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 such as reachability and bisimulation.

...

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.

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.

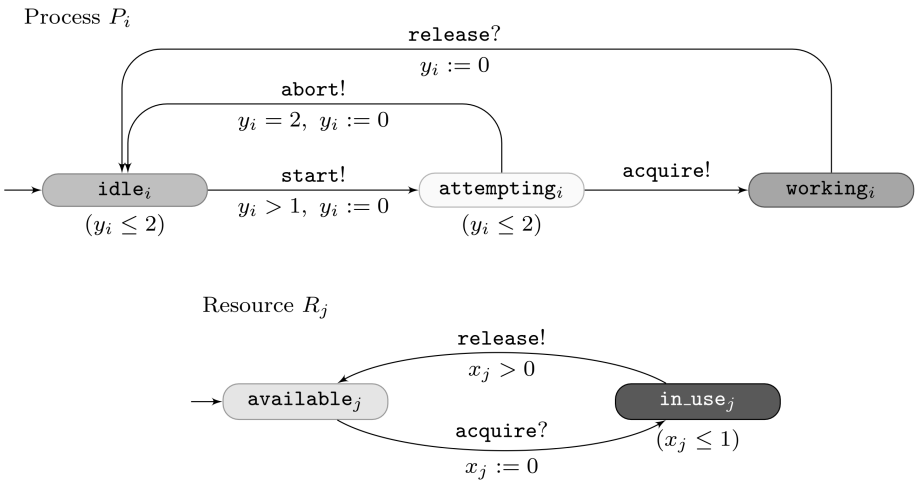


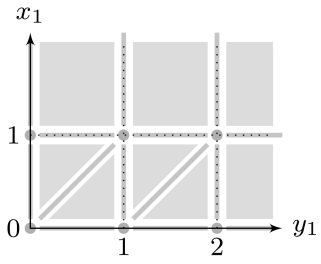
Figure 1: Two TA modeling processes which can use resources.
Figure taken from [CHVB18, Chapter 29.2]

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 the synchronous[1] and asynchronous transitions. The point is that systems can be defined in a compositional way.

Region Equivalence Informally, two valuations are region equivalent if a TA cannot differentiate between them and we write[2] this relation $\cong_M$. This notion is extended to states of the TA by defining $(\ell, v) \cong_M (\ell', v')$ iff $v \cong_M v'$ and $\ell = \ell'$. The equivalence class of (ℓ, v) is denoted $[\ell, v]_{\cong_M}$. As an example, we give a represention of clock regions for the parallel composition of P_1 and R_1 in Fig. 2.



P_1	R_1	f_1
start!	–	start
abort!	–	abort
acquire!	acquire?	acquire
release!	release?	release

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]

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

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. $\cong_M$:

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

The region automaton $\mathcal{R}_{\cong_M}(\mathcal{A})$ is a finite automaton whose size is exponential compared with the size of $\mathcal{A}$. It can be used to check, reachability properties and "one of the most fundamental theorems" [CHVB18, Chapter 29.2, Theorem 1] states that the reachability problem (or equivalently emptiness) in TA is PSPACE-complete [AD94, Section 4.5].

Any binary relation R is a *timed simulation* if for any two equivalent states $s_1\ R\ s_2$, and any transition $s_1 \xrightarrow{d,a} s_1'$, s_2 can reach a state s_2' with the same transition $s_2 \xrightarrow{d,a} s_2'$ such that $s_1'\ R\ s_2'$. It is a *timed bisimulation* if it is a timed simulation and symmetric.

The relation $\cong_M$ has the weaker property of *time-abstracted bisimulation*, which informally means that for a binary relation R on states, from any two equivalent states the TA can take the same transitions, expect that the delays might differ.

An important result [CHVB18, Chapter 29.4.2] is that decidability of both timed and time-abstracted (bi)similarity is ensured for TA.

Extensions Many extensions of TA were proposed in the literature, such as allowing *diagonal constraints*[3], *updatable* TA[4] or adding *urgency* requirements. However, it has been shown that such extensions are no more expressive than the original class of TA. Furthermore, most attempted extensions generally lead to undecidability of the reachability problem, e.g. by simply adding linear constraints of the form $\lambda x + \mu y \leq k$, or more complex classes such as *parametrized* or *hybrid* automata.

References

[AD94] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

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

[3] Constraints of the form $x - y \odot k$.

[4] Clocks can be updated to any value instead of being reset to 0.