Efficient model-checking of dense-time systems with time-convexity analysis

Farn Wang

PII: S0304-3975(12)00869-9
DOI: 10.1016/j.tcs.2012.09.019
Reference: TCS 9050

To appear in: *Theoretical Computer Science*

Received date: 17 December 2010
Revised date: 30 June 2012
Accepted date: 17 September 2012

Efficient Model-Checking of Dense-Time Systems with Time-Convexity Analysis [*]

Farn Wang
Dept. of Electrical Engineering & Graduate Institute of Electronic Engineering
National Taiwan University
Research Center for Information Technology Innovation, Academia Sinica
farn@cc.ee.ntu.edu.tw; http://cc.ee.ntu.edu.tw/~farn

RED 8 is available at http://sourceforge.net/projects/redlib

June 30, 2012

Abstract

To overcome the sky-rocketing verification cost of embedded software, symbolic model-checking technology of dense-time automata has been proposed as an automated solution. The construction of timed precondition is a central component in the technology. The general formulation for *timed precondition operator* needs to check the continuity of time progress and usually results in high complexity in the construction. However, when the state space characterized by the path condition is convex, we can use a more efficient *convex timed precondition operator*. In this work, we discuss the concept of *time-convexity* that allows us to relax the restrictions on the application of the convex timed precondition operator in place of the general one. We present examples in model-checking that engenders non-time-convex space of time progress. Nevertheless, we have also identified a class of *Timed Computation Tree Logic* (*TCTL*) formulas that only characterize time-convex state spaces. This class includes several important types of TCTL formulas, including some timed inevitabilities with deadlines. We then present a new formulation for the efficient evaluation of general timed inevitabilities with non-time-convex path conditions. The new formulation also leads to a new technique for the approximate evaluation of timed inevitabilities with better precision. Finally, we report our implementation and experiments.

Keywords: timed automaton, timed precondition, model-checking, TCTL, verification, timed inevitability, convex, time-convex

1 Introduction

As the verification cost of embedded software now sky-rocketing to more than half of the development budget [23], software engineers and project managers are now in need of automated support in verifying their software. A popular framework for the automated verification of embedded systems [4, 8, 11, 12, 16, 17, 21, 22, 24, 26, 27, 29, 31–33, 39, 42, 43] is the symbolic model-checking of *timed computation-tree logic* (*TCTL*) [4, 24]. In this framework, we are given a *timed automaton* (*TA*) [5] as a dense-time system description and a TCTL formula [4]

[*]The work is partially supported by NSC, Taiwan, ROC under grants NSC 95-2221-E-002-067 and NSC 95-2221-E-002-072. Part of the materials appeared in proceedings of International Symposium of Automated Technology on Verification and Analysis (ATVA) 2008, LNCS 5311, Springer-Verlag and proceedings of IEEE Real-Time System Symposium (RTSS) 2008, IEEE Computer Society.

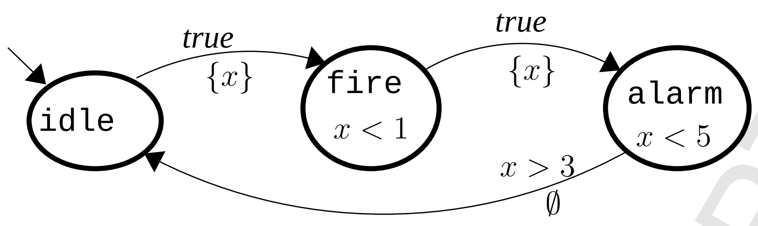


Figure 1: An example TA for a fire alarm

as the specification and we want to answer whether the TA satisfies the TCTL formula. A TA is a finite-state automaton extended with dense-value clock variables and atomic propositions. The semantics of TA is defined with the interleaved execution of discrete transitions and continuous time transitions. In a discrete transition, propositions may change values and clock variables can be reset. In continuous time transitions, proposition values stay unchanged while all clock variable values increase with the same rate.

Example 1 *In* Fig. 1, we present a TA description of a fire alarm system. The ovals represent the control locations while the arcs represent discrete state transitions. The TA is initially in the location labeled "`idle`" from where it may go to the location labeled "`fire`" on the occurrence of a fire. There, it can stay less than 1 time unit before making a transition to the location labeled "`alarm`" where after spending 3 to 5 time units, the TA goes back to the initial location.

In the TA description, there are three atomic propositions respectively for location `idle`, `fire`, and `alarm`. Variable x is a clock with non-negative real values. In a location oval, we may put down a constraint on the clock values. Along a discrete transition arc, we may write the triggering condition and the set of clocks to reset during the transition. Only when its triggering condition is satisfied, a transition may execute. ■

As a specification language, TCTL supports specification of common timing properties for embedded computation, including safety and inevitabilities with response time or deadline constraints.

Example 2 *For* the TA in Example 1, we may want to specify that after a fire is detected, an alarm is signaled in 5 to 10 time units. In TCTL, such a property can be written as "$\forall\Box\left(\texttt{fire} \rightarrow \forall\Diamond_{[5,10]}\texttt{alarm}\right)$." Here $\forall$ is a path quantifier that says "*for all paths*." Modal operator $\Box$ is a state quantifier that says "*along a given path, for all states*." Modal operator $\Diamond$ is the dual of $\Box$ and says "*along a given path, there exists some states*." Together, the formula says that "*for all paths and all states along the paths, if* `fire` *is detected, then along all paths from the fire detection and all states along the paths,* `alarm` *is on in 5 to 10 time units.*"

In the literature, formulas starting with modality $\forall\Box$ are traditionally called *safety* properties while those with $\forall\Diamond$ are called *inevitability* properties. ■

Over the years, symbolic technique [24] has been proven powerful in checking many TCTL properties. The core idea of symbolic TCTL model-checking is characterizing states satisfying a temporal logic formula with a Boolean combination of atomic propositions and difference inequalities involving clock variables. For convenience, such a Boolean combination is called a *state predicate* in this manuscript. Then the evaluation of a TCTL formula is carried out by inductively constructing state predicates characterizing states satisfying the subformulas. To achieve the promise of symbolic TCTL model-checking, the importance of performance enhancement of related algorithms can not be over-emphasized. One important algorithm in this regard is the timed precondition evaluation algorithm. For simplicity, we focus on the backward timed precondition operation. However, the ideas discussed in this work should also apply to the forward counterpart. Usually we are given two state predicates α and β respectively as the *path condition* and the *destination condition* for a continuous time transition. For symbolic model-checking, we then want to construct a state predicate, out of α and β, that characterizes those states that can go to states satisfying β through a continuous time transition in space characterized by α. In [24], Henzinger et al. presented a quantified state predicate template with argument α and β for this purpose. For convenience, we call their template the formulation $\mathsf{T}_{pre}(\alpha, \beta)$. Since the formulation is for general path conditions, $\mathsf{T}_{pre}(\alpha, \beta)$ includes subformulas for checking the continuity of the time transition. Specifically, it needs to check that every state in the continuous time transition satisfies $\alpha \vee \beta$. For convenience, we call $\alpha \vee \beta$ a *time-passage condition*. The following example shows how state space can cause discontinuity in the time transition.

Example 3 *Suppose* that we want to specify that "*the alarm must be on when clock x reads between 5 and 10.*" Following the standard algorithm in model-checking, we evaluate the negation of TCTL formulas. Thus the above specification is first negated and then translated to the following state predicate for states in the continuous time transition: "$x < 5 \vee \neg\mathtt{alarm} \vee x > 10$." As can be seen, alarm states with value of clock x less than 5 can not go to a state with x greater than 10 through the continuous time transition in this state space since when clock x advances to between 5 and 10, the location requirement of "$\neg\mathtt{alarm}$" is violated. ■

This continuity checking component in T_{pre} involves existential quantifications [20], complementation, and conjunctions of state-predicates with dense variables. As a result, the evaluation of T_{pre} is usually expensive. Since the timed precondition algorithm is fundamental to TCTL model-checking, such expensive computation usually results in significant performance degradation.

One way to enhance the evaluation efficiency of formulation T_{pre} is to take the shape of the time-passage condition state space into consideration. Since we are primarily interested at developing efficient techniques for timed precondition evaluation, we shall focus on state spaces in which each proposition variable is of the same value for all states in a state space. For convenience, such a state space S is said *stable* if and only if for any two states $\nu, \nu' \in S$ and proposition p, the values of p at ν and ν' are the same. We propose to take advantage of the convexity of stable spaces of path conditions for efficient timed precondition evaluation. A space is *convex* if for any two points in the space, every *convex combination* of the two points is also in the space. Formally, a convex combination of two points $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ is $(t \cdot x_1 + (1-t) \cdot y_1, \ldots, t \cdot x_n + (1-t) \cdot y_n)$ for some real number $t \in [0, 1]$. For convenience, from now on, a constraint is said to be convex if it characterizes a convex state space.

Example 4 *Time* condition "$x < 5 \vee \neg\texttt{alarm} \vee x > 10$" in Example 3 is non-convex. An evidence for the non-convexity are two states ν and ν' that respectively satisfy "$x = 3 \wedge \texttt{alarm}$" and "$x = 11 \wedge \texttt{alarm}$." Then it is clear that their middle point satisfies "$x = 7 \wedge \texttt{alarm}$" and is not in the space characterized by this time-passage condition. ■

The reason that the convexity of the condition in T_{pre} ensures the continuity of time transitions is as follows. We assume that a starting state ν and a destination state ν' both in a convex and stable space restricted by the time-passage condition. Then all states that happen during the continuous time transition from ν to ν' are all convex combinations of ν and ν'. Intuitively, those states actually form a straight-line segment between ν and ν' in the time-passage state space. According to the definition of convexity, then all states in this straight-line segment (and the continuous time transition) must also be in the same space. This observation suggests that we can skip the expensive continuity checking in evaluating T_{pre} with a convex and stable time-passage condition. We can then design a simplified and sound formulation T^c_{pre} for calculating timed precondition in a convex space by skipping the continuity checking for better efficiency.

In fact, this efficient formulation T^c_{pre} has been used in many verification tools for the safety analysis of embedded and real-time systems [8, 33, 43]. For safety analysis, we are given a TA and a state predicate η and want to check whether all reachable states of the TA satisfy η. In such a framework, the convexity of declared location invariance condition guarantees that all time-passage conditions in timed precondition evaluation are convex. However, it is not clear how we can apply formulation T^c_{pre} beyond the framework of safety analysis. It will be interesting to see to what extent in TCTL model-checking [4, 40], we can use procedure T^c_{pre} in place of T_{pre} for better model-checking performance. Specifically, one important class of TCTL formulas, called *timed inevitabilities*, are of the form $\forall\Diamond_{\langle c,d\rangle}\phi$, where $\langle c, d\rangle$ is an interval of non-negative real numbers, and are important in specifying that some good behavior should happen within a deadline. For example, the TCTL property in Example 2 contains a timed inevitability.

In this work, we have the following contributions.

- We found that the concept of convexity is too narrow for the continuity of time transitions. As long as the discontinuity does not happen along the direction of continuous time transitions, the applicability of T^c_{pre} is not affected. We thus propose the idea of *time-convexity* to relax the applicability of T^c_{pre} to non-convex time-passage conditions. For convenience, from now on, a constraint is said time-convex if it characterizes a time-convex state space.
- We show that the time-convexity of the location invariance conditions of a TA is sufficient in ensuring that all time transition conditions used in the reachability analysis are also time-convex.
- However, the time-convexity of the location invariance conditions does not ensure the continuity property

in time transitions of timed precondition evaluation in the general setting of TCTL model-checking. In addition to the property in Example 2, we present several evidences in TCTL model-checking [4] that entail the computation of timed precondition through *non-time-convex* time-passage conditions even when all the location invariance conditions of the TA are time-convex.

- The evidences in the above motivated us in finding situations in which we can skip the expensive continuity checking in timed precondition evaluation. For this purpose, we identified a class, called TCXTL , of TCTL formulas that only characterize time-convex state spaces. The syntax of TCXTL can be relatively efficiently checked. TCXTL itself may not be general enough for writing full specifications. But its design purpose is to help us in efficiently identifying those subformulas in a full specification that can induce efficient timed precondition evaluations.
- Unfortunately, the time-passage conditions involved in the evaluation of the important class of timed inevitabilities of TCTL does not fall in the scope of TCXTL . For the property in Example 2, discontinuity happens respectively at the boundary between time intervals $[0, 5)$ and $[5, 10]$ and at the boundary between $[5, 10]$ and $(10, \infty)$. We need a different way in handling the continuity checking across the two boundaries. For this purpose, we present lemmas that identify time-convexity across the boundary of two time-convex state spaces. Then based on the lemmas, we propose a new formulation for the evaluation of timed inevitabilities that can avoid the use of T_{pre} for special subclasses of timed inevitabilities. The new formulation involves breaking a time-passage into at most three path segments and allows us to take the time-convexity of each segment into consideration for efficient timed precondition evaluation.
- In examining the new formulation in the above, we found that it also offers an opportunity in approximate evaluation of the preconditions for timed inevitabilities. In [40], an approximation for timed inevitabilities was proposed by omitting the expensive evaluation of non-Zeno requirement of the timed runs. As mentioned in the previous bullet, the new formulation involves breaking a time progress path into three segments with the non-Zeno requirement imposed on the last segment. Thus, we propose to approximate timed inevitabilities, by skipping the non-Zeno requirement on their last segments, and rule out those states that fail to time-progress to their last segments from the approximate evaluation of the timed inevitabilities. Intuitively, this new approximate evaluation adds the condition that there is a timed path to the last segment to the approximate formulation in [40]. As a result, we can prove, in both theory and experiment, that our new approximate formulation can recognize more states that satisfy the timed inevitabilities than the one in [40].
- Finally, we have implemented our techniques and report our experiment with our TA model-checker RED, version 8. The result shows significant performance enhancement against several benchmarks.

We have the following presentation plan. In Section 2, we review related work. Section 3 defines TAs and the TCTL model-checking problem and reviews a symbolic algorithm for the TCTL model-checking problem.

Then in Section 4, we introduce the concept of time-convexity for an efficient evaluation of timed precondition. Section 5 investigates some possibilities of non-time-convex time-passage conditions in reachability analysis and model-checking. We then use Section 5 as motivation for proposing TCXTL in Section 6. However, TCXTL does not express the time-passage condition of an important class of formulas: the timed inevitabilities. Thus in Section 7, we present new formulations for efficiently evaluating timed inevitabilities. Section 8 reports our implementation and experiments. Section 9 concludes the paper by discussing the potentials of our research direction.

2 Related work

In [24], Henzinger et al. presented a fully symbolic algorithm for TCTL model-checking. All steps in the algorithm are for the construction of state predicates characterizing spaces of states satisfying certain properties, including subformulas of a given TCTL property. They also proposed formulation T_{pre} for the calculation of timed precondition for TAs through non-convex path conditions. Various tools for reachability analysis are now available with formulation T^{c}_{pre} based on the convexity assumption of the location invariance condition of TAs [8, 33, 43].

UPPAAL [8] and Kronos [43] are two popular model-checkers for TAs. However, they used matrices to represent convex state spaces called *zones*. In their implementation, a convex and stable state space is represented as a pair of a location name and a matrix. Thus a non-convex state space is represented as a list of such pairs. Thus, timed precondition evaluation can be performed inside a zone or between zones. When timed precondition evaluation is carried out along non-convex path state space, then pairwise manipulations on the zones in the lists are necessary for the quantifications, complementations, and conjunctions.

UPPAAL [8] mainly supports safety analysis and a restricted class of inevitability analysis. With this restriction, usually the convexity of location invariance condition can guarantee the convexity of all path conditions.

In [33], Wang used a data-structure, Clock-Restriction Diagrams (CRD), as unified representation of both convex and non-convex state spaces. The data-structure has been implemented with TCTL model-checker RED [37, 38, 40, 41]. In earlier version of RED, for timed precondition, it used T^{c}_{pre} in safety analysis when the location invariance condition is convex; and T_{pre} otherwise.

In [40], Wang et al. presented a performance-enhancing technique for timed inevitabilities. The authors also presented an early decision technique for the evaluation of inevitabilities. The basic idea is that in evaluating a timed inevitability, we usually evaluate its negation instead with a greatest fixpoint algorithm. Since the greatest fixpoint image shrinks in the iteration of the fixpoint algorithm, if we find that the intersection of the initial condition and the fixpoint image is empty, then we can stop the fixpoint algorithm without having to finish the fixpoint calculation.

In [34], Wang presented a model-checking algorithm for timed inevitabilities with event constraints and weak/strong fairness assumptions. The logic considered in [34] subsumes *Fair TCTL* [4]. Formulas in Fair TCTL are evaluated with time-divergent paths along which certain fairness conditions must be satisfied infinitely many times.

The analysis of space convexity has also been used in other directions related to verification. Especially, Bouyer et al. [13] and Jurdzinski et al. [25] also employed the convexity of state spaces in the context of optimization.

In [19], Engdahl et al. independently developed the concept of time convexity for the minimisation algorithm of zone graphs in their master thesis [10]. They proved that time convexity is closed under intersection and timed preconditions.

3 TA and TCTL model-checking

Let $\mathbb{N}$ be the set of non-negative integers, $\mathbb{Z}$ the set of all integers, and $\mathbb{R}^{\geq 0}$ the set of non-negative reals. Given two sets X and Y, $X \subsetneq Y$ means $X \subset Y$ and $X \neq Y$. For convenience, we may write a constraint like $c \leq \theta \leq d$ as a shorthand for $c \leq \theta \wedge \theta \leq d$. Also '*iff*' means "*if and only if.*"

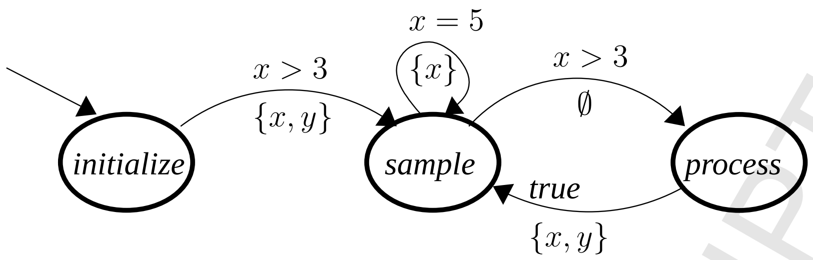
3.1 Timed automata

Given a set Q of atomic propositions and a set X of clocks, a *location predicate* is a Boolean combination of atoms of the forms q and $x \sim c$, where $q \in Q$, $x \in X$, '$\sim$' is one of $\leq, <, =, >, \geq$, and $c \in \mathbb{N}$. The set of all location predicates of Q and X is denoted as $\mathcal{LP}(Q, X)$.

Definition 5 **Timed automaton (TA)** A *TA* is a tuple $\langle \mathcal{Q}, \mathcal{X}, \mathcal{I}, \mathcal{H}, E, \sigma, \delta, \tau, \pi \rangle$ with the following restrictions. $\mathcal{Q}$ is a finite set of control locations. $\mathcal{X}$ is a finite set of clocks. $\mathcal{I} \in \mathcal{LP}(\mathcal{Q}, \mathcal{X})$ is the initial condition. $\mathcal{H} \in \mathcal{LP}(\mathcal{Q}, \mathcal{X})$ is the (location) invariance condition. E is a finite set of directed edges representing the location transitions. Function $\sigma : E \mapsto \mathcal{Q}$ and $\delta : E \mapsto \mathcal{Q}$ respectively specify the source and the destination locations of each transition. Function $\tau : E \mapsto \mathcal{LP}(\emptyset, \mathcal{X})$ defines the triggering condition of each rule execution. For each $e \in E$, $\pi(e) \subseteq \mathcal{X}$ specifies the set of clocks to reset during the transition. ■

For convenience, from now on, without explicit explanation, we assume that we are in the context of a given TA $\mathcal{A} = \langle \mathcal{Q}, \mathcal{X}, \mathcal{I}, \mathcal{H}, E, \sigma, \delta, \tau, \pi \rangle$. For example, when we put down $\mathcal{Q}$ and $\mathcal{X}$, we refer to the $\mathcal{Q}$ and $\mathcal{X}$ components of this given $\mathcal{A}$. Without loss of generality, we require that for each control locaiton $q \in \mathcal{Q}$, there is a null transition e in E such that $\sigma(e) = \delta(e) = q$, $\tau(e) =$ *true*, and $\pi(e) = \emptyset$.

Example 6 *We* may have a signal processor that periodically samples signals in every 5 seconds. It occasionally processes the accumulated signal samples. However, if it has not processed any samples in 15 seconds, it must immediately process the accumulated signal samples. The initialization of the processor needs from 3 to 5 seconds. In Fig. 2, we have the transition diagram of a TA $\mathcal{A}$ for this signal processor. The ovals represent



$$\begin{array}{lcl} \mathcal{I} & \equiv & \textit{initialize} \wedge x \leq 5 \\ \mathcal{H} & \equiv & (\textit{initialize} \wedge x \leq 5) \\ & \vee & (\textit{sample} \wedge x \leq 5 \wedge y \leq 15) \\ & \vee & (\textit{process} \wedge (x \leq 5 \vee y \geq 15)) \end{array}.$$

Figure 2: An example TA $\mathcal{A}$ for a signal processor

control locations *initialize*, *sample*, and *process*. The initial location is *initialize*. The arcs represent transitions between locations. On each arc, we label the triggering condition and the clock reset set. We use clock x to control the periodical sampling of signals and clock y to check whether signal processing is overdue. ■

A *valuation* of a set Y *(domain)* is a mapping from Y to a *codomain*.

Definition 7 **States of a TA** A *clock valuation* of a TA $\mathcal{A}$ is a total valuation from $\mathcal{X}$ to $\mathbb{R}^{\geq 0}$.

A *state* of $\mathcal{A}$ is a pair (q, ν) with $q \in \mathcal{Q}$ and ν a clock valuation of $\mathcal{X}$. A state (q, ν) *satisfies* a location predicate η, in symbols $(q, \nu) \models \eta$, if η is evaluated true when q is interpreted true, all other location names are interpreted false, and all clock variables are interpreted according to ν. We require that $(q, \nu) \models \mathcal{H}$ for all states (q, ν).

Let $\mathcal{ST}$ denote the set of states of $\mathcal{A}$. ■

For any clock valuation ν of a TA $\mathcal{A}$ and $t \in \mathbb{R}^{\geq 0}$, $\nu + t$ is a valuation such that for every $x \in \mathcal{X}$, $(\nu + t)(x) = \nu(x) + t$. Given a set $Y \subseteq \mathcal{X}$, we let $\nu[Y := 0]$ be a valuation that is identical to ν except that all variables in Y are mapped to zero.

Given two states $(q, \nu), (q', \nu')$ and a transition $e \in E$, we say $\mathcal{A}$ *transits with* e from (q, ν) to (q', ν'), in symbols $(q, \nu) \xrightarrow{e} (q', \nu')$, if $\sigma(e) = q$, $\delta(e) = q'$, $(q, \nu) \models \tau(e) \wedge \mathcal{H}$, $\nu[\pi(e) := 0] = \nu'$, and $(q', \nu') \models \mathcal{H}$.

Definition 8 **Runs** A *run* is an infinite sequence of state-time pairs $((q_0, \nu_0), t_0)((q_1, \nu_1), t_1) \ldots ((q_k, \nu_k), t_k) \ldots\ldots$ such that for all $k \geq 0$, the following three restrictions hold. (1) $t_k \leq t_{k+1}$. (2) For all $t \in [0, t_{k+1} - t_k]$, $(q_k, \nu_k + t) \models \mathcal{H}$. (3) Either $(q_k, \nu_k + t_{k+1} - t_k) = (q_{k+1}, \nu_{k+1})$ or there is an $e \in E$ such that $(q_k, \nu_k + t_{k+1} - t_k) \xrightarrow{e} (q_{k+1}, \nu_{k+1})$. The run is *initial* if $(q_0, \nu_0) \models \mathcal{I}$. It is *Zeno* if there is a $c \in \mathbb{N}$ with $\forall k \geq 0 (t_k \leq c)$. ■

3.2 TCTL

Given a set $\mathcal{Q}$ of atomic propositions, a set $\mathcal{X}$ of clocks, and a non-negative integer b, a *state-predicate* within bound b is a Boolean combination of atoms of the forms q and $x - y \sim c$, where $q \in \mathcal{Q}$, $x, y \in \mathcal{X} \cup \{0\}$,

'$\sim$'$\in \{<, \leq, =, \neq, \geq, >\}$, and $c \in \mathbb{Z} \cap [-b, b]$. The set of all state-predicates of $\mathcal{Q}$ and $\mathcal{X}$ within bound b is denoted as $\mathcal{SP}_b(\mathcal{Q}, \mathcal{X})$. The satisfaction of zone predicates by a state can be defined similarly as that of location predicates.

An interval is of the following syntax rule: $[c, d] \mid [c, d) \mid (c, d) \mid (c, d] \mid [c, \infty) \mid (c, \infty)$, where $c, d \in \mathbb{N}$ and $c \leq d$. *Timed Computation Tree Logic* (*TCTL*) is a language for the specification of timing behaviors with branching structures [4]. For a TA $\mathcal{A}$, we adopt TCTL formulas, say ϕ, with the following extended syntax.

$$\phi ::= \eta \mid \phi_1 \vee \phi_2 \mid \neg\phi_1 \mid \exists\Box_\Theta\phi_1 \mid \exists\phi_1\mathbf{U}_\Theta\phi_2$$

Here η is a state-predicate in $\mathcal{SP}_\infty(\mathcal{Q}, \mathcal{X})$ and Θ is an interval. Standard shorthands like *true*, *false*, $\phi_1 \wedge \phi_2$, $\phi_1 \rightarrow \phi_2$, $\exists\Diamond_\Theta\phi_1$, $\forall\Box_\Theta\phi_1$, $\forall\Diamond_\Theta\phi_1$, and $\forall\phi_1\mathbf{U}_\Theta\phi_2$ are also adopted. Also interval $[0, \infty)$ can be conveniently omitted. For modal formulas $\exists\Box_\Theta\phi$ and $\exists\phi\mathbf{U}_\Theta\psi$, ϕ is called the *path condition* while ψ is called the *destination condition*. Given a TCTL formula ϕ, we let c_{max} be the biggest timing constant used in $\mathcal{A}$ and ϕ. Note that our TCTL definition is a slightly extended from [4]. First, we allow intervals instead of inequalities as the subscripts to modal operators. Computationally, this does not affect much in the related algorithms in model-checking. Second, we allow inequalities in $\mathcal{SP}_\infty(\mathcal{Q}, \mathcal{X})$ to appear in formulas. The reason is that according to [24], in the evaluation of nested modal formulas, the evaluation of inner modal formulas may yield predicates in $\mathcal{SP}_{c_{max}}(\mathcal{Q}, \mathcal{X})$. Thus, for the investigation of non-convex time-passage conditions in timed precondition calculation, it makes no difference to have state-predicates in TCTL formulas.

In our semantics of TCTL, we interpret TCTL formulas directly on the sequences of state-time pairs for runs to make it easy to check the properties of the source states of discrete transitions. The satisfaction of TCTL formulas is defined as follows.

- Given a state-predicate η, $\mathcal{A}, (q, \nu) \models \eta$ iff $(q, \nu) \models \eta$.
- $\mathcal{A}, (q, \nu) \models \exists\Box_\Theta\phi$ iff there is a non-Zeno run $((q_0, \nu_0), t_0)((q_1, \nu_1), t_1) \ldots ((q_k, \nu_k), t_k) \ldots\ldots$ such that $(q_0, \nu_0) = (q, \nu)$ and for all $k \geq 0$ and $t \in [0, t_{k+1} - t_k]$, $t + t_k - t_0 \in \Theta$ implies $\mathcal{A}, (q_k, \nu_k + t) \models \phi$.
- $\mathcal{A}, (q, \nu) \models \exists\phi\mathbf{U}_\Theta\psi$ iff there is a non-Zeno run $((q_0, \nu_0), t_0)((q_1, \nu_1), t_1) \ldots ((q_k, \nu_k), t_k) \ldots\ldots$ such that $(q_0, \nu_0) = (q, \nu)$ and there exists a $k \geq 0$ and a $t \in [0, t_{k+1} - t_k]$ with the following restrictions.
  - $t + t_k - t_0 \in \Theta$, $\mathcal{A}, (q_k, \nu_k + t) \models \psi$, and for all $t' \in [0, t)$, $\mathcal{A}, (q_k, \nu_k + t') \models \phi$.
  - For all $h \in [0, k)$ and $t' \in [0, t_{h+1} - t_h]$, $\mathcal{A}, (q_h, \nu_h + t') \models \phi$.

The TCTL *model-checking problem* instance of a TA $\mathcal{A}$ and a TCTL formula ϕ asks if all initial states of $\mathcal{A}$ satisfy ϕ. We also use $[\![\phi]\!]$ to denote the set of states $(q, \nu) \in \mathcal{ST}$ with $\mathcal{A}, (q, \nu) \models \phi$. It is clear that for any $(q, \nu) \in \mathcal{ST}$, $\mathcal{A}, (q, \nu) \models \phi$ iff $(q, \nu) \in [\![\phi]\!]$.

3.3 Symbolic TCTL model-checking

We assume that there are two auxiliary clocks z and w not in $\mathcal{X}$. In the following, we use w to measure the fulfillment of timed inevitabilities and z to maintain the quantification of non-Zeno runs. For convenience, for

Θ	$w \in \Theta$	$w \prec \Theta$	$w \succ \Theta$
$[c,d], c \neq 0$	$w \geq c \wedge w \leq d$	$w < c$	$w > d$
$[c,d), c \neq 0$	$w \geq c \wedge w < d$	$w < c$	$w \geq d$
$(c,d]$	$w > c \wedge w \leq d$	$w \leq c$	$w > d$
(c,d)	$w > c \wedge w < d$	$w \leq c$	$w \geq d$
$[0,d]$	$w \leq d$	*false*	$w > d$
$[0,d)$	$w < d$	*false*	$w \geq d$
$[c,\infty), c \neq 0$	$w \geq c$	$w < c$	*false*
(c,∞)	$w > c$	$w \leq c$	*false*

Here c, d are two integers in $\mathbb{N}$.

Table 1: Shorthands for interval constraints

an interval Θ, we adopt the shorthands in Table 1 for convenience. An *extended state-predicate* of a TA $\mathcal{A}$ is a state-predicate extended with clock inequalities with clocks w and z. According to [24, 40], we only need to manipulate extended state-predicates in $\mathcal{SP}_{c_{max}}(\mathcal{Q}, \mathcal{X} \cup \{w, z\})$ in the symbolic model-checking of TAs. Many model-checkers for TAs are based on symbolic manipulation algorithms of *extended state-predicates* represented in various forms [8, 33, 43]. Moreover, in checking a TA with a TCTL formula ϕ with the symbolic algorithms, the evaluation results of all subformulas of ϕ are *state-predicates* without clock variables w and z [24, 40].

For convenience, given a formula ϕ and a set $X = \{x_1, \ldots, x_n\}$ of variables, we use $\exists X(\phi)$ as the shorthand for $\exists x_1 \ldots \exists x_n(\phi)$. Also, given a $q \in \mathcal{Q}$, we let $\texttt{loc}(q)$ be a shorthand of formula $q \wedge \bigwedge_{q' \in \mathcal{Q}-\{q\}} \neg q'$. Intuitively, predicate $\texttt{loc}(q)$ is used to restrict that only the proposition for location q can be true. Then we have the following formulation for the evaluation of precondition of a discrete transition e to a post-condition β.

$$\texttt{Xt}_{pre}(e, \beta) \overset{\text{def}}{=} \left(\begin{array}{ll} & \texttt{loc}(\sigma(e)) \wedge \mathcal{H} \wedge \tau(e) \\ \wedge & \exists(\pi(e) \cup \mathcal{Q}) \left(\beta \wedge \texttt{loc}(\delta(e)) \wedge \mathcal{H} \wedge \bigwedge_{x \in \pi(e)} x = 0 \right) \end{array} \right)$$

The formulation in the above can be understood in the following way. The predicate inside the existential quantification is for the destination state while the one on top of the existential quantification is for the source state. For the destination state, we require that it is in the space of β, satisfies the location invariance at the destination state (i.e., $\texttt{loc}(\delta(e)) \wedge \mathcal{H}$), and has all clocks in $\pi(e)$ reset to zero. The existential quantification is for removing the effects of the clock reset operations from the precondition. Then on top of the existential quantification, we require that the source state must satisfy the location invariance at the source state (i.e., $\texttt{loc}(\sigma(e)) \wedge \mathcal{H}$) and the triggering condition of e (i.e., $\tau(e)$). Formally speaking, $(q, \nu) \models \texttt{Xt}_{pre}(e, \beta)$ iff there exists a $(q', \nu') \models \beta$ with $(q, \nu) \xrightarrow{e} (q', \nu')$.

Following the notations in [24], we let $\alpha + t$ denote the state predicate obtained from α by replacing every $x \in \mathcal{X}$ in α with $x + t$ [4]. In such state predicates, t occurs as a quantified variable. For example, if $p \wedge x < 3$ is a destination condition, then we may want to see whether there exists a time transition distance $t \geq 0$ with $(p \wedge x < 3) + t$ true. In the manipulation, this can be written as $\exists t(t \geq 0 \wedge (p \wedge x < 3) + t) \equiv \exists t(t \geq 0 \wedge p \wedge x + t < 3)$.

formulas	formulations
$\mathsf{T}_{pre}(\alpha, \beta)$	$\exists t \in \mathbb{R}^{\geq 0} \left(\beta + t \wedge \forall t' \in \mathbb{R}^{\geq 0} \left(t' \leq t \rightarrow (\alpha \vee \beta) + t'\right)\right)$ $\equiv \quad \exists t \in \mathbb{R}^{\geq 0} \left(\beta + t \wedge \neg \exists t' \in \mathbb{R}^{\geq 0} \left(t' \leq t \wedge \neg(\alpha \vee \beta) + t'\right)\right)$
$\mathsf{T}^c_{pre}(\alpha, \beta)$	$(\alpha \vee \beta) \wedge \exists t \in \mathbb{R}^{\geq 0} (\beta + t)$

Table 2: Formulations T_{pre} and T^c_{pre} of timed precondition evaluation.

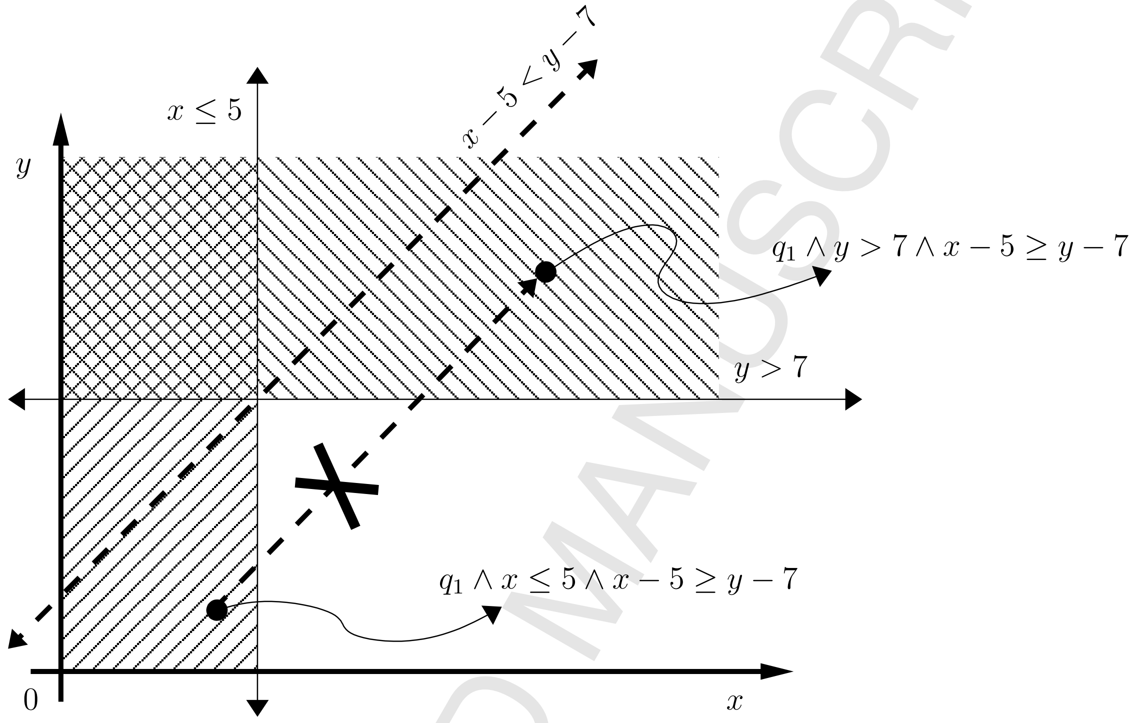

$x - 5 < y - 7$
$x \leq 5$
y
$q_1 \wedge y > 7 \wedge x - 5 \geq y - 7$
$y > 7$
$q_1 \wedge x \leq 5 \wedge x - 5 \geq y - 7$
0
x

Figure 3: The state space of $q_1 \wedge (x \leq 5 \vee y > 7)$

Now we present formulation T_{pre} and T^c_{pre} for the construction of timed precondition in Table 2 according to [24]. For $\mathsf{T}_{pre}(\alpha, \beta)$, the outer quantification on t specifies the "*through a continuous time transition of* t *time units*" part. The inner quantification specifies that every state along the continuous time transition satisfies $\alpha \vee \beta$. For $\mathsf{T}^c_{pre}(\alpha, \beta)$, restrictions $\alpha \vee \beta$ and $\beta + t$ ensure that the source and the destination states respectively are in the convex state space of $\alpha \vee \beta$. As can be seen, formulation T^c_{pre} only uses one existential quantification and avoids the complementation.

Example 9 *Suppose* that we are in a stable state space of two clocks x and y of readings in $\mathbb{R}^{\geq 0}$. For a state with $x = 3$ and $y = 3$, we may use the pair $(x = 3, y = 3)$ to represent the state. We have a path condition $\alpha \equiv x \leq 5 \vee y > 7$. The state space of the path condition when interpreted with a location, say q_1, can be seen in Fig. 3. The condition is non-convex since states $(x = 3, y = 3)$ and $(x = 9, y = 9)$ both satisfy it but their middle point $(x = 6, y = 6)$ does not. Intuitively, there is a gap between $(x = 5, y = 5)$ and $(x = 7, y = 7)$ that can not be stepped into according to α. The time passes by incrementing the values of x and y at the same rate and in the direction of vector $(1, 1)$. Then the union of the two spaces $q_1 \wedge x \leq 5$ and $q_1 \wedge y > 7$ forms a non-

formulas	formulations
q	q
$x - y \sim c$	$x - y \sim c$
$\phi \vee \psi$	$\langle\langle\phi\rangle\rangle \vee \langle\langle\psi\rangle\rangle$
$\neg\phi$	$\neg\langle\langle\phi\rangle\rangle$
$\exists x(\phi)$	$\exists x(\langle\langle\phi\rangle\rangle)$
NZ(ϕ)	$\textbf{gfp}Z.\,(\exists z\,(z = 0 \wedge (\langle\langle\phi\rangle\rangle \rightsquigarrow (Z \wedge z \geq k))))$
$\exists\phi\mathbf{U}_\Theta\psi$	$\exists w\,(w = 0 \wedge (\langle\langle\phi\rangle\rangle \rightsquigarrow (w \in \Theta \wedge \langle\langle\psi\rangle\rangle \wedge \text{NZ}(\textit{true}))))$
$\exists\Box_\Theta\phi$	$\exists w\,(w = 0 \wedge \text{NZ}(w \in \Theta \rightarrow \langle\langle\phi\rangle\rangle))$

Note that the k symbol in the formulation of NZ() is actually a constant parameter chosen by the developers of the model-checker.

Table 3: Traditional formulation for the symbolic evaluation of TCTL formulas

time-convex state space. Time passes between states not in the half plane $x-5 < y-7$ may incur discontinuity. Specifically, the continuous time transitions from a state satisfying $[\![q_1 \wedge x \leq 5 \wedge x-5 > y-7 \wedge \mathcal{H}]\!]$ to a state satisfying $[\![q_1 \wedge y > 7 \wedge x-5 > y-7 \wedge \mathcal{H}]\!]$ have to pass through the space of $x > 5 \wedge y \leq 7$ which is not in the space of the path condition.

With the formulations in Table 2, $\mathsf{T}_{\mathit{pre}}(\alpha, x = 8 \wedge y = 8)$ is $7 < x \leq 8 \wedge 7 < y \leq 8 \wedge x = y$. However, $\mathsf{T}^c_{\mathit{pre}}(\alpha, x = 8 \wedge y = 8)$ is $0 \leq x \leq 8 \wedge 0 \leq y \leq 8 \wedge x = y$ which extends across the gap and is incorrect. ■

Based on this, in the following, we can then present the traditional formulation of backward reachability to destination states satisfying a state-predicate β through a path of states satisfying a state-predicate α.

$$\alpha \rightsquigarrow \beta \stackrel{\text{def}}{=} \textbf{lfp}Z.\left(\mathsf{T}_{\mathit{pre}}(\alpha,\beta) \vee \bigvee_{e\in E} \mathsf{T}_{\mathit{pre}}(\alpha, \mathsf{Xt}_{\mathit{pre}}(e, Z))\right)$$

lfp is the least fixpoint operator. The evaluation of the least fixpoint operator works by initializing Z with $\mathsf{T}_{\mathit{pre}}(\alpha,\beta)$ and then iteratively adding states to Z until no more addition is possible.

Based on the formulation in the above, in Table 3, we present the traditional formulations for the symbolic evaluation algorithm of TCTL formulas from the literature [24, 40]. Given a TCTL formula ϕ, the symbolic algorithm inductively constructs state-predicates that characterize states satisfying subformulas of ϕ. We use $\langle\langle\phi\rangle\rangle$ to denote the state predicate obtained from the algorithm in tables 2 and 3 for a TCTL formula ϕ.

gfp is the greatest fixpoint operator. The evaluation of the greatest fixpoint operator works by first initializing Z to $\mathcal{H}$ and then iteratively eliminating states from Z until no more elimination is possible.

Formulation NZ(ϕ), with $k \geq 1$, characterizes those states that starts a non-Zeno run along which all states satisfy ϕ. Parameter k can be any integer constant no less than one chosen by the developers of the model-checker. For example, if k is chosen to be three, then the greatest fixpoint algorithm for NZ() constructs a characterization for states that start a run segment of > 3 time units. In [40], it was reported that $k = \max(1, c_{\mathit{max}})$ usually yields reasonably good performance.

To check a TA $\mathcal{A}$ against a TCTL formula ϕ, usually we check if $\mathcal{I} \wedge \langle\langle\neg\phi\rangle\rangle$ is satisfiable. $\mathcal{A}$ satisfies ϕ iff

formulas	formulations
$\exists\phi\mathbf{U}_\Theta\psi$	$\exists w\,(w=0\wedge(\langle\!\langle\phi\rangle\!\rangle \rightsquigarrow (w\in\Theta\wedge\langle\!\langle\psi\rangle\!\rangle)))$
$\exists\Box_\Theta\phi$	$\exists w\left(w=0\wedge\mathbf{gfp}Z\left(\mathsf{T}_{pre}\left(w\in\Theta\rightarrow\langle\!\langle\phi\rangle\!\rangle,\bigvee_{e\in E}\mathsf{Xt}_{pre}(e,Z)\right)\right)\right)$

Table 4: Zeno approximate formulation of timed modal formulas

$\mathcal{I}\wedge\langle\!\langle\neg\phi\rangle\!\rangle$ is unsatisfiable. According to [24, 40], we have the following lemma.

Lemma 10 *Given a TCTL formula ϕ and a state $(q,\nu)\in\mathcal{ST}$, $(q,\nu)\models\langle\!\langle\phi\rangle\!\rangle$ iff $\mathcal{A},(q,\nu)\models\phi$.* ■

In the literature [24, 40], inevitabilities of the form $\forall\Diamond_\Theta\phi$ are usually evaluated as the negation of $\exists\Box_\Theta\neg\phi$. It is known that formulation for $\exists\Box_\Theta\phi$ in Table 3 is needed so that the inevitabilities are not refuted by Zeno runs [40]. But for many verification tasks, the formulation NZ(ϕ) nested inside that for $\exists\Box_\Theta\phi$ is expensive to evaluate. It involves a double fixpoint calculation, i.e., a least fixpoint nested inside a greatest fixpoint. According to [40], the formulations in Table 4 can be used as the Zeno approximation of the corresponding formulations in Table 3 if computation resources is limited. The approximation skips the evaluation of NZ(ϕ) and directly uses $\mathcal{H}$ instead. Given a TCTL formula ϕ, we use $\langle\!\langle\phi\rangle\!\rangle^{app}$ to denote the state-predicate obtained for ϕ from the formulation in tables 2 and 3 except that formulas like $\exists\Box_\Theta\phi$ and $\exists\phi\mathbf{U}_\Theta\psi$ are evaluated with the formulations in Table 4. The following lemma shows that this approximation is strictly safe for the evaluation of timed inevitabilities.

Lemma 11 *For all state-predicates η and intervals $\Theta\subseteq\mathbb{R}^{\geq 0}$, $[\![\langle\!\langle\forall\Diamond_\Theta\eta\rangle\!\rangle^{app}]\!]\subseteq[\![\forall\Diamond_\Theta\eta]\!]$. There are also TAs with $[\![\langle\!\langle\forall\Diamond_\Theta\eta\rangle\!\rangle^{app}]\!]\subsetneq[\![\forall\Diamond_\Theta\eta]\!]$.*

Proof : The first sentence is true since the set of runs considered in the approximate formulations in Table 4 includes Zeno runs and those considered in the exact formulations in Table 3 does not.

For the second sentence, we can create a TA with location invariance that implies $x<5$ and without clock reset operations. Then according to the exact formulations in Table 3, $\forall\Diamond$*true* is vacuously satisfied at any state of the TA. However, with formulations in Table 4, the negation of the formula is evaluated true with the Zeno runs. This implies that $[\![\langle\!\langle\forall\Diamond\mathit{true}\rangle\!\rangle^{app}]\!]=\emptyset$. ■

4 Convexity and time-convexity

Suppose that we are given a clock valuation ν of $\mathcal{X}$. For convenience, we write ν as $\{x\mapsto\nu(x)\mid x\in\mathcal{X}\}$.

We define the convexity of a TA state space by checking the clock valuations in the same control locations in the following way. A set $S\subseteq\mathcal{ST}$ of states is *convex* if for any $(q,\nu),(q,\nu')\in S$, any real number $t\in[0,1]$, and any ν'' with $\forall x\in\mathcal{X}(\nu''(x)=t\cdot\nu(x)+(1-t)\cdot\nu'(x))$, (q,ν'') is also in S. The reachable state space of a TA is usually non-convex. Most state-spaces that we need to manipulate in TCTL model-checking are likely non-convex. For convenience, we say a formula ϕ is convex iff $[\![\phi]\!]$ is convex.

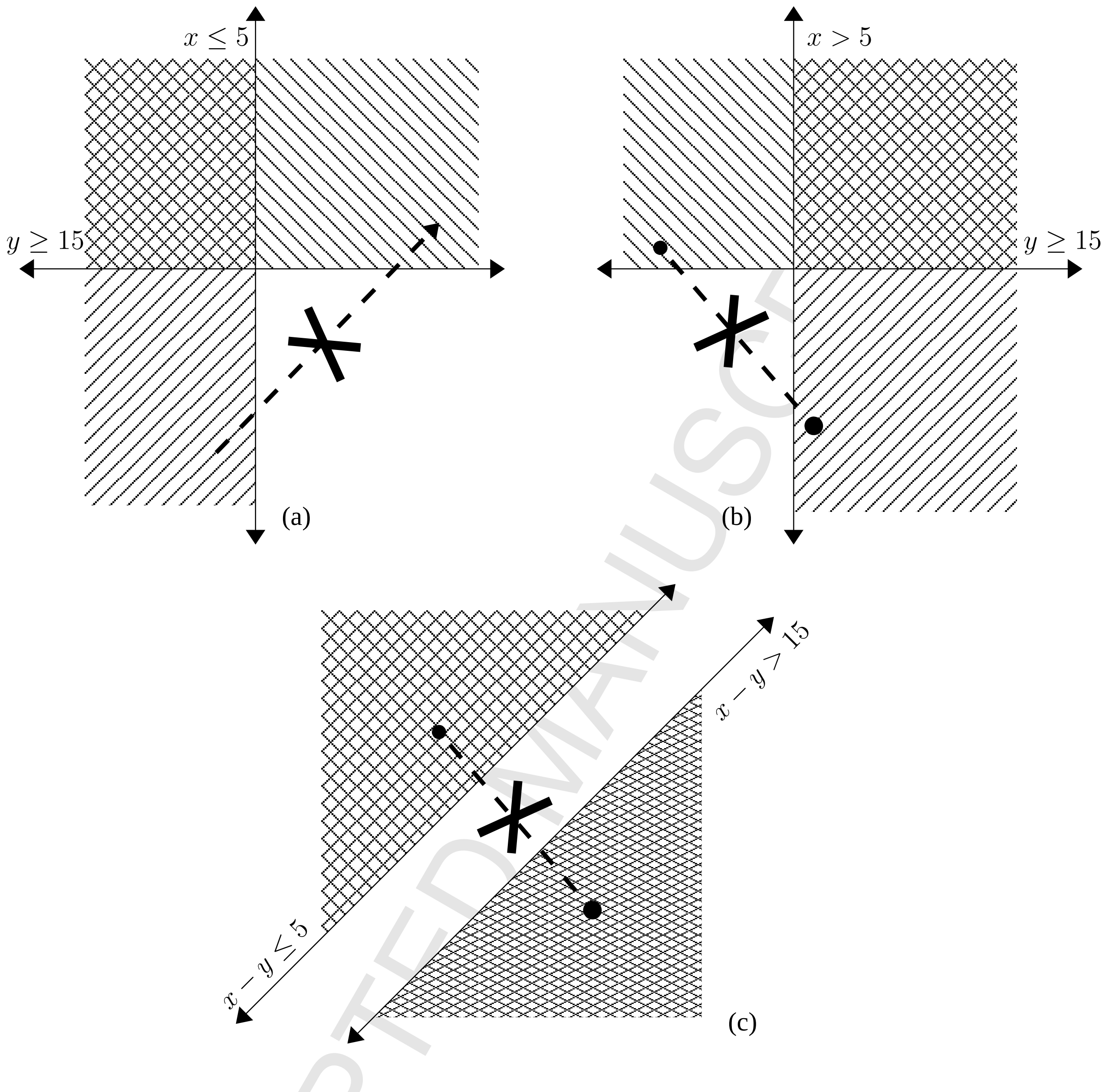


Figure 4: Non-convexities and time-convexities in the state spaces

Example 12 *The* initial condition $\mathcal{I}$ of the TA in Example 6 is convex while the location invariance condition $\mathcal{H}$ is non-convex. Specifically, the following subformula $\dot{H} \equiv \mathit{process} \wedge (x \le 5 \vee y \ge 15)$ is non-convex. For example, we may have two states $(\mathit{process}, \nu_1)$ and $(\mathit{process}, \nu_2)$ with $\nu_1 = \{x \mapsto 4, y \mapsto 4\}$ and $\nu_2 = \{x \mapsto 16, y \mapsto 16\}$. It is clear that $(\mathit{process}, \nu_1) \in [\![\dot{H}]\!]$ and $(\mathit{process}, \nu_2) \in [\![\dot{H}]\!]$. However, the middle point $(\mathit{process}, \{x \mapsto 10, y \mapsto 10\})$ between the two states is not in $[\![\dot{H}]\!]$ as shown in Fig. 4(a).

Figure 4(b) shows another type of non-convexity. The space is characterized by $\ddot{H} \equiv \mathit{process} \wedge (x > 5 \vee y \ge 15)$. As can be seen, although states $(\mathit{process}, \{x \mapsto 6, y \mapsto 12\})$ and $(\mathit{process}, \{x \mapsto 2, y \mapsto 16\})$ are both in the space, there middle point is not.

Convexity may also be absent due to difference constraints between two clock variables. For example, the following state-predicate $\ddot{H} \equiv \mathit{process} \wedge (x-y \leq 5 \vee x-y > 15)$ is also non-convex. Specifically, we may have two states $(\mathit{process}, \nu_3)$ and $(\mathit{process}, \nu_4)$ with $\nu_3 = \{x \mapsto 4, y \mapsto 0\}$ and $\nu_4 = \{x \mapsto 16, y \mapsto 0\}$. It is clear that $(\mathit{process}, \nu_3)$ and $(\mathit{process}, \nu_4)$ are both in $[\![\ddot{H}]\!]$. However the middle point $(\mathit{process}, \{x \mapsto 10, y \mapsto 0\})$ between the two states is not in $[\![\ddot{H}]\!]$ as shown in Fig. 4(c). ■

As explained in Example 9, formulation $\mathsf{T}^c_{\mathit{pre}}$ for timed precondition evaluation can be applied to convex path conditions. Here we relax the restriction of the applicability of formulation $\mathsf{T}^c_{\mathit{pre}}$ with the following concept.

Definition 13 **Time-convexity** A set $S \subseteq \mathcal{ST}$ of states is *time-convex* if and only if for any $(q, \nu) \in S$, $t \in \mathbb{R}^{\geq 0}$, and $t' \in [0, t]$, $(q, \nu + t) \in S$ implies $(q, \nu + t') \in S$. A TCTL formula ϕ is time-convex for $\mathcal{A}$ if $[\![\phi]\!]$ is time-convex. ■

Example 14 *In* Examples 6 and 12, $\mathcal{I}$ is time-convex while $\mathcal{H}$ is non-time-convex. Moreover, state-predicate $\ddot{H} \equiv \mathit{process} \wedge (x-y \leq 5 \vee x-y > 15)$ is non-convex as explained in Fig. 4(c). But we have the following derivation for any state $(\mathit{process}, \nu)$ and real $t \in \mathbb{R}^{\geq 0}$.

$$\begin{array}{ll}
\multicolumn{2}{l}{(\mathit{process}, \nu) \models \mathit{process} \wedge (x-y \leq 5 \vee x-y \geq 15)} \\
\equiv & (\mathit{process}, \nu + t) \models \mathit{process} \wedge ((x-t)-(y-t) \leq 5 \vee (x-t)-(y-t) \geq 15) \\
\equiv & (\mathit{process}, \nu + t) \models \mathit{process} \wedge (x-y \leq 5 \vee x-y \geq 15)
\end{array}$$

Thus it is clear that predicate $\mathit{process} \wedge (x-y \leq 5 \vee x-y \geq 15)$ is time-convex. The intuition is that the non-convexity in this space does not interfere with the continuous time transitions.

Due to similar argument, it should be clear that the non-convex condition $\mathit{process} \wedge (x > 5 \vee y \geq 15)$ is actually time-convex since the non-convexity does not interfere with continuous time transitions. The non-convexity is explained in Fig. 4(b).

Finally, the space of $\mathit{process} \wedge (x \leq 15 \vee y \geq 15)$ in Fig. 4(a) is both non-convex and non-time-convex. ■

Lemma 15 *Given two state-predicates α, β such that $\alpha \vee \beta$ is time-convex, $[\![\mathsf{T}_{\mathit{pre}}(\alpha, \beta)]\!] = [\![\mathsf{T}^c_{\mathit{pre}}(\alpha, \beta)]\!]$.*

Proof : We can prove this lemma in two directions. First, we want to prove that $[\![\mathsf{T}_{\mathit{pre}}(\alpha, \beta)]\!] \subseteq [\![\mathsf{T}^c_{\mathit{pre}}(\alpha, \beta)]\!]$. Given a state $(q, \nu) \models \mathsf{T}_{\mathit{pre}}(\alpha, \beta)$, we have the following derivation.

$$\begin{array}{lll}
\multicolumn{3}{l}{(q, \nu) \models \mathsf{T}_{\mathit{pre}}(\alpha, \beta)} \\
\equiv & (q, \nu) \models \exists t \in \mathbb{R}^{\geq 0} \left(\beta + t \wedge \forall t' \in \mathbb{R}^{\geq 0} \left(t' \leq t \rightarrow (\alpha \vee \beta) + t'\right)\right) & \\
\Rightarrow & (q, \nu) \models \exists t \in \mathbb{R}^{\geq 0} \left(\beta + t \wedge (\alpha \vee \beta) \wedge \forall t' \in \mathbb{R}^{\geq 0} \left(t' \leq t \rightarrow (\alpha \vee \beta) + t'\right)\right) & \text{; instantiating } t' \text{ with } 0 \\
\Rightarrow & (q, \nu) \models \exists t \in \mathbb{R}^{\geq 0} \left(\beta + t \wedge (\alpha \vee \beta)\right) & \text{; restriction relaxation.} \\
\Rightarrow & (q, \nu) \models (\alpha \vee \beta) \wedge \exists t \in \mathbb{R}^{\geq 0} \left(\beta + t\right) & \text{; } \beta \vee \alpha \text{ independent of } t. \\
\equiv & (q, \nu) \models \mathsf{T}^c_{\mathit{pre}}(\alpha, \beta) & \text{; defnition}
\end{array}$$

Now we prove $[\![\mathsf{T}^c_{\mathit{pre}}(\alpha, \beta)]\!] \subseteq [\![\mathsf{T}_{\mathit{pre}}(\alpha, \beta)]\!]$ with the following derivation.

$$\begin{array}{lll}
\multicolumn{3}{l}{(q, \nu) \models \mathsf{T}^c_{\mathit{pre}}(\alpha, \beta)} \\
\equiv & (q, \nu) \models (\alpha \vee \beta) \wedge \exists t \in \mathbb{R}^{\geq 0} \left(\beta + t\right) & \\
\equiv & (q, \nu) \models \exists t \in \mathbb{R}^{\geq 0} \left((\alpha \vee \beta) \wedge \beta + t\right) & \text{; } \alpha \vee \beta \text{ independent of } t. \\
\Rightarrow & (q, \nu) \models \exists t \in \mathbb{R}^{\geq 0} \left(\begin{array}{cc} & (\alpha \vee \beta) \wedge \beta + t \\ \wedge & \forall t' \in \mathbb{R}^{\geq 0} \left(t' \leq t \rightarrow (\alpha \vee \beta) + t'\right)\end{array}\right) & \text{; since } \alpha \vee \beta \text{ is time-convex} \\
\equiv & (q, \nu) \models \exists t \in \mathbb{R}^{\geq 0} \left(\beta + t \wedge \forall t' \in \mathbb{R}^{\geq 0} \left(t' \leq t \rightarrow (\alpha \vee \beta) + t'\right)\right) & \text{; since } 0 \leq t \\
\equiv & (q, \nu) \models \mathsf{T}_{\mathit{pre}}(\alpha, \beta) & \text{; defnition}
\end{array}$$

formulas	formulations	conditions
$\alpha \rightsquigarrow \beta$	$\mathbf{lfp} Z.\left(\mathsf{T}_{pre}(\alpha,\beta) \vee \bigvee_{e\in E} \mathsf{T}_{pre}(\alpha, \mathsf{Xt}_{pre}(e,Z))\right)$	traditional
$\alpha \overset{\supseteq}{\rightsquigarrow} \beta$	$\mathbf{lfp} Z.\left(\beta \vee \bigvee_{e\in E} \mathsf{T}_{pre}(\alpha, \mathsf{Xt}_{pre}(e,Z))\right)$	$\langle\langle\alpha\rangle\rangle \supseteq \langle\langle\beta\rangle\rangle$.
$\alpha \overset{c\supseteq}{\rightsquigarrow} \beta$	$\mathbf{lfp} Z.\left(\beta \vee \bigvee_{e\in E} \mathsf{T}^c_{pre}(\alpha, \mathsf{Xt}_{pre}(e,Z))\right)$	$\langle\langle\alpha\rangle\rangle \supseteq \langle\langle\beta\rangle\rangle$ and time-convex α.
$\alpha \overset{:c}{\rightsquigarrow} \beta$	$\mathbf{lfp} Z.\left(\mathsf{T}_{pre}(\alpha\vee\beta,\beta) \vee \bigvee_{e\in E} \mathsf{T}^c_{pre}(\alpha, \mathsf{Xt}_{pre}(e,Z))\right)$	time-convex α
$\alpha \overset{c:c}{\rightsquigarrow} \beta$	$\mathbf{lfp} Z.\left(\mathsf{T}^c_{pre}(\alpha\vee\beta,\beta) \vee \bigvee_{e\in E} \mathsf{T}^c_{pre}(\alpha, \mathsf{Xt}_{pre}(e,Z))\right)$	α and $\alpha \vee \beta$ both time-convex.

Table 5: Efficient formulations of reachability analysis for special cases of $\alpha \rightsquigarrow \beta$

With the proof for the two directions, we know the lemma is correct. ■

Lemma 15 implies that we can also apply the more efficient T^c_{pre} to non-convex but time-convex path conditions.

Example 16 *In* Example 12, T^c_{pre} was not considered to be applicable to path state-predicate $\ddot{H}$ either. But now, $\ddot{H} \vee \beta \equiv \ddot{H} \vee (\mathit{process} \wedge x = 8 \wedge y = 8)) \equiv \mathit{process} \wedge (x - y \leq 5 \vee x - y \geq 15) \equiv \ddot{H}$. According to Example 14, $\ddot{H}$ is time-convex. Indeed, $\mathsf{T}_{pre}(\ddot{H}, \mathit{process} \wedge x = 8 \wedge y = 8)$ and $\mathsf{T}^c_{pre}(\ddot{H}, \mathit{process} \wedge x = 8 \wedge y = 8)$ both evaluate to predicate $\mathit{process} \wedge x \geq 0 \wedge y \geq 0 \wedge x \leq 8 \wedge y \leq 8 \wedge x - y = 0$. ■

Given a state predicate α, the non-time-convex components in $[\![\alpha]\!]$ can be characterized with the following formula.

$$\mathit{nTConvex}(\alpha) \stackrel{\text{def}}{=} \alpha \wedge \exists t \in \mathbb{R}^{\geq 0}\left(\alpha + t \wedge \exists t' \in \mathbb{R}^{\geq 0}(t' < t \wedge (\neg\alpha) + t')\right)$$

Thus to check whether α is time-convex, we only have to check whether $\mathit{nTConvex}(\alpha)$ is consistent or not.

To take advantage of the time-convexity of path conditions, we have the four variations of $\alpha \rightsquigarrow \beta$ in Table 5 for the efficient evaluation of backward reachability with special cases of α and β. It is obvious that formulations $\alpha \overset{c\supseteq}{\rightsquigarrow} \beta$ and $\alpha \overset{c:c}{\rightsquigarrow} \beta$ can be much more efficient than the other formulations since no execution of T_{pre} is needed. Indeed, in safety analysis and risk analysis of TAs, significant enhancement of verification performance can be achieved with formulation $\alpha \overset{c:c}{\rightsquigarrow} \beta$ since α in the case would be the location invariance conditions of the target TAs which are usually time-convex. One research issue in this work is to efficiently check when we should use efficient formulations in Table 5 for performance in the general setting of TCTL model-checking.

5 Time-convexity in verification problems

In this section, we show two things for TCTL model-checking. First, time-convexity of the location invariance condition $\mathcal{H}$ is good enough to guarantee the time-convexity of all path conditions used in the reachability analysis of $\mathcal{A}$. Second, time-convexity of $\mathcal{H}$ is not sufficient to guarantee the time-convexity of all path conditions in the TCTL model-checking of $\mathcal{A}$.

5.1 For reachability analysis

The most used verification framework is *reachability analysis*. In this framework, we are given a TA $\mathcal{A}$ and a safety predicate η and want to check whether there is an initial run of $\mathcal{A}$ along which some state satisfies $\neg\eta$. The system is *safe* iff $\mathcal{A}$ satisfies $\neg\exists\mathcal{H}\mathbf{U}(\mathcal{H}\wedge\neg\eta)$. The formula means that in the invariance space of $\mathcal{A}$, it is not true that $\neg\eta$ may become true along an initial run. According to Table 3, we find that all the path conditions used in the timed precondition evaluation is exactly $\mathcal{H}$. Thus we have the following lemma.

Lemma 17 *Given a time-convex $\mathcal{H}$ and a safety predicate η, $\mathcal{A}$ is safe iff $\mathcal{I}\wedge(\mathcal{H}\overset{c\supseteq}{\rightsquigarrow}(\mathcal{H}\wedge\neg\eta))$ with the efficient formulation $\overset{c\supseteq}{\rightsquigarrow}$ of reachability analysis in Table 5 is unsatisfiable.*

Proof : The reason is that $\mathcal{H}\vee(\mathcal{H}\wedge\neg\eta)\equiv\mathcal{H}$ which is time-convex. ■

5.2 For TCTL model-checking

We have identified some generic cases in Example 18 through 22 that can break the time-convexity of path conditions in model-checking.

Example 18 Disjunction in the path conditions in modal formulas. We may have a TCTL formula: $\exists(q_1\wedge(x\leq 5\vee y>7))\mathbf{U}q_2$. In the literature [24, 40], this formula can be evaluated as

$$(q_1\wedge(x\leq 5\vee y>7))\rightsquigarrow q_2.$$

For simplicity, we assume that the location invariance $\mathcal{H}$ is *true*. According to Table 3, the path condition is $q_1\wedge(x\leq 5\vee y>7)\vee q_2$. Then the continuous time transitions in the space characterized by $q_1\wedge(x\leq 5\vee y>7)$ in location q_1 can be discontinuous according to Example 9 and Fig. 3 for the state space with only two clocks x and y.

Also, according to Table 3, TCTL formula $\exists\Box(q_1\wedge(x\leq 5\vee y>7))$ may create a non-time-convex path condition. ■

Example 19 Complementation in the path conditions in modal formulas. We have a formula: $\forall\Diamond(q_1\rightarrow(x>5\wedge y\leq 7))$ which can be rewritten as $\neg\exists\Box(q_1\wedge(x\leq 5\vee y>7))$. Similar to the reasoning in Example 18, the path condition $q_1\wedge(x\leq 5\vee y>7)\wedge\mathcal{H}$ is not time-convex when $[\![q_1\wedge x\leq 5\wedge x-5>y-7\wedge\mathcal{H}]\!]\neq\emptyset$ and $[\![q_1\wedge y>7\wedge x-5>y-7\wedge\mathcal{H}]\!]\neq\emptyset$. ■

Note that in TCTL, many abbreviations are defined as the negation of other formulas. As a result, in symbolic model-checking, we usually need to calculate the complement of time-convex state spaces and end up with losing the time-convexity of state spaces.

Example 20 Timing constraints with $\exists\Box$-formulas. Similar to the reasoning in Example 18, formula: $\exists\Box_{\leq 7}x\leq 5$ incurs a path condition $w\leq 7\rightarrow x\leq 5\equiv w>7\vee x\leq 5$. Then following the argument in Example 12 and 14, it is easy to see that this path condition is neither time-convex. ■

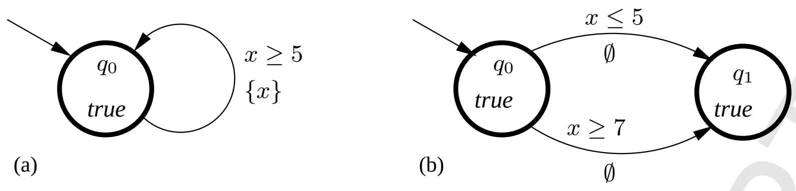


Figure 5: Another example TA

The following two examples show that path condition time-convexity may also be lost due to the structures of TAs.

Example 21 Lack of time-convexities due to TA structures. We may have the example TA $\mathcal{A}$ in Fig. 5(a) and want to check $\mathcal{A}, (q_0, \nu) \models \exists\Box\exists x > 3\,\mathbf{U} x \leq 0$ with $\nu(x) = 0$. Note that $\mathcal{H}$ is time-convex. $(q_0, \nu) \in [\![\exists x > 3\,\mathbf{U} x \leq 0]\!]$ since $x \leq 0$ is immediately fulfilled at (q_0, ν). Also $(q_0, \nu + 4) \in [\![\exists x > 3\mathbf{U} x \leq 0]\!]$ with the firing of the transition at $\nu + 5$. But it is clear that $(q_0, \nu + 2) \notin [\![\exists x > 3\,\mathbf{U} x \leq 0]\!]$. ■

According to the original definition of TCTL [4], only propositions may appear as atoms. Thus we might argue that the above-mentioned formulas in Example 18 to 21 might not happen in the original TCTL definition. The following example is interesting in showing that time-convexity may not be guaranteed even with the original TCTL definition.

Example 22 Nested $\exists$U-formulas with modal timing constraints. Now we may want to check the TA in Fig. 5(b) for a formula $\exists\Box\exists q_0 \mathbf{U}_{<1} q_1$ at a state (q_0, ν) with $\nu(x) = 5$. Then $(q_0, \nu) \in [\![\exists q_0 \mathbf{U}_{<1} q_1]\!]$ and $(q_0, \nu + 2) \in [\![\exists q_0 \mathbf{U}_{<1} q_1]\!]$. However, it is clear that $(q_0, \nu + 1) \notin [\![\exists q_0 \mathbf{U}_{<1} q_1]\!]$. ■

6 Time-convexity checking from the syntax of TCTL formulas

Although there were some new formulations for timed precondition evaluation in [36], such new formulations do not lead to consistent performance enhancement over T_{pre}. In this section, we present techniques for avoiding such checking by recognizing the syntax structures in TCTL that yield only time-convex state-predicates.

Two TCTL formulas ϕ and ψ are *location-disjoint* if two states respectively satisfying ϕ and ψ must not be in the same location. Specifically, ϕ and ψ are location-disjoint if there is a $Q' \subseteq \mathcal{Q}$ such that $[\![\phi]\!] \subseteq [\![\bigvee_{q \in Q'} q]\!]$ and $[\![\psi]\!] \subseteq [\![\bigvee_{q \in \mathcal{Q} - Q'} q]\!]$.

The following lemmas help us recognizing time-convex TCTL path conditions from their syntax structures. First, we are inspired by Example 21 which shows that time-convexity can be violated due to the immediate fulfillment of the destination requirement ψ in a formula of the form $\exists\phi\mathbf{U}\psi$. If the destination formula ψ is not fulfilled immediately, then $\exists\phi\mathbf{U}\psi$ is evaluated with path constraint $\phi \wedge \mathcal{H}$. Thus, with careful restrictions on the immediate fulfillment of ψ, we may have time-convexity from such a formula. The following lemmas 23 and 25 and corollary 24 are established with this inspiration.

Lemma 23 *Assume that we have an interval $\langle c, \infty) \subseteq \mathbb{R}^{\geq 0}$ and two TCTL formulas ϕ and ψ such that $\phi \wedge \mathcal{H}$ and $\psi \wedge \mathcal{H}$ are location-disjoint and both time-convex for $\mathcal{A}$. $[\![\exists\phi\mathbf{U}_{\langle c,\infty)}\psi]\!]$ is time-convex.*

Proof : We assume that there is a $Q' \subseteq \mathcal{Q}$ such that $[\![\phi \wedge \mathcal{H}]\!] \subseteq [\![\bigvee_{q\in\mathcal{Q}} q]\!]$ and $[\![\psi \wedge \mathcal{H}]\!] \subseteq [\![\bigvee_{q\in\mathcal{Q}-Q'} q]\!]$. We assume that there is a state (q, ν) and a $t \in \mathbb{R}^{\geq 0}$ such that (q, ν) and $(q, \nu + t)$ are both in $[\![\exists\phi\mathbf{U}_{\langle c,\infty)}\psi]\!]$. There are two cases to analyze.

- $\underline{q \in Q'}$: This implies that both (q, ν) and $(q, \nu+t)$ are in $[\![\phi\wedge\mathcal{H}]\!]$. Since $[\![\phi\wedge\mathcal{H}]\!]$ is time-convex, we know that for every $t' \in [0, t]$, $(q, \nu+t') \in [\![\phi\wedge\mathcal{H}]\!]$ also. Then the concatenation of the continuous time transition from $(q, \nu + t')$ to $(q, \nu + t)$ and a non-Zeno run from $(q, \nu + t)$ for the proof of $(q, \nu + t) \in [\![\exists\phi\mathbf{U}_{\langle c,\infty)}\psi]\!]$ can be used to support that $(q, \nu + t') \in [\![\exists\phi\mathbf{U}_{\langle c,\infty)}\psi]\!]$.
- $\underline{q \in \mathcal{Q} - Q'}$: This is possible only when $\langle c, \infty)$ is $[0, \infty)$ since (q, ν) does not satisfy ϕ in $\mathcal{A}$. Thus (q, ν) and $(q, \nu + t)$ are both in $[\![\psi \wedge \mathcal{H}]\!]$. Since $[\![\psi \wedge \mathcal{H}]\!]$ is time-convex, we know that for every $t' \in [0, t]$, $(q, \nu + t') \in [\![\psi \wedge \mathcal{H}]\!]$ also. This implies that $\mathcal{A}, (q, \nu + t') \in [\![\exists\phi\mathbf{U}_{[0,\infty)}\psi]\!]$.

Thus the lemma is proven. ■

Corollary 24 *Assume that we have an interval $\langle c, \infty) \subseteq \mathbb{R}^{\geq 0}$ and a TCTL formula ϕ such that $\mathcal{H} \wedge \phi$ and $\mathcal{H} \wedge \neg\phi$ are location-disjoint and both time-convex for $\mathcal{A}$. Then $[\![\exists\Diamond_{\langle c,\infty)}\phi]\!]$ is time-convex.*

Proof : We can prove that $[\![\exists(\neg\phi)\mathbf{U}_{\langle c,\infty)}\phi]\!] = [\![\exists\Diamond_{\langle c,\infty)}\phi]\!]$. Then the corollary follows Lemma 23. ■

As argued in the above, time-convexity can be maintained with a formula $\exists\phi\mathbf{U}\psi$ if we carefully restrict the immediate fulfillment of the destination constraint ψ. Lemma 25 in the following uses this inspiration to forbid the immediate fulfillment of ψ at time distance zero.

Lemma 25 *Two TCTL formulas ϕ and ψ, and an interval $\langle c, \infty) \subseteq (0, \infty)$, if $[\![\mathcal{H}\wedge\phi]\!]$ is time-convex, $[\![\exists\phi\mathbf{U}_{\langle c,\infty)}\psi]\!]$ is time-convex.*

Proof : We assume that $\mathcal{H} \wedge \phi$ is time-convex and there is a state (q, ν) and a $t \in \mathbb{R}^{\geq 0}$ such that (q, ν) and $(q, \nu + t)$ are both in $[\![\exists\phi\mathbf{U}_{\langle c,\infty)}\psi]\!]$. Since $\langle c, \infty) \neq [0, \infty)$, we know that $0 \notin \langle c, \infty)$ which implies that (q, ν) and $(q, \nu + t)$ are both in $[\![\mathcal{H} \wedge \phi]\!]$. Since $[\![\mathcal{H} \wedge \phi]\!]$ is time-convex, we know that for every $t' \in [0, t]$, $(q, \nu + t') \in [\![\mathcal{H} \wedge \phi]\!]$ also. Suppose that there is a non-Zeno run ρ that satisfies $\phi\mathbf{U}_{\langle c,\infty)}\psi$ in $\mathcal{A}$ with a state fulfilling ψ at a state $\hat{t}$ time units from $(q, \nu + t')$ with $\hat{t} \in \langle c, \infty)$. Then the concatenation of the continuous time transition from $(q, \nu + t')$ to $(q, \nu + t)$ and ρ can be used to support that $(q, \nu + t') \in [\![\exists\phi\mathbf{U}_{\langle c,\infty)}\psi]\!]$ since $t - t' + \hat{t} \geq \hat{t}$ and $t - t' + \hat{t} \in \langle c, \infty)$ also. ■

Lemma 26 *Assume that we have an interval $[0, d\rangle \subseteq \mathbb{R}^{\geq 0}$ and two TCTL formulas ϕ and ψ such that $\phi \wedge \mathcal{H}$ and $\psi \wedge \mathcal{H}$ are location-disjoint and both time-convex for $\mathcal{A}$. $[\![\forall\phi\mathbf{U}_{[0,d\rangle}\psi]\!]$ is time-convex.*

Proof : We assume that there is a $Q' \subseteq \mathcal{Q}$ such that $[\![\phi \wedge \mathcal{H}]\!] \subseteq [\![\bigvee_{q\in Q'} q]\!]$ and $[\![\psi \wedge \mathcal{H}]\!] \subseteq [\![\bigvee_{q\in\mathcal{Q}-Q'} q]\!]$. We assume that there is a state (q, ν) and a $t \in \mathbb{R}^{\geq 0}$ such that (q, ν) and $(q, \nu + t)$ are both in $[\![\forall\phi\mathbf{U}_{[0,d\rangle}\psi]\!]$. There are two cases to analyze.

- $\underline{q \in Q'}$: This implies that both (q,ν) and $(q,\nu+t)$ are in $[\![\mathcal{H} \wedge \phi]\!]$. Moreover, both (q,ν) and $(q,\nu+t)$ are not in $[\![\mathcal{H} \wedge \psi]\!]$. This further implies that $t \in [0,d\rangle$. Otherwise, the continuous time transition from (q,ν) to $(q,\nu+t)$ refutes $(q,\nu) \in [\![\forall\phi\mathbf{U}_{[0,d\rangle}\psi]\!]$. Since $[\![\mathcal{H} \wedge \phi]\!]$ is time-convex, we know that for every $t' \in [0,t]$, $(q,\nu+t') \in [\![\mathcal{H} \wedge \phi]\!]$ and $(q,\nu+t') \notin [\![\mathcal{H} \wedge \psi]\!]$ also. With some arithmetic on the time length to the fulfillment of ψ, we know that for every $t' \in [0,t]$, $(q,\nu+t') \in [\![\forall\phi\mathbf{U}_{[0,\lfloor d-t'\rfloor\rangle}\psi]\!]$. Since $[0,\lfloor d-t'\rfloor\rangle \subseteq [0,d\rangle$, we know that $(q,\nu+t') \in [\![\forall\phi\mathbf{U}_{[0,d\rangle}\psi]\!]$ according to the semantics of TCTL.
- $\underline{q \in \mathcal{Q} - Q'}$: In this case, we know that (q,ν) and $(q,\nu+t)$ are both in $[\![\mathcal{H} \wedge \psi]\!]$. Since $[\![\mathcal{H} \wedge \psi]\!]$ is time-convex, we know that for every $t' \in [0,t]$, $(q,\nu+t') \in [\![\mathcal{H} \wedge \psi]\!]$ also. This implies that $\mathcal{A},(q,\nu+t') \in [\![\forall\phi\mathbf{U}_{[0,d\rangle}\psi]\!]$.

Thus the lemma is proven. ■

The following corollary is important since it captures the time-convexity of the important class of timed inevitabilities with deadlines in TCTL formulas.

Corollary 27 *Assume that we have an interval $[0,d\rangle \subseteq \mathbb{R}^{\geq 0}$ and a TCTL formula ϕ such that $\mathcal{H} \wedge \phi$ and $\mathcal{H} \wedge \neg\phi$ are location-disjoint and both time-convex for $\mathcal{A}$. Then $[\![\forall\Diamond_{[0,d\rangle}\phi]\!]$ is time-convex.*

Proof : We can prove that $[\![\forall(\neg\phi)\mathbf{U}_{[0,d\rangle}\phi]\!] = [\![\forall\Diamond_{[0,d\rangle}\phi]\!]$. Then the corollary follows Lemma 26. ■

The lemma can be useful when we have a $\forall\Diamond$ modal operator in a path constraint. The reason is that the standard formulation for the evaluation of a formula like $\forall\Diamond_{[0,d\rangle}\phi$ is $\neg\exists\Box_{[0,d\rangle}\neg\phi$. Thus, even if we know that $[\![\exists\Box_{[0,d\rangle}\neg\phi]\!]$ is time-convex, it could still be difficult to check if its complement is. But this lemma tells us that we do not have to check it if the lemma can be applied.

Example 28 *For* the TA in Example 6, suppose that we want to specify that location *process* will always happen in 15 seconds. The property can be written as $\forall\Diamond_{[0,15]}$*process*. According to corollary 27, the space of the property is time-convex. ■

Lemma 29 *Given a TCTL formula ϕ for $\mathcal{A}$ and an interval $\langle c,\infty) \subseteq \mathbb{R}^{\geq 0}$, $[\![\forall\Box_{\langle c,\infty)}\phi]\!]$ is time-convex.*

Proof : We assume that there is a state (q,ν) and a $t \in \mathbb{R}^{\geq 0}$ such that (q,ν) and $(q,\nu+t)$ are both in $[\![\forall\Box_{\langle c,\infty)}\phi]\!]$. For every $t' \in [0,t]$, every run from $(q,\nu+t')$ is a tail of a run from (q,ν). There are two cases to analyze.

- $\underline{t' \notin \langle c,\infty)}$: This implies that $(q,\nu+t') \in [\![\forall\Box_{\langle\lceil c-t'\rceil,\infty)}\phi]\!]$. It is easy to see that $\langle c,\infty) \subseteq \langle\lceil c-t'\rceil,\infty)$. Thus we know $(q,\nu+t') \in [\![\forall\Box_{\langle c,\infty)}\phi]\!]$ according to the semantics of TCTL.
- $\underline{t' \in \langle c,\infty)}$: This implies $(q,\nu+t') \in [\![\forall\Box_{[0,\infty)}\phi]\!]$. This further implies that $(q,\nu+t') \in [\![\forall\Box_{\langle c,\infty)}\phi]\!]$ according to the semantics of TCTL.

Thus the lemma is proven. ■

Interestingly, Lemma 29 says that $[\![\forall\Box_{\langle c,\infty)}\phi]\!]$ is time-convex even if $[\![\phi]\!]$ is not. Moreover, based on an argument similar to the one for the usefulness of corollary 27, we feel that Lemma 29 can also be useful in deciding the time-convexity of the complement of evaluation of formulas like $\neg\exists\Diamond_{\langle c,\infty)}\neg\phi$.

Lemma 30 *Given a time-convex TCTL formula ϕ for $\mathcal{A}$ and a non-empty interval $[0,d\rangle \subseteq \mathbb{R}^{\geq 0}$, $[\![\exists\Box_{[0,d\rangle}\phi]\!]$ is time-convex.*

Proof : We assume that there is a state (q,ν) and a $t \in \mathbb{R}^{\geq 0}$ such that (q,ν) and $(q,\nu+t)$ both satisfy $[\![\exists\Box_{[0,d\rangle}\phi]\!]$. This implies that $(q,\nu) \in [\![\phi]\!]$ and $(q,\nu+t) \in [\![\phi]\!]$. Since $[\![\phi]\!]$ is time-convex, we know that for every $t' \in [0,t]$, $(q,\nu+t') \in [\![\phi]\!]$. Moreover, there is a non-Zeno run ρ from $(q,\nu+t)$ that satisfies $\Box_{[0,d\rangle}\phi$. Thus the concatenation of the continuous time transition from $(q,\nu+t')$ to $(q,\nu+t)$ and ρ has an initial run segment of length $t-t'+d \geq d$ along which all states are in $[\![\phi]\!]$. According to the semantics of TCTL, this means that $(q,\nu+t') \in [\![\exists\Box_{[0,d\rangle}\phi]\!]$. ■

Based on the lemmas in the above, we can define a new class of TCTL formulas that yield only time-convex state spaces. This class is called *time-convex Tree Logic*, denoted `TCXTL` , and is defined inductively as follows.

Definition 31 `TCXTL` Only TCTL formulas of the following forms are members of `TCXTL` .

- η, a state-predicate such that *nTConvex*$(\eta \wedge \mathcal{H})$ is unsatisfiable.
- $\phi_1 \wedge \phi_2$, with $\phi_1 \in$ `TCXTL` $\wedge\, \phi_2 \in$ `TCXTL`.
- $\phi_1 \vee \phi_2$, with location-disjoint `TCXTL` formulas ϕ_1 and ϕ_2.
- $\exists\phi_1\mathbf{U}_{\langle c,\infty)}\phi_2$ with location-disjoint `TCXTL` formulas ϕ_1 and ϕ_2.
- $\exists\phi_1\mathbf{U}_{\langle c,\infty)}\phi_2$ with $\phi_1 \in$ `TCXTL` and $\langle c,\infty) \subseteq (0,\infty)$.
- $\forall\phi_1\mathbf{U}_{[0,d\rangle}\phi_2$ with location-disjoint `TCXTL` formulas ϕ_1 and ϕ_2.
- $\forall\Box_{\langle c,\infty)}\phi$ with $\langle c,\infty) \subseteq \mathbb{R}^{\geq 0}$ and $\phi \in$ `TCXTL`.
- $\exists\Box_{[0,d\rangle}\phi$ with $[0,d\rangle \subseteq \mathbb{R}^{\geq 0}$ and $\phi \in$ `TCXTL`. ■

Note that `TCXTL` itself may not support the full specification of some properties. But it helps identifying subformulas in a full specification that yield only time-convex path conditions for efficient timed precondition evaluation. For example, in our experiment reported in Section 8, some benchmark formulas are not in `TCXTL` while they all contain an inner `TCXTL` path condition. According to our experiment, significant performance enhancement in model-checking was indeed observed by taking advantage of the time-convexity of the corresponding path conditions.

7 New formulations of timed inevitabilities

As can be seen from Example 2 and 3, the time-passage conditions used in evaluating timed inevitabilities in general do not belong to TCXTL and may even be non-time-convex. Consider the timed inevitability in Example 2.

$$\forall\Box\left(\texttt{fire} \rightarrow \forall\Diamond_{[5,10]}\texttt{alarm}\right).$$

This property is usually evaluated with the following traditional formulation [24, 40]:

$$\neg\exists \mathit{true}\mathbf{U}\left(\texttt{fire} \wedge \exists w\left(w=0 \wedge \exists\Box(w<5 \vee w>10 \vee \neg\texttt{alarm})\right)\right),$$

where w is a clock variable not used in the model. Let (q, ν) denote a state in a location q and with a clock valuation ν. Suppose that we are given three valuations ν_1, ν_2, ν_3 with $\nu_1(w) = 0, \nu_2(w) = 6$, and $\nu_3(w) = 12$. $(\mathtt{alarm}, \nu_2)$ is the middle point between $(\mathtt{alarm}, \nu_1)$ and $(\mathtt{alarm}, \nu_3)$. $(\mathtt{alarm}, \nu_1)$ and $(\mathtt{alarm}, \nu_3)$ both satisfy $w < 5 \vee w > 10 \vee \neg\mathtt{alarm}$ while $(\mathtt{alarm}, \nu_2)$ does not. Thus it is obvious that such a formulation creates a non-convex path condition $w < 5 \vee w > 10 \vee \neg\mathtt{alarm}$. This implies that formulation T^c_{pre} can not be straightforwardly applied in the evaluation of such timed inevitabilities.

Note that in the above TCTL formula, the inner modal formula, i.e., $\exists\Box_{[5,10]}\neg\mathtt{alarm}$ can not be correctly evaluated with the approximate formulation in Table 4 which does not take non-Zenoness into consideration. For example, the property can be violated with a Zeno run that does not progress more than 5 time units after $\mathtt{fire}$ is true. Thus, to correctly check such properties, we can not use the approximate formulation in Table 4.

In this section, we are especially interested at the time-convexity of path conditions with negated timed inevitabilities of the following form $\exists\Box_\Theta\phi$. According to the formulation in Table 3, the path condition is $\mathcal{H} \wedge (w \in \Theta \rightarrow \langle\langle\phi\rangle\rangle) \equiv \mathcal{H} \wedge (w \notin \Theta \vee \langle\langle\phi\rangle\rangle)$. Please be reminded that w is a clock variable that does not appear in ϕ and $\mathcal{A}$. Then this path condition can be further rewritten as

$$\mathcal{H} \wedge (w \prec \Theta \vee (w \in \Theta \wedge \langle\langle\phi\rangle\rangle) \vee w \succ \Theta)$$

As can be seen, continuous time transitions in such a path condition can be broken into three segments, the first in $\mathcal{H} \wedge w \prec \Theta$, the second in $\mathcal{H} \wedge w \in \Theta \wedge \langle\langle\phi\rangle\rangle$, and the third in $\mathcal{H} \wedge w \succ \Theta$. In Subsection 7.1, we show that the time-convexity of $\mathcal{H}$ and $\langle\langle\phi\rangle\rangle$ guarantees the time-convexity between the first two segments or the last two segments. Then with special arrangement in Subsection 7.2, the time-convexity of $\mathcal{H}$ and $\langle\langle\phi\rangle\rangle$ in turn ensures that we can correctly evaluate such timed inevitabilities only with T^c_{pre}.

Later in Subsection 7.3, we shall present a new approximate formulation with enough precision to correctly and efficiently evaluate many timed inevitabilities.

7.1 Time-convexity across the boundary of two time-convex spaces

We need the following concepts. A state predicate η is in *positive normal form* (*PNF*) if the negations in η only appears right in front of atomic propositions. It is clear that by properly rewriting the inequalities and de Morgan's law, we can translate every state predicate to its PNF easily. Clock inequalities of the form $x \geq c$, $x > c$, $-x < -c$, and $-x \leq c$ are called *lower-bound inequalitie*s. Clock inequalities of the form $x \leq c$, $x < c$, $-x > -c$, and $-x \geq c$ are called *upper-bound inequalitie*s. Given a PNF state predicate η, we let *diff*(η) be a state predicate obtained from η by replacing every lower-bound and upper-bound inequalities in η with *true*. In a continuous time transition, a difference inequality $x - y \sim d$ does not change its truth value since the increment of x's reading cancels with that of y's reading in the inequality. It is easy to see that *diff*(η) is always time-convex.

The following lemma establishes the time-convexity of continuous time transitions between the last two segments.

Lemma 32 *Suppose that $\mathcal{H}$ is time-convex and we are given an interval Θ and a TCTL formula ϕ such that $\langle\!\langle\phi\rangle\!\rangle$ is PNF and time-convex and w is not used in ϕ and $\mathcal{A}$. Then $\mathcal{H} \wedge (w \succ \Theta \vee (w \in \Theta \wedge \langle\!\langle\phi\rangle\!\rangle))$ is time-convex.*

Proof : We assume that there is a state (q, ν) and a $t \in \mathbb{R}^{\geq 0}$ such that (q, ν) and $(q, \nu + t)$ both satisfy $\mathcal{H} \wedge (w \succ \Theta \vee (w \in \Theta \wedge \langle\!\langle\phi\rangle\!\rangle))$. Then we have the following case analysis.

- Case (q, ν) and $(q, \nu + t)$ both satisfies $\mathcal{H} \wedge w \succ \Theta$: Since $\mathcal{H}$ and $w \succ \Theta$ are both time-convex, their conjunction is still time-convex. This implies that for every $t' \in [0, t]$, $(q, \nu + t') \models \mathcal{H} \wedge (w \succ \Theta \vee (w \in \Theta \wedge \langle\!\langle\phi\rangle\!\rangle))$.
- Case (q, ν) and $(q, \nu + t)$ both satisfies $\mathcal{H} \wedge w \in \Theta \wedge \langle\!\langle\phi\rangle\!\rangle$: Since $\mathcal{H}$, $w \in \Theta$, and $\langle\!\langle\phi\rangle\!\rangle$ are all time-convex, their conjunction is still time-convex. This implies that for every $t' \in [0, t]$, $(q, \nu + t') \models \mathcal{H} \wedge (w \succ \Theta \vee (w \in \Theta \wedge \langle\!\langle\phi\rangle\!\rangle))$.
- Case $(q, \nu) \models \mathcal{H} \wedge w \in \Theta \wedge \langle\!\langle\phi\rangle\!\rangle$ and $(q, \nu + t) \models \mathcal{H} \wedge w \succ \Theta$: Since *diff*$(\langle\!\langle\phi\rangle\!\rangle)$ is weaker than $\langle\!\langle\phi\rangle\!\rangle$, we know that $(q, \nu) \models \mathcal{H} \wedge w \in \Theta \wedge$ *diff*$(\langle\!\langle\phi\rangle\!\rangle)$. In a continuous time transition, a difference inequality $x - y \sim d$ does not change its truth value since the increment of x's reading cancels with that of y's reading in the inequality. Thus it is clear that $(q, \nu + t) \models \mathcal{H} \wedge$ *diff*$(\langle\!\langle\phi\rangle\!\rangle)$. Then the convexities of $\mathcal{H}$ and *diff*$(\langle\!\langle\phi\rangle\!\rangle)$ imply that for every $t' \in [0, t]$, $(q, \nu + t') \models \mathcal{H} \wedge$ *diff*$(\langle\!\langle\phi\rangle\!\rangle)$.

  We now prove $(q, \nu + t') \models \mathcal{H} \wedge (w \succ \Theta \vee (w \in \Theta \wedge \langle\!\langle\phi\rangle\!\rangle))$ by contradiction. If the above satisfaction is false, it means that $(q, \nu + t') \models \mathcal{H} \wedge$ *diff*$(\langle\!\langle\phi\rangle\!\rangle) \wedge w \in \Theta \wedge \neg\langle\!\langle\phi\rangle\!\rangle$. Since $t' \geq 0$ and $\langle\!\langle\phi\rangle\!\rangle$ and *diff*$(\langle\!\langle\phi\rangle\!\rangle)$ share the same difference inequalities, this implies that there is an upper-bound $x < d$ or $x \leq d$ in $\langle\!\langle\phi\rangle\!\rangle$ that is violated by $(q, \nu + t')$. We assume that this upper-bound inequality is $x < d$ without loss of generality. Then $(q, \nu) \models x < d$ while $(q, \nu + t') \not\models x < d$. This means that there is a lower-bound inequality on x in $\mathcal{H} \wedge w \succ \Theta$, say $x \geq a$, that is satisfied at $(q, \nu + t)$. This implies that $a > d$ and further implies that $(q, \nu + t')$ can not satisfy $\mathcal{H} \wedge w \succ \Theta$. This suggests that x can not be w since between $w \in \Theta$ and $w \succ \Theta$ there is no such a gap. But for all other clock variable x, this leads to a contradiction since $\mathcal{H}$ characterizes a space that contains the space characterized by $\mathcal{H} \wedge$ *diff*$(\langle\!\langle\phi\rangle\!\rangle) \wedge \neg\langle\!\langle\phi\rangle\!\rangle$. Thus the lemma is also proven in this case.

With the proof of the three cases, the lemma is proven. ■

Similarly to Lemma 32, we can establish the following lemma for the time-convexity of time transitions between the first two segments.

Lemma 33 *Suppose that $\mathcal{H}$ is time-convex and we are given an interval Θ and a TCTL formula ϕ such that $\langle\!\langle\phi\rangle\!\rangle$ is PNF and time-convex and w is not used in ϕ and $\mathcal{A}$. Then $\mathcal{H} \wedge (w \prec \Theta \vee (\langle\!\langle\phi\rangle\!\rangle \wedge w \in \Theta))$ is time-convex.*

Proof : The proof is similar to that for Lemma 32. ■

Θ		
$[0,$	$\infty)$	formulations
No	No	$\exists w \left(w = 0 \wedge \left(\begin{array}{l} (w \prec \Theta \vee (\langle\langle\phi\rangle\rangle \wedge w \in \Theta)) \\ \overset{c\supseteq}{\rightsquigarrow} \left(w \in \Theta \wedge \left(\begin{array}{l} ((\langle\langle\phi\rangle\rangle \wedge w \in \Theta) \vee w \succ \Theta) \\ \overset{c\supseteq}{\rightsquigarrow} (w \succ \Theta \wedge \mathrm{NZ}(\mathit{true})) \end{array} \right) \right) \end{array} \right) \right)$
Yes	No	$\exists w \left(w = 0 \wedge \left(((\langle\langle\phi\rangle\rangle \wedge w \in \Theta) \vee w \succ \Theta) \overset{c\supseteq}{\rightsquigarrow} (w \succ \Theta \wedge \mathrm{NZ}(\mathit{true})) \right) \right)$
No	Yes	$\exists w \left(w = 0 \wedge \left((w \prec \Theta \vee w \in \Theta) \overset{c\supseteq}{\rightsquigarrow} (w \in \Theta \wedge \mathrm{NZ}(\langle\langle\phi\rangle\rangle)) \right) \right)$

Under the assumption that $\langle\langle\phi\rangle\rangle$ is a zone-structure predicate and $\mathcal{H}$ is time-convex.

Table 6: Our formulations for the exact analysis of $\exists\Box_\Theta\phi$

7.2 New exact formulation

We can see that for a non-Zeno run ρ to satisfy $\Box_\Theta\phi$, it means that ρ can be decomposed into three run segments ρ_1, ρ_2, ρ_3 with the following restrictions.

- States in ρ_1 happen at time before Θ and satisfy $\mathcal{H}$.
- States in ρ_2 happen at time in Θ and satisfy $\phi \wedge \mathcal{H}$.
- ρ_3 is a non-Zeno run with all states happening at time after Θ and satisfy $\mathcal{H}$.

This observation leads to the new formulation for a formula like $\exists\Box_\Theta\phi$ in Table 6. The advantage of the new formulation is that it may happen that for each segment of ρ_1, ρ_2, ρ_3, its individual path condition could be time-convex and the corresponding timed precondition evaluation can be done with the more efficient formulation T^c_{pre} in place of T_{pre}. For example, for $\exists\Box_\Theta\phi$, we have the following mapping between the syntax structure of its top formulation (case (No,No)) in Table 6 and the three run segments.

- The path condition of the outer "$\overset{c\supseteq}{\rightsquigarrow}$" formula characterizes the time progress from ρ_1 to ρ_2. It can be evaluated with "$\overset{c\supseteq}{\rightsquigarrow}$" due to the following two reasons.
  - The time-convexity of $\mathcal{H} \wedge (w \prec \Theta \vee (w \in \Theta \wedge \langle\langle\phi\rangle\rangle))$ which follows from Lemma 33 and the time-convexity of $\mathcal{H}$ and $\langle\langle\phi\rangle\rangle$.
  - The subsumption of the corresponding destination condition $w \in \Theta \wedge \langle\langle\phi\rangle\rangle$ by the path condition $w \prec \Theta \vee (w \in \Theta \wedge \langle\langle\phi\rangle\rangle)$.
- The destination condition of the outer "$\overset{c\supseteq}{\rightsquigarrow}$" formula and the path condition of the inner "$\overset{c\supseteq}{\rightsquigarrow}$" formula together characterize the time progress from ρ_2 to ρ_3. It can be evaluated with "$\overset{c\supseteq}{\rightsquigarrow}$" due to the following two reasons.
  - The time-convexity of $\mathcal{H} \wedge ((w \in \Theta \wedge \langle\langle\phi\rangle\rangle) \vee w \succ \Theta)$ which follows from Lemma 32 and the time-convexity of $\mathcal{H}$ and $\langle\langle\phi\rangle\rangle$.
  - The subsumption of the corresponding destination condition $w \succ \Theta$ by the path condition $(w \in \Theta \wedge \langle\langle\phi\rangle\rangle) \vee w \succ \Theta$.
- The destination condition of the inner "$\rightsquigarrow$" formula and $\mathrm{NZ}(\mathit{true})$ together characterize the time progress

Θ		
$[0,$	$\infty)$	formulations
No	No	$\exists w\left(w=0\wedge\left(\begin{array}{l}(w\prec\Theta\vee(\langle\langle\phi\rangle\rangle\wedge w\in\Theta))\\ \overset{c\supseteq}{\rightsquigarrow}\left(w\in\Theta\wedge\left(((\langle\langle\phi\rangle\rangle\wedge w\in\Theta)\vee w\succ\Theta)\overset{c\supseteq}{\rightsquigarrow}w\succ\Theta\right)\right)\end{array}\right)\right)$
Yes	No	$\exists w\left(w=0\wedge\left(((\langle\langle\phi\rangle\rangle\wedge w\in\Theta)\vee w\succ\Theta)\overset{c\supseteq}{\rightsquigarrow}w\succ\Theta\right)\right)$
No	Yes	$\exists w\left(w=0\wedge\left((w\prec\Theta\vee w\in\Theta)\overset{c\supseteq}{\rightsquigarrow}(w\in\Theta\wedge\langle\langle\phi\rangle\rangle)\right)\right)$

Under the assumption that $\langle\langle\phi\rangle\rangle$ is a zone-structure predicate and $\mathcal{H}$ is time-convex.

Table 7: Our formulations for the Zeno approximation of $\exists\Box_\Theta\eta$

along ρ_3. It can be evaluated with formulation T^c_{pre} if $\mathcal{H}$ is time-convex.

Given a TCTL formula ϕ, we use $\langle\langle\phi\rangle\rangle^{new}$ to denote the state predicate obtained for ϕ from the algorithm in tables 2 and 3 except that formulation in Table 6 is used for formulas like $\exists\Box_\Theta\phi$. The following lemma establishes the correctness of the new formulation in Table 6.

Lemma 34 *Given a TCTL formula ϕ, $[\![\phi]\!]=[\![\langle\langle\phi\rangle\rangle^{new}]\!]$.*

Proof : The evaluation of $\langle\langle\phi\rangle\rangle^{new}$ differs from that of $\langle\langle\phi\rangle\rangle$ only in the formulations used for $\exists\Box$-formulas. For convenience, we assume that Θ does not contain zero and does not extend to infinity. The other cases can be proven in a similar way. According to the semantics of TCTL, a formula $\exists\Box_\Theta\phi$ is satisfied if there is a run as the concatenation of a head run segment ρ_1, a middle run segment ρ_2, and a non-Zeno tail run segment ρ_3 as described in the above. Then by a structural analysis, we find the followings are true.

- The inner-most modal formula NZ$(\textit{true})$ is satisfied iff there is such a non-Zeno tail run segment ρ_3.
- The inner $\rightsquigarrow$ formula is satisfied iff there is such a middle run segment ρ_2 followed by a non-Zeno tail run segment ρ_3.
- The outer $\rightsquigarrow$ formula is satisfied iff there is such a head run segment ρ_1 followed by a middle run segment ρ_2 and then by a non-Zeno tail run segment ρ_3.

Thus it is clear that the lemma is true. ■

7.3 New approximate formulation

In Section 8, we will see that for the benchmarks, the exact formulation in Table 6 indeed leads to much better verification performance. However, there is an additional advantage of the formulation in Table 6 since it links to a safe approximation of timed inevitabilities with better precision. Intuitively, we may replace NZ(ϕ) in Table 6 directly with $\langle\langle\phi\rangle\rangle$ as an efficient approximation. Specifically, we present the new Zeno approximate formulation in Table 7 for formulas like $\exists\Box_\Theta\phi$. We use $\langle\langle\phi\rangle\rangle^{new-app}$ to denote the state predicate obtained for ϕ from the algorithm in tables 2 and 3 except that formulation in Table 7 is used for formulas like $\exists\Box_\Theta\phi$. The following lemma establishes the correctness of the new formulation in Table 7.

Lemma 35 *Given a state-predicate η and an interval $\Theta \subseteq \mathbb{R}^{\geq 0}$.*

$$[\![\langle\!\langle \forall\Diamond_\Theta \eta \rangle\!\rangle^{app}]\!] \subsetneq [\![\langle\!\langle \forall\Diamond_\Theta \eta \rangle\!\rangle^{new-app}]\!] \subsetneq [\![\forall\Diamond_\Theta \eta]\!].$$

Proof : The proof is similar to the one for Lemma 11. Since $\forall\Diamond_\Theta \eta \equiv \neg\exists\Box_\Theta\neg\eta$, the lemma is equivalent to two assertions $[\![\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{app}]\!] \supsetneq [\![\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{new-app}]\!]$ and $[\![\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{new-app}]\!] \supsetneq [\![\exists\Box_\Theta\neg\eta]\!]$.

For $[\![\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{app}]\!] \supsetneq [\![\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{new-app}]\!]$, note that the set of runs considered with $\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{app}$ is a superset of those considered with $\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{new-app}$. The extra runs considered with formulation $\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{app}$ are those Zeno runs with time converges to a value in before the end of Θ. Thus it is clear that this part is true with the semantics of $\forall\Diamond$-formulas.

For $[\![\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{new-app}]\!] \supsetneq [\![\exists\Box_\Theta\neg\eta]\!]$, note that the set of runs considered with formulation $\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{new-app}$ is a superset of those considered with the semantics of $\exists\Box_\Theta\neg\eta$. The extra runs considered with $\langle\!\langle \exists\Box_\Theta\neg\eta \rangle\!\rangle^{new-app}$ are those Zeno runs that consist of the following.

- A head segment with states of time preceding Θ from the start of the run.
- A middle segment with states of time in Θ from the start of the run.
- A tail Zeno run segment.

Thus the second assertion is also true. With the two assertions proven, we know the lemma is proven. ■

This lemma shows that the new approximate formulation in Table 7 is not only safe but also is strictly more precise than the one in Table 4 for evaluating timed inevitabilities. For example, we may want to check $\forall\Diamond_{(5,8)} q_1$ on the TA in Fig. 6. The formula is first converted to $\neg\exists\Box_{(5,8)}\neg q_1$. On one hand, $\langle\!\langle \exists\Box_{(5,8)}\neg q_1 \rangle\!\rangle^{app}$ characterizes

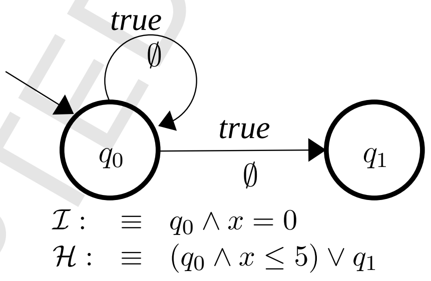

true
∅
true
∅
q_0
q_1
$\mathcal{I}: \equiv q_0 \wedge x = 0$
$\mathcal{H}: \equiv (q_0 \wedge x \leq 5) \vee q_1$

Figure 6: An example TA with Zeno runs

those states that start runs along which any state that is more than five time units and less than 8 time units from the starting state of the run satisfies $\neg q_1$. One such run is Zeno since it stays in q_0 with time not progressing beyond 5. Thus it is easy to see that all states in the TA satisfy $\langle\!\langle \exists\Box_{(5,8)}\neg q_1 \rangle\!\rangle^{app}$. This in turn means that the TA does not satisfy $\langle\!\langle \forall\Diamond_{(5,8)} q_1 \rangle\!\rangle^{app}$. However this is counter-intuitive.

On the other hand, $\langle\!\langle \exists\Box_{(5,8)}\neg q_1 \rangle\!\rangle^{new-app}$ characterizes those states that start a run segment of 5 time units followed by a run segment of 3 time units always satisfying $\neg q_1$. It is easy to see that no state of the TA satisfies this characterization since any run longer than 5 time units must enter q_1 after 5 time units. This in turn means that all initial states of the TA satisfy $\langle\!\langle \forall\Diamond_{(5,8)} q_1 \rangle\!\rangle^{new-app}$.

(A)	$\forall\Box(\texttt{waiting}_1 \rightarrow \exists(\exists\Diamond_{[19,\infty)}\texttt{critical}_1)\mathbf{U}\texttt{idle}_1))$
(B)	$\forall\Box\left((\texttt{ready}_1 \wedge x_1 = 0) \rightarrow \exists(\exists\Box_{[0,10)}\texttt{ready}_1)\mathbf{U}\texttt{waiting}_1\right)$
(C)	$\forall\Box\left((\texttt{transm}_1 \wedge \exists i \in [2,m], \texttt{transm}_i) \rightarrow \exists(\forall\Diamond_{[0,10)}\neg\texttt{transm}_1)\mathbf{U}\texttt{retry}_1))\right)$
(D)	$\forall(\forall i \in [2,m], \forall\Box_{[m,\infty)}\texttt{parent}_i \neq 0)\mathbf{U}(\texttt{parent}_1 = 0 \wedge \forall i \in [2,m], \texttt{parent}_i \neq 0))$
(E)	$\forall\Box\left((\texttt{waiting}_1 \wedge x_1 = 0) \longrightarrow \forall\Diamond_{[0,19]}\neg\texttt{waiting}_1\right)$
(F)	$\forall\Box\left((\exists i \in [1,m], (\texttt{transm}_i \wedge \exists j \in [i+1,m], \texttt{transm}_j) \rightarrow \forall\Diamond_{[0,26]}\texttt{bus\_idle}\right)$
(G)	$\forall\Diamond_{[0,\lceil\log(m)\rceil]}(\texttt{parent}_1 = 0 \wedge \forall i \in [2,m], \texttt{parent}_i \neq 0)$

Table 8: Seven formulas used in the experiment

Our experiment report in the next section corroborates Lemma 35 since against all our timed inevitability benchmarks, the approximate formulation in Table 7 yields correct evaluation while the one in Table 4 does not.

8 Implementation and experiments

We have implemented our ideas in `RED` 8, a model-checker/simulation-checker for TAs [5] and a parametric safety analyzer for LHAs (linear hybrid automata) [3] built on top of REDLIB, a library based on CRD and HRD (Hybrid-Restriction Diagram) technology [33, 35]. We first explain our benchmarks in Subsection 8.1. Subsection 8.2 reports the experiment with the techniques presented in Section 6 for identifying time-convex path conditions for the efficient evaluation of timed preconditions. Subsection 8.3 reports the experiment with the new formulations for timed inevitability evaluations.

8.1 Benchmarks

For the two experiments, we use the following parameterized benchmarks so that we can observe how our techniques scale with the benchmark concurrency sizes.

1. *Fischer's timed mutual exclusion algorithm* [33]: The system uses a shared lock and a local clock per process. The participating processes need to set the lock in order to gain access to the critical section. Parameter m in the performance data table represents the number of processes. Three timing constants used are 10, 19, and 30. Formulas (A) and (B) in Table 8 are for the corresponding experiment in Subsection 8.2 while (E) is for 8.3.
2. *CSMA/CD* [43]: This is essentially the Ethernet bus arbitration protocol with collision-and-retry. In total, there is one bus process and m sender processes. The timing constants used are 26, 52, and 808. Formulas (C) in Table 8 is for the corresponding experiment in Subsection 8.2 while (F) is for 8.3.
3. *Leader election* [33]: In the benchmark, there are m processes that want to form a tree network with the root process as the master and all the other processes as the slaves. In each iteration, the processes without a parent use time-bounded binary interaction to engage in a parent-child relation. Each iteration takes 2 time units. In $\log_2 m$ iterations, all but one process will have its parent. The one process remains without a parent becomes the master of the network. Formulas (D) in Table 8 is for the corresponding experiment

		Zeno approximate			non-Zeno exact		
bm's	m	no TCXTL	TCXTL	$\frac{\mathsf{T}^c_{pre}}{\mathsf{T}_{pre}}$	no TCXTL	TCXTL	$\frac{\mathsf{T}^c_{pre}}{\mathsf{T}_{pre}}$
(A)	7	0.980s/16.8M	0.672s/10.6M	$\frac{84}{85}$	0.676s/16.8M	0.540s/10.6M	$\frac{84}{85}$
	9	1.88s/56.1M	0.992s/31.6M	$\frac{108}{109}$	2.06s/56.1M	1.10s/31.6M	$\frac{108}{109}$
	11	16.5s/206M	6.42s/100M	$\frac{132}{133}$	16.8s/206M	6.56s/100M	$\frac{132}{133}$
(B)	7	0.720s/9.80M	0.436s/8.70M	$\frac{20}{21}$	0.564s/16.0M	0.532s/13.3M	$\frac{68}{69}$
	9	0.836s/29.8M	0.764s/26.2M	$\frac{26}{27}$	2.55s/51.4M	1.32s/43.0M	$\frac{88}{89}$
	11	5.88s/93.2M	4.58s/81.1M	$\frac{32}{33}$	18.5s/178M	12.6s/149M	$\frac{108}{109}$
(C)	2	0.704s/298k	0.528s/298k	$\frac{55}{56}$	1.90s/321k	1.92s/321M	$\frac{248}{249}$
	4	7.24s/88.2M	4.13s/47.6M	$\frac{314}{315}$	n/a		
	6	n/a	442s/1.73G	$\frac{732}{733}$	n/a		
(D)	5	0.748s/6.46M	0.520s/5.13M	$\frac{93}{94}$	0.696s/9.19M	0.620s/7.28M	$\frac{123}{124}$
	7	5.38s/75.9M	4.22s/59.6M	$\frac{447}{448}$	9.19s/107M	5.98s/86.7M	$\frac{552}{553}$
	9	128s/693M	92.4s/543M	$\frac{1304}{1305}$	176s/883M	149s/753	$\frac{1520}{1521}$

data collected on a Pentium 4 1.7GHz with 2G memory running LINUX;
n/a: not available; s: seconds;
k: kilobytes, M: megabytes, G: gigabytes of memory in diagram data-structure.

Table 9: Time convexities with TCXTL formulas

in Subsection 8.2 while (G) is for 8.3.

8.2 Time convexity with TCXTL formulas

In this subsection, we use benchmarks (A), (B), (C), and (D) to experiment with the techniques of using `TCXTL` path conditions to speed up the evaluation of timed preconditions. These four benchmarks all consist of an $\exists\mathbf{U}$-formula with a `TCXTL` path condition. The four path conditions are:

$$\exists\Diamond_{[19,\infty)}\texttt{critical}_1 \qquad \exists\Box_{[0,10)}\texttt{ready}_1$$
$$\forall\Diamond_{[0,10)}\neg\texttt{transm}_1 \qquad \forall i\in[2,m],\forall\Box_{[m,\infty)}\texttt{parent}_i\neq 0$$

They represent four types of `TCXTL` modalities. The performance data can be found in Table 9. We experimented with the exact and approximate formulations respectively in tables 3 and 4. For each benchmark, we collected performance data for CPU time, memory used for the state-space representations, and the ratios of the number of T^c_{pre} executed over those of T_{pre}. As can be seen, through identifying `TCXTL` path constraints, indeed significant run time can be saved. Specifically, for every benchmark formula in every experiment configuration, only one timed precondition has to be evaluated with T_{pre}. All the other timed preconditions can be done with T^c_{pre}. In fact, according to the result in Subsection 7.1, even this remaining execution of T_{pre} can be replaced with T^c_{pre} if the boundary between the path constraint and the destination constraint of $\rightsquigarrow$ formulation satisfies the properties presented in Subsection 7.1.

8.3 New formulations of timed inevitabilities

We use formulas (E), (F), and (G) in Table 8 for experiment reported in this subsection. We have collected performance data for the exact and approximate formulations in tables 3, 4, 6, and 7 for the evaluation of timed

		Zeno approximate			non-Zeno exact		
bm's	m	Table 4	Table 7 (new)	$\frac{\mathsf{T}^c_{pre}}{\mathsf{T}_{pre}}$	Table 3	Table 6 (new)	$\frac{\mathsf{T}^c_{pre}}{\mathsf{T}_{pre}}$
(E)	3	N/0.568s/4.03M	Y/0.228s/2.85M	$\frac{16}{58}$	Y/0.896s/26.3M	Y/0.892s/15.3M	$\frac{181}{141}$
	4	N/1.38s/34.8M	Y/0.676s/25.4MM	$\frac{32}{94}$	n/a	Y/32.6s/298M	$\frac{673}{n/a}$
	5	N/35.5s/346M	Y/25.2s/285M	$\frac{54}{138}$	n/a		
(F)	3	N/1.13s/669k	Y/0.308s/669M	$\frac{17}{130}$	Y/111s/577M	Y/95.1s/593M	$\frac{818}{898}$
	4	N/6.37s/81.6M	Y/0.476s/1.49M	$\frac{26}{252}$			
	5	N/73.3s/456M	Y/1.73s/36.7M	$\frac{37}{419}$	n/a		
	6	n/a	Y/16.5s/204M	$\frac{50}{n/a}$			
	7	n/a	Y/139s/909M	$\frac{65}{n/a}$			
(G)	6	N/0.568s/1.91M	Y/5.18s/102M	$\frac{61}{0}$	Y/5.73s/87.3M	Y/1.07s/29.8M	$\frac{92}{91}$
	7	N/0.396s/3.85M	Y/80.5s/599M	$\frac{106}{0}$	Y/50.7s/363M	Y/7.61s/114M	$\frac{149}{148}$
	8	N/0.448s/8.27M	n/a	$\frac{n/a}{0}$	Y/357s/1532M	Y/63.0s/436M	$\frac{226}{225}$
	9	N/0.708s/20.5M	n/a	$\frac{n/a}{0}$	n/a	Y/404s/1.80G	$\frac{326}{n/a}$

data collected on a Pentium 4 1.7GHz with 2G memory running LINUX;
n/a: not available; s: seconds;
k: kilobytes, M: megabytes, G: gigabytes of memory in diagram data-structure.

Table 10: Timed inevitability evaluation

inevitabilities. The performance data is reported in Table 10. The answers of model-checking, CPU time used, and the total memory consumption for the state-space representations are reported. For the three benchmark formulas, our new formulations in tables 6 and 7 are able of replacing all T_{pre} execution with T^c_{pre} execution. However, the new formulations may incur more timed precondition evaluations than the traditional ones in tables 3 and 4. Thus we also report the number of T^c_{pre} executed with our new formulations and the number of T_{pre} executed with the traditional formulations. Following the suggestion in [40], we use $k = c_{max}$ for all the configurations. Also we use the early decision techniques for the greatest fixpoint evaluation [40].

In general, our new exact formulations yield better performance than the traditional exact formulations do for timed inevitabilities. As for the approximation techniques, although the traditional approximate formulation in Table 4 sometimes yields good performance, its verification result is always incorrect. In contrast, our new approximate formulation in Table 7 is precise and efficient enough to correctly check the three timed inevitabilities. This may imply that when computing budget is a concern, our new approximate formulation seems a good choice for economic and effective verification configuration.

9 Concluding remarks

There is a high demand for tool support for verifying embedded systems in the industry. For this demand, the verification framework of TCTL model-checking [4, 5] was conceived in the society of theoretical computer sciences and popularly accepted as a solid theoretical foundation for the algorithmic verification of embedded systems. Any progress in algorithm performance of the framework will likely benefit numerous related research areas in embedded system analysis, including scheduling [1, 2], network protocol verification [7, 14, 18, 30], controller synthesis [6, 9, 28], and performance analysis [15]. Certainly serious investigation in TCTL model-checking algorithm improvement toward the verification of practical embedded systems is justified. In this work, we set forth such an investigation by re-examining algorithmic components of TCTL model-checking and rigorously identifying cases where an efficient algorithm for timed preconditions can be soundly used in place of the general one. We wish that our result could encourage our colleagues in the theoretical computer science community to research for algorithm improvement in TA model-checking. In fact, only through more insight and improvement to the related algorithmic components, we can have a chance in fulfilling the promise of TCTL model-checking to the industry.

References

[1] Y. Abdeddaim, E. Asarin, and O. Maler. Scheduling with timed automata. *Theoretical Computer Science*, 354(2), March 2006. A preliminary version appears in proceedings of Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS), 2003.

[2] K. Altisen, G. Gössler, and J. Sifakis. Scheduler modeling based on the controller synthesis paradigm. *Real-time system journal (RTSJ)*, 23:55–84, 2002.

[3] R. Alur, C.Courcoubetis, T. A. Henzinger, and P.-H. Ho. Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems. In *Workshop on Theory of Hybrid Systems*, volume LNCS 736. Springer-Verlag, 1993.

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

[5] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[6] E. Asarin, O. Maler, A. Pnueli, and J. Sifakis. Controller synthesis for timed automata. In *IFAC Symposium on System Structure and Control*, pages 469–474. Elsevier, 1998.

[7] J. Bengtsson, W. Griffioen, K. Kristoffersen, K. Larsen, F. Larsson, P. Pettersson, and W. Y. Verification of an audio protocol with bus collision using uppaal. In *Conference on Computer Aided Verification (CAV)*, volume LNCS 1119. Springer-Verlag, 1996.

[8] J. Bengtsson, K. G. Larsen, F. Larsson, P. Pettersson, and W. Yi. Uppaal - a tool suite for automatic verification of real-time systems. In *Workshop on Tools and Algorithms for the Construction and Analysis of Systems*, volume LNCS 1055, pages 431–434. Springer-Verlag, 27-29 March 1996.

[9] B. Bonakdarpour and S. S. Kulkarni. Automated incremental synthesis of timed automata. In *Formal Methods: Applications and Technology*, volume LNCS 4346, pages 261–276, 2007.

[10] A. Bouajjani, J. C. Fernandez, N. Halbwachs, and P. Raymond. Minimal state graph generation. *Science of Computer Programming*, 18(3):247V269, 1992.

[11] A. Bouajjani, S. Tripakis, and S. Yovine. On-the-fly symbolic model-checking for real-time systems. In *IEEE Real-Time System Symposium (RTSS)*. IEEE Computer Society, 1997.

[12] H. Boucheneb, G. Gardey, and O. H. Roux. TCTL model checking of time Petri nets. *Journal of Logic and Computation*, 19(6):1509–1540, 2009.

[13] P. Bouyer, T. Brihaye, V. Bruyere, and J. francois Raskin. On the optimal reachability problem. *Formal Methods in System Design*, 2006.

[14] C. Daws, M. Kwiatkowska, and G. Norman. Automatic verification of the ieee-1394 root contention protocol with kronos and prism. *Electronic Notes in Theoretical Computer Science*, 66(2), 2002.

[15] L. de Alfaro, M. Kwiatkowska, G. Norman, D. Parker, and R. Segala. Symbolic model checking of probabilistic processes using mtbdds and the kronecker representation. In *Proceedings of the 2000 Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume LNCS 1785, pages 395–410. Springer, March 2000.

[16] J. K. Deka, P. Dasgupta, and P. Chakrabarti. An efficiently checkable subset of TCTL for formal verification of transition systems with delays. In *International Conference on VLSI Design*, page 294. IEEE Computer Society, 1999.

[17] M. Dickhöfer and T. Wilke. The automata-theoretic method works for TCTL model checking. In *International Colloquium on Automata, Langauges, and Programming (ICALP)*, volume LNCS 1443. Springer-Verlag, 1998.

[18] B. Dutertre and M. Sorea. Modeling and verification of a fault-tolerant real-time startup protocol using calendar automata. In *International Conference on Formal Modelling and Analysis of Timed Systems and International Symposium on Formal Techniques in Real-Time and Fault Tolerant Systems (FORMATS/FTRTFT)*. Springer-Verlag, 2004.

[19] R. J. Engdahl and A. M. M. Haugstad. State space reduction for probablistic timed automata. Master's thesis, The faculty of Engineering and Science, Aalborg University, April 2008.

[20] J. B. J. Fourier. In *(reported in:) Analyse des travaux de l'Académie Royale des Sciences pendant l'année 1824*. Partie Mathématique, 1827.

[21] G. Gardey, D. Lime, M. Magnin, and O. H. Roux. Romeo: A tool for analyzing time Petri nets. In *17th Conference on Computer Aided Verification (CAV)*, volume LNCS 3576. Springer-Verlag, 2005.

[22] R. Hadjidja and H. Boucheneb. On-the-fly TCTL model checking for time Petri nets. *Theoretical Computer Science*, 410(42):4241–4261, September 2009.

[23] M. J. Harrold. Testing: A roadmap. In *In The Future of Software Engineering, 22nd International Conference on Software Engineering*, pages 61–72. ACM Press, 2000.

[24] T. A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. *Information and Computation*, 111:193–244, 1994. A preliminary version appeared in the Proceedings of the Seventh Annual Symposium on Logic in Computer Science (LICS), IEEE Computer Society Press, 1992, pp. 394-406.

[25] M. Jurdzinski, M. Z. Kwiatkowska, G. Norman, and A. Trivedi. Concavely-priced probabilistic timed automata. In *CONCUR*, pages 415–430. Springer-Verlag, 2009.

[26] K. G. Larsen, P. Petterson, and W. Yi. Compositional and symbolic model checking of real-time systems. In *IEEE Real-Time Systems Symposium*. IEEE Computer Society, 1995.

[27] N. Markey and P. Schnoebelen. Symbolic model checking of simply-timed systems. Technical report, Lab. Specification and Verification, ENS de Cachan, Cachan, France, October 2003.

[28] S. Panek, O. Stursberg, and S. Engell. Efficient synthesis of production schedules by optimization of timed automata. *Control Engineering Practice*, 14(10):1183–1197, 2006.

[29] W. Penczek, B. Woźna, and A. Zbrzezny. Towards bounded model checking for the universal fragment of TCTL. In *International Symposium on Formal Techniques in Real-Time and Fault Tolerant Systems (FTRTFT)*, volume LNCS 2469. Springer-Verlag, 2002.

[30] N. Petalidis. Verification of a fieldbus scheduling protocol using timed automata. *Computing and Informatics*, 28:655–672, 2009.

[31] O. Sokolsky and S. A. Smolka. Local model checking for real-time systems. In *Conference on Computer Aided Verification (CAV)*, volume LNCS 939. Springer-Verlag, 1995.

[32] R. F. L. Spelberg and W. J. Toetenel. Parametric real-time model checking using splitting trees. *Nordic Journal of Computing*, 8(1):88–120, 2001.

[33] F. Wang. Efficient verification of timed automata with BDD-like data-structures. *International Journal of Software Tools for Technology Transfer (STTT)*, 6(1), 2004. special issue for the 4th International Con-

ference on Verification, Model Checking, and Abstract Interpretation (VMCAI), Jan. 2003, LNCS 2575, Springer-Verlag.

[34] F. Wang. Model-checking distributed real-time systems with states, events, and multiple fairness assumptions. In *International Conference on Algebraic Methodology and Software Technology (AMAST)*, volume LNCS 3116. Springer-Verlag, 2004.

[35] F. Wang. Symbolic parametric safety analysis of linear hybrid systems with BDD-like data-structures. *IEEE Transactions on Software Engineering*, 31(1):38–51, 2005. A preliminary version is in proceedings of 16th Conference on Computer Aided Verification(CAV), 2004, LNCS 3114, Springer-Verlag.

[36] F. Wang. Time-progress evaluation for dense-time automata with concave path conditions. In *Automated Technology for Verification and Analysis (ATVA)*, volume LNCS 5311. Springer-Verlag, 2008.

[37] F. Wang. Symbolic verification of distributed real-time systems with complex synchronizations. In *7th International Conference on Formal Engineering Methods (ICFEM)*, volume LNCS 3785. Springer-Verlag, November 2005.

[38] F. Wang. Symbolic simulation checking of dense-time automata. In *5th International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS)*, volume LNCS 4763. Springer-Verlag, October 2007.

[39] F. Wang and P.-A. Hsiung. Efficient and user-friendly verification. *IEEE Transactions on Computers*, January 2002.

[40] F. Wang, G.-D. Huang, and F. Yu. TCTL inevitability analysis of dense-time systems: From theory to engineering. *IEEE Transactions on Software Engineering*, 32(7), 2006. A preliminary version of the work appears in the proceedings of 8th Conference on Implementation and Application of Automata (CIAA), July 2003, Santa Barbara, CA, USA; LNCS 2759, Springer-Verlag.

[41] F. Wang, L.-W. Yao, and Y.-L. Yang. Efficient verification of distributed real-time systems with broadcasting behaviors. *Real-Time Systems Journal (RTSJ)*, 47(4):285–318, July 2011.

[42] B. Woźna and A. Zbrzezny. Bounded model checking for the existential fragment of TCTL$_{-G}$ and diagonal timed automata. *Fundamenta Informaticae*, 79(1-2):229–256, 2007.

[43] S. Yovine. Kronos: A verification tool for real-time systems. *International Journal of Software Tools for Technology Transfer (STTT)*, 1(1/2), October 1997.