Symbolic Model Checking of Real-Time Systems

G. Logothetis and K. Schneider
University of Karlsruhe
Institute for Computer Design and Fault Tolerance (Prof. Dr.-Ing. D. Schmid)
P.O. Box 6980, 76128 Karlsruhe, Germany
email: {logo,schneide}@informatik.uni-karlsruhe.de
http://goethe.ira.uka.de/fmg

Abstract

We present a new real-time temporal logic for the specification and verification of discrete quantitative temporal properties. This logic is an extension of the well-known logic CTL. Its semantics is defined on discrete time transition systems which are in turn interpreted in an abstract manner instead of the usual stuttering interpretation. Hence, our approach directly supports abstractions of real-time systems by ignoring irrelevant qualitative properties, but without loosing any quantitative information. We analyse the complexity of the presented model checking algorithm and furthermore present a fragment of the logic that can be efficiently checked.

1. Introduction

Formal verification methods have been developed to reason about the correctness of a system with respect to a given specification. In particular, model checking [4, 15] of temporal logics has become one of the most successful verification techniques. Using this technique requires to adequately model a system by a finite state transition system so that specifications given in temporal logics [8] can be checked for that model.

In general, model checking procedures suffer from the so-called *state explosion problem*: The size, i.e., the number of states of the system can exponentially grow with the size of the implementation description. It is therefore often necessary to use abstraction techniques like those given in [6, 12, 14] to neglect irrelevant details so that the verification can concentrate on the necessary facts. As sets of states are thereby collected into abstract states, this means that the number of transitions to reach a certain state from another one is changed. As a consequence, however, information about quantitative time consumption is lost.

Real-time systems must perform certain actions within limited time bounds or should start actions only after some point of time. It is therefore natural to label the transitions of the abstract transition system by numbers that denote the time required to move from one state to another one. In general, there are two possible interpretations of these timed transition systems:

Interpretation I_1**:** A transition from state s_1 to state s_2 with label $k \in \mathbb{N}$ means that at any time t_0, where we are in state s_1, we can perform an atomic action that requires k units of time. The action terminates at time t_0+k, where we are in state s_2. *There is no information about the intermediate points of time* $t_0 < t < t_0 + k$.

Interpretation I_2**:** A transition from state s_1 to state s_2 with label $k > 1$ is seen as abbreviation for a stuttering sequence $s_1 \xrightarrow{1} s_{1,1} \xrightarrow{1} \ldots \xrightarrow{1} s_{1,k-1} \xrightarrow{1} s_2$ where all the states $s_{1,i}$ have the same variable assignment as state s_1.

Clearly, only interpretation I_1 can be used in a setting where more powerful abstraction techniques than stuttering simulations are used. It is therefore surprising that none of the previous real-time extensions of CTL is based on interpretation I_1, although this is the more general (expressive) one!

The development of discrete real-time extensions of CTL has been initiated in [9], where the temporal operators have been extended by time bounds to limit the number of fixpoint iterations required to evaluate the considered temporal expression. The models used in [9] were still traditional finite-state transition systems where each transition requires a single unit of time.

In order to represent real-time systems in a more compact way, [3] introduced timed transition systems, where transitions are labeled by natural numbers that denote the time consumption of the action associated with the transition. The meaning of these timed transitions is in our terms the stuttering interpretation I_2.

In [10] a new real-time temporal logic was introduced, where both interpretations I_1 and I_2 were unfortunately mixed: Different temporal operators of this logic interpret

transitions differently, i.e., either according to I_1 or I_2. As the meaning of timed transitions therefore depends on the context, it is impossible to reason about the meaning of timed transitions. In particular, it is not possible to define composition of structures.

Becoming aware of the misleading semantics of [10] (see page 11 of [16]), another temporal logic that is solely based on interpretation I_2 has been presented in [16]. In principle, the logic of [16] is defined on unit delay transition systems that are obtained by expanding (cf. definition 6) a given timed transition system. However, there are different expansion algorithms that yield in different transition systems that are not bisimilar to each other. Hence, the verification results obtained for an expanded structure can not be easily transferred back to the timed transition system.

To summarize, the mentioned real-time extensions of CTL have the following drawbacks:

- None of the previous real-time extensions of CTL is based on interpretation I_1 that is necessary to benefit from abstraction techniques.
- None of the previous real-time extensions of CTL has a time bounded next state operator to express facts about actions that correspond with a single transition. As a consequence, facts as the following one can not be expressed: *'Is there a non-stop flight from New York to Paris with a duration of at most 9 hours?'*
- The problem to compute the set of states of a timed transition system $\mathcal{K}$ where a real-time CTL formula holds, can not be easily translated to an equivalent CTL model checking problem on a unit delay structure: Expansions of timed transition systems may yield in different results [13], and furthermore different expansions are not equivalent to each other [13].

Hence, there has been much confusion and misconception about the definition of a sound and reasonable real-time extension of the famous temporal logic CTL. All mentioned previous approaches [3, 10, 16] are – from a logical perspective – questionable.

In [13], we have presented a real-time extension JCTL of CTL that is based on interpreting timed transition systems with *interpretation* I_1. This directly supports abstractions of real-time systems by ignoring their irrelevant qualitative properties, but without loosing their quantitative ones. For example, we can model processes that compute some values within a certain limit of time with a single transition, that does not state anything about the values of the variables *during the computation*. Moreover, JCTL has a next-state operator equipped with time bounds, so that one can reason about real-time constraints of atomic actions.

In this paper, we reconsider the logic JCTL and its model checking algorithm, and analyse its complexity. It turns out that the complexity to compute the set of states where a given JCTL formula holds is the same as for previous approaches like [10]. However, we additionally present a non-trivial fragment $\mathsf{JCTL}^{\leq}$ of JCTL that can be more efficiently checked. In particular, less fixpoints need to be evaluated in this fragment, and the model checking algorithm can ignore the time bounds of the transitions for many of its computations.

This paper is organized as follows: In the next section, we define our version of timed transitions systems and our real-time temporal logic JCTL. We will then proceed with the definition of a symbolic model checking procedure for JCTL. After this, we analyze its complexity, and present the fragment $\mathsf{JCTL}^{\leq}$ of JCTL that can be checked more efficiently.

Though not closely related to this paper, we want to finally mention that beneath the real-time extensions of CTL that are defined on discrete time models, there are also very successful approaches that are based on a continuous model of time [1, 11, 7, 2]. These approaches usually rely on timed automata, i.e., on finite state automata that are endowed by a finite set of real-valued clocks. Most verification procedures based on timed automata require the construction of a so-called region graph to reduce the infinite state space of timed automata to a finite state problem. However, the construction of the region graph is very expensive in practice. Approaches for discrete time models do not suffer from the need of such a construction as they directly use finite state transition systems to model the systems.

2. Syntax and Semantics of the Logic JCTL

2.1. Timed Kripke Structures

We consider systems modeled as timed Kripke structures[1] over some set of variables $\mathcal{V}$. These timed Kripke structures are formally defined as follows:

Definition 1 (Timed Kripke Structures (TKS)) *A timed Kripke structure over the variables $\mathcal{V}$ is a tuple $(\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$, such that $\mathcal{S}$ is a finite set of states, $\mathcal{I} \subseteq \mathcal{S}$ is the set of initial states, and $\mathcal{R} \subseteq \mathcal{S} \times \mathbb{N} \times \mathcal{S}$ is the set of transitions. For any state $s \in \mathcal{S}$, the set $\mathcal{L}(s) \subseteq \mathcal{V}$ is the set of variables that hold on s. We furthermore demand that for any $(s, t, s') \in \mathcal{R}$, we have $t > 0$ and that for any $s \in \mathcal{S}$, there must be a $t \in \mathbb{N}$ and a $s' \in \mathcal{S}$ such that $(s, t, s') \in \mathcal{R}$ holds.*

Timed Kripke structures may be pictorially drawn as given in Figure 1, where initial states are drawn with double lines.

[1] Timed transition systems have been introduced by many authors with different names like timed transitions graphs [3], quantitative temporal structures [10], or timed temporal structures in [16]. Following the CTL notations, we prefer the name timed Kripke structures.

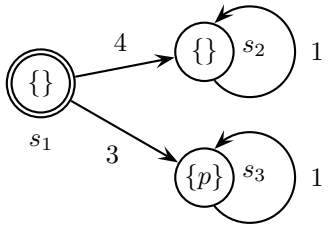

Figure 1. A Timed Kripke Structure

Some approaches, e.g. [3] label transitions with intervals $[a, b]$ of time. It is easily seen that our TKSs subsume these models since we can add for any $t \in [a, b]$ a new transition between the considered two states.

It is crucial to understand what is modeled by a TKS. We use interpretation I_1: A transition from state s to state s' with label $k \in \mathbb{N}$ means that at any time t_0, where we are in state s, we can perform an atomic action that requires k units of time. The action terminates at time t_0+k, where we are in state s'. *In particular, there is no information about the intermediate points of time t with $t_0 < t < t_0 + k$.*

2.2. JCTL as a Real-Time Extension of CTL

To write down specifications in a formal way, we use JCTL, a real-time extension of the temporal logic CTL. For its definition below, we only use a small subset of logical operators that will be extended below by some abbreviations.

Definition 2 (Syntax of JCTL) *Given a set of variables $\mathcal{V}$, the set of JCTL formulas is the least set satisfying the following rules, where φ and ψ denote arbitrary JCTL formulas, and $a, b \in \mathbb{N}$ are arbitrary natural numbers:*

- $\mathcal{V} \subseteq$ JCTL, *i.e, any variable is a JCTL formula*
- $\neg\varphi$, $\varphi \wedge \psi \in$ JCTL
- $\mathsf{EX}^{[a,b]}\varphi \in$ JCTL
- $\mathsf{EX}^{\geq a}\varphi \in$ JCTL
- $\mathsf{E}[\varphi\ \underline{\mathsf{U}}^{[a,b]}\ \psi] \in$ JCTL
- $\mathsf{E}[\varphi\ \underline{\mathsf{U}}^{\geq a}\ \psi] \in$ JCTL
- $\mathsf{EG}^{[a,b]}\varphi \in$ JCTL
- $\mathsf{EG}^{\geq a}\varphi \in$ JCTL

The semantics of JCTL is defined with respect to a TKS. For the definition of the semantics, we need the notion of paths. A *path π through a timed Kripke structure* is a function $\pi : \mathbb{N} \to \mathcal{S}$ such that $\forall i \in \mathbb{N}.\exists t \in \mathbb{N}.(\pi^{(i)}, t, \pi^{(i+1)}) \in \mathcal{R}$ holds (we write the function application with a superscript). Hence, $\pi^{(i)}$ is the $(i+1)th$ state on path π. For a given path π, we define an *associated time consumption function* τ_π, so that π and τ_π satisfy the condition $\forall i \in \mathbb{N}.\exists t \in \mathbb{N}.(\pi^{(i)}, \tau_\pi^{(i)}, \pi^{(i+1)}) \in \mathcal{R}$. Note that τ_π is not uniquely defined for a fixed path π, since we may have more than one transition between two states that are labeled with different numbers. The set of paths starting in a state s is furthermore denoted as $\mathsf{Paths}_\mathcal{K}(s)$.

Definition 3 (Semantics of JCTL) *Given a TKS $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$, and $s \in \mathcal{S}$, then the semantics of the logic is recursively defined as follows:*

- $\mathcal{K}, s \models p$ *iff* $p \in \mathcal{L}(s)$ *for any* $p \in \mathcal{V}$
- $\mathcal{K}, s \models \neg\varphi$ *iff* $(\mathcal{K}, s) \not\models \varphi$
- $\mathcal{K}, s \models \varphi \wedge \psi$ *iff* $\mathcal{K}, s \models \varphi$ *and* $\mathcal{K}, s \models \psi$
- $\mathcal{K}, s \models \mathsf{EX}^{[a,b]}\varphi$ *iff there is a path* $\pi \in \mathsf{Paths}_\mathcal{K}(s)$ *with associated duration function* τ_π *with*

$$\left(a \leq \tau_\pi^{(0)} \leq b\right) \wedge \left(\mathcal{K}, \pi^{(1)} \models \varphi\right)$$

- $\mathcal{K}, s \models \mathsf{EX}^{\geq a}\varphi$ *iff there is a path* $\pi \in \mathsf{Paths}_\mathcal{K}(s)$ *with associated duration function* τ_π *with*

$$\left(a \leq \tau_\pi^{(0)}\right) \wedge \left(\mathcal{K}, \pi^{(1)} \models \varphi\right)$$

- $\mathcal{K}, s \models \mathsf{E}[\varphi\ \underline{\mathsf{U}}^{[a,b]}\ \psi]$ *iff there is a path* $\pi \in \mathsf{Paths}_\mathcal{K}(s)$ *with associated duration function* τ_π *and an* $i \in \mathbb{N}$ *with*

$$\left(a \leq \sum_{j=0}^{i-1} \tau_\pi^{(j)} \leq b\right) \wedge \left(\mathcal{K}, \pi^{(i)} \models \psi\right) \wedge \left(\forall j < i.\ \mathcal{K}, \pi^{(j)} \models \varphi\right)$$

- $\mathcal{K}, s \models \mathsf{E}[\varphi\ \underline{\mathsf{U}}^{\geq a}\ \psi]$ *iff there is a path* $\pi \in \mathsf{Paths}_\mathcal{K}(s)$ *with associated duration function* τ_π *and an* $i \in \mathbb{N}$ *with*

$$\left(a \leq \sum_{j=0}^{i-1} \tau_\pi^{(j)}\right) \wedge \left(\mathcal{K}, \pi^{(i)} \models \psi\right) \wedge \left(\forall j < i.\ \mathcal{K}, \pi^{(j)} \models \varphi\right)$$

- $\mathcal{K}, s \models \mathsf{EG}^{[a,b]}\varphi$ *iff there is a path* $\pi \in \mathsf{Paths}_\mathcal{K}(s)$ *with associated duration function* τ_π*, such that for all* $i \in \mathbb{N}$*, we have*

$$\left(a \leq \sum_{j=0}^{i-1} \tau_\pi^{(j)} \leq b\right) \rightarrow \left(\mathcal{K}, \pi^{(i)} \models \varphi\right)$$

- $\mathcal{K}, s \models \mathsf{EG}^{\geq a}\varphi$ *iff there is a path* $\pi \in \mathsf{Paths}_\mathcal{K}(s)$ *with associated duration function* τ_π*, such that for all* $i \in \mathbb{N}$*, we have*

$$\left(a \leq \sum_{j=0}^{i-1} \tau_\pi^{(j)}\right) \rightarrow \left(\mathcal{K}, \pi^{(i)} \models \varphi\right)$$

Given a TKS $\mathcal{K}$ and a JCTL *formula φ, we denote the set of states of $\mathcal{K}$ where φ holds as $[\![\varphi]\!]_{\mathcal{K}}$.*

Intuitively, $\mathcal{K}, s \models \mathsf{EX}^{[a,b]}\varphi$ means that the state s has a direct successor state s' that satisfies φ and can be reached in time $t \in [a, b]$. $\mathcal{K}, s \models \mathsf{EX}^{\geq a}\varphi$ means that the state s has a direct successor state s' that satisfies φ and can be reached in time $t \geq a$.

$\mathcal{K}, s \models \mathsf{E}[\varphi\, \underline{\mathsf{U}}^{[a,b]}\, \psi]$ means that there is a path π starting in $\pi^{(0)} = s$ and a number $i \in \mathbb{N}$ so that for the first i states $\pi^{(0)}, \pi^{(1)}, \ldots, \pi^{(i-1)}$ the property φ holds, and ψ holds on $\pi^{(i)}$, and the time $t := \sum_{j=0}^{i-1} \tau_\pi^{(j)}$ *required to reach* state $\pi^{(i)}$ satisfies the numerical relations $a \leq t$ and $t \leq b$.

$\mathcal{K}, s \models \mathsf{E}[\varphi\, \underline{\mathsf{U}}^{\geq a}\, \psi]$ means that there is a path π starting in $\pi^{(0)} = s$ and a number $i \in \mathbb{N}$ so that for the first i states $\pi^{(0)}, \pi^{(1)}, \ldots, \pi^{(i-1)}$ the property φ holds, and ψ holds on $\pi^{(i)}$, and the time $t := \sum_{j=0}^{i-1} \tau_\pi^{(j)}$ *required to reach* state $\pi^{(i)}$ satisfies the numerical relation $a \leq t$.

$\mathcal{K}, s \models \mathsf{EG}^{[a,b]}\varphi$ means that there is a path π starting in $\pi^{(0)} = s$, such that for any state $\pi^{(i)}$ that is reached within a time $t := \sum_{j=0}^{i-1} \tau_\pi^{(j)}$ with $t \in [a, b]$, we have $\pi^{(i)}$. Hence, φ holds in the interval $[a, b]$.

Finally, $\mathcal{K}, s \models \mathsf{EG}^{\geq a}\varphi$ means that there is a path π starting in $\pi^{(0)} = s$, such that for any state $\pi^{(i)}$ that is reached within a time $t := \sum_{j=0}^{i-1} \tau_\pi^{(j)}$ with $t \geq a$, we have $\pi^{(i)}$. Hence, φ holds for all states on π that are reached at time a or after time a.

In the above definition, we have only used basic operators of the logic. Of course, we must introduce some further operators to express some properties directly. For this reason, we give the following abbreviations:

Definition 4 (Further Temporal Operators) *We define further temporal operators in* JCTL *as follows, where p is an arbitrary variable:*

- *Boolean Operators:*
  - $1 := p \vee \neg p$
  - $0 := p \wedge \neg p$
  - $\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi)$
  - $\varphi \rightarrow \psi := \neg\varphi \vee \psi$
- EX *Operators:*
  - $\mathsf{EX}\varphi := \mathsf{EX}^{\geq 0}\varphi$
  - $\mathsf{EX}^{\leq k}\varphi := \mathsf{EX}^{[0,k]}\varphi$
  - $\mathsf{EX}^{<k}\varphi := \mathsf{EX}^{[0,k-1]}\varphi$
  - $\mathsf{EX}^{>k}\varphi := \mathsf{EX}^{\geq k+1}\varphi$
  - $\mathsf{EX}^{=k}\varphi := \mathsf{EX}^{[k,k]}\varphi$
- $\mathsf{E}[\cdot\, \underline{\mathsf{U}}\, \cdot]$ *Operators:*
  - $\mathsf{E}[\varphi\, \underline{\mathsf{U}}\, \psi] := \mathsf{E}[\varphi\, \underline{\mathsf{U}}^{\geq 0}\, \psi]$
  - $\mathsf{E}[\varphi\, \underline{\mathsf{U}}^{\leq k}\, \psi] := \mathsf{E}[\varphi\, \underline{\mathsf{U}}^{[0,k]}\, \psi]$
  - $\mathsf{E}[\varphi\, \underline{\mathsf{U}}^{<k}\, \psi] := \mathsf{E}[\varphi\, \underline{\mathsf{U}}^{[0,k-1]}\, \psi]$
  - $\mathsf{E}[\varphi\, \underline{\mathsf{U}}^{>k}\, \psi] := \mathsf{E}[\varphi\, \underline{\mathsf{U}}^{\geq k+1}\, \psi]$
  - $\mathsf{E}[\varphi\, \underline{\mathsf{U}}^{=k}\, \psi] := \mathsf{E}[\varphi\, \underline{\mathsf{U}}^{[k,k]}\, \psi]$
- EG *Operators:*
  - $\mathsf{EG}\varphi := \mathsf{EG}^{\geq 0}\varphi$
  - $\mathsf{EG}^{\leq k}\varphi := \mathsf{EG}^{[0,k]}\varphi$
  - $\mathsf{EG}^{<k}\varphi := \mathsf{EG}^{[0,k-1]}\varphi$
  - $\mathsf{EG}^{>k}\varphi := \mathsf{EG}^{\geq k+1}\varphi$
  - $\mathsf{EG}^{=k}\varphi := \mathsf{EG}^{[k,k]}\varphi$
- *Let κ be any time constraint, i.e., $[a, b]$, $\sim k$ with $\sim \in \{<, \leq, =, \leq, >\}$, or the empty constraint. Then, we define the following operators:*
  - $\mathsf{AX}^{\kappa}\varphi := \neg\mathsf{EX}^{\kappa}\neg\varphi$
  - $\mathsf{A}[\varphi\, \underline{\mathsf{U}}^{\kappa}\, \psi] := \neg\mathsf{E}[(\neg\psi)\, \underline{\mathsf{U}}^{\kappa}\, \neg(\varphi \vee \psi)] \wedge \neg\mathsf{EG}^{\kappa}\neg\psi$
  - $\mathsf{EF}^{\kappa}\varphi := \mathsf{E}[1\, \underline{\mathsf{U}}^{\kappa}\, \varphi]$
  - $\mathsf{AF}^{\kappa}\varphi := \mathsf{A}[1\, \underline{\mathsf{U}}^{\kappa}\, \varphi]$
  - $\mathsf{AG}^{\kappa}\varphi := \neg\mathsf{EF}^{\kappa}(\neg\varphi)$
  - $\mathsf{E}[\varphi\, \underline{\mathsf{B}}^{\kappa}\, \psi] := \mathsf{E}[(\neg\psi)\, \underline{\mathsf{U}}^{\kappa}\, (\varphi \wedge \neg\psi)]$
  - $\mathsf{A}[\varphi\, \underline{\mathsf{B}}^{\kappa}\, \psi] := \mathsf{A}[(\neg\psi)\, \underline{\mathsf{U}}^{\kappa}\, (\varphi \wedge \neg\psi)]$
  - $\mathsf{E}[\varphi\, \mathsf{U}^{\kappa}\, \psi] := \neg\mathsf{A}[(\neg\varphi)\, \underline{\mathsf{B}}^{\kappa}\, \psi]$
  - $\mathsf{A}[\varphi\, \mathsf{U}^{\kappa}\, \psi] := \neg\mathsf{E}[(\neg\varphi)\, \underline{\mathsf{B}}^{\kappa}\, \psi]$
  - $\mathsf{E}[\varphi\, \mathsf{B}^{\kappa}\, \psi] := \neg\mathsf{A}[(\neg\varphi)\, \underline{\mathsf{U}}^{\kappa}\, \psi]$
  - $\mathsf{A}[\varphi\, \mathsf{B}^{\kappa}\, \psi] := \neg\mathsf{E}[(\neg\varphi)\, \underline{\mathsf{U}}^{\kappa}\, \psi]$

The definitions of the further EX, $\mathsf{E}[\cdot\, \underline{\mathsf{U}}\, \cdot]$, and EX operators should be clear. $\mathsf{EF}^{\kappa}\varphi$ holds in state s iff a state can be reached where φ holds and that state can be reached within a time that satisfies the the time constraint κ.

$\mathsf{E}[\varphi \underline{\mathsf{B}}^{\kappa} \psi]$ means that there must be a path π and a number $i \in \mathbb{N}$ such that φ holds on $\pi^{(i)}$, and all states $\pi^{(j)}$ with $j \leq i$ do not satisfy ψ, and the time required to reach $\pi^{(i)}$ satisfies the time constraint κ (hence, φ holds *before* ψ).

We also have weak variants $\mathsf{E}[\cdot\, \mathsf{U}\, \cdot]$ and $\mathsf{E}[\cdot\, \mathsf{B}\, \cdot]$ of $\mathsf{E}[\cdot\, \underline{\mathsf{U}}\, \cdot]$ and $\mathsf{E}[\cdot\, \underline{\mathsf{B}}\, \cdot]$, respectively, that do not demand that the events that are awaited for must actually occur. $\mathsf{E}[\varphi\, \underline{\mathsf{U}}^{\kappa}\, \psi]$ implies $\mathsf{E}[\varphi\, \mathsf{U}^{\kappa}\, \psi]$, but $\mathsf{E}[\varphi\, \mathsf{U}^{\kappa}\, \psi]$ may also hold, if there is a path π where ψ can not be reached in time. In this case, φ must hold on those states on the path that can be reached within a time that satisfies the constraint κ. $\mathsf{E}[\cdot\, \mathsf{B}\, \cdot]$ is defined in a similar way. Finally, the versions with the A path quantifier are defined such that the corresponding path property must hold for all paths leaving the state.

As can be seen, the $\mathsf{EG}^{[a,b]}$ operator states that some property holds for *all* states that can be reached within $[a, b]$, while $\mathsf{EF}^{[a,b]}\cdot$ states a property for *some* point of time in that interval. Nevertheless, the following lemma holds, which shows that the $\mathsf{EG}^{\leq k}$ operator is a somehow hybrid operator that makes both a universal and an existential statement (compare G_2 and G_3 in the following lemma). This is due to the fact, that the equation $\mathsf{EG}^{\leq k}\varphi = \mathsf{E}[\varphi\, \underline{\mathsf{U}}^{>k}\, 1]$ is valid:

Lemma 1 (Semantics of $\mathsf{EG}^{\leq k}\varphi$) *Given a* JCTL *formula φ and a number $k \in \mathbb{N}$. Then, the following properties are equivalent for any TKS $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$:*

(G_1) $\mathcal{K}, s \models \mathsf{EG}^{\leq k}\varphi$

(G_2) *there is a path* $\pi \in \mathsf{Paths}_{\mathcal{K}}(s)$ *starting in state* s*, such that for all numbers* $i \in \mathbb{N}$*, we have*

$$\left(\sum_{j=0}^{i-1} \tau_\pi^{(j)} \leq k\right) \rightarrow \left(\mathcal{K}, \pi^{(i)} \models \varphi\right)$$

(G_3) *there is a path* $\pi \in \mathsf{Paths}_{\mathcal{K}}(s)$ *starting in state* s *and a number* $i \in \mathbb{N}$ *such that*

$$\left(\sum_{j=0}^{i-1} \tau_\pi^{(j)} \leq k < \sum_{j=0}^{i} \tau_\pi^{(j)}\right) \wedge \left(\forall j \leq i.\ \mathcal{K}, \pi^{(j)} \models \varphi\right)$$

(G_4) *there is a path* $\pi \in \mathsf{Paths}_{\mathcal{K}}(s)$ *starting in state* s *and a number* $i \in \mathbb{N}$ *such that*

$$\left(\sum_{j=0}^{i-1} \tau_\pi^{(j)} > k\right) \wedge \left(\forall j < i.\ \mathcal{K}, \pi^{(j)} \models \varphi\right)$$

(G_5) $\mathcal{K}, s \models \mathsf{E}[\varphi\ \underline{\mathsf{U}}^{>k}\ 1]$

The proof of the above lemma is not very difficult. We just make use of the well-ordering of natural numbers, i.e., if there is a number with some property, then there is also a least number with the same property.

2.3. Expansion of TKSs

Normal Kripke structures are special cases of TKS that are obtained by restricting TKSs so that $(s, t, s') \in \mathcal{R}$ implies $t = 1$. To avoid confusion, we call the 'normal Kripke structures' unit delay structures in the following:

Definition 5 (Unit Delay Structure (UDS)) *A TKS* $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$ *is called to be unit delay structure iff* $\mathcal{T}_{\mathcal{K}} := \{t \mid (s, t, s') \in \mathcal{R}\} = \{1\}$.

The previously mentioned approaches to define real-time logics that rely on interpretation I_2 are forced to expand TKSs in order to define the semantics of their logics. For example, the following expansion algorithm could be used for that purpose:

Definition 6 (Expansion of TKS) *Given a TKS* $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$*, we compute* $(\mathcal{I}_e, \mathcal{S}_e, \mathcal{R}_e) = \mathsf{expand}(\mathcal{I}, \mathcal{S}, \mathcal{R})$ *with the function* expand *as defined in Figure 2. Moreover, we define for any* $(s, u) \in \mathcal{S}_e$ *the label function* $\mathcal{L}_e((s, u)) := \mathcal{L}(s)$*. The expansion of* $\mathcal{K}$ *is then the unit delay structure* $\mathcal{K}_e := (\mathcal{I}_e, \mathcal{S}_e, \mathcal{R}_e, \mathcal{L}_e)$.

As can be seen, the expansion relies on interpretation I_2, since we defined $\mathcal{L}_e((s, u)) := \mathcal{L}(s)$, i.e., the states of $\mathcal{K}_e$ have the same variable assignments as the corresponding states of $\mathcal{K}$. For conciseness, we use the following definition:

```
function expand(I, S, R)
  S_e := {(s,1) | s ∈ S};
  R_e := {};
  for (s,t,s') ∈ R do
    for i := 2 to t do
      S_e := S_e ∪ {(s,i)};
      R_e := R_e ∪ {((s,i-1),(s,i))};
    end for;
  R_e := R_e ∪ {((s,t),(s',1))};
  end for;
  I_e := {(s,1) | s ∈ I};
  return (I_e, S_e, R_e);
end function
```

Figure 2. Expansion of TKS to UDS

Definition 7 (Tracks of a State) *Given a TKS* $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$*, its expanded structure* $\mathcal{K}_e := (\mathcal{I}_e, \mathcal{S}_e, \mathcal{R}_e, \mathcal{L}_e)$*, and a state* $s \in \mathcal{S}$*. Then, we define* $\mathsf{Track}_{\mathcal{K}}(s) = \{(s', u) \in \mathcal{S}_e \mid s = s'\}$.

We emphasize that expansions of TKS can be performed in many different ways that are not equivalent to each other [13]. Furthermore, real-time model checking problems can not be simply reduced to ordinary CTL model checking problems, although this is widely believed [13]. For this reason, we must extend the usual CTL model checking procedure to capture JCTL. This is shown in the next section.

3. Real Time Model Checking on TKSs

In this section, we present a model checking algorithm for our real-time logic JCTL. We have implemented this algorithm in our verification tool JERRY by using the CUDD BDD library [17]. The underlying algorithms for the basic operators are given in Figure 4.

The essential idea to reason about the real-time constraints is to move fronts of tracks on a virtual expanded structure. However, we emphasize that *we do never expand the structure*. Furthermore, we do not run into semantic problems since the result of any evaluation of a logical operator is *a set of states* instead of a set of tracks. *Hence, all calculations are independent of the virtual expansion.* This is achieved by abstraction of the set of tracks, so that the semantics of the evaluated temporal operator is respected. In general, there are two possibilities: On the one hand, a state s belongs to the result if its main track $(s, 1)$ belongs to the track set, on the other hand it may be sufficient if anyone of its tracks (s, t) belongs to the track set. The choice between the two possibilities depends on the semantics of the considered temporal operator (see Figure 4).

The key function for the evaluation of all real-time constraints is the function MoveFront. Given a set of tracks

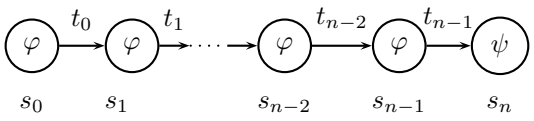

Figure 3. Correctness of MoveFront

$\mathcal{T}_\psi$ and a set of states $\mathcal{S}_\varphi$, this function computes the set of tracks that have a path of a certain length through the tracks $\mathcal{S}_\varphi \times \mathbb{N}$. The precise specification is as follows:

Lemma 2 (Correctness of MoveFront**)** *Given a TKS $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$, a set of states $\mathcal{S}_\varphi$, and a set of tracks $\mathcal{T}_\psi$, the function* MoveFront *as given in Figure 4 satisfies the following equations for $\sim \in \{=, \geq\}$ (cf. Figure 3):*

$$(s_0, t) \in \mathsf{MoveFront}(\sim, k, \mathcal{S}_\varphi, \mathcal{T}_\psi) \Leftrightarrow \left(\begin{array}{l} \exists s_1, \ldots, s_{n-1} \in \mathcal{S}_\varphi . \exists (s_n, d) \in \mathcal{T}_\psi . \\ \quad \exists t_0, \ldots, t_{n-1} \in \mathbb{N}. \\ \qquad \bigwedge_{i=0}^{n-1} (s_i, t_i, s_{i+1}) \in \mathcal{R} \wedge \\ \qquad k \sim \left(\sum_{i=0}^{n-1} t_i \right) + d - t \wedge \\ \qquad t \leq t_0 \end{array} \right)$$

Hence, $\mathsf{MoveFront}(\sim, k, \mathcal{S}_\varphi, \mathcal{T}_\psi)$ *computes the set of tracks that have a path through any expanded structure of length ℓ with $\ell \sim k$ to a track in $\mathcal{T}_\psi$ which runs only through tracks of $\mathcal{S}_\varphi \times \mathbb{N}$.*

Proof: The correctness easily follows by induction on k, when we observe that our algorithm and the right hand sides of the above equivalence both satisfy the following recursion equations (note that primitive recursive definitions are uniquely determined):

- $\mathsf{MoveFront}(\sim, 0, \mathcal{S}_\varphi, \mathcal{T}_\psi) = \mathcal{T}_\psi$
- $\mathsf{MoveFront}(=, k+1, \mathcal{S}_\varphi, \mathcal{T}_\psi)$ $= (\mathcal{S}_\varphi \times \mathbb{N}) \cap \mathsf{preTracks}(\mathsf{MoveFront}(=, k, \mathcal{S}_\varphi, \mathcal{T}_\psi))$
- $\mathsf{MoveFront}(\geq, k+1, \mathcal{S}_\varphi, \mathcal{T}_\psi)$ $= \left(\begin{array}{l} \mathsf{let}\ \mathcal{T}_0 := \mathsf{MoveFront}(\geq, k, \mathcal{S}_\varphi, \mathcal{T}_\psi) \\ \mathsf{in}\ \ \mathcal{T}_0 \cup ((\mathcal{S}_\varphi \times \mathbb{N}) \cap \mathsf{preTracks}(\mathcal{T}_0)) \end{array} \right)$

Using these equations, we can easily prove that $\mathcal{T}_0 = \mathsf{MoveFront}(\sim, i, \mathcal{S}_\varphi, \mathcal{T}_\psi)$ is an invariant of the loop in the algorithm given in Figure 4 for MoveFront. This directly implies the correctness of the above lemma. ■

Please note that in the algorithm given in Figure 4 to implement the function MoveFront, we use $(i \neq k) \wedge (\mathcal{T}_0 \neq \mathcal{T}_1)$ as loop condition instead of $(i \neq k)$ (which would also be correct). The reason for this is that whenever $\mathcal{T}_0 = \mathcal{T}_1$ holds for an iteration $i < k$, then it follows that all fronts $\mathsf{MoveFront}(\sim, i, \mathcal{S}_\varphi, \mathcal{T}_\psi)$, …, $\mathsf{MoveFront}(\sim, k, \mathcal{S}_\varphi, \mathcal{T}_\psi)$ would be identical, so that we already have the result in this case.

Now consider the function $\mathsf{StatesEU}^{[a,b]}$. We first compute the set of tracks $\mathcal{T}_0$ that can reach a track of $\mathcal{S}_\psi \times \{1\}$ in exactly a steps (where only tracks of the states $\mathcal{S}_\varphi$ are traversed). After this, we move the tracks $\mathcal{T}_0$ by further $b - a$ unit delay steps through the tracks of the states $\mathcal{S}_\varphi$. By the above lemma, we then obtain

$$(s'_0, t') \in \mathcal{T}_1 \Leftrightarrow \left(\begin{array}{l} \exists s'_1, \ldots, s'_m \in \mathcal{S}_\varphi . \exists t'_0, \ldots, t'_{m-1} \in \mathbb{N}. \\ \exists s_0, \ldots, s_{n-1} \in \mathcal{S}_\varphi . \exists t_0, \ldots, t_{n-1} \in \mathbb{N}. \\ \exists s_n \in \mathcal{S}_\psi . \exists t \in \mathbb{N}. \\ \quad \bigwedge_{i=0}^{m-1} (s'_i, t'_i, s'_{i+1}) \in \mathcal{R} \wedge \\ \quad (s'_m = s_0) \in \mathcal{R} \wedge \\ \quad \bigwedge_{i=0}^{n-1} (s_i, t_i, s_{i+1}) \in \mathcal{R} \wedge \\ \quad a = \left(\sum_{i=0}^{n-1} t_i \right) + 1 - t \wedge \\ \quad b \geq \left(\sum_{i=0}^{m-1} t'_i \right) + \left(\sum_{i=0}^{n-1} t_i \right) + 1 - t' \wedge \\ \quad t \leq t_0 \wedge t' \leq t'_0 \end{array} \right)$$

Note that $\left(\sum_{i=0}^{m-1} t'_i \right) + \left(\sum_{i=0}^{n-1} t_i \right) + 1 - t'$ is the number of unit delay steps that are required to reach track $(s_n, 1) \in \mathcal{S}_\psi \times \{1\}$ from track (s'_0, t'). By the above result, it follows that this time is in the interval $[a, b]$ (consider the case $m = 0$, where the second MoveFront went along a single transition, and the case where $m > 0$ holds). The final step is to translate this result (given for tracks) to sets of states. If $(s'_0, 1) \in \mathcal{T}_1$ holds, then we clearly see that state s_0 belongs to $\left[\!\left[\mathsf{E}[\varphi\ \underline{\mathsf{U}}^{[a,b]}\ \psi] \right]\!\right]_{\mathcal{K}}$. If, on the other hand, $(s'_0, t') \in \mathcal{T}_1$ holds for some $t' > 1$, but $(s'_0, 1)$ is not included in $\mathcal{T}_1$, then it follows by the above formula that the time to reach $(s_n, 1)$ from $(s'_0, 1)$ is larger than b, so that $s'_0 \notin \left[\!\left[\mathsf{E}[\varphi\ \underline{\mathsf{U}}^{[a,b]}\ \psi] \right]\!\right]_{\mathcal{K}}$.

The correctness of $\mathsf{EG}^{[a,b]}\varphi$ is seen as follows: $\mathcal{T}_0$ is the set of tracks that have a path of length $b - a$ through $\mathcal{S}_\varphi \times \mathbb{N}$ in the expanded structure. $\mathcal{T}_1$ is the set of tracks of $\mathcal{S}_\varphi \times \mathbb{N}$ that can reach $\mathcal{T}_0$ while only traversing tracks of $\mathcal{S}_\varphi \times \mathbb{N}$. Hence, $\mathcal{T}_1$ is the set of tracks that have a path of length $\geq b - a$ through $\mathcal{S}_\varphi \times \mathbb{N}$ in the expanded structure. Finally, $\mathcal{T}_2$ is the set of tracks of $\mathcal{S} \times \mathbb{N}$ that can reach $\mathcal{T}_1$ in exactly a steps. Clearly, we only should consider those tracks with time stamp 1 of $\mathcal{T}_2$.

The correctness of the other two functions to evaluate $\mathsf{E}[\varphi\ \underline{\mathsf{U}}^{\geq a}\ \psi]$ and $\mathsf{EG}^{\geq a}\varphi$ are proved in a similar way. Therefore, we obtain the following correctness result:

Theorem 1 (Correctness of Function States**)** *For any TKS $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$ and any* JCTL *formula φ, the function* States *given in Figure 4 satisfies the equation* $\mathsf{States}(\varphi) = [\![\varphi]\!]_{\mathcal{K}}$.

function preStates($\mathcal{S}_0$)
  $\mathcal{S}_1 := \left\{ s \in \mathcal{S} \;\middle|\; \begin{array}{r} \exists s' \in \mathcal{S}_0 . \exists t \in \mathbb{N}. \\ (s,t,s') \in \mathcal{R} \end{array} \right\}$;
  return $\mathcal{S}_1$;
end function

function preTracks($\mathcal{T}$)
  $\mathcal{T}_1 := \{(s,t) \mid (s,t+1) \in \mathcal{T}\}$;
  $\mathcal{T}_2 := \{(s,t) \mid \exists (s',1) \in \mathcal{T} . (s,t,s') \in \mathcal{R}\}$;
  return $\mathcal{T}_1 \cup \mathcal{T}_2$;
end function

function MoveFront($\sim, k, \mathcal{S}_\varphi, \mathcal{T}_\psi$)
  $\mathcal{T}_\varphi := \mathcal{S}_\varphi \times \mathbb{N}$;
  $\mathcal{T}_0 := \mathcal{T}_\psi$;
  $\mathcal{T}_1 := \{\}$;
  $i := 0$;
  while $(i \neq k) \wedge (\mathcal{T}_0 \neq \mathcal{T}_1)$ **do**
    $\mathcal{T}_1 := \mathcal{T}_0$;
    $\mathcal{T}_0 := \mathcal{T}_\varphi \cap$ preTracks($\mathcal{T}_1$);
    if equal($\sim, \geq$) **then** $\mathcal{T}_0 := \mathcal{T}_1 \cup \mathcal{T}_0$ **endif**;
    $i := i + 1$;
  end;
  return $\mathcal{T}_0$;
end function

function StatesEU$^{[a,b]}$($\mathcal{S}_\varphi, \mathcal{S}_\psi$)
  $\mathcal{T}_0 :=$ MoveFront($=, a, \mathcal{S}_\varphi, \mathcal{S}_\psi \times \{1\}$);
  $\mathcal{T}_1 :=$ MoveFront($\geq, b-a, \mathcal{S}_\varphi, \mathcal{T}_0$);
  return $\{s \in \mathcal{S} \mid (s,1) \in \mathcal{T}_1\}$;
end function

function StatesEU$^{\geq a}$($\mathcal{S}_\varphi, \mathcal{S}_\psi$)
  $\mathcal{T}_0 :=$ MoveFront($=, a, \mathcal{S}_\varphi, \mathcal{S}_\psi \times \{1\}$);
  $\mathcal{S}_0 := \{s \in \mathcal{S} \mid \exists t \in \mathbb{N} . (s,t) \in \mathcal{T}_0\}$;
  repeat
    $\mathcal{S}_1 := \mathcal{S}_0$;
    $\mathcal{S}_0 := \mathcal{S}_0 \cup (\mathcal{S}_\varphi \cap$ preStates($\mathcal{S}_1$)) ;
  until $(\mathcal{S}_0 = \mathcal{S}_1)$;
  return $\mathcal{S}_0$;
end function

function StatesEG$^{[a,b]}$($\mathcal{S}_\varphi$)
  $\mathcal{T}_0 :=$ MoveFront($=, (b-a)+1, \mathcal{S}_\varphi, \mathcal{S} \times \{1\}$);
  $\mathcal{T}_1 :=$ MoveFront($\geq, -1, \mathcal{S}_\varphi, \mathcal{T}_0$);
  $\mathcal{T}_2 :=$ MoveFront($=, a, \mathcal{S}, \mathcal{T}_1$);
  return $\{s \in \mathcal{S} \mid (s,1) \in \mathcal{T}_2\}$;
end function

function StatesEG$^{\geq a}$($\mathcal{S}_\varphi$)
  $\mathcal{S}_0 := \mathcal{S}_\varphi$;
  repeat
    $\mathcal{S}_1 := \mathcal{S}_0$;
    $\mathcal{S}_0 := \mathcal{S}_\varphi \cap$ preStates($\mathcal{S}_1$);
  until $\mathcal{S}_0 = \mathcal{S}_1$;
  (* *we now have* $S_0 = [\![\mathsf{EG}\varphi]\!]_\mathcal{K}$ *)
  $\mathcal{T}_0 :=$ MoveFront($\geq, a, \mathcal{S}, \mathcal{S}_0 \times \{1\}$);
  return $\{s \in \mathcal{S} \mid (s,1) \in \mathcal{T}_0\}$;
end function

function States(Φ)
  case Φ **of**
    is_var(Φ) : **return** $\{s \in \mathcal{S} \mid \Phi \in \mathcal{L}(s)\}$;
    $\neg\varphi$: $\mathcal{S}_\varphi :=$ States(φ);
      return $\mathcal{S} \setminus \mathcal{S}_\varphi$;
    $\varphi \wedge \psi$: $\mathcal{S}_\varphi :=$ States(φ); $\mathcal{S}_\psi :=$ States(ψ);
      return $\mathcal{S}_\varphi \cap \mathcal{S}_\psi$;
    $\mathsf{EX}^{[a,b]}\varphi$: $\mathcal{S}_\varphi :=$ States(φ);
      $\mathcal{S}_0 := \left\{ s \in \mathcal{S} \;\middle|\; \begin{array}{r} \exists s' \in \mathcal{S}_\varphi . \exists t \in [a,b]. \\ (s,t,s') \in \mathcal{R} \end{array} \right\}$;
      return $\mathcal{S}_0$;
    $\mathsf{EX}^{\geq a}\varphi$: $\mathcal{S}_\varphi :=$ States(φ);
      $\mathcal{S}_0 := \left\{ s \in \mathcal{S} \;\middle|\; \begin{array}{r} \exists s' \in \mathcal{S}_\varphi . \exists t \geq a. \\ (s,t,s') \in \mathcal{R} \end{array} \right\}$;
      return $\mathcal{S}_0$;
    $\mathsf{E}[\varphi\ \underline{\mathsf{U}}^{[a,b]}\ \psi]$: $\mathcal{S}_\varphi :=$ States(φ); $\mathcal{S}_\psi :=$ States(ψ);
      return StatesEU$^{[a,b]}$($\mathcal{S}_\varphi, \mathcal{S}_\psi$);
    $\mathsf{E}[\varphi\ \underline{\mathsf{U}}^{\geq a}\ \psi]$: $\mathcal{S}_\varphi :=$ States(φ); $\mathcal{S}_\psi :=$ States(ψ);
      return StatesEU$^{\geq a}$($\mathcal{S}_\varphi, \mathcal{S}_\psi$);
    $\mathsf{EG}^{[a,b]}\varphi$: $\mathcal{S}_\varphi :=$ States(φ);
      return StatesEG$^{[a,b]}$($\mathcal{S}_\varphi$);
    $\mathsf{EG}^{\geq a}\varphi$: $\mathcal{S}_\varphi :=$ States(φ);
      return StatesEG$^{\geq a}$($\mathcal{S}_\varphi$);
end function

Figure 4. Model Checking of JCTL Formulas on a Timed Kripke Structure $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$

4. Complexity of JCTL

In this section, we analyze the complexity of the presented model checking algorithm for JCTL. For this reason, we first note that there is a model checking procedure for CTL, e.g. the one given in [5], that runs in time $O(|\varphi|\,(|\mathcal{R}|+|\mathcal{S}|))$. This procedure is able to evaluate any CTL operator in time $O(|\varphi|\,(|\mathcal{R}|+|\mathcal{S}|))$.

It is easily seen that if we expand a TKS to a UDS, the number of states and transition is multiplied with the maximum delay time $\hat{\tau}_{\mathcal{K}}$ that appears in $\mathcal{K}$. However, the runtime of the JCTL model checking procedure is not in $O(\hat{\tau}_{\mathcal{K}}\,|\varphi|\,(|\mathcal{R}|+|\mathcal{S}|))$, since the number of iterations does also depend on the time constraints of the temporal operators that may enforce more than $\hat{\tau}_{\mathcal{K}}\,|\mathcal{S}|$ iterations. The crucial part of our complexity analysis is the complexity of the MoveFront function. To this end, we note that the following holds:

Lemma 3 (Complexity of MoveFront (I)) *Given a TKS $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$, a set of states $\mathcal{S}_\varphi$, and a set of tracks $\mathcal{T}_\psi$. Let moreover be $\mathcal{K}_e = (\mathcal{I}_e, \mathcal{S}_e, \mathcal{R}_e, \mathcal{L}_e)$ the expanded structure of $\mathcal{K}$ according to definition 6, and φ and ψ formulas so that the equations $\mathcal{S}_\varphi \times \mathbb{N} \cap \mathcal{S}_e = [\![\varphi]\!]_{\mathcal{K}_e}$ and $\mathcal{T}_\psi = [\![\psi]\!]_{\mathcal{K}_e}$ hold. Then, we have*

$$\mathsf{MoveFront}(\sim, k, \mathcal{S}_\varphi, \mathcal{T}_\psi) = [\![\Phi(\sim, k, \varphi, \psi)]\!]_{\mathcal{K}_e},$$

where the formula $\Phi(\sim, k, \varphi, \psi)$ is recursively defined as follows:

- $\Phi(\sim, 0, \varphi, \psi) = \psi$
- $\Phi(=, k+1, \varphi, \psi) = \varphi \wedge \mathsf{EX}\Phi(=, k, \varphi, \psi)$
- $\Phi(\geq, k+1, \varphi, \psi) = \begin{pmatrix} \mathsf{let} & y = \Phi(\geq, k, \varphi, \psi) \\ \mathsf{in} & y \vee \varphi \wedge \mathsf{EX}y \\ \mathsf{end} & \end{pmatrix}$

Moreover, if common subformulas are shared, it is easily seen that $|\Phi(\sim, k, \varphi, \psi)| \in O(k)$ holds. Hence, $\mathsf{MoveFront}(=, k, \mathcal{S}_\varphi, \mathcal{T}_\psi)$ can be computed in time $O(k(|\mathcal{R}_e| + |\mathcal{S}_e|))$, and $\mathsf{MoveFront}(\geq, k, \mathcal{S}_\varphi, \mathcal{T}_\psi)$ even in time $O(\min\{k, |\mathcal{S}_e|\}(|\mathcal{R}_e| + |\mathcal{S}_e|))$.

The complexity can also be directly derived from the implementation of MoveFront: Note that the value of $\mathcal{T}_0$ monotonically grows in function calls of $\mathsf{MoveFront}(\geq, k, \ldots)$, but not for calls of $\mathsf{MoveFront}(=, k, \ldots)$. Therefore, $\mathsf{MoveFront}(\geq, k, \ldots)$ runs in time $O(\min\{k, |\mathcal{S}_e|\}(|\mathcal{R}_e| + |\mathcal{S}_e|))$, but $\mathsf{MoveFront}(=, k, \ldots)$ requires time $O(k(|\mathcal{R}_e| + |\mathcal{S}_e|))$. For this reason, we have the following result:

Theorem 2 (Complexity of JCTL) *For any TKS $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$ and any JCTL formula φ, the function States given in Figure 4 runs in time $O(\hat{k}_\varphi\,|\varphi|\,\hat{\tau}_{\mathcal{K}}(|\mathcal{R}| + |\mathcal{S}|))$, where $\hat{\tau}_{\mathcal{K}} := \max\{t \mid \exists s, s'.(s, t, s') \in \mathcal{R}\}$ is the maximum delay time of $\mathcal{K}$, and $\hat{k}_\varphi$ is the maximal number used in time constraints in φ.*

The proof can be obtained by induction along the JCTL formulas. The induction steps are thereby obtained by the following facts, where $\mathsf{Time}(f)$ denotes the runtime of function f:

- $\mathsf{Time}(\mathsf{preStates}) \in O(|\mathcal{R}| + |\mathcal{S}|)$
- $\mathsf{Time}(\mathsf{preTracks}) \in O(\hat{\tau}_{\mathcal{K}}(|\mathcal{R}| + |\mathcal{S}|))$
- $\mathsf{Time}(\mathsf{StatesEU}^{[a,b]}) \in O(k\hat{\tau}_{\mathcal{K}}(|\mathcal{R}| + |\mathcal{S}|))$, where $k := \max\{a, \min\{b-a, \hat{\tau}_{\mathcal{K}}\,|\mathcal{S}|\}\} \leq b$
- $\mathsf{Time}(\mathsf{StatesEU}^{\geq a}) \in O(a\hat{\tau}_{\mathcal{K}}(|\mathcal{R}| + |\mathcal{S}|))$
- $\mathsf{Time}(\mathsf{StatesEG}^{[a,b]}) \in O(k\hat{\tau}_{\mathcal{K}}(|\mathcal{R}| + |\mathcal{S}|))$, where $k := \max\{b-a, a, 1\} \leq b$
- $\mathsf{Time}(\mathsf{StatesEG}^{\geq a}) \in O(k\hat{\tau}_{\mathcal{K}}(|\mathcal{R}| + |\mathcal{S}|))$, where $k := \min\{a, \hat{\tau}_{\mathcal{K}}\,|\mathcal{S}|\,, 1\} \leq a$

Hence, all operators can be evaluated in time $O(\hat{k}_\varphi\hat{\tau}_{\mathcal{K}}(|\mathcal{R}|+|\mathcal{S}|))$. As a formula φ may contain $|\varphi|$ operators, the above theorem follows.

Hence, one key to define a more efficient fragment of JCTL is to avoid calls of the form $\mathsf{MoveFront}(=, k, \ldots)$, since this is not as efficient as $\mathsf{MoveFront}(\geq, k, \ldots)$. Using a specialized algorithm similar to the one given in [5], we can even improve the complexity for computing $\mathsf{MoveFront}(\geq, k, \ldots)$:

Lemma 4 (Complexity of MoveFront (II)) *Given a TKS $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$, a set of states $\mathcal{S}_\varphi$, and a set of tracks $\mathcal{T}_\psi$. Let moreover be $\mathcal{K}_e = (\mathcal{I}_e, \mathcal{S}_e, \mathcal{R}_e, \mathcal{L}_e)$ the expanded structure of $\mathcal{K}$ according to definition 6, and φ and ψ formulas so that the equations $\mathcal{S}_\varphi \times \mathbb{N} \cap \mathcal{S}_e = [\![\varphi]\!]_{\mathcal{K}_e}$ and $\mathcal{T}_\psi = [\![\psi]\!]_{\mathcal{K}_e}$ hold. Then, there is an algorithm to compute $\mathsf{MoveFront}(\geq, k, \mathcal{S}_\varphi, \mathcal{T}_\psi)$ in time $O(|\mathcal{R}_e| + |\mathcal{S}_e|)$.*

The specialized algorithm is similar to the one given in [5], but it needs to additionally take care of the lengths of the paths that have reached a certain track. For this reason, we maintain for any track $s_i \in \mathcal{S}_e$ a list $T_i = [s_{i,1}, \ldots, s_{i,m_i}]$ such that $(s_{i,j}, s_i)$ is a transition in $\mathcal{R}_e$. Any track s_i will be marked with a number m_i later on that is the minimal length of a path to reach the set $\mathcal{T}_\psi$ from s_i. Furthermore, we create lists L_ℓ during the computations that contain the tracks that are marked with the number ℓ. The algorithm performs then the following steps:

Step 1: We eliminate all $s_{i,j}$ in each list T_i that does not belong to $\mathcal{T}_\psi$. This can be done in time $O(|\mathcal{R}_e|)$, since we look at each transition once.

Step 2: We mark the tracks $\mathcal{T}_\psi$ with the number 0, and list them in our first list L_0. Let $\ell := 0$. This step is performed in time $O(|\mathcal{S}_e|)$, since we look at each track at most once.

Step 3**:** For each track s_i in L_ℓ, and each track $s_{i,j}$ of T_i, we mark $s_{i,j}$ with $\ell+1$ if it is not already marked, and put $s_{i,j}$ in list $L_{\ell+1}$ in this case. We eliminate $s_{i,j}$ from T_i. We then increment ℓ, and repeat this step until $\ell = k$ holds or no transitions are left in the lists T_i.

Clearly, the repeated execution of step 3 does look at each transition at most once, so that $O(|\mathcal{R}_e|)$ is an upper bound for its complexity. Hence, we see that the entire algorithm runs in time $O(|\mathcal{R}_e| + |\mathcal{S}_e|)$.

Note that the above sketched algorithm works with a depth-first search and therefore performs – in theory – better than breadth-first searches like those used by symbolic model checking based on BDDs. However, in practice, the latter approaches are in general superior.

The next step is to define a subset of JCTL that can be computed without calls $\mathsf{MoveFront}(=, k, \cdot, \cdot)$. Clearly, we must therefore forbid the interval constraints, and therefore are restricted to the basic operators $\mathsf{EX}^{[a,b]}$, $\mathsf{EX}^{\geq a}$, $\mathsf{E}[\cdot\ \underline{\mathsf{U}}^{[0,a]}\ \cdot]$, and $\mathsf{EG}^{\geq a}$ only. Of course, we can still use all macro operators that can be defined from these basic operators like, for example, $\mathsf{E}[\varphi\ \underline{\mathsf{U}}^{\leq a}\ \psi] := \mathsf{E}[\varphi\ \underline{\mathsf{U}}^{[0,a]}\ \psi]$, $\mathsf{EF}^{\leq a}\varphi := \mathsf{E}[1\ \underline{\mathsf{U}}^{\leq a}\ \varphi]$, and $\mathsf{EG}^{\leq a}\varphi := \mathsf{E}[\varphi\ \underline{\mathsf{U}}^{>a}\ 1]$ (cf. lemma 1). The following interesting theorem allows us however to further extend this set of formulas:

Theorem 3 *The following equations are valid, and therefore allow to reduce $\geq$ constraints to $\leq$ constraints:*

- $\mathsf{E}[\varphi\ \mathsf{U}^{\geq k+1}\ \psi] = \mathsf{EF}^{\leq k}(\psi \vee \mathsf{EXE}[\varphi\ \mathsf{U}\ \psi])$
- $\mathsf{E}[\varphi\ \mathsf{B}^{\geq k+1}\ \psi] = \mathsf{EF}^{\leq k}(\varphi \vee \mathsf{EXE}[\varphi\ \mathsf{B}\ \psi])$
- $\mathsf{EG}^{\geq k+1}\varphi = \mathsf{EF}^{\leq k}\mathsf{EXEG}\varphi$
- $\mathsf{A}[\varphi\ \underline{\mathsf{U}}^{\geq k+1}\ \psi] = \mathsf{AG}^{\leq k}(\varphi \wedge \mathsf{AXA}[\varphi\ \underline{\mathsf{U}}\ \psi])$
- $\mathsf{A}[\varphi\ \underline{\mathsf{B}}^{\geq k+1}\ \psi] = \mathsf{AG}^{\leq k}(\neg\psi \wedge \mathsf{AXA}[\varphi\ \underline{\mathsf{B}}\ \psi])$
- $\mathsf{AF}^{\geq k+1}\varphi = \mathsf{AG}^{\leq k}\mathsf{AXAF}\varphi$

Moreover, constraints ≥ 0 can be simply omitted due to the equations $\mathsf{E}[\varphi\ \mathsf{U}^{\geq 0}\ \psi] = \mathsf{E}[\varphi\ \mathsf{U}\ \psi]$ and $\mathsf{E}[\varphi\ \mathsf{B}^{\geq 0}\ \psi] = \mathsf{E}[\varphi\ \mathsf{B}\ \psi]$, and $\mathsf{EG}^{\geq 0}\varphi = \mathsf{EG}\varphi$.

The above equations allow us to split the temporal expressions on the left hand side into a quantitative and a qualitative part. The latter is a simple CTL formula, and can therefore be directly checked on the Kripke structure that is obtained by omitting the time constraints of the TKS. As this is a normal CTL model checking problem, it follows that this can be computed in time $O(|\mathcal{R}| + |\mathcal{S}|)$. The other fixpoint, i.e., $\mathsf{EF}^{\leq k}\ldots$, has a time constraint, but this is now of type $\leq$. Observe now that $\mathsf{EF}^{\leq k}\varphi$ is equivalent to $\mathsf{E}[1\ \underline{\mathsf{U}}^{[0,k]}\ \varphi]$, and we already know how to compute that in time $O(|\mathcal{R}_e| + |\mathcal{S}_e|)$, i.e., in time $O(\hat{\tau}_\mathcal{K}(|\mathcal{R}| + |\mathcal{S}|))$.

For the proof, note that the last three equation (starting with A quantifiers are dual to the first three ones, so that it is sufficient to prove the first three ones. The third one can be reduced to the first one by the equation $\mathsf{EG}^{\geq k+1}\varphi = \mathsf{E}[\varphi\ \mathsf{U}^{\geq k+1}\ 0]$. Also, the second equation can be reduced to the first via the equation $\mathsf{E}[\varphi\ \mathsf{B}^{\geq k+1}\ \psi] = \mathsf{E}[(\neg\psi)\ \mathsf{U}^{\geq k+1}\ (\varphi \wedge \neg\psi)]$. Hence, we only need to prove the first equation, or equivalently, the fourth: For this reason, it is convenient to rewrite the right hand side to $\mathsf{AG}^{\leq k}\varphi \wedge \mathsf{AG}^{\leq k}\mathsf{AXA}[\varphi\ \underline{\mathsf{U}}\ \psi]$.

$\mathsf{A}[\varphi\ \underline{\mathsf{U}}^{\geq k+1}\ \psi]$ holds in a particular state s iff for all paths π starting in s, there is a number $i \in \mathbb{N}$ such that (1) $\sum_{j=0}^{i-1} \tau_\pi^{(j)} \geq k+1$, (2) $\mathcal{K}, \pi^{(i)} \models \psi$, and (3) $\forall j < i.\ \mathcal{K}, \pi^{(j)} \models \varphi$ holds. We may choose for any path π the least number i. In particular, let j be the least number for a considered path π such that (1) holds. Then, we have $j \leq i$, and therefore φ must hold on each state $\pi^{(0)}, \ldots, \pi^{(j-1)}$, which is captured in the first conjunct ($\mathsf{AG}^{\leq k}\varphi$) of the right hand side. In state $\pi^{(j)}$, we must furthermore demand that $\mathsf{A}[\varphi\ \underline{\mathsf{U}}\ \psi]$ holds, and this is captured by the second conjunct.

Hence, we can now define the following subset $\mathsf{JCTL}^{\leq}$ of JCTL that consist of the formulas that we have found to be efficiently checkable so far (we only list E quantified formulas to avoid too much redundancies).

Definition 8 (The Fragment $\mathsf{JCTL}^{\leq}$ of JCTL) *Given a set of variables $\mathcal{V}$, the set of $\mathsf{JCTL}^{\leq}$ formulas is the least set satisfying the following rules, where φ and ψ denote arbitrary $\mathsf{JCTL}^{\leq}$ formulas, and $a, b \in \mathbb{N}$ are arbitrary natural numbers, and $\sim \in \{<, \leq, \geq, >\}$:*

- $\mathcal{V} \subseteq \mathsf{JCTL}^{\leq}$*, i.e, any variable is a* $\mathsf{JCTL}^{\leq}$ *formula*
- $\neg\varphi$*,* $\varphi \wedge \psi$*,* $\varphi \vee \psi \in \mathsf{JCTL}^{\leq}$
- $\mathsf{EX}^{[a,b]}\varphi \in \mathsf{JCTL}^{\leq}$
- $\mathsf{EX}^{\sim a}\varphi \in \mathsf{JCTL}^{\leq}$
- $\mathsf{E}[\varphi\ \mathsf{U}^{\sim a}\ \psi] \in \mathsf{JCTL}^{\leq}$
- $\mathsf{EG}^{\sim a}\varphi \in \mathsf{JCTL}^{\leq}$
- $\mathsf{E}[\varphi\ \underline{\mathsf{U}}^{<a}\ \psi] \in \mathsf{JCTL}^{\leq}$ *and* $\mathsf{E}[\varphi\ \underline{\mathsf{U}}^{\leq a}\ \psi] \in \mathsf{JCTL}^{\leq}$
- $\mathsf{EF}^{<a}\varphi \in \mathsf{JCTL}^{\leq}$ *and* $\mathsf{EF}^{\leq a}\varphi \in \mathsf{JCTL}^{\leq}$

Note that $\mathsf{JCTL}^{\leq}$ still contains CTL, but we believe that it is strictly less expressive than JCTL, although we have not yet a proof for this. As each temporal operator of $\mathsf{JCTL}^{\leq}$ can be evaluated in time $O(|\mathcal{R}_e| + |\mathcal{S}_e|)$, i.e., $O(\hat{\tau}_\mathcal{K}(|\mathcal{R}| + |\mathcal{S}|))$, we immediately have the following result:

Theorem 4 (Complexity of $\mathsf{JCTL}^{\leq}$) *For any TKS $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$ and any $\mathsf{JCTL}^{\leq}$ formula φ, there is an algorithm to compute $[\![\varphi]\!]_\mathcal{K}$ in time $O(|\varphi|\ \hat{\tau}_\mathcal{K}(|\mathcal{R}| + |\mathcal{S}|))$, where $\hat{\tau}_\mathcal{K}$ is defined as in theorem 2.*

In comparison to the complexity of JCTL as given in theorem 2, the factor $\hat{k}_\varphi$ disappeared. This is due to the fact that the temporal operators that belong to $\mathsf{JCTL}^{\leq}$ are all monotonic, so that the iterations can stop at least when all tracks have been visited once.

The more important part is, however, that the equations of theorem 3 enable us to reduce a considerable part of the

model checking to qualitative fixpoints. This is due to the fact, that we can separate between a quantitative and a qualitative aspect in these formulas. Note again, that the qualitative fixpoint is computed on the abstract structure with an ordinary CTL model checking procedure.

5. Conclusions

We have presented a new real-time temporal logic JCTL that is directly defined on timed Kripke structures using interpretation I_1. This logic overcomes some semantic problems of previous real-time extensions of CTL and allows us to use abstractions without loosing quantitative information. We analysed the complexity of the JCTL model checking problem and presented a non-trivial fragment $\mathsf{JCTL}^{\leq}$ that can be checked more efficiently than JCTL. Our first experimental results are promising and allow to check practical $\mathsf{JCTL}^{\leq}$ model checking problems in runtimes that are comparable to CTL model checking.

References

[1] R. Alur, C. Courcoubetis, and D. Dill. Model Checking in Dense Real-time. Technical report, Stanford University, University of Crete, 1991.

[2] J. Bengtsson, K. Larsen, F. Larsson, P. Pettersson, and W. Yi. UPPAAL in 1995. In *Tools and Algorithms for the Construction and Analysis of Systems*, number 1055 in Lecture Notes In Computer Science, pages 431–434. Springer-Verlag, March 1996.

[3] S. Campos and E. 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*, AMAST Series in Computing. World Scientific Press, AMAST Series in Computing, May 1994.

[4] E. Clarke and E. Emerson. Design and Synthesis of Synchronization Skeletons using Branching Time Temporal Logic. In D. Kozen, editor, *Workshop on Logics of Programs*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71, Yorktown Heights, New York, May 1981. Springer-Verlag.

[5] E. Clarke, E. Emerson, and A. Sistla. Automatic Verification of Finite-State Concurrent Systems Using Temporal Logic Specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, April 1986.

[6] E. Clarke, O. Grumberg, and D. Long. Model checking and abstraction. *ACM Transactions on Programming Languages and systems*, 16(5):1512–1542, September 1994.

[7] C. Daws, A. Olivero, S. Tripakis, and S. Yovine. The tool KRONOS. In *Hybrid Systems III*, volume 1066 of *Lecture Notes in Computer Science*. Springer, 1996.

[8] E. Emerson. Temporal and Modal Logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 996–1072, Amsterdam, 1990. Elsevier Science Publishers.

[9] E. Emerson, A. Mok, A. Sistla, and J. Srinivasan. Quantitative Temporal Reasoning. *Journal of Real-Time Systems*, 4:331–352, 1992.

[10] J. Frößl, J. Gerlach, and T. Kropf. An Efficient Algorithm for Real-Time Model Checking. In *European Design and Test Conference (EDTC)*, pages 15–21, Paris, France, March 1996. IEEE Computer Society Press (Los Alamitos, California).

[11] T. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. Symbolic Model Checking for Real-Time Systems. In *IEEE Symposium on Logic in Computer Science (LICS)*, pages 394–406, Santa-Cruz, California, June 1992. IEEE Computer Society Press.

[12] G. Logothetis and K. Schneider. Abstraction from counters: An application on real-time systems. In *Design, Automation and Test in Europe (DATE'2000)*, Paris, France, March 2000. IEEE Computer Society Press.

[13] G. Logothetis and K. Schneider. A new approach to the specification and verification of real-time systems. In *EUROMICRO Conference on Real-Time Systems*. IEEE/ACM, 2001. http://goethe.ira.uka.de/fmg/ps/LoSc01.ps.gz.

[14] C. Loiseaux, S. Graf, J. Sifakis, A. Bouajjani, and S. Bensalem. Property preserving abstractions for the verification of concurrent systems. *Formal Methods in System Design*, 6:1–35, February 1995.

[15] J. Queille and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *International Symposium in Programming*, 1981.

[16] J. Ruf and T. Kropf. Using MTBDDs for discrete timed symbolic model checking. *Multiple-Valued Logic – An International Journal*, 1998. Special Issue on Decision Diagrams.

[17] F. Somenzi. CUDD: CU decision diagram package, release 2.3.0, 1998. ftp://vlsi.colorado.edu/pub/ and http://vlsi.Colorado.EDU/.