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

F. Laroussinie, Ph. Schnoebelen, and M. Turuani

Lab. Spécification & Vérification
ENS de Cachan & CNRS UMR 8643
61, av. Pdt. Wilson, 94235 Cachan Cedex France
email: {fl,phs,turuani}@lsv.ens-cachan.fr

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 introducing freeze quantifiers leads to a complexity blow-up.

1 Introduction

Temporal logic is widely used as a formal language for specifying the behaviour of reactive systems (see [Eme90]). This approach allows *model checking*, i.e. the automatic verification that a finite state system satisfies its expected behavourial specifications. 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.* 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. [EMSS91] showed that *RTCTL* (i.e. *CTL* plus bounded modalities "$\mathsf{A}\_\mathsf{U}_{\leq k}\_$" and "$\mathsf{E}\_\mathsf{U}_{\leq k}\_$" in the Kripke structure framework) still enjoys the bilinear model checking time complexity of *CTL*.

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

The first extension, called $TCTL_s$, s for "subscripts", is basically the most general logic along the lines of the *RTCTL* proposal : it allows combining "$\leq k$", "$\geq k$" and "$= k$" (so that modalities counting w.r.t. intervals are possible). We show this brings real improvements in expressive power, and model checking is still in polynomial time. This extends results for *RTCTL* beyond the increased expressivity: we use a finer measure for size of formula ($\mathsf{EF}_{=k}$ has size in $O(\log k)$ and not k) and do not require that one step uses one unit of time.

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

The results on expressive power formalize natural intuitions which (as far as we know) have never been proven formally, even in the dense time framework [1]. Furthermore, in our discrete time framework our results on expressive power must be stated in terms of how succinctly can one logic express this or that property. Such proofs are scarce in the literature (one example is [Wil99]).

Related work. $TCTL_s$ and $TCTL_c$ are similar to (and inspired from) logics used in dense real-time frameworks (though, in the discrete framework we use here, their behaviour is quite different). Our results on complexity of model checking build on ideas from [DS98,KVW98,AL99,ET99].

Other branching-time extensions of *RTCTL* have been considered. Counting with regular patterns makes model checking intractable [ET97]. Merging different time scales makes model checking NP-complete [ET99]. Allowing parameters makes model checking exponential in the number of parameters [ET99].

Another extension with freeze variables can be found in [YMW97] where richer constraints on number of occurrences of events can be stated (rending satisfiability undecidable). On the other hand, 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]). Complete proofs appear in a full version of the paper, available from the authors.

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,

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

Definition 2.1. *A Kripke structure (a "KS") is a tuple $S = \langle Q_S, R_S, l_S\rangle$ where $Q_S = \{q_1, \ldots\}$ 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 notations whenever no ambiguity will arise. A *computation* in a KS is an infinite sequence π of the form $q_0 q_1 \ldots$ 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: $q_0 q_1, \ldots, q_i$). 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 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 [2]. In pictures, we use different grey levels to distinguish *tick* states from non-*tick* ones.

Given a computation $\pi = q_0 q_1 \ldots$ 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 π.

2.1 $TCTL_{\mathrm{s}}$

Syntax. $TCTL_{\mathrm{s}}$ formulae are given by the following grammar:

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

where I can be any finite union $[a_1, b_1[\cup \cdots \cup [a_n, b_n[$ 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 $\top, \bot, \varphi \vee \psi, \varphi \Rightarrow \psi, \ldots$ 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 U corresponds to $\mathsf{U}_{<\omega}$.

Semantics. Figure 1 defines when a state q in some KS S, satisfies a $TCTL_{\mathrm{s}}$ formula φ, written $q \models \varphi$, by induction over the structure of φ.

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 $TCTL_{\mathrm{s}}[<]$ (with the proviso that our KS's have *tick*'s).

2.2 $TCTL_{\mathrm{c}}$

$TCTL_{\mathrm{c}}$ 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.

[2] Thus KS's with *tick*'s can be seen as *discrete timed structures*, i.e. KS's where edges $(q, q') \in R$ are labeled by a natural number: the time it takes to follow the edge. While discrete timed structures are more natural, KS's with *tick* are an essentially equivalent framework where technicalities are simpler since they do not need labels on the edges.

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

Fig. 1. Semantics of $TCTL_s$

Syntax. For a set $Cl = \{x, y, \ldots\}$ of *clocks*, $TCTL_c$ formulae are given by the following grammar:

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

where $\sim \in \{=, \leq, <, \geq, >\}$ and $k \in \mathbb{N}$. Constraints referring to clocks are restricted to the simple form $x \sim k$, in the spirit of $TCTL_s$.

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.

Semantics. $TCTL_c$ formulae are interpreted over a state of a KS S together with a *valuation* $v : Cl \to \mathbb{N}$ of the clocks free in φ.

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

Fig. 2. Semantics of $TCTL_c$

Figure 2 defines when $q, v \models \varphi$ in some KS S by induction over the structure of φ. 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.

Clearly the $TCTL_s$ operators can be defined with $TCTL_c$ operators:

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

where, for I of the form $[a_1, b_1[\cup \cdots \cup [a_n, b_n[$, $I(x)$ denotes the *clocks constraint* $\bigvee_{i=1}^{n} \Big((a_i \leq x) \wedge (x < b_i)\Big)$. Hence $TCTL_s$ can be seen as a fragment of $TCTL_c$ where only one formula clock is allowed (and used in restricted ways).

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_φ denote the largest constant appearing in φ (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 2.2. *If $v \equiv_m v'$ and $m \geq M_\varphi$, then $q, v \models \varphi$ iff $q, v' \models \varphi$.*

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

Remark 2.3. A related property is used by Emerson et al. in their study of $RTCTL$: when checking whether $q \models \varphi$ inside some KS with $|Q| = m$ states, it is possible to replace by m any constant k larger than m in the subscripts of φ. We emphasize that this property does not hold for $TCTL_s[=]$ (it does hold for $TCTL_s[<,>]$). $\square$

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

3 Expressivity

Formally, $TCTL_s$ or $TCTL_c$ do not add expressive power to CTL:

Theorem 3.1. *Any closed $TCTL_c$ (or $TCTL_s$) formula is equivalent to a CTL formula.*

Proof. With any $TCTL_c$ formula φ, 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*. Then, if φ has no free clock variables, any $(\varphi)^v$ is a CTL equivalent to φ. The definition of $(\varphi)^v$ is given by the following rewrite rules:

$$(\varphi \wedge \psi)^v \stackrel{\text{def}}{=} \varphi^v \wedge \psi^v \qquad (x \sim k)^v \stackrel{\text{def}}{=} \begin{cases} \top & \text{if } v(x) \sim k, \\ \bot & \text{otherwise} \end{cases}$$

$$(\neg\varphi)^v \stackrel{\text{def}}{=} \neg\varphi^v$$

$$(A)^v \stackrel{\text{def}}{=} A \qquad (x \ \underline{\mathrm{in}}\ \varphi)^v \stackrel{\text{def}}{=} \varphi^{v[x \leftarrow 0]}$$

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

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

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

This gives a well-founded definition for $(\_)^v$ since in the right-hand sides either $(\_)^v$ is recursively applied over 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. Then it is a routine matter to check that the correctness invariant (i.e., "$q, v \models \varphi$ iff $q \models (\varphi)^v$") is preserved by these rules. $\square$

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 $TCTL_{\mathrm{s}}[<]$ to $TCTL_{\mathrm{s}}$ to ... allows writing new formulae that have no succinct equivalent at the previous level.

Theorem 3.2. *1. $TCTL_{\mathrm{s}}[<]$ can be exponentially more succinct than CTL,*
2. $TCTL_{\mathrm{s}}[<,>]$ can be exponentially more succinct than $TCTL_{\mathrm{s}}[<]$.

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. One easily shows (by structural induction over φ) that for any CTL formula φ, $ht(\varphi) \leq i$ implies $\alpha_i \models \varphi$ iff $\alpha_{i+1} \models \varphi$. On the other hand, $\alpha_j \models \mathsf{EF}_{<n}A$ iff $j < n$. Thus any CTL equivalent to $\mathsf{EF}_{<n}A$

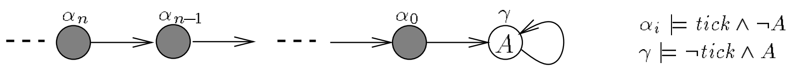

Fig. 3. $\alpha_n \models \mathsf{EF}_{<n+1}\,A$ and $\alpha_{n+1} \not\models \mathsf{EF}_{<n+1}\,A$

must have temporal height larger than 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. One easily shows (by structural induction over φ) that for any formula φ 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 $\beta_{i+1} \models \varphi$. On the other hand, $\alpha_j \models \mathsf{EF}_{>n}\,A$ iff $j > n$. Thus any $TCTL_{\mathrm{s}}[<]$ equivalent to $\mathsf{EF}_{>n}\,A$ must have temporal height larger than n. $\square$

Let us mention two (natural) conjectures that would allow separating further fragments:

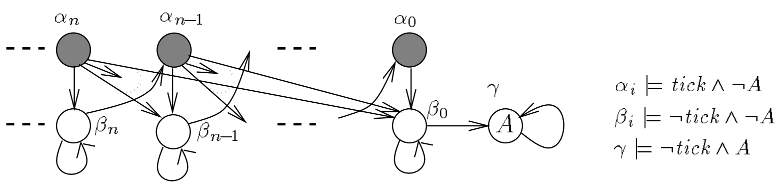

Fig. 4. $\alpha_n \not\models \mathsf{EF}_{>n}\, A$ and $\alpha_{n+1} \models \mathsf{EF}_{>n}\, A$

Conjecture 3.5. 1. $TCTL_{\mathrm{s}}[<,>,=]$ can be exponentially more succinct than $TCTL_{\mathrm{s}}[<,>]$,
2. $TCTL_{\mathrm{c}}$ can be exponentially more succinct than $TCTL_{\mathrm{s}}$.

We have not yet been able to find the required proofs, which are hard to build. The first point is based on the conjecture that any $TCTL_{\mathrm{s}}[<,>]$ formula equivalent to $\mathsf{EF}_{=k}\, A$ has temporal height at least k. For the second one, we conjecture that any $TCTL_{\mathrm{s}}$ formula equivalent to $x \ \underline{\texttt{in}}\ \mathsf{EF}\big(A \wedge \mathsf{EF}(B \wedge x = k)\big)$ has size at least k.

We have explained how $TCTL_{\mathrm{s}}$ becomes more and more expressive when we allow subscripts with $<$, then also with $>$, then also with $=$. Subscripts of the form "$= k$" are the main difference between *RTCTL* and our proposal. They enhance expressivity and make model checking more complex (see § 4).

Once we have $TCTL_{\mathrm{s}}[<,>,=]$, subscripts with intervals are just a convenient shorthand:

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

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

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

$$\mathsf{A}\,\alpha\,\mathsf{U}_I\,\beta \equiv \begin{cases} \mathsf{A}\,\alpha\,\mathsf{U}_{=a_1}\,(\mathsf{A}\,\alpha\,\mathsf{U}_{I-a_1}\,\beta) & \text{if } a_1 > 0,\\[2ex] \quad \neg\mathsf{E}(\neg\beta)\mathsf{U}_{<b_1}\,(\neg\,\alpha\ \wedge\neg\beta) & \\ \wedge\,\neg\mathsf{E}(\neg\beta)\mathsf{U}_{=b_1}\,\Big(\neg\mathsf{A}\,\alpha\,\mathsf{U}_{=a_2-b_1}\,(\mathsf{A}\,\alpha\,\mathsf{U}_{I-a_2}\,\beta)\Big) & \text{otherwise} \end{cases}$$

Correctness is easy to check. The size of $\tilde{\varphi}$, seen as a dag, is linear in the size of φ seen as a dag [4]. □

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 φ a temporal formula. In

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

this section we analyse the complexity of model checking problems for $TCTL_{\rm s}$ and $TCTL_{\rm c}$.

Given a KS S and a formula φ, the complexity of model checking can be evaluated in term of $|S|$ and $|\varphi|$. But more discriminating information can be obtained by also looking at the *program complexity* of model checking (i.e., the complexity when φ is fixed and S, q is the only input) and the *formula complexity* (i.e., when S, q is fixed and φ is the only input).

While $TCTL_{\rm s}$ model checking can be done efficiently, this is not true for $TCTL_{\rm c}$ (even when considering a fixed KS).

Theorem 4.1. *Let $S = \langle Q, R, l\rangle$ be a KS and φ a $TCTL_{\rm s}$ formula. There exists a model checking algorithm running in time $O\big((|Q|^3 + |R|)\times|\varphi|\big)$. Moreover if φ belongs to $TCTL_{\rm s}[<,>]$, the algorithm runs in time $O\big((|Q| + |R|)\times|\varphi|\big)$.*

Proof (Idea). The algorithm extends the classical algorithms for CTL and $RTCTL$ (see [EMSS91]) with procedures dealing with $TCTL_{\rm s}[<,=,>]$ operators (as seen in Theorem 3.6, formulae with interval subscripts can be decomposed). The most expensive procedure concerns the $\mathsf{EU}_=$ case where we compute the transitive closures of relations, hence the (quite naive) $O(|Q|^3 + |R|)$. The $TCTL_{\rm s}[<,>]$ fragment uses only procedures in $O\big((|Q| + |R|)\times|\varphi|\big)$. □

Theorem 4.2. *The model checking problem for $TCTL_{\rm c}$ is PSPACE-complete. The formula complexity of $TCTL_{\rm c}$ model checking is PSPACE-complete.*

Proof. To prove this result, it is sufficient to show that $TCTL_{\rm c}$ model checking is in PSPACE [5] and that the formula complexity is PSPACE-hard. The proof of this last point relies on ideas from [AL99]: let P be an instance of QBF (Quantified Boolean Formula, a PSPACE-complete problem). W.l.o.g. P is some $Q_1 p_1 \ldots Q_n p_n . \varphi$ (with $Q_i \in \{\exists, \forall\}$ and φ a propositional formula over $p_1, \ldots, p_n$). We reduce P to a model checking problem $S, q \models \Phi$ where S is the simple KS $(\{q\}, \{q \to q\}, \{l(q) = tick\})$ and Φ is the following $TCTL_{\rm c}$ formula:

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

where O_i is $\mathsf{EF}_{\leq 1}$ (resp. $\mathsf{EG}_{\leq 1}$) if Q_i is $\exists$ (resp. $\forall$) and $\tilde{\varphi}$ is φ where occurrences of p_i have been replaced by $x_i = n+1-i$. Observe that any clock x_i is reset at time i or $i+1$ and depending on this reset time the atomic propositions p_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. Clearly Φ is valid iff $S, q \models \Phi$. □

In practice, one can easily use any CTL model checker for model checking $TCTL_{\rm c}$ formulae, and the resulting algorithm runs in time $O(|S| . M^{|Cl|} . |\varphi|)$. For

[5] This uses standard arguments, see the long version for details.

example, with SMV, one just adds one variable for each formula clock and update them in the obvious way. This is much more practical than an approach based on Theorem 3.1 and the complexity is not too frightening for formulae with $|Cl| = 1$ (only one clock), a fragment already more expressive than $TCTL_s$.

A theoretical view. The following table gives a synthetic summary of complexity measures for model checking CTL, $TCTL_s$ and $TCTL_c$, showing that model checking the full $TCTL_s$ is as tractable as model checking CTL in both arguments. On the other hand, model checking $TCTL_c$ requires polynomial space even for a fixed Kripke structure.

	CTL	$TCTL_s$	$TCTL_c$
Complexity of model checking	P-complete		PSPACE-complete
Formula complexity	LOGSPACE		PSPACE-complete
Program complexity	NLOGSPACE-complete		

Filling the table. Model checking $TCTL_s$ is in P as we just saw. P-hardness results from the obvious reading of the circuit-value problem (with proper alternation) as a model checking problem for the EX fragment of CTL. The formula complexity of model checking CTL is LOGSPACE and this result can be easily extended to $TCTL_s$. The program complexity of model checking $TCTL_s$ and $TCTL_c$ is NLOGSPACE-complete since we proved (Theorem 3.1) that these logics can be translated into CTL, for which the NLOGSPACE-complete complexity is given in [KVW98].

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 [KVW98], this is also true for $TCTL_s$ and $TCTL_c$:

Theorem 4.3. *The symbolic model checking problem for $TCTL_s$ and $TCTL_c$ is PSPACE-complete.*

5 Conclusion

We investigated the expressive power and the complexity of model checking for $TCTL_s$ and $TCTL_c$, two quantitative extensions of CTL along the lines of $RTCTL$ [EMSS91,ET99].

The expressive power must be measured in a framework where, strictly speaking, everything can be translated into CTL.

We showed that $TCTL_s$, while more succinct than $RTCTL$, still allows an efficient model checking algorithm. By contrast $TCTL_c$, the extension of CTL with freeze quantifiers leads to 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.

[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.

[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.

[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.

[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.

[KVW98] O. Kupferman, M. Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking, 1998. Full version of the CAV'94 paper, accepted for publication in J. ACM.

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

[Wil99] T. Wilke. 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*. 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.