Model Checking Real-Time Systems

Written Abstract for the Seminar "Recent Advances in Model Checking"

Vincent Trélat

Organizational information

This abstract is based on Chapter 29 of the Handbook of Model Checking [CHVB18]. Section 1 first introduces and motivates model checking applied to real-time systems, building on [CHVB18, Chapters 29.1 and 29.2]. Section 2 gives some formal definitions from [CHVB18, Chapter 29.2] about timed-automata and related notions.

...

1 Introduction

The most basic problem regarding timed automata is reachability, i.e. given a timed automaton $\mathcal{A}$, is a set of locations F of $\mathcal{A}$ reachable or not.

...

2 Timed Automata

Preliminaries In this chapter, time values are equated with non-negative real numbers of $\mathbb{R}_{\geq 0}$. A *time sequence* is a finite or infinite non-decreasing sequence of time values. A *timed word* over $\Sigma \times \mathbb{R}_{\geq 0}$ is a word over the alphabet Σ sequentially paired with a time sequence. If the time sequence of a timed word is upper-bounded or converging, the timed word is said to be *converging* and *divergent* otherwise.

Let C be a finite set of variables called *clocks*. A *valuation* over C is a mapping $v \colon C \to \mathbb{R}_{\geq 0}$. The set of valuations over C is denoted $\mathbb{R}_{\geq 0}^C$ and $\mathbf{0}_C$ denoted the valuation assigning 0 to every clock of C.

For any valuation v and any time value t, the valuation $v + t$ denotes the valuation obtained by shifting all values of v by t. For any subset r of C, $v[r]$ is the valuation obtained by resetting all clocks of r in v.

A *constraint* φ over C is recursively defined by the following grammar:

$$\varphi ::= x \odot k \mid \varphi \wedge \varphi$$

where $x \in C$, $k \in \mathbb{Z}$ and $\odot \in \{<, \leq, =, \geq, >\}$. The set of constraints over C is denoted $\Phi(C)$. We say that a valuation v over C satisfies $x \odot k$ when $v(x) \odot k$, and when v satisfies a constraint φ, we write $v \models \varphi$. The set of valuations satisfying a constraint φ is denoted $[\![\varphi]\!]_C$.

Timed Automata A timed automaton is basically a finite automaton with (real-time) constraints on the states. The following formal definition is a reformulation of [CHVB18, Chapter 29.2, Definition 1].

Definition 1. A *Timed Automaton* (TA) $\mathcal{A}$ is the data $(L, l_0, C, \Sigma, I, E)$ where L is a finite set of *locations* with initial location $l_0 \in L$, C is a finite set of *clocks*, Σ is a finite set of *actions*, $I \colon L \to \Phi(C)$ is an *invariant mapping* and $E \subseteq L \times \Phi(C) \times \Sigma \times 2^C \times L$ is a set of edges.

Any edge $(\ell, \varphi, a, r, \ell') \in E$ is denoted $\ell \xrightarrow{\varphi,a,r} \ell'$ where φ is a *guard*, and r is a subset of clocks that are set to zero after taking the transition.

An example of TA is given in Fig. 1.

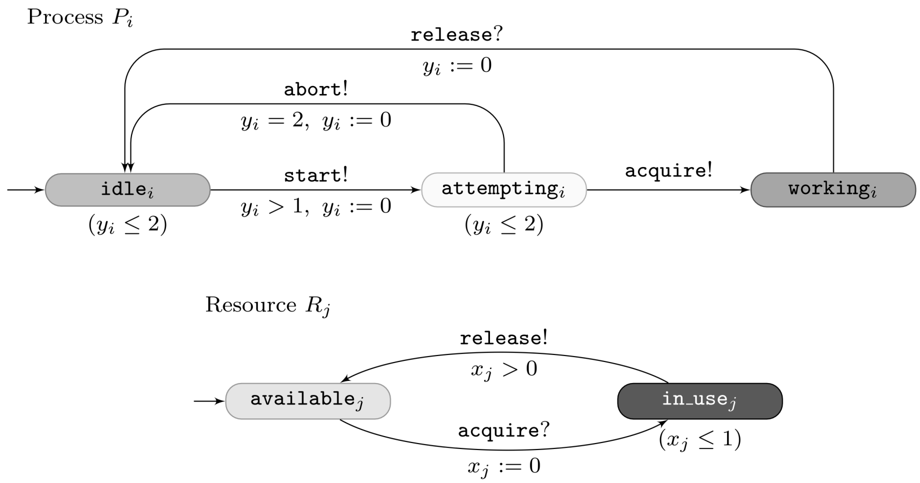


Figure 1: Two TA modeling processes which can use resources.
Figure taken from [CHVB18, Chapter 29.2]

Semantics The *operational semantics* of a TA $\mathcal{A} = (L, \ell_0, C, \Sigma, I, E)$ is the infinite-state timed transition system $[\![A]\!] = (S, s_0, \mathbb{R}_{\geq 0} \times \Sigma, T)$, where

$$S := \{(\ell, v) \in L \times \mathbb{R}_{\geq 0}, v \models I(\ell)\}, \quad s_0 := (\ell_0, \mathbf{0}_C),$$
$$T := \{(\ell, v) \xrightarrow{d,a} (\ell', (v+d)[r]) \mid d \in \mathbb{R}_{\geq 0}, \forall d' \in [0,d], v + d' \models I(\ell) \wedge \exists \ell \xrightarrow{\varphi,a,r} \ell' \in E, v + d \models \varphi\}$$

Parallel Composition It is possible to compose several TA in parallel. Informally, parallel composition roughly consists in pairing the automata and taking the conjunction of their invariants, distinguishing two types of transitions, namely he synchronous[1] and asynchronous transitions.

Region Equivalence Informally, two valuations v and v' are region equivalent if a TA $\mathcal{A}$ cannot differentiate between them and we write[2] $v \cong_M v'$. This notion is extended to states of $\mathcal{A}$ by defining $(\ell, v) \cong_M (\ell', v')$ if $v \cong_M v'$ and $\ell = \ell'$. The equivalence class of (ℓ, v) is denoted $[\ell, v]_{\cong_M}$.

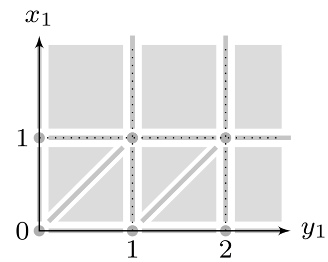
Such a relation has the property of *time-abstracted bisimulation*, which informally means that from any two equivalent states (for any binary relation), the TA can take the same transitions, expect that the delays might differ.

From this notion, we can define the *region automaton* $\mathcal{R}_{\cong_M}(\mathcal{A}) = (S, s_0, \Sigma, T)$ which basically consists in quotienting $\mathcal{A}$ w.r.t. $\cong_M$:

$$S = (L \times \mathbb{R}_{\geq 0})_{/\cong_M} \quad s_0 = [\ell_0, \mathbf{0}_C]_{\cong_M} \quad T = \{[\ell, v]_{\cong_M} \xrightarrow{a} [\ell', v']_{\cong_M} \mid \exists d, (\ell, v) \xrightarrow{d,a} (\ell', v')\}$$

As an example, we give a represention of clock regions for the parallel composition of P_1 and R_1 in Fig. 2.

[1] Synchronous transitions are given by a *synchronization function*.
[2] $M := (M_x)_{x \in C}$ where M_x is the maximal constant x is compared to in $\mathcal{A}$.



P_1	R_1	f_1
`start!`	–	$\texttt{start}_1$
`abort!`	–	$\texttt{abort}_1$
`acquire!`	`acquire?`	$\texttt{acquire}_1$
`release!`	`release?`	$\texttt{release}_1$

Figure 2: Clock regions for $(P_1||R_1)_{f_1}$ where f_1 is given by its table of values.

Figure taken from [CHVB18, Chapter 29.3]

References

[CHVB18] Edmund M. Clarke, Thomas A. Henzinger, Helmut Veith, and Roderick Bloem. *Handbook of Model Checking*. Springer Publishing Company, Incorporated, 1st edition, 2018.