SUP(T) Decides Timed Automata Reachability

Timed automata [AlurDill94,BouyerEtAl04] are finite automata extended by clock variables. The clock variables are used in linear arithmetic formulas to encode location invariants and guards for transitions. All clocks are initially zero and can be reset to zero after having taken a transition. They increase simultaneously at a constant rate as long as location invariants are satisfied.

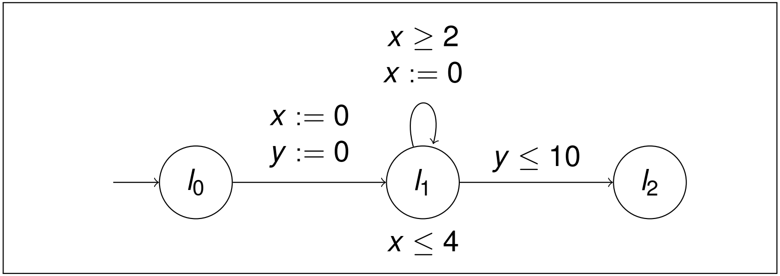

Figure: (8.1) A timed automaton

Transitions of the automaton are encoded by BSH(LA) (Bernays-Schoenfinkel Horn) clauses.
Translating the transition from l_0 to l_1 and the state invariant for state l_1 results in the respective BSH(LA) clauses

(1) $x' = 0, y' = 0, x' \leq 4 \parallel \mathrm{Reach}(l_0, x, y) \rightarrow \mathrm{Reach}(l_1, x', y')$
(2) $t \geq 0, x'' = x' + t, y'' = y' + t, x'' \leq 4 \parallel \mathrm{Reach}(l_1, x', y') \rightarrow \mathrm{Reach}(l_1, x'', y'')$.

Performing superposition on $\mathsf{Reach}(l_1, x', y')$ yields

$$x' = 0, y' = 0, x' \leq 4, t \geq 0, x'' = x' + t, y'' = y' + t, x'' \leq 4 \;\|\; \\ \mathsf{Reach}(l_0, x, y) \rightarrow \mathsf{Reach}(l_1, x'', y'')$$

which can be simplified by LA constraint simplification (elimination of t, x', and y') to

$$(3) \quad x'' \geq 0, x'' = y'', x'' \leq 4 \;\|\; \mathsf{Reach}(l_0, x, y) \rightarrow \mathsf{Reach}(l_1, x'', y'').$$

8.9.1 Definition (Timed Automaton)

A *timed automaton* T is a tuple $T = (L, \mathsf{start}, X, \{\mathsf{inv}_l\}_{l \in L}, E, \mathsf{final})$ where L is a finite set of *locations* with *initial location* $\mathsf{start} \in L$ and final location $\mathsf{final} \in L$, X is a finite set of clock variables, $\mathsf{inv}_l \in B(X)$ is the *invariant* of location l; $E \subseteq L \times B(X) \times 2^X \times L$ is a finite set of *edges*. An edge (l, ϕ, Z, l') represents a transition from location l to location l'. The constraint ϕ determines when the edge is enabled, and the set Z contains the clocks to be reset to zero when taking the edge.

A *state* of a timed automaton is a tuple (l, ν) consisting of a location $l \in L$ and a valuation $\nu \in X \to \mathbb{Q}^+$ for all clocks. The initial state is (l_0, ν_0) where ν_0 assigns zero to all clocks. In a location, the values of all clocks can increase at a fixed rate as long as the location's invariant is satisfied. This induces a relation $(l, \nu) \xrightarrow{\delta} (l, \nu + \delta)$ between states, where both ν and $\nu + \delta$ satisfy inv_l. We call the relation $\bigcup_{\delta \in \mathbb{Q}^+} \xrightarrow{\delta}$ the *timed reachability relation*. For example, in the timed automaton from Figure 8.1, we have $(l_1, \{x \mapsto 0, y \mapsto 0\}) \xrightarrow{4} (l_1, \{x \mapsto 4, y \mapsto 4\})$.

When the valuation satisfies the guard of an edge, the edge can be taken, and a new clock valuation is determined according to Z. These discrete steps in turn induce a transition relation $(l, \nu) \overset{\mathrm{d}}{\to} (l', \nu')$ between states, where ν satisfies the guard of an edge between l and l', and ν' is the result of applying the corresponding clock resets. We call this relation the *discrete-step reachability relation*. Considering again the timed automaton from Figure 8.1, we have $(l_1, \{x \mapsto 3, y \mapsto 3\}) \overset{\mathrm{d}}{\to} (l_1, \{x \mapsto 0, y \mapsto 3\})$.

The *reachability relation* $\rightarrow$ is defined as the union of the timed and discrete-step reachability relations. By $\rightarrow^*$ we denote its reflexive transitive closure. We say that a state l is *reachable* if $(l_0, \nu_0) \rightarrow^* (l, \nu)$ from some ν. It is easy to see that a state l is reachable if and only if there exists a finite sequence $\pi = (l_0, \nu_0), (l_1, \nu_1), (l_2, \nu_2), \ldots, (l_n, \nu_n)$ of states where $(l_n, \nu_n) = (l, \nu)$ and any two consecutive states are contained either in the timed reachability relation or in the discrete-step reachability relation. We call such a sequence a *run* (of T). A timed automaton T is *non-empty* if the final state is reachable.

8.9.2 Theorem (Timed Automata are decidable [AlurDill94])

It is decidable, PSPACE-complete, whether a timed automaton T is non-empty.

Encoding Reachability in FOL(LA)

Let $T = (L, \mathsf{start}, X, \{\mathsf{inv}_l\}_{l \in L}, E, \mathsf{final})$ be a timed automaton, let the vector $\vec{x}$ contain the variables in X.
The following clause represents reachability of the initial state:

$$\vec{x} = 0 \;\|\; \rightarrow \mathrm{Reach}(\mathsf{start}, \vec{x}).$$

For every location $l \in L$, the following clause encodes time-reachability:

$$t \geq 0,\ \vec{x}' = \vec{x} + t,\ \mathsf{inv}_l[\vec{x}'] \parallel \mathsf{Reach}(l, \vec{x}) \rightarrow \mathsf{Reach}(l, \vec{x}').$$

For every edge $(l, \phi, Z, l') \in E$, the following clause encodes discrete-step reachability:

$$\phi,\ \vec{x}' = \mathsf{reset}_Z(\vec{x}),\ \mathsf{inv}_{l'}[\vec{x}'] \parallel \mathsf{Reach}(l, \vec{x}) \rightarrow \mathsf{Reach}(l', \vec{x}').$$

where

$$\mathsf{reset}_Z(x) = \begin{cases} 0 & \text{if } x \in Z, \\ x & \text{otherwise.} \end{cases}$$

8.9.4 Theorem (SUP(LA) Decides Timed Automata [FietzkeWeidenbach12])

Let N be the clause set resulting from the translation from a timed automaton T. Then T is non-empty iff $N_E = N \cup \{ \parallel \mathsf{Reach}(\mathsf{final}, \vec{x}) \rightarrow \}$ is unsatisfiable. The saturation of N_E by SUP(LA) terminates.

One important trick of the proof is to consider reachability backwards by adding an extra argument of a free sort to Reach making the right hand sides of any implaction maximal in the ordering. For example, the clause

$t \geq 0, x' = x + t, y' = y + t, x' \leq 4 \parallel \mathrm{Reach}(l_1, x, y) \rightarrow \mathrm{Reach}(l_1, x', y')$

becomes

$t \geq 0, x' = x + t, y' = y + t, x' \leq 4 \parallel \mathrm{Reach}(z, l_1, x, y) \rightarrow \mathrm{Reach}(g(z), l_1, x', y')$