June 1991*, volume 600 of *Lecture Notes in Computer Science*, pages 74–106. Springer, 1992.

[AH94] R. Alur and T. A. Henzinger. A really temporal logic. *Journal of the ACM*, 41(1):181–203, 1994.

[AL99] L. Aceto and F. Laroussinie. Is your model checker on time? In *Proc. 24th Int. Symp. Math. Found. Comp. Sci. (MFCS'99), Szklarska Poreba, Poland, Sep. 1999*, volume 1672 of *Lecture Notes in Computer Science*, pages 125–136. Springer, 1999.

[BCG88] M. C. Browne, E. M. Clarke, and O. Grumberg. Characterizing finite Kripke structures in propositional temporal logic. *Theoretical Computer Science*, 59(1–2):115–131, 1988.

[BCM+92] J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and L. J. Hwang. Symbolic model checking: $10^{20}$ states and beyond. *Information and Computation*, 98(2):142–170, 1992.

[CGP99] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, 1999.

[DS98] S. Demri and Ph. Schnoebelen. The complexity of propositional linear temporal logics in simple cases (extended abstract). In *Proc. 15th Ann. Symp. Theoretical Aspects of Computer Science (STACS'98), Paris, France, Feb. 1998*, volume 1373 of *Lecture Notes in Computer Science*, pages 61–72. Springer, 1998.

[Eme90] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, vol. B*, chapter 16, pages 995–1072. Elsevier Science, 1990.

[EMSS91] E. A. Emerson, A. K. Mok, A. P. Sistla, and J. Srinivasan. Quantitative temporal reasoning. In *Proc. 2nd Int. Workshop Computer-Aided Verification (CAV'90), New Brunswick, NJ, USA, June 1990*, volume 531 of *Lecture Notes in Computer Science*, pages 136–145. Springer, 1991.

[EMSS92] E. A. Emerson, A. K. Mok, A. P. Sistla, and J. Srinivasan. Quantitative temporal reasoning. *Real-Time Systems*, 4(4):331–352, 1992.

[Ers58] A. P. Ershov. On programming of arithmetic operations. *Communications of the ACM*, 1(8):3–6, 1958.

[ET97] E. A. Emerson and R. J. Trefler. Generalized quantitative temporal reasoning: An automata-theoretic approach. In *Proc. 7th Int. Joint Conf. Theory and Practice of Software Development (TAPSOFT'97), Lille, France, Apr. 1997*, volume 1214 of *Lecture Notes in Computer Science*, pages 189–200. Springer, 1997.

[ET99] E. A. Emerson and R. J. Trefler. Parametric quantitative temporal reasoning. In *Proc. 14th IEEE Symp. Logic in Computer Science (LICS'99), Trento, Italy, July 1999*, pages 336–343, 1999.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. Freeman, 1979.

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

[KVW00] O. Kupferman, M. Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. *Journal of the ACM*, 47(2):312–360, 2000.

[Lam83] L. Lamport. What good is temporal logic? In *Information Processing'83. Proc. IFIP 9th World Computer Congress, Sep. 1983, Paris, France*, pages 657–668. North-Holland, 1983.

[Lew90] H. R. Lewis. A logic of concrete time intervals (extended abstract). In *Proc. 5th IEEE Symp. Logic in Computer Science (LICS'90), Philadelphia, PA, USA, June 1990*, pages 380–389, 1990.

[Pap94] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[VW86] M. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proc. 1st IEEE Symp. Logic in Computer Science (LICS'86), Cambridge, MA, USA, June 1986*, pages 332–344, 1986.

[Wil99] T. Wilke. $\mathrm{CTL}^+$ is exponentially more succint than CTL. In *Proc. 19th Conf. Found. of Software Technology and Theor. Comp. Sci. (FST&TCS'99), Chennai, India, Dec. 1999*, volume 1738 of *Lecture Notes in Computer Science*, pages 110–121. Springer, 1999.

[YMW97] J. Yang, A. K. Mok, and F. Wang. Symbolic model checking for event-driven real-time systems. *ACM Transactions on Programming Languages and Systems*, 19(2):386–412, 1997.