Efficient timed model checking for discrete-time systems

F. Laroussinie, N. Markey and Ph. Schnoebelen

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

Abstract

We consider model checking of timed temporal formulae in *durational transition graphs (DTGs)*, i.e., Kripke structures where transitions have integer durations. Two semantics for DTGs are presented and motivated. We consider timed versions of *CTL* where subscripts put quantitative constraints on the time it takes before a property is satisfied.

We exhibit an important gap between logics where subscripts of the form "$= c$" (exact duration) are allowed, and simpler logics that only allow subscripts of the form "$\leq c$" or "$\geq c$" (bounded duration).

Without exact durations, model checking can be done in polynomial time, but with exact durations, it becomes Δ_2^{P}-complete or PSPACE-complete depending on the considered semantics.

Keywords: Model checking; Timed automata; Timed Transition Graphs; Durational Kripke Structures; Quantitative Temporal Logics.

1 Introduction

Model checking (the automatic verification that a model fulfills temporal logic specifications) is widely used when designing and debugging critical reactive systems [CGP99,BBF$^+$01]. During the last decade, model checking has been extended to *real-time systems*, where quantitative information about timings is required [EMSS92,ACD93,HNSY94,CC99].

Timed models. Real-time model checking has been mostly studied and developed in the framework of Alur and Dill's *Timed Automata* [AD94]. There now exists a large body of theoretical knowledge and practical experience for this class of systems, and it is agreed that their main drawback is the complexity blowup induced by timing constraints: All model checking problems are at least PSPACE-hard[1] over Timed Automata [Alu91,CY92,ACD93,AL02].

However, there exist simpler families of timed models, for which polynomial-time model checking is possible. Usually, these are based on classical, discrete, Kripke Structures (KSs). In this case, there is no inherent concept of time (contrary to clocks in Timed Automata (TA)) and the elapsing of time is encoded by events. For example, each transition of a KS can be viewed as taking exactly one time unit. This simple and natural assumption is used in, e.g., [EMSS92,CCMM95]. A small extension consists in allowing also "instantaneous" transitions, that take zero time unit, as is done in [CTM$^+$99,LST03]. Finally the Timed Transition Graphs (TTGs) [CC95] extends the previous models by associating arbitrary integer durations with transitions.

The TTG framework is less expressive than TAs, but it is conceptually simpler, may allow efficient model checking algorithms, and is convenient in many situations (see examples in [CCMM95,CC01]). Moreover this approach easily lends itself to BDD-based symbolic model checking [CC95,CC99,MS04].

Timed specifications. It is often necessary to verify *real-time* properties over timed systems. Such properties can involve the minimal or maximal delay to reach some particular configuration, or the duration of a given property along a path [CCM$^+$94,CY92]. A flexible approach for specifying these properties is to extend classical temporal logics with the ability to express timing aspects of computation (see [AH92] for a survey). There are two main popular approaches for such extensions: First, the use of *freeze variables* (also *formula clocks*) in temporal formulae allows the comparison of delays between events [AH94]. The resulting logics are very expressive but often have hard model checking problems (because they make it possible to combine the timings of several different events in arbitrary ways); The second approach, which is simpler, is the use of timing constraints tagging temporal modalities [Koy90,ACD90]. For example, the formula $\mathsf{EF}_{<10}\, A$ states that it is possible to reach a state satisfying A ("$\mathsf{EF}\, A$" in *CTL*) in less than 10 time units. These constraints are less expressive than freeze variables but they lead to more readable formulae, and sometimes allow more efficient model checking algorithms.

Timing constraints can have three main forms: "$\leq c$" and "$\geq c$", where c is

[1] Here, and in other places in the article, we make the convenient assumption that there are no collapses between major complexity classes like PTIME, NP, PSPACE, etc.

some integer constant, set a lower or upper bound for durations, while "$=c$" requires a precise value. *TCTL* is the extension of *CTL* with all three types of constraints, and we write $TCTL_{\leq,\geq}$ for the fragment of *TCTL* where the "$=c$" constraints are forbidden. Other classical temporal logics (e.g., CTL^* or *LTL*) can be extended in the same way, and we call $TCTL^*$, $TLTL_{\leq,\geq}$, etc., the resulting formalisms.

Model checking *TCTL* over Kripke structures can be done in time[2] $O(|S|^3 \cdot |\varphi|)$ [EMSS92]. This is in sharp contrast with model checking over Timed Automata (PSPACE-complete [ACD93]) and with model checking *CTL* extended by freeze variables (PSPACE-complete over KSs [LST03]).

Thus it appears that, for timed properties of timed systems, polynomial-time model checking is possible if one picks the right logic (e.g., *TCTL*) and the adequate models (e.g., KSs).

Our contribution. In this article, we aim at defining extensions of KSs for handling real-time aspects in such a way that model checking remains efficient (polynomial-time). We propose and study *durational transition graphs* (DTGs), a very natural extension of KSs. As illustrated in Fig. 1, a DTG is a KS where transitions have possible durations specified by an interval

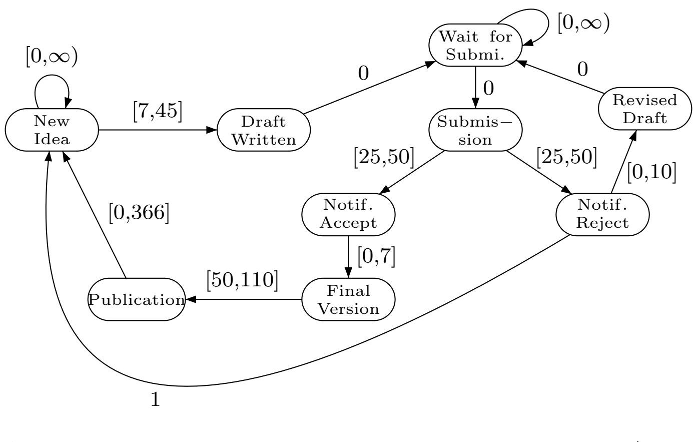


Fig. 1. A DTG modeling publications by one researcher (time in days)

of integers. Such structures generalize the models where every transition is considered as taking 0 or 1 time unit and provide a higher-level viewpoint. For example, steps having long durations can be modeled without long sequences of transitions. Also, the size of a DTG is mostly insensitive to a change of time scale. We study two semantics for DTGs. Indeed time elapsing can be interpreted in different manner: Either transitions are atomic, and time elapses abruptly, all in one step — then the duration of a transition can be seen as a

[2] In such statements, $|S|$ denotes the size of the structure, and $|\varphi|$ the length of the temporal formula.

cost with this "jump" semantics; Or time elapses (semi-)continuously, i.e., we stay in the source state for the duration of the transition before going to the target state, and we call this one the "continuous" semantics.

Our main results are two polynomial-time algorithms for model checking $TCTL_{\leq,\geq}$ properties with respect to both semantics. The algorithm for the "continuous" semantics is much more intricate than the one for the "jump" semantics. This extends the positive results from [EMSS92,LST03] to a more expressive class of models.

Allowing exact duration constraints increases the complexity of model checking: We show that model checking *TCTL* over DTGs is PSPACE-complete or Δ_2^{p}-complete depending on the semantics for DTGs. This last result is technically involved, and it is also quite surprising since Δ_2^{p}, the class P^{NP} of problems that can be solved by a deterministic polynomial-time Turing machine that has access to an NP oracle [Sto76,Pap94], does not contain many natural complete problems [Pap84,Wag87,Kre88]. Indeed, the only known Δ_2^{p}-complete problems from the field of temporal model checking have only been recently identified [LMS01,RS05].

We also consider logics that do not admit polynomial-time model checking algorithms (*TLTL* and $TCTL^*$), and we show that, for these too, exact duration constraints induce a similar complexity blowup when model checking DTGs.

Related work. Quantitative logics for the more expressive TA are now well-known and many results are available regarding their expressive power, or their satisfiability and model checking [AH94,ACD93,AH93,AFH96,Hen98]. That exact durations may induce harder model checking complexity was already observed in the case of *TLTL* and Timed Automata [AFH96].

The literature contains several models that are close to DTGs. Emerson *et al.* give polynomial time algorithms for model checking *TCTL* over discrete KSs in [EMSS92] and $TCTL_{\leq}$ over *tight* DTGs (all intervals are singletons, see section 7.2) with the jump semantics in [ET99, section 4]. They also study model checking for quantitative logics with more complex constraints in [ET97,ET99]. Model checking *TCTL* over *small-steps* DTGs (i.e., with transition durations in $\{0, 1\}$, see section 7.2) is considered in [LST03] where the expressive power of constraints is investigated. Algorithms for maximal and minimal delays and condition counting are given in [CTM+99] for small-steps DTGs.

The Timed Transition Graphs introduced in [CC95] correspond to our DTGs with the jump semantics (see section 2). An algorithm based on BDDs for bounded *TCTL* is given in [CC95], it uses an unfolding of temporal formula with respect to timing constraints which makes its complexity very sensitive to a change of time scale. Algorithms for minimal or maximal delays and for condition counting in TTGs are given in [CCM$^+$94]. DTGs with the jump semantics have also been studied in [LMS02] where complexity of $TCTL$ model checking is addressed, and in [MS04] where an algorithm based on BDDs is given for $TCTL$ model checking (and implemented on top of NuSMV).

[LS05] introduces *probabilistic* DTGs, a model exhibiting both nondeterministic and stochastic behavior, and addresses timed model checking for these systems.

The literature also contains several models based on more expressive discrete-time structures [Lew90,YMW97]. These works do not explicitly look for polynomial-time verification algorithms. Sometimes linear-time logics are considered [Ost90,AH94], but model checking is shown to be at least PSPACE-hard in those cases.

Plan of the article. We first define DTGs (section 2) and the quantitative temporal logic we use (section 3). For $TCTL$ and $TCTL_{\leq,\geq}$, model checking of DTGs assuming the jump semantics is addressed in section 4, and assuming the continuous semantics in section 5. Finally we consider $TLTL$ and $TCTL^*$ in section 6, while other possible semantics are addressed in section 7.

2 Durational Transition Graphs

We write $\mathbb{N}$ for the set of natural numbers, and $\mathcal{I}_{\mathbb{N}}$ (or just $\mathcal{I}$) for the set of intervals over $\mathbb{N}$. An interval $\rho \in \mathcal{I}$ is either finite (of the form "$[n,m]$" with $n \leq m$) or right-open and infinite (of the form "$[n,\infty)$").

Let AP be a countable set $\{P_1, P_2, \ldots\}$ of *atomic propositions.*

Definition 2.1 *A* Durational Transition Graph *(DTG for short) is a 4-tuple $\mathcal{S} = \langle Q, q_{init}, R, l\rangle$ where Q is a set of* states*, $q_{init} \in Q$ is the initial state, $R \subseteq Q \times \mathcal{I} \times Q$ is a total* transition relation with duration *and $l : Q \to 2^{AP}$ labels every state with a subset of AP.*

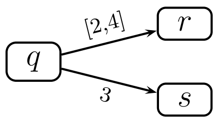
Below we only consider *finite* DTGs, such that Q, R and all $l(q)$ are finite sets. Graphically, a DTG is a directed graph where a triple $(q, \rho, q') \in R$ is depicted as a ρ-labeled edge from q to q'. The interval ρ specifies the possible durations of the transition.

Example 2.2 *The DTG of Fig. 1 models the publication process of one busy researcher, assuming time is counted in days. (This example does not distin-*

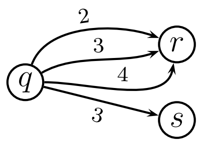
guish between the name of the states and their labeling by propositions. Also, singleton intervals "$[n, n]$" are written simply as "n".)

We consider several natural semantics for DTGs. Indeed the intended meaning of an edge (q, ρ, q') in a DTG is that it is possible to move from q to q' with any *integer* duration d belonging to the interval ρ. This can be interpreted in different manners.

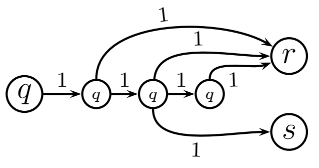
- First we consider the *jump semantics*: moving from q to q' takes d time units and there are no intermediary states. Hence, if the system is in q at time t, then it is in q' at time $t{+}d$; there is no position for time $t{+}1, \ldots, t{+}d{-}1$. This semantics corresponds to the semantics of Timed Transition Graph [CC95].
- Then we consider the *continuous semantics*: the system waits for $d - 1$ time units in state q before performing the transition. This is the semantics used in Timed Automata of Alur and Dill [ACH94] when discrete time is assumed.



A DTG $\mathcal{S}_{ex}$



Behavior of $\mathcal{S}_{ex}$ assuming the jump semantics



Behavior of $\mathcal{S}_{ex}$ assuming the continuous semantics

Fig. 2. Two different semantics for a DTG

Fig. 2 gives an intuitive representation of those two semantics. A third one, called *continuous early*, will be briefly addressed at the end of this article.

Timed Transition Systems. A DTG $\mathcal{S}$ is used as a symbolic description of the behavior of a process. This is formalized by associating a *Timed Transition System* (TTS) with $\mathcal{S}$ (actually, we do this in two different ways, see sections 4 and 5). A TTS is a labeled transitions system with fairness, and where every transition has a fixed integer duration. Formally, a TTS is a 5-tuple $T = \langle S, s_{\text{init}}, \rightarrow, l, F\rangle$ where S is a (possibly infinite) set of states, $s_{\text{init}} \in S$ is the initial state, $\rightarrow \subseteq S \times \mathbb{N} \times S$ is a total transition relation with integer durations, $l\colon S \rightarrow 2^{AP}$ labels every state with a subset of AP and $F \subseteq S$ is a fairness condition. A transition $(s_1, d, s_2) \in \rightarrow$ is denoted by $s_1 \xrightarrow{d} s_2$.

Remark 2.3 *This notion of TTSs is a variant of the one classically used in the semantics of Timed Automata, the main difference being that their TTSs are usually defined over dense time domain and assume special properties like time-determinism and time-additivity.*

A sequence $\pi = s_0 \xrightarrow{d_0} s_1 \xrightarrow{d_1} s_2 \ldots$ of transitions in a TTS is called a *path* if it is finite and a *run* if it is infinite. For a run (resp. path) π, $\pi_{|n}$ is the prefix path obtained by only considering the first n steps in π, and $\pi_{\geq n}$ is the suffix run (resp. path) obtained by removing the first n steps. A *simple* path is a path where no state is visited twice.

Let $\mathsf{Inf}(\pi)$ be the set of states that occur infinitely many times along a run π. We say that π is a *fair run* if $\mathsf{Inf}(\pi) \cap F \neq \varnothing$. For $s \in S$, we let $\mathrm{Exec}^{\mathrm{F}}(s)$ denote the set of fair runs starting from s. Note that for any $n \in \mathbb{N}$, for any run π, π is fair iff $\pi_{\geq n}$ is. Fairness conditions are used in the definition of the continuous semantics (section 5).

The *size* (or *length*) of a path $\pi = s_0 \xrightarrow{d_0} s_1 \xrightarrow{d_1} s_2 \cdots s_n$ is n (the number of steps), and its *duration*, denoted by $\mathsf{Time}(\pi)$, is $d_0 + \cdots + d_{n-1}$.

Size of a DTG. We assume the constants used to denote intervals are encoded in binary. The size of a transition $(q, [l, u], q')$ of a DTG $\mathcal{S}$ is defined as $1 + \lfloor \log(l+1) \rfloor + \lfloor \log(u+1) \rfloor$ and the size of $(q, [l, \infty), q')$ as $1 + \lfloor \log(l+1) \rfloor$. Then the size of $\mathcal{S}$ is defined as its number of states, $|Q|$, plus the sum of the sizes of its transitions.

3 Quantitative temporal logic

TCTL is a quantitative extension of *CTL* where temporal modalities are subscripted with constraints on duration [ACD93]. Here it is interpreted over TTSs states.

Definition 3.1 (Syntax of *TCTL*) *TCTL formulae are given by the following grammar:*

$$\varphi, \psi \ ::= \ P_1 \mid P_2 \mid \ldots \mid \neg\varphi \mid \varphi \wedge \psi \mid \mathsf{EX}\varphi \mid \mathsf{E}\varphi\mathsf{U}_{\sim c}\,\psi \mid \mathsf{A}\varphi\mathsf{U}_{\sim c}\,\psi$$

where $\sim$ can be any comparator in $\{<, \leq, =, \geq, >\}$ and c any natural number.

Definition 3.2 (Semantics of *TCTL*) *The following clauses define when a state s of some TTS $T = \langle S, s_{init}, \rightarrow, l, F\rangle$ satisfies a TCTL formula φ, written*

$s \models_T \varphi$, by induction over the structure of φ (clauses for Boolean operators are omitted).

$$
\begin{array}{lll}
s \models_T \mathsf{EX}\varphi & \textit{iff} & \exists \pi \in \mathit{Exec}^F(s) \textit{ s.t. } \pi = s \xrightarrow{d_0} s_1 \xrightarrow{d_1} s_2 \dots \textit{ and} \\
 & & \quad s_1 \models_T \varphi, \\
s \models_T \mathsf{E}\varphi\mathsf{U}_{\sim c}\,\psi & \textit{iff} & \exists \pi \in \mathit{Exec}^F(s) \textit{ s.t. } \pi = s \xrightarrow{d_0} s_1 \xrightarrow{d_1} s_2 \dots \textit{ and} \\
 & & \quad \exists n \textit{ s.t. } \mathsf{Time}(\pi_{|n}) \sim c, \\
 & & \qquad s_n \models_T \psi \textit{ and} \\
 & & \qquad s_i \models_T \varphi,\ \forall\, 0 \le i < n \textit{ (with } s_0 = s\textit{)}, \\
s \models_T \mathsf{A}\varphi\mathsf{U}_{\sim c}\,\psi & \textit{iff} & \forall \pi \in \mathit{Exec}^F(s) \textit{ s.t. } \pi = s \xrightarrow{d_0} s_1 \xrightarrow{d_1} s_2 \dots, \\
 & & \quad \exists n \textit{ s.t. } \mathsf{Time}(\pi_{|n}) \sim c, \\
 & & \qquad s_n \models_T \psi \textit{ and} \\
 & & \qquad s_i \models_T \varphi,\ \forall\, 0 \le i < n \textit{ (with } s_0 = s\textit{)},
\end{array}
$$

We write $T \models \varphi$ whenever $s_{\mathit{init}} \models_T \varphi$.

Thus, in $\mathsf{E}\varphi\mathsf{U}_{\sim c}\psi$, the classical until is extended by requiring that ψ be satisfied within a duration (from the current state) satisfying the constraint "$\sim c$".

Note that the modality EX deals with a step of the TTS. We will see that, depending on how TTSs are associated with a DTG $\mathcal{S}$, i.e., depending on the semantics of DTGs, such a TTS step may correspond to a delay transition of the DTG, where the control location remains unchanged. We could use another semantics for EX and require that it concerns the action transitions of $\mathcal{S}$: this would not change the complexity results presented in this article, and our algorithms can easily be adapted to handle this case.

Standard abbreviations include $\top$, $\bot$, $\varphi \vee \psi$, $\varphi \Rightarrow \psi$, ... as well as $\mathsf{AX}\varphi$ (for $\neg\mathsf{EX}\neg\varphi$), $\mathsf{EF}_{\sim c}\,\varphi$ (for $\mathsf{E}\top\mathsf{U}_{\sim c}\,\varphi$), $\mathsf{AF}_{\sim c}\,\varphi$ (for $\mathsf{A}\top\mathsf{U}_{\sim c}\,\varphi$), $\mathsf{EG}_{\sim c}\,\varphi$ (for $\neg\mathsf{AF}_{\sim c}\,\neg\varphi$) and $\mathsf{AG}_{\sim c}\,\varphi$ (for $\neg\mathsf{EF}_{\sim c}\,\neg\varphi$). Further, the modalities U, F and G without subscripts are shorthand for $\mathsf{U}_{\geq 0}$, etc. The size $|\varphi|$ of a formula φ is defined in the standard way, with constants c written in binary notation.

Equivalence between formulae. We write $\varphi \equiv \psi$ when φ and ψ are *equivalent* (i.e., when every state of every TTS satisfies $\varphi \Leftrightarrow \psi$). The following equivalences hold:

$$\mathsf{A}\,\varphi\,\mathsf{U}_{\leq c}\,\psi \equiv \mathsf{AF}_{\leq c}\,\psi \ \wedge\ \neg\mathsf{E}(\neg\psi)\mathsf{U}(\neg\varphi \wedge \neg\psi) \tag{E1}$$

$$\mathsf{A}\,\varphi\,\mathsf{U}_{\geq c}\,\psi \equiv \mathsf{AG}_{< c}\left(\varphi \ \wedge\ \mathsf{A}\,\varphi\,\mathsf{U}_{>0}\,\psi\right) \qquad \text{if } c > 0 \tag{E2}$$

The proof of Equivalence (E1) is the following:

- $\Rightarrow$: Assume $s \models \mathsf{A}\varphi\mathsf{U}_{\leq c}\,\psi$. First $\mathsf{AF}_{\leq c}\,\psi$ holds clearly for s. Moreover we also have $s \models \mathsf{A}\varphi\mathsf{U}\psi$ and the classical *CTL* equivalence $\mathsf{A}\varphi\mathsf{U}\psi \equiv \mathsf{AF}\psi \wedge \neg(\mathsf{E}(\neg\psi)\mathsf{U}(\neg\varphi \wedge \neg\psi))$ entails $s \models \neg(\mathsf{E}(\neg\psi)\mathsf{U}(\neg\varphi \wedge \neg\psi))$.

- $\Leftarrow$: $s \models \neg(\mathsf{E}(\neg\psi)\mathsf{U}(\neg\varphi\wedge\neg\psi))$ means that there is no run from s along which $\neg\varphi$ precedes ψ, this entails that every run from s satisfies either $\varphi\mathsf{U}\psi$ or $\mathsf{G}\neg\psi$. Then if s also satisfies $\mathsf{AF}_{\leq c}\,\psi$, we have $s \models \mathsf{A}\varphi\mathsf{U}_{\leq c}\,\psi$.

For Equivalence (E2), we can argue as follows:

- $\Rightarrow$: Assume that $s \models \mathsf{A}\ \varphi\ \mathsf{U}_{\geq c}\,\psi$, and consider $\pi = s_0(= s) \xrightarrow{d_0} s_1 \xrightarrow{d_1} \ldots \in \mathrm{Exec}^{\mathrm{F}}(s)$ and n s.t. $\mathsf{Time}(\pi_{|n}) < c$. Then clearly φ holds for any s_i with $0 \leq i \leq n$. Moreover for any fair run σ from s_i, $\pi_{|i} \cdot \sigma \in \mathrm{Exec}^{\mathrm{F}}(s)$ and then there exists a state $s_{i,\sigma}$ satisfying ψ s.t. $\mathsf{Time}(s_0 \xrightarrow{d_0} \ldots \xrightarrow{d} s_{i,\sigma}\ ) \geq c$ and then $\mathsf{Time}(s_i \xrightarrow{d_i} \ldots \xrightarrow{d} s_{i,\sigma}\ ) > 0$ for any $0 \leq i \leq n$ and any state between s_i and $s_{i,\sigma}$ satisfies φ. This gives the result.
- $\Leftarrow$: Assume that $s \models \mathsf{AG}_{<c}\,(\varphi\ \wedge\ \mathsf{A}\ \varphi\ \mathsf{U}_{>0}\,\psi)$, and consider $\pi = s_0(= s) \xrightarrow{d_0} s_1 \xrightarrow{d_1} \ldots \in \mathrm{Exec}^{\mathrm{F}}(s)$. Consider the minimal n s.t. $\mathsf{Time}(\pi_{|n}) \geq c$ (such a n exists because any state s_i with $\mathsf{Time}(\pi_{|i}) < c$ satisfies $\mathsf{A}\varphi\mathsf{U}_{>0}\,\psi$ and then there is some $j > i$ with $\mathsf{Time}(s_i \xrightarrow{d_i} \ldots \xrightarrow{d_{j-1}} s_j\ _|) > 0$). Moreover we have $n > 0$. For any $0 \leq i < n$, $\mathsf{Time}(\pi_{|i}) < c$ and we have $s_i \models \varphi \wedge \mathsf{A}\varphi\mathsf{U}_{>0}\,\psi$. Then $s_{n-1} \models \mathsf{A}\varphi\mathsf{U}_{>0}\,\psi$ and there exists $j \geq n$ s.t. $s_j \models \psi$ and $\forall n < l < j$, $s_l \models \varphi$ and $\mathsf{Time}(\pi_{|j}) \geq \mathsf{Time}(\pi_{|n}) \geq c$.

The rest of the article formally defines how a TTS $T(\mathcal{S})$ is associated with a DTG $\mathcal{S}$ and considers the *model checking problem*: Given a DTG $\mathcal{S}$ and a *TCTL* formula φ, does $T(\mathcal{S}) \models \varphi$? We consider several possibilities for defining $T(\mathcal{S})$, starting with the jump semantics.

4 The *jump* semantics

4.1 Definition

Let $\mathcal{S} = \langle Q, q_{\mathrm{init}}, R, l\rangle$ be a DTG. The *jump* semantics of $\mathcal{S}$ is defined as the TTS $T_j(\mathcal{S}) = \langle S, s_{\mathrm{init}}, \rightarrow, l, F\rangle$ with:

- $S = Q$ and $s_{\mathrm{init}} = q_{\mathrm{init}}$;
- $s_1 \xrightarrow{d} s_2$ iff there exists $(s_1, \rho, s_2) \in R$ and $d \in \rho$;
- $F = Q$.

Observe that any state $s \in S$ is labeled as it is in $\mathcal{S}$. For any formula φ, we write $\mathcal{S} \models_j \varphi$ iff $T_j(\mathcal{S}) \models \varphi$.

$T_j(\cdots)$ is the most basic semantics for DTGs. Indeed the only difference between $\mathcal{S}$ and $T_j(\mathcal{S})$ is that a transition labeled by some interval ρ in R has been replaced by a (possibly infinite) set of transitions corresponding to all durations in ρ. Any run is a fair run. This semantics makes the DTGs equivalent to the Timed Transition Graphs of [CC95].

4.2 Model checking DTGs with the jump semantics

In DTGs where durations belong to $\{0,1\}$, model checking can be done in polynomial time [EMSS92,LST03]. But when dealing with arbitrary durations, a complexity blow-up occurs and NP-hard problems appear for simple formulae and many variants of weighted graphs [NU02]. Indeed we have:

Proposition 4.1 *Model checking formulae of the form* $\mathsf{EF}_{=c}\, P$ *over DTGs with the jump semantics is* NP*-hard.*

PROOF. By reduction from SUBSET-SUM [GJ79, p. 223]: An instance is a finite set $A = \{a_1, \dots, a_n\}$ of natural numbers and some number D. One asks whether there exists a subset A' of A such that $D = \sum_{a \in A'} a$. This is the case iff $\mathcal{S} \models_j \mathsf{EF}_{=D}\, P$ where $\mathcal{S}$ is the DTG depicted on Figure 3. $\square$

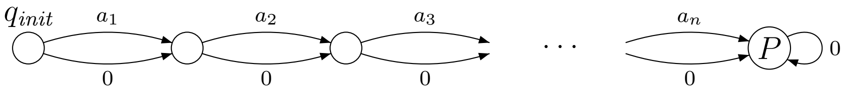


Fig. 3. The DTG associated with an instance of SUBSET-SUM

Therefore model checking *TCTL* over DTGs is NP-hard and coNP-hard for the jump semantics. In fact model checking the fragment of *TCTL* using only the $\mathsf{EF}_{\sim c}$ and $\mathsf{AF}_{\sim c}$ modalities, is already Θ_2^{p}-complete for tight DTGs [Sch03].

For *TCTL*, we have:

Theorem 4.2 *Model checking TCTL over DTGs with the jump semantics is* Δ_2^{p}*-complete.*

See appendix A for the proof. Recall that Δ_2^{p} (resp. Θ_2^{p}) is the complexity class P^{NP} (resp. $\mathsf{P}^{\mathsf{NP}[O(\log n)]}$) of problems that can be solved by a polynomial time Turing machine having access to an NP oracle (resp. and making $O(\log n)$ adaptive queries to the oracle) [Pap94]. Both classes lie between NP and PSPACE. *TCTL* model checking over DTGs with the jump semantics is the second verification problem shown to be complete for Δ_2^{p}.

The hardness part of Theorem 4.2 crucially relies on exact duration constraints. Without them, polynomial-time model checking is possible:

Theorem 4.3 *Verifying whether $T_j(\mathcal{S}) \models \Phi$, for $\mathcal{S}$ a DTG and Φ a $TCTL_{\leq,\geq}$ formula, can be done in time $O(|\mathcal{S}|^2 \cdot |\Phi|)$.*

PROOF. Let $\mathcal{S} = \langle Q, q_{\text{init}}, R, l\rangle$ be a DTG. We extend the standard *CTL* model checking algorithm with labeling procedures running in time $O(|\mathcal{S}|^2 \cdot \lceil \log c \rceil)$ for subformulas of the form $\mathsf{E}\,\varphi\,\mathsf{U}_{\sim c}\,\psi$ and $\mathsf{A}\,\varphi\,\mathsf{U}_{\sim c}\,\psi$.

- $\xi = \mathsf{E}\,\varphi\,\mathsf{U}_{\leq c}\,\psi$: We restrict to the subgraph where only states satisfying $\mathsf{E}\,\varphi\,\mathsf{U}\,\psi$ have been kept, and where we only consider the minimal duration on every transition. Then for every state q we compute the duration c_q of the shortest path leading to some ψ-state. This can be done in time $O(|Q| \cdot |R|)$ using a classical *single-source shortest path* algorithm [CLR90]. Then $q \models \xi$ iff $c_q \leq c$.
- $\xi = \mathsf{E}\,\varphi\,\mathsf{U}_{\geq c}\,\psi$: First we introduce a new proposition $P_{\mathsf{SCC}^+(\varphi)}$ to label every node belonging to a strongly connected set of nodes satisfying φ and where at least one edge allows a strictly positive duration. Labeling states for $P_{\mathsf{SCC}^+(\varphi)}$ can be done in time $O(|\mathcal{S}|)$.

  There are two ways a state can satisfy ξ. Either a simple path is enough, or a path with loops is required so that a long enough duration is reached. We check the existence of a path of the first kind with a variant of the earlier shortest paths method, this times geared towards *longest acyclic paths*. We check for the existence of a path of the second kind by verifying the *CTL* formula $\mathsf{E}\,\varphi\mathsf{U}(P_{\mathsf{SCC}^+(\varphi)} \wedge \mathsf{E}\,\varphi\,\mathsf{U}\,\psi)$. This provides an algorithm running in time $O(|Q| \cdot |R|)$.
- $\xi = \mathsf{A}\,\varphi\,\mathsf{U}_{\leq c}\,\psi$: We first label with a new atomic proposition $P_{\mathsf{SCC}^0(\neg\psi)}$ the states of strongly connected components where one can loop on $\neg\psi$-states using transitions allowing zero durations. We then reduce to the previous cases using equivalence (E1) and $\mathsf{AF}_{\leq c}\psi \equiv \neg\mathsf{E}\neg\psi\,\mathsf{U}_{>c}\top \wedge \neg\mathsf{E}\neg\psi\mathsf{U}\,P_{\mathsf{SCC}^0(\neg\psi)}$.
- $\xi = \mathsf{A}\,\varphi\,\mathsf{U}_{\geq c}\,\psi$: We reduce to the previous cases using equivalence (E2) and $\mathsf{AG}_{<c}\Psi \equiv \neg\mathsf{EF}_{<c}\neg\Psi$. A procedure for the subformula $\mathsf{A}\varphi\mathsf{U}_{>0}\,\psi$ can be easily defined. □

This algorithm for $TCTL_{\leq,\geq}$ is then a simple extension of the one for *CTL* with shortest path procedures. From *CTL* we also inherit a lower bound for complexity: model checking $TCTL_{\leq,\geq}$ is PTIME-complete.

5 The *continuous* semantics

5.1 *Definition*

Given a state q of $\mathcal{S}$, we define $\delta_{\max}(q) \in \mathbb{N} \cup \{\infty\}$ as the upper bound of the intervals labeling outgoing transitions from q. Formally, $\delta_{\max}(q) = \infty$ if there exists an outgoing transition (q, ρ, q') with $\rho = [l, \infty)$, and otherwise $\delta_{\max}(q)$ is $\max\{u \mid (q, [l, u], q') \in R\}$. The *continuous* semantics of $\mathcal{S}$ is defined as the (possibly infinite) TTS $T_c(\mathcal{S}) = \langle S, s_{\text{init}}, \rightarrow, l, F\rangle$ with:

- $S = \{(q, i) \mid q \in Q \text{ and } 0 < i < \delta_{\max}(q)\} \cup \{(q, 0) \mid q \in Q\}$ and $s_{\text{init}} = (q_{\text{init}}, 0)$;
- The transition relation $\rightarrow$ is defined as follows:
  - · *action* transitions:
    - – $(q, 0) \xrightarrow{0} (q', 0)$ if $\exists (q, \rho, q') \in R$ and $0 \in \rho$;
    - – $(q, i) \xrightarrow{1} (q', 0)$ if $\exists (q, \rho, q') \in R$ and $i + 1 \in \rho$;
  - · *delay* transitions:
    - – $(q, i) \xrightarrow{1} (q, i + 1)$ if $i + 1 < \delta_{\max}(q)$;
- The states (q, i) are labeled by the atomic propositions labeling q;
- $F = Q \times \{0\}$.

The delay transitions let time elapse in the current state, and the fairness condition forbids waiting forever in a state: An action transition has to be taken eventually.

Note that this semantics allows for defining parallel composition of the underlying TTS. Indeed, in this case, the behavior of a synchronized product of DTGs consists in synchronizing several TTSs where transitions have durations 0 or 1. But from the complexity point of view, parallel composition entails a blow-up for model checking: Verification of parallel compositions of KSs or TA or DTGs, has the same complexity [AL02].

The continuous semantics is inspired from the semantics of TAs. Indeed, a DTG with the continuous semantics can be seen as a timed automaton with a single clock, with $\mathbb{N}$ as underlying time domain, and where the clock is only used to time transitions and is reset after each move. Model checking TAs with one clock has been studied in [LMS04] where it is shown that it admits the same complexity as model checking DTGs (but the algorithms for one-clock TAs are trickier).

Remark 5.1 *From any state of $T_c(\mathcal{S})$ there exists at least one fair run. This is based on the fact that R is left-total and on the definition of T_c (the set of states, the relation $\rightarrow$ and the fair condition). As a consequence, any path can be extended to a fair run.*

We observe that the *continuous* semantics is not equivalent to the *jump* semantics on two grounds: it makes nondeterministic choices "later" and has more intermediary states (a finer granularity).

Hence, if one considers the following (untimed) formula:

$$\Psi \overset{\text{def}}{=} \mathsf{EF}(q \wedge \neg \mathsf{EF}\, s)$$

then $T_c(\mathcal{S}) \models \Psi$ and $T_j(\mathcal{S}) \not\models \Psi$ for the DTG $\mathcal{S}$ displayed on Fig. 2.

See appendix C for a comparison between the continuous semantics and the jump semantics.

5.2 Model checking DTGs with the continuous semantics

NP-hardness (Prop. 4.1) also holds for the continuous semantics, and here again, there is no hope for efficient model checking algorithm with exact durations. The problem is even harder (assuming PSPACE differs from Δ_2^{P}):

Theorem 5.2 *Model checking TCTL over DTGs with the continuous semantics is* PSPACE*-complete.*

Appendix B contains the proof of this statement. In fact, the proof only involves $\mathsf{EF}_{\sim c}$- and $\mathsf{AF}_{\sim c}$-modalities, and the complexity result also holds for the logic $\mathrm{B}(\mathsf{F})$.

Here again, if we restrict to $TCTL_{\leq,\geq}$, we can have an efficient algorithm for model checking:

Theorem 5.3 *Verifying whether $T_c(\mathcal{S}) \models \Phi$, for $\mathcal{S}$ a DTG and Φ a $TCTL_{\leq,\geq}$ formula, can be done in time $O(|\mathcal{S}|^3 \cdot |\Phi|^3)$.*

PROOF.

Assume $\mathcal{S} = \langle Q, q_{\text{init}}, R, l\rangle$. Let $T_{\mathcal{S}}$ be $T_c(\mathcal{S})$. We design an algorithm for labeling every state (q,i) of $T_{\mathcal{S}}$ with the set of subformulae of Φ it satisfies: Given a state q and a subformula φ of Φ, we define $\mathsf{Sat}[q,\varphi]$ as the sequence of integer intervals $S_j = [\alpha_j, \beta_j)$ such that:

- Any $T_{\mathcal{S}}$ state (q,i) satisfies φ **iff** $i \in \bigcup_j S_j$.
- For any S_j, we have
  - $[\alpha_j, \beta_j) \subseteq [0, \delta_{\max}(q))$,
  - $\alpha_j < \beta_j$, and

· $\beta_j < \alpha_{j+1}$ if $\mathsf{Sat}[q, \varphi]$ contains at least $j + 1$ items.

For any q and φ, this clearly defines a unique set $\mathsf{Sat}[q, \varphi]$. Its number of intervals in $\mathsf{Sat}[q, \varphi]$ is the *size* of $\mathsf{Sat}[q, \varphi]$ (denoted by $|\mathsf{Sat}[q, \varphi]|$).

In the sequel, we write $\mathsf{Sat}[q, \varphi]$ for the union $\bigcup_j S_j$.

We define procedures for inductively computing $\mathsf{Sat}[q, \xi]$ for all subformulas of a given $TCTL_{\leq,\geq}$ formula Φ and for all states $q \in Q$. Along with these procedures, we show that

- $|\mathsf{Sat}[q, \xi]|$ is finite and bounded by $|\xi| \cdot |R^q_{\mathcal{S}}|$, where $R^q_{\mathcal{S}}$ is the set of $\mathcal{S}$-transitions from q, and
- The $\mathsf{Sat}[q, \xi]$ (for all states q and a given ξ) can be computed in time $O(|\xi|^2 \cdot |R|^3)$.

This will globally ensure that the whole algorithm runs in time $O(|\Phi|^3 \cdot |R|^3)$.

Before going further, we introduce some new notations: For a given integer interval $\rho = [l, u)$, we write $\rho - 1$ for the interval $[\max(0, l - 1), \infty)$ if $u = \infty$, and $[\max(0, l - 1), u - 1)$ otherwise. We also define $\overleftarrow{\rho}$ as the interval ρ itself if it equals the singleton $[0, 1)$, and as $\rho - 1$ otherwise.

We now describe our procedures and prove the above statements. The cases of atomic propositions and Boolean connectives are straightforward and clearly satisfy the requirements w.r.t. the size of $\mathsf{Sat}[q, \varphi]$. We now consider the remaining cases:

- Case $\xi = \mathsf{EX}\psi$: We have to deal with the two kinds of transitions[3]:
  · For action transitions: Given a transition $(q, \rho, q') \in R$, if $0 \in \mathsf{Sat}[q', \psi]$, then we add $\overleftarrow{\rho}$ to $\mathsf{Sat}[q, \xi]$;
  · For delay transitions: For any $\rho \in \mathsf{Sat}[q, \psi]$, we add $\rho - 1$ to $\mathsf{Sat}[q, \xi]$.
- Case $\xi = \mathsf{E}\varphi\mathsf{U}_{\leq c}\,\psi$: For each state (q, i) in the TTS $T_c(\mathcal{S})$, we compute the duration of the shortest path (if any) witnessing the property $\mathsf{E}\varphi\mathsf{U}\psi$, and compare it to c. That path will necessary be a prefix of a fair run, thus fairness is not an issue here.

  For each state q in Q, assuming $\mathsf{Sat}[q, \varphi]$ and $\mathsf{Sat}[q, \psi]$ have already been computed, we first refine these intervals by computing the smallest list of intervals $L(q) = \bigcup_{j=1..l}[a_j, b_j)$ s.t.:

(1) For any j, $a_j < b_j$, and $b_j \leq a_{j+1}$ if $j + 1 \leq l$;
(2) For any i, we have $i \in L(q) \;\Leftrightarrow\; i \in \mathsf{Sat}[q, \varphi] \cup \mathsf{Sat}[q, \psi]$, and each interval $[a_j, b_j)$ is either included in one of $\mathsf{Sat}[q, \psi]$'s intervals, or disjoint with $\mathsf{Sat}[q, \psi]$.

[3] Remember that the EX-operator deals with any (action- and delay-) transition of the TTS.

(3) Intervals in $L(q)$ are *homogeneous* w.r.t. action transitions: For any transition $(q, \rho, q') \in R$, for any j, either $[a_j, b_j) \subseteq \overleftarrow{\rho}$ or $[a_j, b_j) \cap \overleftarrow{\rho} = \varnothing$.

(4) The special interval $[0, 1)$ is handled separately: If $0 \in \mathsf{Sat}[q, \varphi] \cup \mathsf{Sat}[q, \psi]$, then it is the first interval in $L(q)$.

Building $L(q)$ is easy from $\mathsf{Sat}[q, \varphi]$ and $\mathsf{Sat}[q, \psi]$: Computing the special union of condition 2 yields at most $|\mathsf{Sat}[q, \varphi]| + 2|\mathsf{Sat}[q, \psi]|$ intervals. Then, by condition 3, any transition (q, ρ, q') might split one of these intervals into two or three smaller ones, i.e., add two intervals. Last, condition 4 possibly adds another one. Thus $|L(q)| \leq |\mathsf{Sat}[q, \varphi]| + 2|\mathsf{Sat}[q, \psi]| + 2|R^q_{\mathcal{S}}| + 1$.

Let $\delta^{\psi}_{q,i}$ be the duration of the shortest paths satisfying φ and leading to some ψ-state. Clearly $(q, i) \models \xi$ iff $\delta^{\psi}_{q,i} \leq c$. Let $[a, b)$ be an interval in $L(q)$. Since any point in $[a, b)$ may fire the same set of action transitions, the function $i \mapsto \delta^{\psi}_{q,i}$ is non increasing over $[a, b)$: any execution starting by an action transition (leading to some $(q', 0)$) enabled from (q, i) is also enabled from $(q, i + 1)$ if $i, i + 1 \in [a, b)$. Figure 4 describes an example of such duration function.

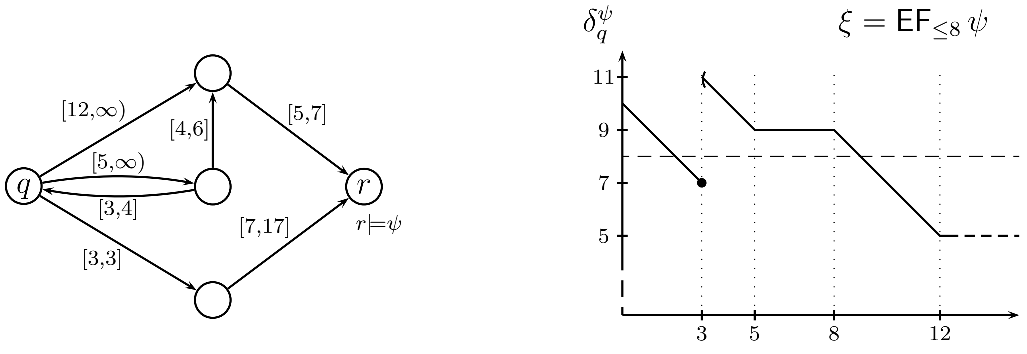


Fig. 4. An example of the duration function for a simple DKS

We have the following important properties:

- Assume that $\delta^{\psi}_{q,a}$ is known for every left-end point a of the intervals in $L(q)$, then it is possible to deduce easily $\delta^{\psi}_{q,i}$ for any $i \in L(q)$. Indeed, for $[a, b)$ in $L(q)$, we have:
  - Either there is an interval in $L(q)$ of the form $[b, b')$. Then for any position $i \in [a, b)$, a shortest path leading to ψ may start either by an action transition — and then $\delta^{\psi}_{q,i} = \delta^{\psi}_{q,a}$ — or by letting time elapse until the interval $[b, b')$ — and then $\delta^{\psi}_{q,i} = \delta^{\psi}_{q,b} - b - i$.
  - Or there is no interval $[b, b')$ in $L(q)$. Then for any $i \in [a, b)$, we have $\delta^{\psi}_{q,i} = \delta^{\psi}_{q,a}$, since in that case, the shortest path necessarily begins with an action transition.
- A shortest path from some (q, a) with $[a, b) \in L(q)$ starts by an action transition or by a delay transition of at least $b - a$ time units: It is never pertinent to wait before performing an enabled action transition when considering shortest paths. Time elapsing only occurs when it is necessary to reach the next interval of $L(q)$.

Therefore it is sufficient to compute the duration of shortest paths from the left-end point of any interval of $L(q)$, and we can consider a jump-semantics point of view restricted to left-end points: The intermediary states (inside the intervals) are not relevant for this. Consider the DTG $G = (V_G, \rightarrow_G, l_G)$ as follows:

· $V_G = \{(q, [a, b)) \mid [a, b) \in L(q)\}$;

· $l_G \colon V_G \rightarrow \{\psi, \varphi \wedge \neg\psi\}$ labels each state (q, ρ) depending on whether $\rho \subseteq \mathsf{Sat}[q, \psi]$;

· Transitions $\rightarrow_G$ are computed as follows:

  – Consider $(q, \rho, q') \in R$ s.t. $[0, 1) \in L(q')$. We have: $(q, [a, b)) \xrightarrow{1}_G (q', [0, 1))$ whenever $[a, b) \in L(q)$ and $a + 1 \in \rho$. Moreover we have $(q, [0, 1)) \xrightarrow{0}_G (q', [0, 1))$ whenever $[0, 1) \in L(q)$ and $0 \in \rho$.

  – If $[a, b), [b, b') \in L(q)$, then we have $(q, [a, b)) \xrightarrow{b-a}_G (q, [b, b'))$.

Then we have: $|G| \stackrel{\text{def}}{=} |V_G| + |\rightarrow_G| \leq \sum_{q \in Q} |L(q)| + \sum_{q \in Q} |L(q)| \cdot (|R| + 1)$. Now we can adapt the procedure described for Theorem 4.3 to get the duration of shortest paths leading to ψ for any G state $(q, [a, b))$, and it corresponds precisely to $\delta^{\psi}_{q,a}$. This can be achieved in time $O(|V_G| \cdot |\rightarrow_G|)$.

Now it remains to compute $\mathsf{Sat}[q, \mathsf{E}\varphi\mathsf{U}_{\leq c}\,\psi]$ from $\delta^{\psi}_{q,a}$ and c. If $\delta^{\psi}_{q,a} \leq c$, we have $[a, b) \subseteq \mathsf{Sat}[q, \xi]$. Otherwise if, for some b', $[b, b') \in L(q)$ and $\delta^{\psi}_{q,b} \leq c$, then $[b - (c - \delta^{\psi}_{q,b}), b) \subseteq \mathsf{Sat}[q, \xi]$. Then we merge the intervals in $\mathsf{Sat}[q, \xi]$ in order to fulfill its requirements.

The size of $\mathsf{Sat}[q, \xi]$ can be bounded by $|\mathsf{Sat}[q, \psi]| + |\mathsf{Sat}[q, \varphi]| + |R^q_{\mathcal{S}}|$. Indeed, $\mathsf{Sat}[q, \mathsf{E}\varphi\mathsf{U}\psi]$ contains at most $|\mathsf{Sat}[q, \psi]| + |\mathsf{Sat}[q, \varphi]|$ intervals. Now, as explained above, we may have to split these intervals depending on the length of the shortest path. Two cases may arise:

· the splitting occurs while the length of the shortest path is decreasing (and thus becomes smaller than c). This case occurs when we are waiting for a transition to be enabled, i.e., it is bound to a constraint $x \geq i$. Thus one transition contains at most one such constraint, and thus may give rise to at most one such splitting;

· the splitting occurs at a point where the shortest path is increasing, i.e., the shortest path is longer than c after that splitting. This may only happen when a transition becomes disabled, that is, it is bound to a constraint $x \leq i$. Here again, one transition may give rise to at most one such splitting.

Thus one transition (q, ρ, q') may at most add one interval in $\mathsf{Sat}[q, \xi]$. Finally, we get $|\mathsf{Sat}[q, \xi]| \leq |\mathsf{Sat}[q, \psi]| + |\mathsf{Sat}[q, \varphi]| + |R^q_{\mathcal{S}}|$.

- Case $\xi = \mathsf{E}\varphi\mathsf{U}_{\geq c}\,\psi$: We assume $c > 0$ — the case $c = 0$ corresponds to the standard *CTL* modality. We use similar techniques as in the previous case. Now in $L(q)$ we distinguish the sub-intervals satisfying $\varphi \wedge \neg\psi$, $\varphi \wedge \psi$ or $\neg\varphi \wedge \psi$. Moreover we replace every interval $[a, b)$ labeled by $\neg\varphi \wedge \psi$ with $[a, a + 1)$ because only the point a may witness ξ. We have $|L(q)| \leq 2 \cdot (|\mathsf{Sat}[q, \varphi]| + |\mathsf{Sat}[q, \psi]| + |R^q_{\mathcal{S}}|)$. We also build a DTG $G = (V_G, \rightarrow_G, l_G)$

with $V_G = \{(q,[a,b)) \mid [a,b) \in L(q)\}$ and $l_G\colon V_G \to \{\varphi\wedge\psi, \varphi\wedge\neg\psi, \neg\varphi\wedge\psi\}$. But now we look for maximal durations $\Delta^{\psi}_{q,a}$ to reach ψ and we distinguish finite intervals and unbounded intervals:

· For finite intervals in $L(q)$, we only consider the right-end points because as soon as a long path goes through the interval $[a,b)$ with $b<\infty$, it goes through the point $b-1$. And we have $\Delta^{\psi}_{q,i} = \Delta^{\psi}_{q,b-1} + b - 1 - i$ for any $i \in [a,b)$.

· For unbounded interval $[a,\infty)$ in $L(q)$, we have $\Delta^{\psi}_{q,i} = \Delta^{\psi}_{q,j}$ for any $i,j \in [a,\infty)$ — and then $(q,i) \models \xi$ iff $(q,j) \models \xi$ — therefore we can restrict ourself to look for the truth value of ξ in the point a.

We then define the transitions of G in order to represent these right-end points of finite intervals and the left-end point of unbounded intervals; the aim is to use the algorithm defined for the jump semantics to compute the maximal durations. We define $\to_G$ as follows:

· Consider $(q,\rho,q') \in R$ s.t. $[0,1) \in L(q')$. We have: $(q,[a,b)) \xrightarrow{1}_G (q',[0,1))$ whenever $[a,b) \in L(q)$ and $[a,b) \subseteq \rho$. Moreover we have $(q,[0,1)) \xrightarrow{0}_G (q',[0,1))$ whenever $[0,1) \in L(q)$ and $\rho = [0,0]$.

· For any $[a,b), [a',b')$ in $L(q)$ s.t. $b = a'$, we have $(q,[a,b)) \xrightarrow{b'-b}_G (q,[b,b'))$ (resp. $(q,[a,b)) \xrightarrow{1}_G (q,[b,\infty))$) if $b' < \infty$ (resp. $b' = \infty$).

· If $[a,\infty) \in L(q)$, we have $(q,[a,\infty) \xrightarrow{1}_G (q,[a,\infty))$.

A state $(q,[a,b))$ with $b<\infty$ of G stands for the state $(q,b-1)$ in $\mathcal{S}$ while a state $(q,[a,\infty))$ in G stands for (q,a) in $\mathcal{S}$. The third kind of transition is used to represent time elapsing in unbounded intervals.

Note that a G transition $(q,[a,b)) \xrightarrow{1} (q,[b,b'))$ with $b' < \infty$ represents the path $(q,b-1) \xrightarrow{1} (q,b) \xrightarrow{1} \ldots (q,b'-1)$ in $T_{\mathcal{S}}$. Then the labeling of intermediary states is given by the target node (contrary to the case where nodes correspond to the left-end points), but this does not matter for $\mathsf{E}\varphi\mathsf{U}\psi$ modality because these intermediary states exist iff $b' > b+1$ and this entails $(q,[b,b')) \subseteq \mathsf{Sat}[q,\varphi]$.

The procedure for the jump semantics of Theorem 4.3 can be used and we assume that it returns maximal durations for G states, and ∞ (resp. $-\infty$) is used when the longest path until ψ is arbitrary long (resp. there is no path reaching ψ). The algorithm runs in time $O(|V_G| \cdot |{\to_G}|)$.

It remains to merge contiguous intervals in order to get $\mathsf{Sat}[q,\xi]$. As in the previous case, we end up with at most $|\mathsf{Sat}[q,\psi]| + |\mathsf{Sat}[q,\varphi]| + |R^q_{\mathcal{S}}|$ intervals.

- Case $\xi = \mathsf{A}\varphi\mathsf{U}_{\leq c}\,\psi$: We reduce to the previous cases using equivalence (E1) and $\mathsf{AF}_{\leq c}\,\psi \equiv \neg\mathsf{E}\neg\psi\mathsf{U}_{>c}\top \wedge \neg\mathsf{E}\neg\psi\mathsf{U}\,P_{\mathsf{SCC}^0(\neg\psi)}$. Here $P_{\mathsf{SCC}^0(\neg\psi)}$ labels strongly connected components where one can loop on $\neg\psi$-states using only transitions allowing zero durations. Note that runs staying in $P_{\mathsf{SCC}^0(\neg\psi)}$ will necessarily be fair.
- Case $\xi = \mathsf{A}\varphi\mathsf{U}_{\geq c}\,\psi$: We reduce to the previous cases using equivalence (E2)

and $\mathsf{AG}_{<c}\,\varphi \equiv \neg\mathsf{EF}_{<c}\,\neg\varphi$ and

$$\mathsf{A}\varphi\mathsf{U}_{>0}\,\psi \equiv \mathsf{AG}_{\leq 0}\ (\varphi \wedge \mathsf{AX}(\mathsf{A}\varphi\mathsf{U}\psi)) \wedge \mathsf{AF}_{\geq 1}\,\top.$$

Here formula $\mathsf{AF}_{\geq 1}\,\top$ means that there is no run of null duration (we may assume that $c \geq 1$, since otherwise $\xi \equiv \mathsf{A}\varphi\mathsf{U}\psi$), and is equivalent to $\neg\mathsf{EF}_{\leq 0}\,P_{\mathsf{scc}^0(\top)}$.

Now we can show that $|\mathsf{Sat}[q,\varphi]|$ being bounded by $|\varphi|\cdot|R^q_{\mathcal{S}}|$ is preserved along the algorithm. This entails that the DTGs G built for $\mathsf{E}\varphi\mathsf{U}_{\sim c}\,\psi$ are such that $|V_G|$ is in $O(|\xi|\cdot|R|)$ and $|{\rightarrow_G}|$ is in $O(|\xi|\cdot|R|^2)$; thus the procedures run in time $O(|\xi|^2\cdot|R|^3)$. $\square$

6 Other temporal logics

In this section we consider how exact duration subscripts do or do not increase the cost of model checking when the models are DTGs and the logic is a timed variant of classic temporal logics like *LTL* or *CTL**.

We write *TLTL* and *TCTL** for the timed variants of the logics *LTL* and *CTL** and will let $TLTL_{\leq,\geq}$ and $TCTL^*_{\leq,\geq}$ denote the fragments where exact duration constraints are not allowed. The formal definitions of *LTL* and *CTL** are omitted (see [Eme90]), here we only point out the main characteristics of these logics.

6.1 Model checking TLTL over DTGs

TLTL is the linear-time timed temporal logic where formulae are built with atomic propositions, Boolean combinators and the modalities X and $\mathsf{U}_{\sim c}$. *TLTL* formulae are *path* formulae and are interpreted over runs in a DTG. As usual in this case (see, e.g., [SC85]), we consider *existential model checking*, that is the problem of deciding for a DTG S, a state q and a formula φ, whether there exists a path from q satisfying φ in the TTS associated with the DTG by the selected semantics.

Theorem 6.1 *For both the jump and the continuous semantics:*

(1) Model checking TLTL over DTGs is EXPSPACE*-complete.*
(2) Model checking $TLTL_{\leq,\geq}$ over DTGs is PSPACE*-complete.*

PROOF. [Sketch] The proof uses the results obtained in the timed framework [AH94,AFH96].

(1): EXPSPACE-hardness: it is possible to describe with a *TLTL* formula the accepting runs of a Turing Machine that runs in space 2^n (see, e.g., [AH94]). As usual, a run of the TM is seen as a sequence of instantaneous descriptions (i.d.). Here each i.d. has length 2^n. One easily writes that any two consecutive i.d.'s agree with the TM rules by means of the $\mathsf{F}_{=2^n}$ modality, a modality of size $O(n)$. This holds for any considered semantics, and the underlying DTG only uses "$\xrightarrow{1}$"-transitions.

Membership in EXPSPACE: It can be seen as a special case of the EXPSPACE upper bound for *TPTL* [AH94], a logic more expressive than *TLTL* interpreted over "timed state graphs" (a model in which one can encode DTGs with continuous semantics). More precisely one can show that there is an algorithm running within space polynomial in the size of the DTG and exponential in the size of the formula to be verified.

(2): PSPACE-hardness is inherited from PSPACE-hardness of *LTL* model checking.

Membership in PSPACE: [AFH96] shows that model checking $MITL_{0,\infty}$ (a logic equivalent to $TLTL_{\leq,\geq}$) over Timed Automata can be done in PSPACE. Since Timed Automata easily encode DTGs with continuous semantics, the upper bound follows. □

6.2 Model checking $TCTL^*$ over DTGs

$TCTL^*$ extends both *TCTL* and *TLTL*: the path quantifiers E and A are allowed to express properties over states, and the modalities $\mathsf{U}_{\sim c}$ may be embedded with no restriction as in *TLTL* to express complex properties over executions. $TCTL^*$ formulae are interpreted over pairs (π, s) corresponding to a state along an execution.

Theorem 6.2 *For both the jump and the continuous semantics, we have:*

(1) Model checking $TCTL^*$ over DTGs is EXPSPACE*-complete.*
(2) Model checking $TCTL^*_{\leq,\geq}$ over DTGs is PSPACE*-complete.*

PROOF. [Sketch]

First consider the case of the jump semantics. Here the results are a direct consequence of Theorem 6.1: the techniques from [EL87] produce an algorithm for $TCTL^*$ under the form of a simple polynomial-time labeling algorithm that calls an oracle for $TLTL$ model checking. Hence model checking belongs to $\mathsf{P}^{\mathsf{EXPSPACE}}$, that is $\mathsf{EXPSPACE}$. More precisely the algorithm runs in space polynomial in the size of the DTG and exponential in the size of the formula.

The same reasoning applies to $TCTL^*_{\leq,\geq}$ and yields a $\mathsf{P}^{\mathsf{PSPACE}}$, i.e., PSPACE, algorithm.

Now we consider the continuous semantics. Let $\mathcal{S}$ be a DTG and let $M_{\mathcal{S}}$ be the maximal integer constant occurring in $\mathcal{S}$. We aim at deciding whether $T_c(\mathcal{S})$ satisfies some formula Φ by reducing to a model checking instance for the jump semantics. The first step consists in replacing $T_c(\mathcal{S})$ by a synchronized product $(\mathcal{S}' \times C_0 \times \ldots \times C_l)$ with $l = \lceil \log(M_{\mathcal{S}} + 1) \rceil$. Every C_i is used to encode the i-th bit of the value v associated with the corresponding $T_c(\mathcal{S})$ state (q, v). The C_is are two-states KSs and $\mathcal{S}'$ represents the control part of $\mathcal{S}$. The synchronized product allows to increase the value v according to time elapsing and specifies when a transition is enabled or not depending on the guards in $\mathcal{S}$ and the current value v. The only difference between $T_c(\mathcal{S})$ and the TTS generated by $(\mathcal{S}' \times C_0 \times \ldots \times C_l)$ is that states (q, v) with $v > M_{\mathcal{S}}$ are merged in a unique state $(q, M_{\mathcal{S}} + 1)$, this clearly does not change the truth value of formulae. Moreover note that the size of the underlying TTS is exponential in $|\mathcal{S}|$.

First assume $\Phi \in TCTL^*$. The model checking algorithm for $TCTL^*$ and DTGs with the jump semantics can be adapted to decide $TCTL^*$ formulae over TTS. As it runs in space polynomial in the size of the DTG and exponential in the size of the formula, it provides an algorithm for deciding $T_c(\mathcal{S}) \models \Phi$ running in space exponential in $|\mathcal{S}|$ and $|\Phi|$.

Now assume Φ is a $TCTL^*_{\leq,\geq}$ formula. Let M_Φ be the maximal constant in Φ. Using the ideas of [AFH96] one can show that verifying subformulae of the form $\varphi \mathsf{U}_{\leq c}\, \psi$ or $\varphi \mathsf{U}_{\geq c}\, \psi$ can be done by adding to the model **one** extra clock for each such subformula. Indeed if we want to verify the property $\varphi \mathsf{U}_{\leq c}\, \psi$ for several configurations $s_1, s_2 \ldots$ along a run, it is sufficient to reset the clock $x_{\varphi \mathsf{U}_{\leq c} \psi}$ when the first configuration s_1 is visited and to verify that $x_{\varphi \mathsf{U}_{\leq c} \psi} \leq c$ when a configuration t satisfying ψ is reached. Then any configuration located between s_1 and t will also satisfy $\varphi \mathsf{U}_{\leq c}\, \psi$. Note that this contrasts with the modalities $\_\mathsf{U}_{=c}\,\_$ for which one clock is not sufficient. Therefore we can add k clocks (k is the number of modalities $\_\mathsf{U}_{\sim c}\,\_$), namely k sets of $\log(M_\Phi + 1)$ bits (encoded as the C_is). The synchronized product is then composed by $1 + \lceil \log(M_{\mathcal{S}}) \rceil + k \cdot \lceil \log(M_\Phi) \rceil$ processes (with $k \leq |\Phi|$). The synchronization is then defined in order to increase the different counters according to time elapsing. Then it remains to verify that some $\overline{\Phi}$ holds for the parallel composition where $\overline{\Phi}$ is a simple translation of Φ into CTL^* including special atomic propositions to handle timing constraints.

This proves membership is PSPACE since model checking CTL^* formulae over products of KSs is PSPACE-complete [KVW00]. $\square$

Remark 6.3 *For $TCTL^+$, the timed variant of CTL^+, model checking is Δ_2^{p}-complete for the jump semantics [LMS02]: Δ_2^{p}-hardness comes from the fact that CTL^+ model checking is already Δ_2^{p}-hard in the untimed case, and membership in Δ_2^{p} is based on an extension of Lemmas A.3 and A.4 in Appendix A for formulae of the form $\mathsf{E}\Big(\bigwedge_i P_i \,\mathsf{U}_{\sim c_i} P'_i \wedge \bigwedge_j \neg(P_j \,\mathsf{U}_{\sim c_j} P'_j)\Big)$. For the continuous semantics, $TCTL^+$ model checking is clearly Δ_2^{p}-hard and* PSPACE*-easy.*

7 Variants of DTGs

In this section we consider another possible semantics for DTGs. We also consider two natural restricted subclasses of DTGs. We discuss how these choices impact on the complexity of model checking.

7.1 Continuous early semantics

Another notion of *continuous semantics* could have been used in the previous section: We call it the *continuous early* semantics[4]. In that semantics, there are intermediary states, but when entering such an intermediary state, the system commits itself to taking a fixed transition and cannot change the destination state.

Given a transition (q, ρ, q') of $\mathcal{S}$, we define $\delta_{\max}(q \xrightarrow{\rho} q')$ as u (resp. ∞) if $\rho = [l, u]$ (resp. $\rho = [l, \infty)$). The *continuous early* semantics (written *c.e.*-semantics) of $\mathcal{S}$ is defined as the TTS $T_{ce}(\mathcal{S}) = \langle S, s_{\text{init}}, \rightarrow, l\rangle$ with:

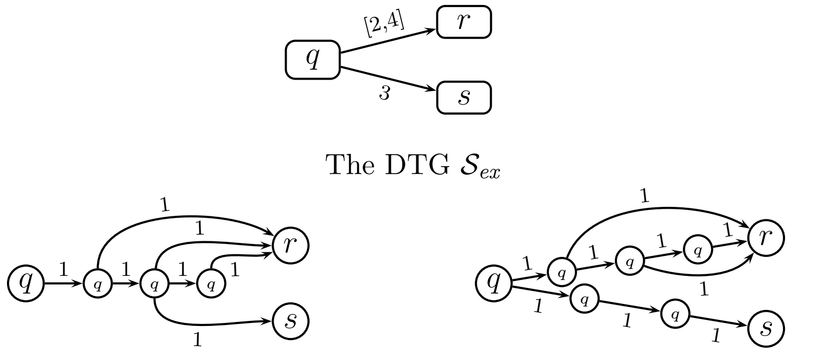
- $S = Q \cup \{(q \xrightarrow{\rho} q', i) \mid (q, \rho, q') \in R \ \wedge\ 1 \leq i < \delta_{\max}(q \xrightarrow{\rho} q')\}$ and $s_{\text{init}} = q_{\text{init}}$;
- The transition relation $\rightarrow$ is defined as follows:
  - · $q \xrightarrow{0} q'$ if $\exists(q, \rho, q') \in R$ and $0 \in \rho$;
  - · $q \xrightarrow{1} q'$ if $\exists(q, \rho, q') \in R$ and $1 \in \rho$;
  - · $q \xrightarrow{1} (q \xrightarrow{\rho} q', 1)$ if $1 < \delta_{\max}(q \xrightarrow{\rho} q')$;
  - · $(q \xrightarrow{\rho} q', i) \xrightarrow{1} (q \xrightarrow{\rho} q', i+1)$ if $i + 1 < \delta_{\max}(q \xrightarrow{\rho} q')$;
  - · $(q \xrightarrow{\rho} q', i) \xrightarrow{1} q'$ if $i + 1 \in \rho$.
- The states $q \in Q$ of T_{ce} are in $\mathcal{S}$; the states of the form $(q \xrightarrow{\rho} q', i)$ are labeled by the atomic propositions associated with q;
- $F = Q$.

[4] And, for improved clarity, we now call *continuous late* the semantics defined in section 5.1.

With this semantics, we distinguish between two kinds of transition: Those leading to a new control state, called the *action* transitions, and those corresponding to a simple delay of one time unit along a transition, called the *delay* transitions. The number of states is infinite when there exist transitions with unbounded duration in $\mathcal{S}$.

Observe that the fairness condition F allows one to rule out runs with a suffix of the form $(q \xrightarrow{[l,\infty)} q', i) \xrightarrow{1} (q \xrightarrow{[l,\infty)} q', i+1) \xrightarrow{1} (q \xrightarrow{[l,\infty)} q', i+2) \xrightarrow{1} \ldots$ Indeed a transition $(q, [l, \infty), q') \in R$ means that the transition from q to q' can take an arbitrary *finite* amount of time (beyond l).

Figure 5 illustrates the difference between the two "continuous" semantics.



The DTG $\mathcal{S}_{ex}$

Behavior of $\mathcal{S}_{ex}$ assuming the continuous late sem.

Behavior of $\mathcal{S}_{ex}$ assuming the continuous early sem.

Fig. 5. Continuous late and continuous early semantics for DTGs

These semantics are not equivalent bisimilar as may be seen with the (untimed) formula $\Psi \overset{\text{def}}{=} \mathsf{E}[\,(\mathsf{EG}\neg r)\,\mathsf{U}\,r\,]$ stating that one can reach r by a path where r is never inevitable. Ψ holds in (the initial state of) $\mathcal{S}_{ex}$ with the *continuous late* semantics, but not with the *continuous early* semantics, because the execution is committed into the transition towards r and the subformula $\mathsf{EG}\ \neg r$ does not hold anymore. See appendix C for more comparison between the three semantics.

As regards algorithmic issues, model checking DTGs under the continuous early semantics can be reduced to the continuous late semantics. Formally, we have:

Lemma 7.1 *Given a DTG $\mathcal{S} = \langle Q, q_{init}, R, l\rangle$, there exists a DTG $\overline{\mathcal{S}} = \langle \overline{Q}, \overline{q_{init}}, \overline{R}, \overline{l}\rangle$ such that for any TCTL formula φ, we have:*

$$\mathcal{S} \models_{ce} \varphi \text{ iff } \overline{\mathcal{S}} \models_{cl} \varphi.$$

Furthermore, $\overline{\mathcal{S}}$ can be build in logarithmic space from $\mathcal{S}$.

The DTG $\overline{\mathcal{S}}$ is defined as follows:

- $\overline{Q} \overset{\text{def}}{=} Q \cup \{(q \xrightarrow{\rho} q') \mid \exists (q,\rho,q') \in R \text{ and } \rho \cap [2,\infty) \neq \varnothing\}$
- $\overline{q_{\text{init}}} \overset{\text{def}}{=} q_{\text{init}}$
- $\bar{l}(q) = l(q)$ and $\bar{l}((q \xrightarrow{\rho} q')) = l(q)$.
- $\overline{R}$ is the following set:

$$\left\{ \begin{array}{l} (q, 1, (q \xrightarrow{\rho} q')), \\ ((q \xrightarrow{\rho} q'), [\max(1, l-1), u-1], q') \end{array} \middle| \exists (q, [l,u], q') \in R, \text{ and } u \geq 2 \right\}$$

$$\bigcup \left\{ \begin{array}{l} (q, 1, (q \xrightarrow{[l,\infty)} q')), \\ ((q \xrightarrow{[l,\infty)} q'), [\max(1, l-1), \infty), q') \end{array} \middle| \exists (q, [l,\infty), q') \in R \right\}$$

$$\bigcup \left\{ (q, 0, q') \mid \exists (q,\rho,q') \in R \text{ and } 0 \in \rho \right\}$$

$$\bigcup \left\{ (q, 1, q') \mid \exists (q,\rho,q') \in R \text{ and } 1 \in \rho \right\}.$$

Figure 6 gives an example of $\mathcal{S}$ and $\overline{\mathcal{S}}$. The equivalence of truth value for $TCTL$ formulae is straightforward: In $\overline{\mathcal{S}}$ it is not possible any more to wait for more than 1 time unit in a node of $\mathcal{S}$, the process has to choose a transition and additional states $(q \xrightarrow{\rho} q')$ behaves as intermediary states of the *c.e.*-semantics.

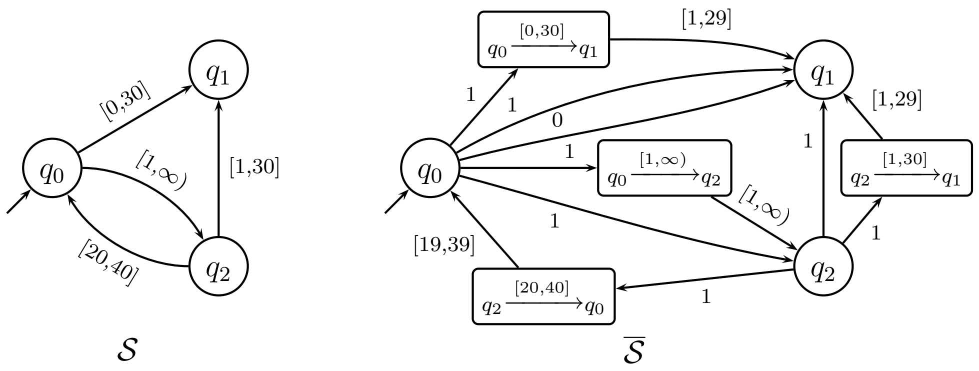

q_0
q_1
q_2
[0,30]
[1,∞)
[1,30]
[20,40]
$\mathcal{S}$
$q_0 \xrightarrow{[0,30]} q_1$
[1,29]
1
1
0
1
$q_0 \xrightarrow{[1,\infty)} q_2$
[1,∞)
$q_2 \xrightarrow{[1,30]} q_1$
[1,29]
1
1
1
[19,39]
$q_2 \xrightarrow{[20,40]} q_0$
1
$\overline{\mathcal{S}}$

Fig. 6. Reduction from *continuous early* to *continuous late* semantics

This entails:

Theorem 7.2 *Model checking $TCTL_{\leq,\geq}$ assuming the c.e.-semantics can be done in polynomial time.*

As regards full $TCTL$, the proof for the continuous late semantics (Appendix B) can easily be adapted to this semantics, and thus model checking $TCTL$ over DTGs with the continuous early semantics is shown to be PSPACE-complete.

Tight DTGs. Instead of allowing duration intervals in transitions of DTGs, we could restrict the durations to be an integer value. This restriction does not change the complexity results: The lower bounds in Theorems 4.2, 5.2, 6.1 and 6.2 have been shown for such *tight* DTGs.

DTGs with duration 0/1. An interesting subclass of DTG is the *small-steps DTG* ($\text{DTG}^{0/1}$) where every transition (q, ρ, q') has an interval $\rho \subseteq [0, 1]$. These DTGs have fundamental properties. First the choice of semantics does not matter: given a $\text{DTG}^{0/1}\mathcal{S}$, $T_j(\mathcal{S})$, $T_{ce}(\mathcal{S})$ and $T_{cl}(\mathcal{S})$ are isomorphic, and are a TTS where every transition has a duration in $\{0, 1\}$. In such a TTS, time progresses smoothly along paths: A path π of duration c can always be decomposed into two subpaths $\pi = \pi' \cdot \pi''$ with $\mathsf{Time}(\pi') = \lfloor \frac{c}{2} \rfloor$ and $\mathsf{Time}(\pi'') = \lceil \frac{c}{2} \rceil$. Moreover the duration of a simple path is bounded by $|Q|$ while in general TTS the duration of a simple path is exponential in $|\mathcal{S}|$.

Observe that the model used in [LST03] is very close to small-steps DTGs, but the duration information, "0 or 1 time unit", is carried by the nodes. In many works (for ex. [EMSS92,CCM+94]), Kripke structures where the duration of every transition is exactly 1 time unit, are used to model real time systems. The two properties (smooth time elapsing and polynomial durations) allow efficient model checking algorithms [EMSS92,LST03].

Acknowledgments. We thank Jeremy Sproston for discussing the proof of Theorem 5.3 with us, and the anonymous referees for their useful remarks.

8 Conclusion

Figure 7 summarizes our results on model checking quantitative temporal logics over DTGs.

A general pattern is that exact duration constraints make model checking harder. Without them, polynomial-time model checking is possible if one uses *TCTL* specifications, and this holds for the three semantics we considered. Another, less interesting, way to efficient model checking goes through the restriction to small-steps DTGs.

		$\mathrm{DTG}^{0/1}$	jump sem.	cont. sem.
TCTL	$\leq, \geq$	PTIME-complete		
	$\leq, \geq, =$	PTIME-complete [EMSS92,LST03]	Δ_2^p-complete	PSPACE-complete
TLTL	$\leq, \geq$	PSPACE-complete		
	$\leq, \geq, =$	EXPSPACE-complete		
TCTL*	$\leq, \geq$	PSPACE-complete		
	$\leq, \geq, =$	EXPSPACE-complete		

Fig. 7. Overview of results

References

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

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

[ACH94] R. Alur, C. Courcoubetis, and T. A. Henzinger. The observational power of clocks. In *Proc. 5th Int. Conf. Theory of Concurrency (CONCUR'94), Uppsala, Sweden, Aug. 1994*, volume 836 of *Lecture Notes in Computer Science*, pages 162–177. Springer, 1994.

[AD94] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[AFH96] R. Alur, T. Feder, and T. A. Henzinger. The benefits of relaxing punctuality. *Journal of the ACM*, 43(1):116–146, 1996.

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

[AH93] R. Alur and T. A. Henzinger. Real-time logics: Complexity and expressiveness. *Information and Computation*, 104(1):35–77, 1993.

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

[AL02] L. Aceto and F. Laroussinie. Is your model checker on time? On the complexity of model checking for timed modal logics. *Journal of Logic and Algebraic Programming*, 52–53:7–51, 2002.

[Alu91] R. Alur. *Techniques for Automatic Verification of Real-Time Systems.* PhD thesis, Stanford Univ., August 1991. Available as Tech. Report STAN-CS-91-1378.

[BBF$^+$01] B. Bérard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and Ph. Schnoebelen. *Systems and Software Verification. Model-Checking Techniques and Tools.* Springer, 2001.

[CC95] S. Campos and E. M. Clarke. Real-time symbolic model checking for discrete time models. In T. Rus and C. Rattray, editors, *Theories and Experiences for Real-Time System Development*, volume 2 of *AMAST Series in Computing*, pages 129–145. World Scientific, 1995.

[CC99] S. Campos and E. M. Clarke. Analysis and verification of real-time systems using quantitative symbolic algorithms. *Journal of Software Tools for Technology Transfer*, 2(3):260–269, 1999.

[CC01] S. Campos and E. M. Clarke. The Verus language: representing time efficiently with BDDs. *Theoretical Computer Science*, 253(1):95–118, 2001.

[CCM$^+$94] S. Campos, E. M. Clarke, W. R. Marrero, M. Minea, and H. Hiraishi. Computing quantitative characteristics of finite-state real-time systems. In *Proc. 15th IEEE Real-Time Systems Symposium (RTSS'94), San Juan, Puerto Rico, Dec. 1994*, pages 266–270. IEEE Comp. Soc. Press, 1994.

[CCMM95] S. Campos, E. M. Clarke, W. R. Marrero, and M. Minea. Timing analysis of industrial real-time systems. In *Proc. 1st Workshop on Industrial-Strength Formal Specification Techniques, Boca-Raton, FL, USA*, pages 97–107, 1995.

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

[CLR90] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to algorithms.* MIT Press, 1990.

[CTM$^+$99] S. Campos, M. Teixeira, M. Minea, A. Kuehlmann, and E. M. Clarke. Model checking semi-continuous time models using BDDs. In *Proc. 1st Int. Workshop on Symbolic Model Checking (SMC'99), Trento, Italy, July 1999*, volume 23(2) of *Electronic Notes in Theor. Comp. Sci.* Elsevier Science, 1999.

[CY92] C. Courcoubetis and M. Yannakakis. Minimum and maximum delay problems in real-time systems. *Formal Methods in System Design*, 1(4):385–415, 1992.

[EL87] E. A. Emerson and Chin-Laung Lei. Modalities for model checking: Branching time logic strikes back. *Science of Computer Programming*, 8(3):275–306, 1987.

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

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

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

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

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

[Hen98] T. A. Henzinger. It's about time: real-time logics reviewed. In *Proc. 9th Int. Conf. Concurrency Theory (CONCUR'98), Nice, France, Sep. 1998*, volume 1466 of *Lecture Notes in Computer Science*, pages 439–454. Springer, 1998.

[HNSY94] T. A. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic model checking for real-time systems. *Information and Computation*, 111(2):193–244, 1994.

[HS85] M. Hennessy and C. Stirling. The power of the future perfect in program logics. *Information and Control*, 67(1–3):23–52, 1985.

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

[Kre88] M. W. Krentel. The complexity of optimization problems. *Journal of Computer and System Sciences*, 36(3):490–509, 1988.

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

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

[LMS01] F. Laroussinie, N. Markey, and Ph. Schnoebelen. Model checking CTL^+ and $FCTL$ is hard. In *Proc. 4th Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS 2001), Genova, Italy, Apr. 2001*, volume 2030 of *Lecture Notes in Computer Science*, pages 318–331. Springer, 2001.

[LMS02] F. Laroussinie, N. Markey, and Ph. Schnoebelen. On model checking durational Kripke structures (extended abstract). In *Proc. 5th Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS 2002), Grenoble, France, Apr. 2002*, volume 2303 of *Lecture Notes in Computer Science*, pages 264–279. Springer, 2002.

[LMS04] F. Laroussinie, N. Markey, and Ph. Schnoebelen. Model checking timed automata with one or two clocks. In *Proc. 15th Int. Conf. Concurrency Theory (CONCUR 2004), London, UK, Aug.-Sep. 2004*, volume 3170 of *Lecture Notes in Computer Science*, pages 387–401. Springer, 2004.

[LS05] F. Laroussinie and J. Sproston. Model checking durational probabilistic systems. In *Proc. 8th Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS 2005), Edinburgh, UK, Apr. 2005*, volume 3441 of *Lecture Notes in Computer Science*, pages 140–154. Springer, 2005.

[LST03] F. Laroussinie, Ph. Schnoebelen, and M. Turuani. On the expressivity and complexity of quantitative branching-time temporal logics. *Theoretical Computer Science*, 297(1–3):297–315, 2003.

[MS04] N. Markey and Ph. Schnoebelen. Symbolic model checking of simply-timed systems. In *Proc. Joint Conf. Formal Modelling and Analysis of Timed Systems (FORMATS 2004) and Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT 2004), Grenoble, France, Sep. 2004*, volume 3253 of *Lecture Notes in Computer Science*, pages 102–117. Springer, 2004.

[NU02] M. Nykänen and E. Ukkonen. The exact path length problem. *Journal of Algorithms*, 42(1):41–53, 2002.

[Ost90] J. S. Ostroff. Deciding properties of timed transition models. *IEEE Transactions on Parallel and Distributed Systems*, 1(2):170–183, 1990.

[Pap84] C. H. Papadimitriou. On the complexity of unique solutions. *Journal of the ACM*, 31(2):392–400, 1984.

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

[RS05] A. Rabinovich and Ph. Schnoebelen. BTL_2 and the expressive power of $ECTL^+$. *Information and Computation*, 2005. To appear.

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

[Sch03] Ph. Schnoebelen. Oracle circuits for branching-time model checking. In *Proc. 30th Int. Coll. Automata, Languages, and Programming (ICALP 2003), Eindhoven, NL, July 2003*, volume 2719 of *Lecture Notes in Computer Science*, pages 790–801. Springer, 2003.

[Sto76] L. J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976.

[Wag87] K. W. Wagner. More complicated questions about maxima and minima, and some closures of NP. *Theoretical Computer Science*, 51(1–2):53–80, 1987.

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

A Model checking *TCTL* over DTGs with the jump semantics

In this appendix, we prove the following Theorem:

Theorem A.1 *Model checking TCTL over DTGs with the jump semantics is Δ_2^{P}-complete.*

A.1 Membership in Δ_2^{P}

Allowing both exact durations *and* general DTGs makes model checking harder (Prop. 4.1) but this is not enough to make the problem PSPACE-complete for the jump semantics. Indeed, we have:

Proposition A.2 *Model checking TCTL over DTGs with the jump semantics is in Δ_2^{P}.*

The standard model-checking algorithm for branching-time logics computes, for each subformula ψ of the formula at hand, the set of states in the DTG that satisfy ψ. This algorithm is in Δ_2^{P} if evaluating a basic modality in a given state can be done in NP. Theorem 4.3 provides deterministic polynomial-time solutions for modalities where exact durations are not used. Therefore it remains to provide NP routines for modalities of the form $\mathsf{E}\, P_1 \,\mathsf{U}_{=c}\, P_2$ and $\mathsf{A}\, P_1 \,\mathsf{U}_{=c}\, P_2$. We do this through the following two lemmas:

Lemma A.3 *Model checking the formula $\mathsf{E}P_1\mathsf{U}_{=c}P_2$ over DTGs with the jump semantics is in NP.*

PROOF. Let $S = \langle Q, q_{\text{init}}, R, l\rangle$ be a DTG. We first deal with the simpler case where S is tight (all intervals labeling R are singletons).

Assume there exists a path $\pi = q_0 \xrightarrow{d_0} q_1 \xrightarrow{d_1} q_2 \cdots q_n$ in S witnessing $q_0 \models \mathsf{E}\, P_1 \,\mathsf{U}_{=c}\, P_2$. We can assume $n < c \cdot |Q|$ since any null duration loop can be removed from π, but this is not enough to guarantee that π has size polynomial in $|S| + \lceil \log c \rceil$.

With π we associate the Parikh image of its transitions, that is, the map $\Phi_\pi \colon R \mapsto \mathbb{N}$ that counts the number of times each transition appears in π. Such a Φ also counts the number of times each node is entered and left: $\Phi^{\mathrm{i}}(q) = \sum\{\Phi(t) \mid t \text{ enters } q\}$ and $\Phi^{\mathrm{o}}(q) = \sum\{\Phi(t) \mid t \text{ leaves } q\}$.

Obviously, Φ_π satisfies the following properties:

(1) $\Phi^{\mathrm{i}}_\pi(q) = \Phi^{\mathrm{o}}_\pi(q)$ for any q different from q_0 and q_n. Furthermore, if $q_0 = q_n$, then $\Phi^{\mathrm{i}}_\pi(q_0) = \Phi^{\mathrm{o}}_\pi(q_0)$, otherwise $\Phi^{\mathrm{o}}_\pi(q_0) - \Phi^{\mathrm{i}}_\pi(q_0) = 1 = \Phi^{\mathrm{i}}_\pi(q_n) - \Phi^{\mathrm{o}}_\pi(q_n)$.
(2) The subgraph of S induced by the transitions $t \in R$ with $\Phi_\pi(t) > 0$ is connected.
(3) Φ_π has duration c, i.e., $c = \sum\{d \cdot \Phi(t) \mid t = (q, [d, d], q') \in R\}$.
(4) $q_n \models P_2$ and $q \models P_1$ for any state q such that $\Phi^{\mathrm{o}}_\pi(q) > 0$.

Conversely, if some Φ (with q_0, q_n) fulfills conditions 1 and 2, then by Euler circuit theorem, Φ is Φ_π for some path π from q_0 to q_n in S. If conditions 3 and 4 also hold, then π proves that $q_0 \models \mathsf{E}\, P_1 \,\mathsf{U}_{=c}\, P_2$.

If we assume $n < c \cdot |Q|$, then Φ can be encoded in polynomial-size, conditions 1 to 4 can be checked in polynomial-time, and Φ (with q_n) can be used as the polynomial-size witness we need for an NP algorithm.

Now, if we remove the assumption that S is tight, it is enough to replace condition 3 by

$$\sum_{t=(q,\rho,q')} \min(\rho) \cdot \Phi(t) \;\le\; c \;\le\; \sum_{t=(q,\rho,q')} \max(\rho) \cdot \Phi(t).$$

□

Lemma A.4 *Model checking the formula* $\mathsf{A}\, P_1 \,\mathsf{U}_{=c}\, P_2$ *over DTGs with the jump semantics is in* coNP.

PROOF. [Sketch] Since $\mathsf{A} P_1 \mathsf{U}_{=c} P_2 \;\equiv\; \mathsf{A} P_1 \mathsf{U}_{\ge c} P_2 \wedge \neg \mathsf{EG}_{=c} \neg P_2$, it is enough to show that model checking formulae of the form $\mathsf{EG}_{=c}\, P$ can be done in NP. This is done using techniques similar to the previous Lemma. (One difference is that we have to consider two cases: the path visits duration c, or it avoids it.) □

This completes the proof of Prop. A.2.

A.2 Hardness for Δ_2^{p}

We now show that model checking *TCTL* over DTGs with the jump semantics is Δ_2^{p}-hard, and hence Δ_2^{p}-complete. This means that there is no essentially better way for model checking *TCTL* over DTGs than the labeling algorithm used in Prop. A.2.

Proving Δ_2^{p}-hardness is difficult in part because there exist very few natural problems that are Δ_2^{p}-complete and that could be used in reductions to *TCTL* model checking. Here we capitalize on our proof that model-checking *FCTL* is Δ_2^{p}-complete [LMS01] and follow its pattern. However, this pattern must be altered and we have to encode Boolean problems in numerical problems. Since model-checking *TCTL* becomes polynomial-time when the numerical constants are written in unary, the Δ_2^{p}-hardness proof has to encode information in the bits of the numbers used in the DTG and the *TCTL* formula.

A Δ_2^{p}-complete problem. We start with the definition of SNSAT, "*Sequentially Nested* SATisfiability", a Δ_2^{p}-complete logic problem we use in our reduction [LMS01]. An instance $\mathcal{I}$ of SNSAT has the form

$$\mathcal{I} = \begin{bmatrix} x_1 := \exists Z_1\ F_1(Z_1), \\ x_2 := \exists Z_2\ F_2(x_1, Z_2), \\ \vdots \\ x_n := \exists Z_n\ F_n(x_1, \ldots, x_{n-1}, Z_n) \end{bmatrix}$$

where each F_i is a Boolean expression, each Z_i is a set of (auxiliary) Boolean variables, and the x_i are the main variables. We write X for $\{x_1, \ldots, x_n\}$, Z for $Z_1 \cup \cdots \cup Z_n$, and assume the sets $X, Z_1, \ldots, Z_n$ are pairwise disjoint. *Var* denotes $X \cup Z$ and $p = |Z|$.

W.l.o.g., we assume every F_i is a 3-CNF of the form $\bigwedge_l \bigvee_{m=1}^{3} \alpha_{i,l,m}$ where the $\alpha_{i,l,m}$ are literals. With every disjunct $\bigvee_m \alpha_{i,l,m}$ we associate a clause $C_{i,l}$ of the form $\overline{x_i} \vee \bigvee_m \alpha_{i,l,m}$ and write $Cl = \{C_1, \ldots, C_r\}$ for the resulting set of clauses.

$\mathcal{I}$ defines a unique valuation $v_{\mathcal{I}}$ of the variables in X where $v_{\mathcal{I}}(x_i) = \top$ iff $F_i(v_{\mathcal{I}}(x_1), \ldots, v_{\mathcal{I}}(x_{i-1}), Z_i)$ is satisfiable. The computational problem called SNSAT is, given an instance $\mathcal{I}$ as above, to decide whether $v_{\mathcal{I}}(x_n) = \top$. Therefore $\mathcal{I}$ can be seen as a sequence of n satisfiability problems where the ith problem depends on the answers of the earlier problems.

With this in mind, we say a valuation w of Var is:

safe: if, for all $i = 1, \ldots, n$,

$$w(x_i) \text{ implies } F_i(w(x_1), \ldots, w(x_{i-1}), w(Z_i)),$$

correct: if, for all $i = 1, \ldots, n$,

$$w(x_i) = F_i(w(x_1), \ldots, w(x_{i-1}), w(Z_i)),$$

admissible: if w is correct and coincide with $v_{\mathcal{I}}$ over X.

A correct valuation is safe and is also consistent for negative values assigned to some x_i. Still, this does not guarantee that the values of variables in Z are best possible, i.e., that w is admissible. An arbitrary valuation over Z extends into a correct valuation in a unique way, and checking that a given w is correct can be done in polynomial-time.

An admissible valuation is just a valuation for Z that yields $v_{\mathcal{I}}$ for X. Hence it is optimal over Z. Clearly, admissible valuations exist for any SNSAT instance, positive ($v_{\mathcal{I}}(x_n) = \top$) or negative, but checking that a given w is admissible is Δ_2^{p}-complete.

Reducing SNSAT to *TCTL* model checking. Fix some $K \in \mathbb{N}$. To variables $u \in \mathit{Var}$ and clauses $C \in \mathit{Cl}$ we assign weights $s(u)$ and $s(C)$ given by:

$$s(x_i) = K^i, \qquad s(z_i) = K^{n+i}, \qquad s(C_i) = K^{n+p+i}.$$

A multiset $\mathcal{M}$ of variables and clauses ($\mathcal{M} \in \mathbb{N}^{\mathit{Var} \cup \mathit{Cl}}$) has weight $s(\mathcal{M}) = \sum_x s(x) \times \mathcal{M}(x)$. Now if $\mathcal{M}(x) < K$ and $\mathcal{M}'(x) < K$ for all $x \in \mathit{Var} \cup \mathit{Cl}$, then $s(\mathcal{M}) = s(\mathcal{M}')$ iff $\mathcal{M} = \mathcal{M}'$. Therefore, by picking K large enough, we can reduce the equality of small multisets to the equality of their weights.

We now build $\mathcal{S}_{\mathcal{I}}$, a DTG associated with $\mathcal{I}$. See Fig. A.1. Nodes in $\mathcal{S}_{\mathcal{I}}$ are of two kinds: literal nodes (in the upper part of the figure) and filling nodes (in the lower part). With a path through the literal nodes that avoids the vertical "$\overline{x_i} \to x_i$" edges one associates a valuation of Var in the obvious way. The filling nodes are there for accounting purposes (see below).

For a literal α of the form $\pm u$, the duration $d(\alpha)$ is defined as $s(u) + \sum\{s(C) \mid C \in \mathit{Cl}, \alpha \Rightarrow C\}$. Therefore a path through the literal nodes will collect in its duration the weight of all the variables it visits plus the weight of all the clauses these literals satisfy (each clause being counted up to four times since

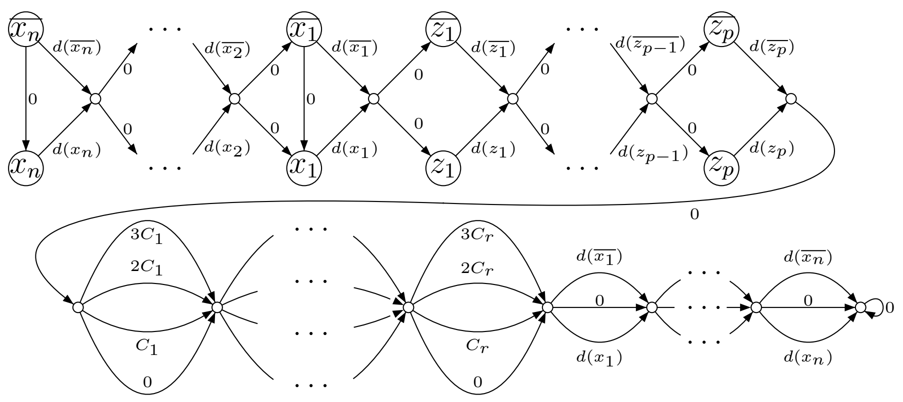


Fig. A.1. Kripke structure $S_\mathcal{I}$ associated with SNSAT instance $\mathcal{I}$

it may be satisfied thanks to four different literals). Then the path visits the filling nodes where it can gather further clause or literal weights.

Now define

$$K' \overset{\text{def}}{=} \sum_{u \in Var} s(u) + 4 \times \sum_{C \in Cl} s(C)$$

and assume K is large enough (here $K > 11$ suffices). Then, for any $u \in Var$, a path π of weight K' must collect $d(u)$ or $d(\overline{u})$ once and only once. Thus π defines a valuation of *Var*. Furthermore π has to gather 4 times the weight of all clauses from *Cl*. Since, for $C \in Cl$, we can only collect $3s(C)$ via filling nodes, π must visit at least one literal that satisfies C.

Hence paths of length K' correspond to valuations that satisfy all the clauses. We rely on this and introduce the following *TCTL* formulae:

$$\begin{aligned} \varphi_0 &\overset{\text{def}}{=} \top, \\ \text{and, for } k > 0, \quad \varphi_k &\overset{\text{def}}{=} \mathsf{E}\Big[P_{\overline{x}} \Rightarrow \mathsf{EX}\big(P_x \wedge \neg\varphi_{k-1}\big)\Big]\mathsf{U}_{=K'}\top, \end{aligned}$$

where P_x (resp. $P_{\overline{x}}$) is an atomic proposition that labels the n positive x_i nodes (resp. the $\overline{x_i}$ nodes).

We can now link $v_\mathcal{I}$ and the φ_k by:

Lemma A.5 *For $k \in \mathbb{N}$ and $r = 1, \ldots, n$:*
(a) if $k \geq 2r - 1$ then $(v_\mathcal{I}(x_r) = \top$ iff $\mathcal{S}_\mathcal{I}, x_r \models \varphi_k)$,
(b) if $k \geq 2r$ then $(v_\mathcal{I}(x_r) = \bot$ iff $\mathcal{S}_\mathcal{I}, \overline{x_r} \models \varphi_k)$.

PROOF. By induction on k. The case $k = 0$ holds vacuously. We now assume that $k > 0$ and that the Lemma holds for $k - 1$.

Proof of the "$\Rightarrow$" direction of both "iff"s. Let w be an admissible valuation. We use w to build a path π that starts at x_r (or $\overline{x_r}$ if $w(x_r) = \bot$), has total duration K', and only visit literals true under w (such a π exists because w is admissible). We claim π proves $x_r \models \varphi_k$ (or $\overline{x_r} \models \varphi_k$). This only requires that all nodes visited by π satisfy $P_{\overline{x}} \Rightarrow \mathsf{EX}(P_x \wedge \neg\varphi_{k-1})$ but on $\mathcal{S}_\mathcal{I}$ this translates into "$w(x_i) = \bot$ for $i \leq r$ implies $x_i \models \neg\varphi_{k-1}$" and is given by the induction hypothesis.

Proof of the "$\Leftarrow$" direction of both "iff"s. Assume $k \geq 2r - 1$ and $x_r \models \varphi_k$ (or $k \geq 2r$ and $\overline{x_r} \models \varphi_k$). Thus there is a path π starting from x_r (or $\overline{x_r}$), with duration K', and only visiting states satisfying $P_{\overline{x}} \Rightarrow \mathsf{EX}(P_x \wedge \neg\varphi_{k-1})$. Since $\mathsf{Time}(\pi) = K'$ the valuation w induced by π satisfies all $C \in Cl$. We further claim that $w(x_i) = v_\mathcal{I}(x_i)$ for $i = 1, \ldots, r$ and prove this by induction over i:

(1) If $w(x_i) = \top$ then $\bigwedge_l \bigvee_m w(\alpha_{i,l,m}) = \top$, so that $F_i(w(x_1), \ldots, w(x_{i-1}), Z_i)$ is satisfiable. By ind. hyp. we get that $F_i(v_\mathcal{I}(x_1), \ldots, v_\mathcal{I}(x_{i-1}), Z_i)$ is satisfiable, so that $v_\mathcal{I}(x_i) = \top$.
(2) If $w(x_i) = \bot$ then $\overline{x_i} \models \mathsf{EX}(P_x \wedge \neg\varphi_{k-1})$, implying $x_i \models \neg\varphi_{k-1}$. If $i < r$ we have $k - 1 \geq 2i - 1$ and, by ind. hyp., $v_\mathcal{I}(x_i) = \bot$. If $i = r$ then we are dealing with the case $k \geq 2r$ and $\overline{x_k} \models \varphi_k$, so that $k - 1 \geq 2i - 1$ and again $v_\mathcal{I}(x_i) = \bot$ by ind. hyp. $\square$

Proposition A.6 *Model checking TCTL over DTGs with the jump semantics is Δ_2^p-hard.*

PROOF. By Lemma A.5, $\mathcal{I}$ is a positive instance iff $S_\mathcal{I}, x_n \models \varphi_{2n-1}$. Observe that $\mathcal{S}_\mathcal{I}$ and φ_{2n-1} can be built in logspace from $\mathcal{I}$. Thus SNSAT, a Δ_2^p-complete [LMS01], reduces to *TCTL* model checking. $\square$

Theorem A.7 *Model checking TCTL over DTGs with the jump semantics is Δ_2^p-complete.*

PROOF. Combine Props A.2 and A.6. $\square$

Remark A.8 *Theorem A.7 can be strengthened in various ways, e.g., observing that $\mathcal{S}_\mathcal{I}$ is a* tight DTG*. Further, we used the* EX *modality in φ_k but this is not necessary (and could be replaced by* $\mathsf{EF}_{\leq 0}$ *). Moreover $\mathcal{S}_\mathcal{I}$ contains transitions with null duration but it is easy to adapt the construction and show that Theorem A.7 still holds over tight DTGs with strictly positive durations.*

B Model checking *TCTL* over DTGs with the continuous semantics

Membership in PSPACE is obtained by adapting algorithms for model checking *TCTL* over Timed Automata (this problem is PSPACE-complete [ACD93]) to DTGs with the continuous semantics.

Lemma B.1 *Model checking TCTL over DTGs with the continuous semantics is* PSPACE*-hard.*

PROOF. Consider an instance $\Phi \overset{\text{def}}{=} Q_0 p_0 Q_1 p_1 \dots Q_{n-1} p_{n-1} \cdot \varphi$ of QBF, "Quantified Boolean Formulae", where for $i = 0, \dots, n-1$, Q_i belongs to $\{\exists, \forall\}$ and p_i is a Boolean variable, and where φ is a propositional formula over the p_i's. The instance Φ is said to be *valid* if there exists a non-empty set V_{Φ_s} of Boolean valuations for $\{p_0, \dots, p_{n-1}\}$ s.t. for any $v \in V_{\Phi_s}$, $v \models \varphi$ and for any i with $Q_i = \forall$ there exists $v' \in V_{\Phi_s}$ s.t. $v'(p_j) = v(p_j)\ \forall j < i$ and $v'(p_i) = \overline{v(p_i)}$. A valuation v for the p_i's can be encoded as an integer $N_v \in [0, 2^n - 1]$ such that the j-th bit of the binary encoding of N_v is 1 iff $v(p_j) = \top$.

Now we reduce the QBF instance Φ to a model checking instance: From Φ, we build the DTG $\mathcal{S}_\Phi$ depicted in Figure B.1.

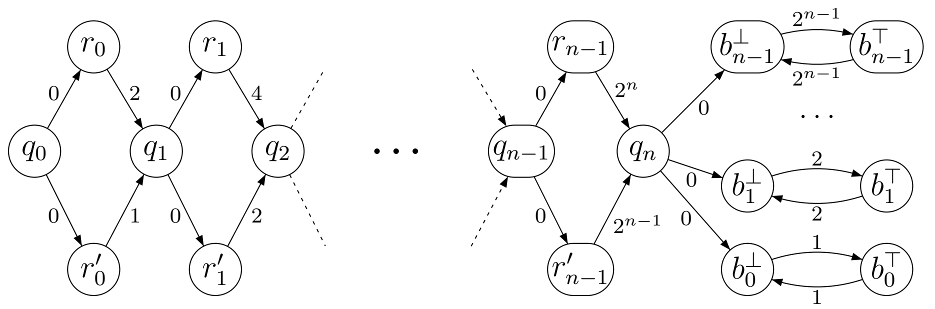


Fig. B.1. DTG $\mathcal{S}_\Phi$ associated with QBF instance Φ

From any state q_i there exist two possible paths leading to q_{i+1}: The upper one, through r_i, with total duration 2^{i+1}, and the lower one, through r'_i, with total duration 2^i. A path from q_0 to q_n can be seen as defining a Boolean valuation for the p_i's with the following convention: Going through the upper (resp. lower) $\mathcal{S}_\Phi$ transition issued from q_i assigns the value **false** (resp. **true**) to p_i.

Let S_i with $1 \le i \le n$ be the set of $T_c(\mathcal{S}_\Phi)$ states located at a distance $\sum_{j=0}^{i-1} 2^j$ from q_0. We can easily show by induction on i that

$$S_i = \{q_i\} \cup \{(r_{i-1} \xrightarrow{2^i} q_i, \alpha) \mid 1 \le \alpha \le 2^{i-1}\} \cup \\ \{(q_{i-1} \xrightarrow{2^{i-1}} r'_i, \alpha) \mid 1 \le \alpha \le 2^{i-1} - 1\}.$$

Moreover note that $|S_i| = 2^i$. Any S_i state has exactly two possible successors in S_{i+1} at a duration 2^i: one is reached by a path going through the $\mathcal{S}_\Phi$ upper transition starting from q_i, and the other one is reached by a path using the lower transition (labeled by 2^i). Therefore given a state s in S_n, there exists exactly one path of duration $\sum_{i=0}^{n-1} 2^i$ leading to s, and this path also defines a Boolean valuation.

Therefore we can associate with a state s in S_n a unique Boolean valuation v_s for the p_i's. Also, we can interpret a *TCTL* formula over s in order to get the value of $v_s(p_j)$, that is the value of the j-th bit of N_{v_s}. Indeed every S_n state is characterized by its distance to q_n, which belongs to $\{0, \ldots, 2^n - 1\}$: the state at distance 0 (i.e., q_n) corresponds to the valuation which assigns $\top$ to every p_j, the state at distance 1 corresponds to the valuation which assigns $\perp$ to p_0 and $\top$ to the other variables etc. An S_n-state s at distance i of q_n corresponds to the valuation v_s with $N_{v_s} = 2^n - 1 - i$. And we have the following property:

$$v_s(p_j) = \top \qquad \text{iff} \qquad s \models \mathsf{EF}_{=2^n-1}\, b_j^\top.$$

Indeed reaching q_n from s takes i t.u. and then it remains to find a path ρ of duration $2^n - 1 - i$ into the loop $b_j^\perp \to b_j^\top \to \ldots$ Clearly if the j-th bit of $2^n - 1 - i$ is 1, then ρ will finish at an intermediary state between $b_j^\top$ and $b_j^\perp$, and such a state satisfies $b_j^\top$. Conversely if the j-th bit is 0, ρ will terminate into an intermediary state between $b_j^\perp$ and $b_j^\top$.

Therefore the propositional formula φ is satisfied by the valuation v_s iff $s \models \varphi[\mathsf{EF}_{=2^n-1}\, b_j^\top / p_j]$.

To encode the QBF instance Φ, it remains to add quantifiers over valuations and we have: $\Phi \overset{\text{def}}{=} Q_0 p_0. Q_1 p_1 \ldots Q_{n-1} p_{n-1}. \varphi$ is valid if, and only if, $O_0\, O_1 \ldots O_{n-1}\, \varphi[\mathsf{EF}_{=2^n-1}\, b_j^\top / p_j]$ where O_i is $\mathsf{EF}_{=2^{i-1}}$ (resp. $\mathsf{AF}_{=2^{i-1}}$) if Q_i is $\exists$ (resp. $\forall$). This is sound because the choice of the i-th upper or lower $\mathcal{S}_\Phi$ transition is actually performed between instant $\sum_{j=0}^{i-1} 2^j$ and $\sum_{j=0}^{i} 2^j$. $\square$

C Comparing the three DTG semantics

One can relate the three semantics for DTGs in formal terms.

On the one hand, the finer granularity and the later timing of nondeterministic choices when moving from *jump* to *continuous early* to *continuous late* can be captured by a notion of timed simulation: We write $\sqsubseteq$ for the largest relation between states of TTSs s.t. for any $q \sqsubseteq r$ and any timed step $q \xrightarrow{d} q'$ from q, there exists a sequence of steps $r \xrightarrow{d_1}\xrightarrow{d_2} \cdots \xrightarrow{d_n} r'$ with $d = d_1 + \cdots + d_n$ and $r \sqsubseteq r'$. Then, for any DTG $\mathcal{S}$:

$$T_j(\mathcal{S}) \sqsubseteq T_{ce}(\mathcal{S}) \sqsubseteq T_c(\mathcal{S}).$$

This entails that $\exists TB(\mathsf{F})$, the fragment of $TCTL$ where only the $\mathsf{F}_{\sim\ldots}$ modalities (arbitrary timing constraints are permitted) and existential path quantification is allowed (but negation is not permitted), is preserved when moving from T_j to T_{ce} to T_c semantics.

Observe that this notion of simulation does not take fairness constraints into account. This is for simplification purposes, and fairness can be accounted for, e.g., along the lines of [HS85].

In the other direction, the wider latitude present in $T_c(\mathcal{S})$ when compared to $T_j(\mathcal{S})$ or $T_{ce}(\mathcal{S})$, does not add fundamentally new behavior. This can be captured with a timed notion of stuttering equivalence between timed runs: Say a timed run $\pi = q_1 \xrightarrow{d_1} q_2 \xrightarrow{d_2} q_3 \cdots \xrightarrow{d_{n-1}} q_n$ is equivalent to $\pi' = q_1 \xrightarrow{d_1} \cdots q_{i-1} \xrightarrow{d_{i-1}+d_i} q_{i+1} \xrightarrow{d_{i+1}} \cdots q_n$, written $\pi \sim_1 \pi'$, if q_i is labeled as q_{i-1} or as q_{i+1}. Then t-s-equivalence, denoted $\sim_t$, is defined by considering the reflexive, symmetric and transitive closure of $\sim_1$. Finally, two TTSs are t-s-equivalent, written $T_1 \sim_t T_2$ if they give rise to the same set of finite runs modulo $\sim_t$. Again, this notion of t-s-equivalence can, and should, be adapted to deal with infinite runs. Then, for any DTG $\mathcal{S}$:

$$T_j(\mathcal{S}) \sim_t T_{ce}(\mathcal{S}) \sim_t T_c(\mathcal{S}).$$

This entails that $LTL - \mathsf{X}$, the untimed fragment of $TLTL$, is preserved across the three semantics.