A New Approach to the Specification and Verification 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,schneider}@informatik.uni-karlsruhe.de
http://goethe.ira.uka.de/fmg

Abstract

We present a new temporal logic for the specification and verification of real-time systems. This logic is defined on discrete time transition systems which are interpreted in an abstract manner instead of the usual stuttering interpretation. Our approach directly allows the abstraction of real-time systems by ignoring irrelevant qualitative properties, but without losing any quantitative information.

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, 13] 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 [7] 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 [5, 11, 12] 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, information about quantitative time consumption is lost.

Real-time systems, however, 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 $s_1 \overset{1}{\rightarrow} s_{1,1} \overset{1}{\rightarrow} \ldots \overset{1}{\rightarrow} s_{1,k-1} \overset{1}{\rightarrow} s_2$ 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 [8], where the temporal operators have been extended by time bounds to limit the number of fix-point iterations required to evaluate the considered temporal expression. The models used in [8] 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 [9] 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. Moreover, the temporal operators of [9] that rely on interpretation $I_2$ have a misleading
semantics as we will outline in detail in section 3.2.

Finally, [14] became aware of the misleading semantics of [9] (see page 11 of [14]), and therefore defined another temporal logic that is solely based on interpretation $I_2$. In principle, their logic is defined on unit delay transition systems that are obtained by expanding (cf. section 3) a given timed transition system. As we will show in section 3.3, different expansions of a timed transition system yield in different transition systems that are not bisimilar to each other, so that the results obtained for an expanded structure can not be easily transferred back to the timed transition system. Hence, the work of [14] is essentially based on expanded, i.e., untimed transition systems, and may therefore be more or less viewed as a variant of Emerson’s work [8].

To summarize, the previous real-time extensions of CTL have the following drawbacks (we discuss them in more detail in section 3):

- 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 $K$ 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, and furthermore different expansions are not equivalent to each other.

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. The previous approaches [3, 9, 14] have all been implemented in efficient symbolic model checking tools, but the underlying formalisms are – from a logical perspective – questionable.

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, 10, 6, 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.

In this paper, we introduce the first real-time extension JCTL of CTL, that is based on interpreting timed transition systems with interpretation $I_1$, so that the logic directly allows the abstraction 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. Our logic has a next-state operator equipped with time bounds, which make our logic powerful enough to reason about real-time constraints of atomic actions. Note that the existing approaches do not allow this. We also present a model checking algorithm for JCTL that we have implemented by using the CUDD BDD library [15] in our model checking tool JERRY.

This paper is organized as follows: In the next section, we define our timed transitions systems and our real-time temporal logic JCTL. Having given the necessary definitions, we then discuss the deficiencies of the above mentioned related approaches in more detail. In particular, we consider problems that occur when timed systems are expanded to 'corresponding' unit delay systems. In general, we show that these expansions yield in misleading and even wrong results. We will then proceed with the definition of a symbolic model checking procedure for JCTL and list first experimental results that we have obtained by our tool.

2. Syntax and Semantics of the Logic JCTL

2.1. Timed Kripke Structures

We consider systems modeled as timed Kripke structures over some set of variables $V$. These timed Kripke structures are formally defined as follows:

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

1Authors of previous work have defined their logics so that these 'wrong' results match with their semantics. As outlined above, and in more detail in section 3, we will however see, that all of these approaches to fix these problems yield in further problems.

2Timed transition systems have been introduced by many authors with different names like timed transitions graphs [3], quantitative temporal structures [9], or timed temporal structures in [14]. Following the CTL notations, we prefer the name timed Kripke structures.
so that \( \pi \) and \( \tau_\pi \) 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 \( \tau_\pi \) is not uniquely defined for a fixed path \( \pi \), 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 \( \text{Paths}_\mathcal{K}(s) \).

**Definition 3 (Semantics of JCTL)** Given a TKS \( \mathcal{K} = (I, S, \mathcal{R}, \mathcal{L}) \), and \( s \in S \), 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 \( \pi \in \text{Paths}_\mathcal{K}(s) \) with associated duration function \( \tau_\pi \) such that
  \[
  \left( a \leq \sum_{j=0}^{i-1} \tau_\pi^{(j)} \leq b \right) \land \left( \mathcal{K}, \pi^{(i)} \models \varphi \right)
  \]
- \( \mathcal{K}, s \models \mathsf{EX}^\omega \varphi \) iff there is a path \( \pi \in \text{Paths}_\mathcal{K}(s) \) with associated duration function \( \tau_\pi \) and an \( i \in \mathbb{N} \) with
  \[
  \left( a \leq \sum_{j=0}^{i-1} \tau_\pi^{(j)} \leq b \right) \land \left( \forall j < i. \mathcal{K}, \pi^{(j)} \models \varphi \right)
  \]
- \( \mathcal{K}, s \models \mathsf{EU}^{[a,b]} \psi \) iff there is a path \( \pi \in \text{Paths}_\mathcal{K}(s) \) with associated duration function \( \tau_\pi \) and an \( i \in \mathbb{N} \) with
  \[
  \left( a \leq \sum_{j=0}^{i-1} \tau_\pi^{(j)} \leq b \right) \land \left( \forall j < i. \mathcal{K}, \pi^{(j)} \models \varphi \right)
  \]
- \( \mathcal{K}, s \models \mathsf{EU}^\omega \psi \) iff there is a path \( \pi \in \text{Paths}_\mathcal{K}(s) \) with associated duration function \( \tau_\pi \), such that for all \( i \in \mathbb{N} \), we have
  \[
  \left( a \leq \sum_{j=0}^{i-1} \tau_\pi^{(j)} \leq b \right) \rightarrow \left( \mathcal{K}, \pi^{(i)} \models \varphi \right)
  \]
- \( \mathcal{K}, s \models \mathsf{EG}^{[a,b]} \varphi \) iff there is a path \( \pi \in \text{Paths}_\mathcal{K}(s) \) with associated duration function \( \tau_\pi \), such that for all \( i \in \mathbb{N} \), we have
  \[
  \left( a \leq \sum_{j=0}^{i-1} \tau_\pi^{(j)} \leq b \right) \rightarrow \left( \mathcal{K}, \pi^{(i)} \models \varphi \right)
  \]

\[
\text{Figure 1. A Timed Kripke Structure}
\]
Given a TKS $K$ and a JCTL formula $\varphi$, we denote the set of states of $K$ where $\varphi$ holds as $[\varphi]_K$.

Intuitively, $K, s \models \mathrm{EX}^{[a,b]} \varphi$ means that the state $s$ has a direct successor state $s'$ that satisfies $\varphi$ and can be reached in time $t \in [a,b]$. $K, s \models \mathrm{EX}^{0} \varphi$ means that the state $s$ has a direct successor state $s'$ that satisfies $\varphi$ and can be reached in time $t \geq a$.

$K, s \models \mathrm{E}[\varphi]^{[0]}_t$ means that there is a path $\pi$ 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 $\varphi$ holds, and $\psi$ holds on $\pi^{(i)}$, and the time $t := \sum_{j=0}^{i-1} \tau_j$ required to reach state $\pi^{(i)}$ satisfies the numerical relations $a \leq t \leq b$.

$K, s \models \mathrm{EG}^{[a,b]} \varphi$ means that there is a path $\pi$ 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 $\varphi$ holds, and $\psi$ holds on $\pi^{(i)}$, and the time $t := \sum_{j=0}^{i-1} \tau_j$ required to reach state $\pi^{(i)}$ satisfies the numerical relation $a \leq t$.

$K, s \models \mathrm{EG}^{[0]}_t \varphi$ means that there is a path $\pi$ starting in $\pi^{(0)} = s$, such that for any state $\pi^{(j)}$ that is reached within a time $t := \sum_{j=0}^{i-1} \tau_j$ with $t \geq a$, we have $\pi^{(i)}$. Hence, $\varphi$ holds for all states on $\pi$ 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**:
  
  - $1 := p \lor \neg p$
  - $0 := p \land \neg p$
  - $\varphi \lor \psi := \neg \neg \varphi \lor \neg \psi$
  - $\varphi \lor \psi := \neg \neg \varphi \lor \psi$

- **EX Operators**:

  - $\mathrm{EX} \varphi := \mathrm{EX}^{0} \varphi$
  - $\mathrm{EX}^{[0,k]} \varphi := \mathrm{EX}^{[0,k]} \varphi$
  - $\mathrm{EX}^{[0,k-1]} \varphi := \mathrm{EX}^{[0,k-1]} \varphi$
  - $\mathrm{EX}^{[0,k]} \varphi := \mathrm{EX}^{[0,k]} \varphi$
  - $\mathrm{EX}^{[0,k+1]} \varphi := \mathrm{EX}^{[0,k+1]} \varphi$
  - $\mathrm{EX}^{[0,k]} \varphi := \mathrm{EX}^{[0,k]} \varphi$

- **E[ U ] Operators**:

  - $E[\varphi] := E[\varphi]^{0}$
  - $E[\varphi]^{[0,k]} := E[\varphi]^{[0,k]}$
  - $E[\varphi]^{[0,k]} := E[\varphi]^{[0,k]}$

**EG Operators**:

- $\mathrm{EG} \varphi := \mathrm{EG}^{0} \varphi$
- $\mathrm{EG}^{[0,k]} \varphi := \mathrm{EG}^{[0,k]} \varphi$
- $\mathrm{EG}^{[0,k-1]} \varphi := \mathrm{EG}^{[0,k-1]} \varphi$
- $\mathrm{EG}^{[0,k]} \varphi := \mathrm{EG}^{[0,k]} \varphi$
- $\mathrm{EG}^{[0,k+1]} \varphi := \mathrm{EG}^{[0,k+1]} \varphi$
- $\mathrm{EG}^{[0,k]} \varphi := \mathrm{EG}^{[0,k]} \varphi$

Let $k$ 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:

- $AX \varphi := \neg \mathrm{EX} \varphi$
- $A[\varphi] := \neg E[\neg \varphi]^{\sim k} \varphi$
- $AE \varphi := E[1] \varphi$
- $AE \varphi := E[1] \varphi$
- $AG \varphi := \neg \mathrm{EX} \varphi$
- $E[\varphi]^{B} := E[\neg \varphi]^{\sim k} \varphi$
- $A[\varphi]^{B} := A[\neg \varphi]^{\sim k} \varphi$
- $E[\varphi]^{B} := \neg A[\neg \varphi]^{B} \varphi$
- $A[\varphi]^{B} := \neg E[\neg \varphi]^{B} \varphi$
- $E[\varphi]^{B} := \neg A[\neg \varphi]^{B} \varphi$
- $A[\varphi]^{B} := \neg E[\neg \varphi]^{B} \varphi$

The definitions of the further EX, E[ U ], and EX operators should be clear. $\mathrm{EG} \varphi$ holds in state $s$ iff a state can be reached where $\varphi$ holds and that state can be reached within a time that satisfies the time constraint $k$. $E[\varphi]^{B} \varphi$ means that there must be a path $\pi$ and numbers $i,j \in \mathbb{N}$ with $j < i$ such that $\varphi$ and $\psi$ hold on $\pi^{(j)}$ and $\pi^{(i)}$, respectively, and the time required to reach $\pi^{(j)}$ satisfies the time constraint $k$ (hence, $\varphi$ must hold before $\psi$). We have also weak variants $E[\varphi]^{U} \varphi$ and $F[\varphi]^{U} \varphi$ of $E[\varphi]^{U} \varphi$ and $F[\varphi]^{U} \varphi$, respectively. $E[\varphi]^{U} \varphi$ means that there must be a path $\pi$ such that for all $i \in \mathbb{N}$ the following must hold: whenever a state $\pi^{(i)}$ satisfying $\psi$ can be reached in time $t$ that satisfies $\kappa$, then $\varphi$ must hold for all states $\pi^{(0)}, \ldots, \pi^{(i-1)}$. $E[\varphi]^{U} \varphi$ 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 $\mathrm{EG}^{[0,k]}$ operator states that some property holds for all states that can be reached within $[a,b]$, while $E[\varphi]^{[0,a]} \varphi$ states a property for some point of time in that interval. Nevertheless, the following lemma holds, which shows that the $\mathrm{EG}^{[k]}$ 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 $\mathrm{EG}^{[k]} \varphi = E[\varphi]^{[k,k]} \varphi$ is valid:

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

$$
(1) \quad K, s \models \mathrm{EG}^{[k]} \varphi
$$
(G_2) there is a path \( \pi \in \text{Paths}_K(s) \) starting in state \( s \), such that for all numbers \( i \in \mathbb{N} \), we have
\[
\left( \sum_{j=0}^{i-1} t_j^{(j)} \leq k \right) \rightarrow (K, \pi^{(i)} \models \varphi)
\]

(G_3) there is a path \( \pi \in \text{Paths}_K(s) \) starting in state \( s \) and a number \( i \in \mathbb{N} \) such that
\[
\left( \sum_{j=0}^{i-1} t_j^{(j)} \leq k < \sum_{j=0}^{i} t_j^{(j)} \right) \land (\forall j < i. K, \pi^{(j)} \models \varphi)
\]

(G_4) there is a path \( \pi \in \text{Paths}_K(s) \) starting in state \( s \) and a number \( i \in \mathbb{N} \) such that
\[
\left( \sum_{j=0}^{i-1} t_j^{(j)} > k \right) \land (\forall j < i. K, \pi^{(j)} \models \varphi)
\]

(G_5) \( K, s \models E[\varphi U^k 1] \)

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.

It is furthermore astonishing that we can replace \( > \) (and \( \geq \)) constraints by \( \leq \) constraints, which can be used to reduce the complexity of the model checking algorithms:

**Theorem 1** The following equations are valid, and therefore allow to reduce \( \geq \) constraints to \( \leq \) constraints:

- \( E[\varphi U^k \psi] := EF^{\leq k}(\psi \lor EX[\varphi U \psi]) \)
- \( E[\varphi B^k \psi] := EF^{\leq k}(\psi \lor EX[\varphi B \psi]) \)
- \( EG^{\geq k} \varphi := EF^{\leq k} \neg \neg \varphi \)

### 3. Related Work

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

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

In order to precisely point out the deficiencies of the previously mentioned approaches that rely on interpretation \( I_2 \), we define a possible expansion of a TKS in Figure 2, that translates a TKS to a 'corresponding' UDS.

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

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

**Definition 7 (Tracks of a State)** Given a TKS \( K = (I, S, R, L) \), its expanded structure \( K_e := (I_e, S_e, R_e, L_e) \), and a state \( s \in S \). Then, we define \( \text{Track}_K(s) = \{(s',u) \in S_e \mid s = s'\} \).

We emphasize that expansions of TKS can be performed in many different ways that are not equivalent to each other (cf. Figure 7). We will discuss this issue in more detail in section 3.3.

#### 3.1. Expansion Problem

One might think that we can simply expand timed Kripke structures to obtain corresponding unit-delay structures in order to use the traditional CTL model checking algorithms. However, we show by the following example, that this is not the case.

Consider the TKS given in Figure 3 and the formula \( EG^{\leq 2} EF^{\leq 1} p \). To evaluate this formula, we successively obtain the following sets of states with our semantics
\[
[p] = \{(s_1, s_2)\}, \quad [EF^{\leq 1} p] = \{(s_0, s_1, s_2)\}, \quad [EG^{\leq 2} EF^{\leq 1} p] = \{(s_0)\}
\]

Now, consider the expanded structure as given in Figure 4. We evaluate again the formula \( EG^{\leq 2} EF^{\leq 1} p \), and obtain \([p] = \{(s_1,1), (s_2,1)\}\) and \([EF^{\leq 1} p] = \{(s_1,1), (s_0,1), (s_2,1), (s_0,3)\}\). To evaluate \([EG^{\leq 2} EF^{\leq 1} p]\), we must now check which of the expansion states have a path so that we are in \([EF^{\leq 1} p]\) as long as the time consumption is \( \leq 2 \). It is not difficult
to see that there is no such state, and hence, we now have $[\text{EG}^{\geq 2}\text{EF}^{\leq 1}p] = \emptyset$.

Figure 4. Expansion of the TKS of Figure 3

Hence, we see by the above example, that TKSs are not equivalent to their expanded UDSs. The example moreover shows an intrinsic problem of approaches like [3, 14] that are based on $I_2$: The evaluation of a formula $\varphi$ may yield in a set of tracks $T_\omega$, where $(s, i) \in T_\omega$ holds for one track $(s, i)$, but $(s, j) \in T_\omega$ does not hold for another track $(s, j)$. As such 'inconsistent' sets of tracks may occur (although the tracks of atomic formulas are consistent) it follows that the approaches based on $I_2$ can not define in which states of the TKS of Figure 3 a formula like $\text{EF}^{\leq 1}p$ holds. This yields in problems when operators are nested, as demonstrated by the above example. Therefore, [14] performs all computations on the expanded structure 4 to compute the set of tracks $T_\omega$ where a formula $\varphi$ holds, and then checks whether $\mathcal{I} \times \{1\} \subseteq T_\omega$ holds.

3.2. Overlapping Time Constraints

In [9], it has been observed that timed transitions may be differently interpreted as already mentioned in the introduction. However, [9] did neither decide to use $I_1$ nor to use $I_2$, and instead mixed both interpretations in that different temporal operators interpret transitions differently, i.e., according to either $I_1$ or $I_2$. For example, [9] defines a temporal operator $\text{EF}^k \varphi$ as follows: $\mathcal{K}, s \models \text{EF}^k \varphi$ iff there is a path $\pi \in \text{Paths}_{n}(s)$ with associated duration function $\tau_\pi$ and an $i \in \mathbb{N}$, so that $(\mathcal{K}, \pi^{(i)}) \models \varphi \land \sum_{j=0}^{i} \tau_\pi^{(j)} > k$ holds. Intuitively, $\text{EF}^k \varphi$ amounts to say that there is a path where $\varphi$ holds after the time $k$ has been consumed. However, this not the case, which can be shown by the following example.

Consider the state $s_0$ of the structure given in Figure 5. For comparison, note first that $\text{EF}^{> 3}p$ means that there must be a path starting in state $s_0$ where we reach some state $s$ in time $> 3$ where $p$ holds. This is not the case for any of the states in Figure 5, and hence, we have $[\text{EF}^{> 3}p] = \emptyset$. However, we have $[\text{EF}^3p] = \{s_0, s_1\}$! The difference between $\text{EF}^k$ and our $\text{EF}^{> k}$ is best seen by the TKS in Figure 6 that contains a single path $\pi$, where $\varphi$ holds only on state $\pi^{(i)}$. Furthermore, assume that $\sum_{j=0}^{i-1} \tau_\pi^{(j)} \leq k < \sum_{j=0}^{i} \tau_\pi^{(j)}$ holds. Then, the formula $\text{EF}^k \varphi$ will be satisfied on $\pi$, while $\text{EF}^{> k} \varphi$ will not be satisfied. The reason is that [9] interpret $\text{EF}^k \varphi$ with interpretation $I_2$ so that it will be satisfied by the intermediate states between $\pi^{(i)}$ and $\pi^{(i+1)}$.

We are not aware of any reasonable application, where additional time constraints of actions are to be considered that are not even taken. But even more severe, the coexistence of both interpretations of timed transitions makes it impossible to define operations on TKS like composition of structures or simulation preorders.

3.3. Branching Problems

Becoming aware of the misleading mixture of semantics used in [9], [14] decided to only use interpretation $I_2$. However, another problem with the approach in [14] is that diff-

---

3In [9], the notation $\text{EXW}^k \varphi$ (weak next) has been chosen, but in analogy to our $\text{EF}$ operator, we prefer the notation above.
different expansion techniques yield in different results. To see this, consider two different expansions of the TKS given in Figure 1 that are given in Figure 7. In the above expansion $K_1$, we have added to every timed transition intermediate states according to the time delay. This is in accordance with [3]. In the second expansion $K_2$, we have shared the intermediate states as long as possible which is the expansion technique of Figure 2 that is currently also preferred by [14].

The two structures $K_1$ and $K_2$ are not equivalent. To see this, consider the formula $\Phi := \mathsf{AF}^{s_1} \mathsf{EF}^{s_1} \mathsf{p}$. In $K_1$, we obtain $\langle \Phi \rangle = \{} (s_1', 2), (s_1', 3), (s_3, 1) \{$. In $K_2$, we however obtain $\langle \Phi \rangle = \{} (s_1, 1), (s_1, 2), (s_1, 3), (s_3, 1) \}$. And therefore $\langle \Phi \rangle = \{} (s_1', 2), (s_1', 3), (s_3, 1) \}$.

Hence, the semantics of [14] depends on the chosen expansion. As a consequence, the approaches followed in [14] and [3] are also not equivalent to each other. Moreover, as the real-time temporal logics used in [14] and [3] both use TKSs as models, it is unsatisfactory to use something else but the TKS to define the semantics.

As the results of the expanded structure moreover seems to have nothing to do with the original problem, i.e., the results obtained for an expanded structure can not be directly transferred back to the timed transition system, we may consider [14] in principle as an approach based on unit-delay Kripke structures. In the next section, we present an algorithm that is able to correctly translate the model checking results obtained after a fixpoint iteration on any expanded structure to the abstract structure so that our interpretation $I_1$ is respected.

Furthermore, it is erroneously argued in [14] that the semantics as given in [9] is not intuitive, since in general, we have $\mathsf{EF}^{s_1} \mathsf{p} \neq \mathsf{EF}^{s_1} \mathsf{EF}^{s_1} \mathsf{p}$. However, both formulas express different things: $\mathsf{EF}^{s_1} \mathsf{EF}^{s_1} \mathsf{p}$ means that we can reach some state in at least 3 units of time where in further 3 units of time, a possibly different state is reached where $p$ holds. This is obviously not the same as $\mathsf{EF}^{s_1} \mathsf{p}$, which means that we can reach a state where $p$ holds in at least 6 units of time. In general, we only have $\mathsf{EF}^{s_1} \mathsf{EF}^{s_1} \mathsf{p} \rightarrow \mathsf{EF}^{s_1} \mathsf{p}$, but not vice versa.

4. 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 [15]. The underlying algorithms for the basic operators are given in Figure 9.

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 9).

The key function for the evaluation of all real-time constraints is the function $\text{MoveFront}$. Given a set of tracks $T_v$ and a set of states $S_v$, this function computes the set of tracks that have a path of a certain length through the tracks $S_v \times \mathbb{N}$. The precise specification is as follows:
function

Proof: Hence, \( \text{MoveFront}(\sim, k, S_{\varphi}, T_{\varphi}) \) computes the set of tracks that have a path through any expanded structure of length \( \ell \) with \( \ell \sim k \) to a track in \( T_{\varphi} \) which runs only through tracks of \( S_{\varphi} \times N \).

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):

- \( \text{MoveFront}(\sim, 0, S_{\varphi}, T_{\varphi}) = T_{\varphi} \)
- \( \text{MoveFront}(\sim, k + 1, S_{\varphi}, T_{\varphi}) = \left( S_{\varphi} \times N \right) \cap \text{preTracks}(\text{MoveFront}(\sim, k, S_{\varphi}, T_{\varphi})) \)
- \( \text{MoveFront}(\geq, k + 1, S_{\varphi}, T_{\varphi}) = \left( \text{let } T_0 := \text{MoveFront}(\geq, k, S_{\varphi}, T_{\varphi}) \text{ in } T_0 \cup ((S_{\varphi} \times N) \cap \text{preTracks}(T_0)) \right) \)

Using these equations, we can easily prove that \( T_0 = \text{MoveFront}(\sim, 1, S_{\varphi}, T_{\varphi}) \) is an invariant of the loop in the algorithm given in Figure 9 for \( \text{MoveFront} \). This directly implies the correctness of the above lemma.

Please note that in the algorithm given in Figure 9 to implement the function \( \text{MoveFront} \), we use \( i \neq k \) and \( T_0 \neq T_1 \) as loop condition instead of \( i = k \) (which would also be correct). The reason for this is that whenever \( T_0 = T_1 \) holds for an iteration \( i < k \), then it follows that all fronts \( \text{MoveFront}(\sim, i, S_{\varphi}, T_{\varphi}), \ldots, \text{MoveFront}(\sim, k, S_{\varphi}, T_{\varphi}) \) would be identical, so that we already have the result in this case.

Now consider the function \( \text{States}E_{\varphi}[a,b] \). We first compute the set of tracks \( T_0 \) that can reach a track of \( S_{\varphi} \times \{1\} \) in exactly \( a \) steps (where only tracks of the states \( S_{\varphi} \) are traversed). After this, we move the tracks \( T_0 \) by further \( b - a \) unit delay steps through the tracks of the states \( S_{\varphi} \). By the above lemma, we then obtain

\[
(s_0', t') \in T_1 \iff \\
\exists s_0', \ldots, s_{m-1}', t_m \in S_{\varphi}, \exists t_0', \ldots, t_{m-1} \in N. \\
\exists s_n \in S_{\varphi}, \exists t \in N. \\
\wedge_{i=0}^{m-1} (s_i', t', s_{i+1}') \in R \land \\
(s_m' = s_0) \in R \land \\
\wedge_{i=0}^{n-1} (s_i, t, s_{i+1}) \in R \land \\
a = \sum_{i=0}^{n-1} t_i + 1 - t \land \\
b \geq \left( \sum_{i=0}^{m-1} t_i' \right) + \left( \sum_{i=0}^{n-1} t_i \right) + 1 - t' \land \\
t \leq t_0 \land t' \leq t_0' 
\]

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 \((s_0', 1) \) from track \((s_0', 1) \). By the above result, it follows that this time is in the interval \([a, b] \) (consider the case \( m = 0 \), where the second \( \text{MoveFront} \) 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 \((s_0', 1) \in T_1 \) holds, then we clearly see that state \( s_0 \) belongs to \( E_{\varphi}[a,b] \). If, on the other hand, \((s_0', t') \in T_1 \) holds for some \( t' > 1 \), but \((s_0', 1) \) is not included in \( T_1 \), 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 \( s_0 \notin E_{\varphi}[a,b] \).

The correctness of \( E_{\varphi}[a,b] \) is seen as follows: \( T_0 \) is the set of tracks that have a path of length \( b - a \) through \( S_{\varphi} \times N \) in the expanded structure. \( T_1 \) is the set of tracks of \( S_{\varphi} \times N \) that can reach \( T_0 \) while only traversing tracks of \( S_{\varphi} \times N \). Hence, \( T_2 \) is the set of tracks that have a path of length \( \geq b - a \) through \( S_{\varphi} \times N \) in the expanded structure. Finally, \( T_2 \) is the set of tracks of \( S \times N \) that can reach \( T_1 \) in exactly \( a \) steps. Clearly, we only should consider those tracks with time stamp 1 of \( T_2 \).

The correctness of the other two functions to evaluate \( E_{\varphi}|_{\sim}^{=a} \psi \) and \( E_{\varphi}^{=a} \psi \) are proved in a similar way. Therefore, we obtain the following correctness result:

**Theorem 2 (Correctness of Function States)** For any TKS \( K = (I, S, R, L) \) and any JCTL formula \( \varphi \), the function States given in Figure 9 satisfies the equation \( \text{States}(\varphi) = \left[ \varphi \right]_K \).

The complexity of States is of order \( O(\gamma_K \hat{k}_{\varphi} |\varphi| (|R| + |S|)) \), where \( \gamma_K := \max t \mid \exists s, s', (s, t, s') \in R \) is the maximum delay time of \( K \), and \( \hat{k}_{\varphi} \) is the maximum number used in time constraints in \( \varphi \).

Table 1 lists some of our experimental results that we have obtained by our our real-time model checking tool JERRY.
Figure 9. Model Checking of JCTL Formulas on a Timed Kripke Structure $\mathcal{K} = (\mathcal{I}, \mathcal{S}, \mathcal{R}, \mathcal{L})$
efficiency of JERRY, we have performed several runs on TKSs that have been randomly generated. As can be seen, we have used the CUDD package for predecessor computations. Our experimental results are promising and show that our tool JERRY can handle large real-time systems within reasonable resource requirements.

5. Conclusions and Future Work

We have shown that many existing approaches to real-time extensions of CTL are misleading. For this reason, we have developed a new real-time temporal logic JCTL that is directly defined on timed Kripke structures using interpretation \( I_t \). We presented an efficient symbolic model checking algorithm for JCTL, so that we are able to benefit from established improvements of symbolic state space traversal (as for example, disjunctive partitioning of transition relations for predecessor computations). Our experimental results are promising and show that our tool JERRY can handle large real-time systems within reasonable resource requirements.

Table 1. Experimental Results of our Real-Time Model Checking Tool JERRY

<table>
<thead>
<tr>
<th>BDD [nodes]</th>
<th>Variables</th>
<th>time [sec]</th>
</tr>
</thead>
<tbody>
<tr>
<td>142405</td>
<td>20</td>
<td>8</td>
</tr>
<tr>
<td>283301</td>
<td>21</td>
<td>8</td>
</tr>
<tr>
<td>806395</td>
<td>24</td>
<td>10</td>
</tr>
<tr>
<td>161934</td>
<td>25</td>
<td>10</td>
</tr>
<tr>
<td>3237846</td>
<td>26</td>
<td>10</td>
</tr>
<tr>
<td>4388566</td>
<td>28</td>
<td>12</td>
</tr>
<tr>
<td>6471472</td>
<td>27</td>
<td>10</td>
</tr>
<tr>
<td>12910202</td>
<td>29</td>
<td>12</td>
</tr>
<tr>
<td>12956728</td>
<td>28</td>
<td>10</td>
</tr>
<tr>
<td>17565319</td>
<td>30</td>
<td>12</td>
</tr>
<tr>
<td>45121373</td>
<td>33</td>
<td>14</td>
</tr>
<tr>
<td>87106038</td>
<td>40</td>
<td>20</td>
</tr>
</tbody>
</table>

All figures are given for a PentiumIII@800Mhz with 500 MByte main memory. For the implementation of JERRY, we have used the CUDD package [15]. To demonstrate the efficiency of JERRY, we have performed several runs on TKSs that have been randomly generated. As can be seen, our algorithms can cope with large real-time systems and obtain a powerful efficiency.

References