Model-Based Analysis (MBA) - Assignment 2

Šimon Stupinský - xstupi00@stud.fit.vutbr.cz

Obtained points:

TIMED AUTOMATON

Clock constraint. A *clock constraint* over set C of clocks is formed according to the grammar

$$g ::= \quad x < c \quad | \quad x \leq c \quad | \quad x > c \quad | \quad x \geq c \quad | \quad g \wedge g$$

where $c \in \mathbb{N}$ and $x \in C$. Let $CC(C)$ denote the set of clock constraint over C. Clock constraints that do not contain any conjunctions are *atomic*. Let $ACC(C)$ denote the set of all atomic clock constraints over C.

Timed Automaton. A *timed automaton* is a tuple $TA = (Loc, Act, C, \hookrightarrow, Loc_0, Inv, AP, L)$ where

- Loc is a finite set of locations,
- $Loc_0 \subseteq Loc$ is a set of initial locations,
- Act is a finite set of actions,
- C is a finite set of clocks,
- $\hookrightarrow \subseteq Loc \times CC(C) \times Act \times 2^C \times Loc$ is a transition relation,
- $Inv : Loc \to CC(C)$ is an invariant-assignment function,
- AP is a finite set of atomic propositions, and
- $L : Loc \to 2^{AP}$ is a labeling function for the locations.

$ACC(TA)$ denotes the set of atomic clock constraints that occur in either a guard or a location invariant of TA.

Elapsed Time on a Path Let TA be a timed automaton with the set Act of actions. The function $ExecTime : Act \cup \mathbb{R}_{>0} \to \mathbb{R}_{\geq 0}$ is defined as

$$ExecTime(\tau) = \begin{cases} 0 & if\ \tau \in Act \\ d & if\ \tau = d \in \mathbb{R}_{>0}. \end{cases}$$

For infinite execution fragment $\rho = s_0 \xrightarrow{\tau_0} s_1 \xrightarrow{\tau_1} s_2 \ldots$ in $TS(TA)$[1] with $\tau_i \in Act \cup \mathbb{R}_{>0}$ let

$$ExecTime(\rho) = \sum_{i=0}^{\infty} ExecTime(\tau_i).$$

The execution time of finite execution fragments is defined analogously. For the path fragment π in $TS(TA)$ induced by ρ, $ExecTime(\pi) = ExecTime(\rho)$.

Time Divergence and Time Convergence The infinite path fragment π is *time-divergent* if $ExecTime(\pi) = \infty$; otherwise. π is *time-convergent*.

---

[1] TS - Transition System of the Timed Automata TA.

1. TASK (2 POINTS)

Assignment: Let us consider the automaton $\mathbb{A}$ in the Figure 1.

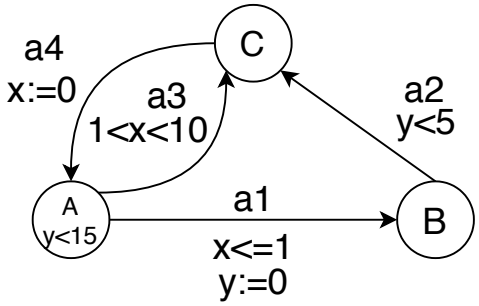

Figure 1. Timed Automaton $\mathbb{A}$

(a) Does this automaton contain the *Zeno Path*? You prove or disprove.

(b) Does this automaton contain the *timelock*? If yes, give a run leading to the *timelock*.

---

a) Does this automaton contain the *Zeno Path*? You prove or disprove

Zeno run. Let **TA** be a timed automaton. The infinite path fragment π in $\mathbf{TS}(\mathbf{TA})$ is *zeno* (or: a *zeno path*) if it is a time-convergent and infinitely many actions $\mathbf{a} \in \mathbf{Act}$ are executed along π.

Zenoness. A timed automaton **TA** is non-zeno if there does not exist an initial zeno path in $\mathbf{TS}(\mathbf{TA})$. This feature of the timed automaton is considered as modelling flaws that should be avoided. The execution of actions $\mathbf{a} \in \mathbf{Act}$ is instantaneous, i.e., actions take no time. Without further restrictions, a timed automaton may perform infinitely many actions in a finite time interval. Thus, we call this phenomenon as *zeno* and represents non-realisable behaviour, since it would require infinitely fast processors.

Proof. Time automaton **TA** is thus *non-zeno* if and only if for every path $\pi \in \mathbf{TS}(\mathbf{TA})$, either π is *time-divergent* or π is *time-convergent* with almost only (i.e., all except for finitely many) delay transitions. However, let us consider the following path of the given timed automaton $\mathbb{A}$ from Figure 1:

$$\pi_{\mathbb{A}} = (A, x=0, y=0) \xrightarrow{a1} (B, x=0, y=0) \xrightarrow{a2} (C, x=0, y=0) \xrightarrow{a4} (A, x=0, y=0) \xrightarrow{a1} (B, x=0, y=0) \xrightarrow{a2} \ldots$$

This shown path $\pi_{\mathbf{A}}$ is the *infinite time-convergent* path fragment and within it are executed the infinitely many actions $\mathbf{a} \in \mathbf{Act}_{\mathbb{A}} \setminus \{\mathbf{a3}\}$. Therefore, we can conclude, that the given timed automaton $\mathbb{A}$ contains the *Zeno Path* $\pi_{\mathbb{A}}$ and thus, it does not *non-zeno*. □

---

b) Does this automaton contain the *timelock*? If yes, give a run leading to the *timelock*.

Time-Divergent Set of Paths. For state **s** in $\mathbf{TS}(\mathbf{TA})$ let: $Paths_{div}(s) = \{\pi \in Paths(s) \mid \pi \text{ is time-divergent}\}$. That is, $\mathbf{Paths_{div}}(\mathbf{s})$ denotes the set of *time-divergent* paths in TS(TA) that starts in **s**. Although *time-convergent* paths are not realistic, their existence cannot be avoided. For the analysis of timed automaton, *time-convergent* paths are simply ignored, e.g., a timed automaton satisfies an invariant when along all its *time-divergent* paths the invariant is satisfied.

Timelock. Let **TA** be a timed automaton. State **s** in $\mathbf{TS}(\mathbf{TA})$ contains a *timelock* if $\mathbf{Paths_{div}}(\mathbf{s}) = \emptyset$. **TA** is *timelock-free* if no state in $\mathbf{Reach}(\mathbf{TS}(\mathbf{TA}))$ contains a timelock. In other words, the state **s** in $\mathbf{TS}(\mathbf{TA})$ contains a *timelock* if there is no *time-divergent* path starting in this state **s**. Such states are unrealistic as time cannot progress forever from these states. Timelocks are considered as undesired and need to be avoided when modelling the time-critical system by means of a timed automaton.

Proof. Any terminal state of a transition system (TS) that results from a timed automaton contains a *timelock*. The Terminal state should no be confused with terminal locations, i.e., locations that have not outgoing edges. The terminal location l with $Inv(l) = true$, e.g., does not a result in a terminal state in the underlying transition system, as time may progress in l forever. Terminal locations thus do not necessarily yield states with timelocks. The given timed automaton $\mathbb{A}$ from Figure 1 does not contain any terminal location and therefore we do not have to deal with this condition. Not only terminal states may contain timelocks. The reachable state $(\mathbf{A}, \mathbf{x} = \mathbf{14}, \mathbf{y} = \mathbf{14})$ is not terminal, e.g., the *time-convergent* path in $\mathbf{TS}(\mathbb{A})$ emanates from it:

$$(A, x = 14, y = 14) \xrightarrow{0.9} (A, x = 14.9, y = 14.9) \xrightarrow{0.09} (A, x = 14.99, y = 14.99) \xrightarrow{0.009} (A, x = 14.999, y = 14.999) \xrightarrow{0.0009} \dots$$

However, $\mathbf{Paths_{div}}((\mathbf{A}, \mathbf{x} = \mathbf{14}, \mathbf{y} = \mathbf{14})) = \emptyset$ as the state $(\mathbf{A}, \mathbf{x} = \mathbf{14}, \mathbf{y} = \mathbf{14})$ has no outgoing discrete transitions (as the guards $\mathbf{x} <= \mathbf{1}$ and $\mathbf{1} < \mathbf{x} < \mathbf{10}$ are violated), and time cannot progress beyond **15** due to $\mathbf{Inv}(\mathbf{A}) = \mathbf{y} < \mathbf{15}$. State $(\mathbf{A}, \mathbf{x} = \mathbf{14}, \mathbf{y} = \mathbf{14})$ in $\mathbf{TS}(\mathbb{A})$ contains a timelock. Timed automaton $\mathbb{A}$ is thus not timelock-free. □

---

2. TASK (2 POINTS)

Assignment: Prove that the languages of timed automaton are closed to union and concatenation operations. Consider the languages over the final words with a set of final locations Loc_{acc}.

Theorem: Timed automatons are closed under *union*.

Proof. Let $\mathbf{A_i} = (\mathbf{Loc_i}, \mathbf{Loc_{0_i}}, \mathbf{Act}, \mathbf{C_i}, \hookrightarrow_\mathbf{i}, \mathbf{Inv_i}, \mathbf{AP_i}, \mathbf{L_i}, \mathbf{Loc_{acc_i}})$, for $1 \leq i \leq n$ be a given set of Timed Automatons with the same actions **Act**. Without loss of generality we assume that the sets of clocks $\mathbf{C_i}$, $\mathbf{C_j}$ and respectively the sets of locations $\mathbf{Loc_i}$, $\mathbf{Loc_j}$ are pairwise dijsoint: $\mathbf{C_i} \cap \mathbf{C_j} = \emptyset$ and $\mathbf{Loc_i} \cap \mathbf{Loc_j} = \emptyset$. Now, we construct the new Timed Automaton **A**, whom language will be the union of $L(A_i),\ for\ 1 \leq i \leq n$.

Let $\mathbf{A} = (\mathbf{Loc}, \mathbf{Loc_0}, \mathbf{Act}, \mathbf{C}, \hookrightarrow, \mathbf{Inv}, \mathbf{AP}, \mathbf{L}, \mathbf{Loc_{acc}})$ be the Timed Automaton defined as follows

$$\bullet\ Loc = \bigcup_{1\leq i\leq n} Loc_i \qquad \bullet\ Inv = \bigcup_{1\leq i\leq n} Inv_i$$
$$\bullet\ Loc_0 = \bigcup_{1\leq i\leq n} Loc_{0_i} \qquad \bullet\ AP = \bigcup_{1\leq i\leq n} AP_i$$
$$\bullet\ C = \bigcup_{1\leq i\leq n} C_i \qquad \bullet\ L = \bigcup_{1\leq i\leq n} L_i$$
$$\bullet\ \hookrightarrow = \bigcup_{1\leq i\leq n} \hookrightarrow_i \qquad \bullet\ Loc_{acc} = \bigcup_{1\leq i\leq n} L_{acc_i}$$

It is clear that $\mathbf{L}(\mathbf{A}) = \bigcup_{\mathbf{1\leq i\leq n}} \mathbf{L}(\mathbf{A_i})$. We denote **A** as $\mathbf{A_1} \cup \mathbf{A_2} \cup \ldots \cup \mathbf{A_n}$. □

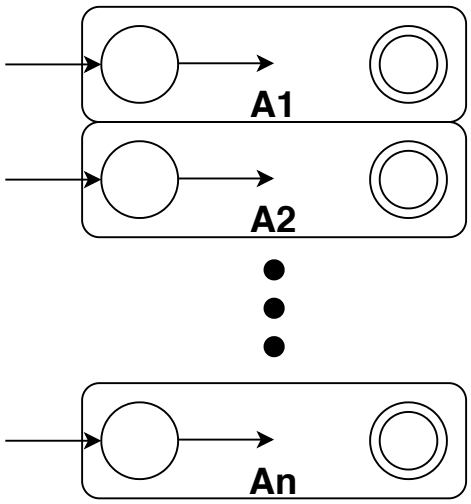

Figure 2. Graphic representation of the *union* of the given set of Timed Automatons $\{A_1, A_2, \ldots, A_n\}$ according to the described way in the proof above.

Theorem: Timed automatons are closed under *concatenation*.

Proof. We propose in this section a basic operation of concatenation of timed automatons. The concatenation of two usual – i.e. untimed – automaton is a classical operation widely used in the modelling of untimed automatons. When we deal with timed automatons, the main different comes from the *clocks*. Assume that we want to concatenate two timed automaton $\mathbb{A}$ and $\mathbb{B}$. Then, entering in $\mathbb{B}$, the two extreme possibilities are either to reset all the clocks of $\mathbb{A}$ or on the contrary to not reset any clock of $\mathbb{A}$.

Let $\mathbb{A} = (\mathbf{Loc}_\mathbb{A}, \mathbf{Loc_{0_\mathbb{A}}}, \mathbf{Act}, \mathbf{C}_\mathbb{A}, \hookrightarrow_\mathbb{A}, \mathbf{Inv}_\mathbb{A}, \mathbf{AP}_\mathbb{A}, \mathbf{L}_\mathbb{A}, \mathbf{Loc_{acc_\mathbb{A}}})$ and $\mathbb{B} = (\mathbf{Loc}_\mathbb{B}, \mathbf{Loc_{0_\mathbb{B}}}, \mathbf{Act}, \mathbf{C}_\mathbb{B}, \hookrightarrow_\mathbb{B}, \mathbf{Inv}_\mathbb{B}, \mathbf{AP}_\mathbb{B}, \mathbf{L}_\mathbb{B}, \mathbf{Loc_{acc_\mathbb{B}}})$ be a Timed Automatons with the same actions **Act**. Without loss of generality we assume that the sets of clocks $\mathbf{C}_\mathbb{A}$, $\mathbf{C}_\mathbb{B}$ and respectively the sets of locations $\mathbf{Loc}_\mathbb{A}$, $\mathbf{Loc}_\mathbb{B}$ are pairwise dijsoint: $\mathbf{C}_\mathbb{A} \cap \mathbf{C}_\mathbb{B} = \emptyset$ and $\mathbf{Loc}_\mathbb{A} \cap \mathbf{Loc}_\mathbb{B} = \emptyset$. Intuitively, the result of the concatenation of two timed automatons is the timed automaton obtained

by first performing actions in $\mathbb{A}$, then resetting all the clocks of $\mathbb{B}$ and finally performing actions in $\mathbb{B}$. Now, we construct the new Timed Automaton $\mathbb{AB}$, whom language will be the concatenation of $\mathbf{L}(\mathbb{A})$ and $\mathbf{L}(\mathbb{B})$.

Let $\mathbb{AB} = (\mathbf{Loc}, \mathbf{Loc_{0_\mathbb{A}}}, \mathbf{Act}, \mathbf{C}, \hookrightarrow, \mathbf{Inv}, \mathbf{AP}, \mathbf{L}, \mathbf{Loc_{acc_\mathbb{B}}})$ be the Timed Automaton defined as follows

- $Loc = Loc_\mathbb{A} \cup Loc_\mathbb{B} \cup \overline{Loc_{acc_\mathbb{A}}}$
- $C = C_\mathbb{A} \cup C_\mathbb{B}$
- $Inv = Inv_\mathbb{A} \cup Inv_\mathbb{B}$
- $AP = AP_\mathbb{A} \cup AP_\mathbb{B}$
- $L = L_\mathbb{A} \cup L_\mathbb{B}$

where $\overline{\mathbf{Loc_{acc_\mathbb{A}}}}$ is a copy of $\mathbf{Loc_{acc_\mathbb{A}}}$, disjoint of $\mathbf{Loc_\mathbb{A}} \cup \mathbf{Loc_\mathbb{B}}$ and $\hookrightarrow$ is defined as follows:

$$
\begin{aligned}
\hookrightarrow = {} & \hookrightarrow_\mathbb{A} \cup \hookrightarrow_\mathbb{B} \\
& \cup \{ (l_1, g_1, a_1, R_1 \cup C_\mathbb{B}, \overline{l_{acc_\mathbb{A}}}) \mid (l_1, g_1, a_1, R_1, l_{acc_\mathbb{A}}) \in \hookrightarrow_\mathbb{A},\ l_{acc_\mathbb{A}} \in Loc_{acc_\mathbb{A}} \} \\
& \cup \{ (\overline{l_{acc_\mathbb{A}}}, g_2, a_2, R_2, l_2) \quad \mid (l_{0_\mathbb{B}}, g_2, a_2, R_2, l_2) \in \hookrightarrow_\mathbb{B},\ l_{0_\mathbb{B}} \in Loc_{0_\mathbb{B}} \}
\end{aligned}
$$

It is clear that $\mathbf{L}(\mathbb{AB}) = \mathbf{L}(\mathbb{A}) \cdot \mathbf{L}(\mathbb{B})$. In this proof, when entering in $\mathbb{B}$, we have not reset any clock of $\mathbb{A}$ and reset all clock of $\mathbb{B}$, obviously. But on many real examples, these extreme possibilities are unsatisfactory. Assume for instance, that we want to construct the complex model under some global timing constraints and that the construction is made in several parts. In most of the cases, each of these parts will have its own timing constraints and also own clocks. Then it is much more natural to allow the concatenation of timing automatons by modelling of the sub-systems to reset some clocks and to not reset some other clocks. To meet this requirement, it is enough to replace the reset clock $\mathbb{C}_\mathbb{B}$ when entering in $\mathbb{B}$, by the hours $\mathbb{C}$, that we get, for example, at the input instance. □

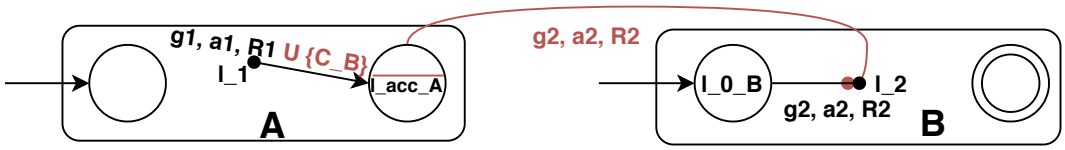

Figure 3. Graphic representation of the *concatenation* of two Timed Automatons $\mathbb{A}$ and $\mathbb{B}$ according to the described way in the proof above. The red colour parts represent the newly added components in the constructed transition relation of the new Timed Automaton.

3. TASK (4 POINTS)

Assignment: Consider the timed automaton $\mathbb{B}$ from Figure 4 with the finite set of atomic propositions $AP = \{\text{init, error, run}\}$ and labelling function L for the locations defined as follows: $L(A) = \{\text{init, run}\}$, $L(B) = L(C) = \{run\}$, $L(D) = \{error\}$.

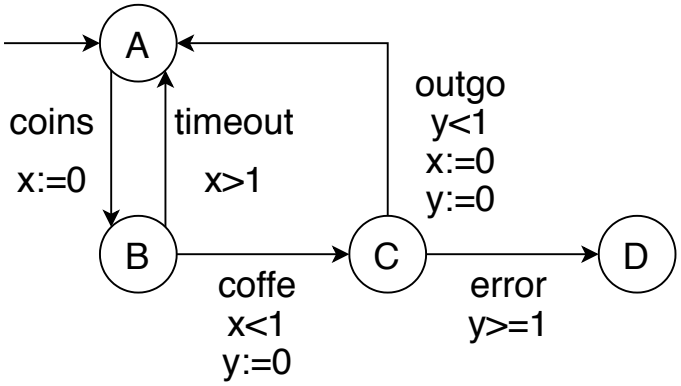

Figure 4. Timed Automaton $\mathbb{B}$.

(a) Construct the region abstraction (just construct the states available from initial configuration).

(b) Decide, whether is reachable the state in which is valid the predicate *error*.

(c) Decide, whether is valid $\mathbb{B} \vDash \exists\,(\text{run } U^{<2} \text{ error})$.

(d) Decide, whether is valid $(B, x = 0, y = 0) \vDash \forall \text{ run } U^{<2} \text{ init})$.

Argument your results.

Construct the region abstraction (just construct the states available from initial configuration)

Clock regions. Each region can be uniquely characterised by a (finite) set of clocks constraints it satisfies. For instance, consider a clock interpretation $\mathbf{t}$ over two clocks with $\mathbf{t}(\mathbf{x}) = .\mathbf{25}$ and $\mathbf{t}(\mathbf{y}) = .\mathbf{5}$. Then, each clock interpretation in $[\mathbf{t}]$ (the clock region to which $\mathbf{t}$ belongs) satisfies the constraint $(\mathbf{0} < \mathbf{x} < \mathbf{y} < \mathbf{1})$, and we will represent this region by $[\mathbf{0} < \mathbf{x} < \mathbf{y} < \mathbf{1}]$. In the following Figure 5 are shown the clock regions of the Timed Automaton from Figure 4.

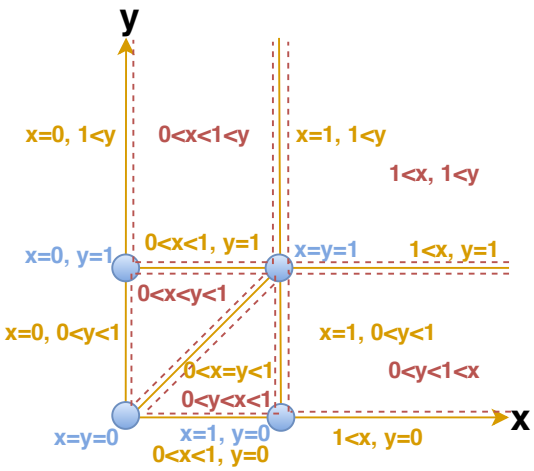

Figure 5. Clock regions of the Timed Automaton $\mathbb{B}$.

Note that there are only a finite number of regions. The constructed clock regions included **4** *Corner points* (blue colour), **9** *Open line segments* (yellow colour) and **5** *Open regions* (red colour):

- **Corner points**: $[(0,0)]$, $[(0,1)]$, $[(1,0)]$, $[(1,1)]$,
- **Open line segments**: $[0<x<1,0]$, $[0<x<1,1]$, $[0,0<y<1]$, $[1,0<y<1]$, $[1<x,0]$, $[1<x,1]$, $[0,1<y]$, $[1,1<y]$, $[0<x=y<1]$,
- **Open regions**: $[0<y<x<1]$, $[0<x<y<1]$, $[0<y<1<x]$, $[0<x<1<y]$, $[1<x,1<y]$.

Region Abstraction. Figure 6 shows the constructed region abstraction of the given Timed Automaton from Figure 4. Note, that only the regions reachable from the initial region $<\mathbf{A},[\mathbf{x}=\mathbf{y}=]>$ are shown. The individual states show the name of the *location* from the original timed automaton $\mathbb{B}$, the atomic propositions (**AP**) of this location according to the given labelling function **L** and the relevant *clock region*. The states with the same colour represent the one location from the original timed automaton $\mathbb{B}$.

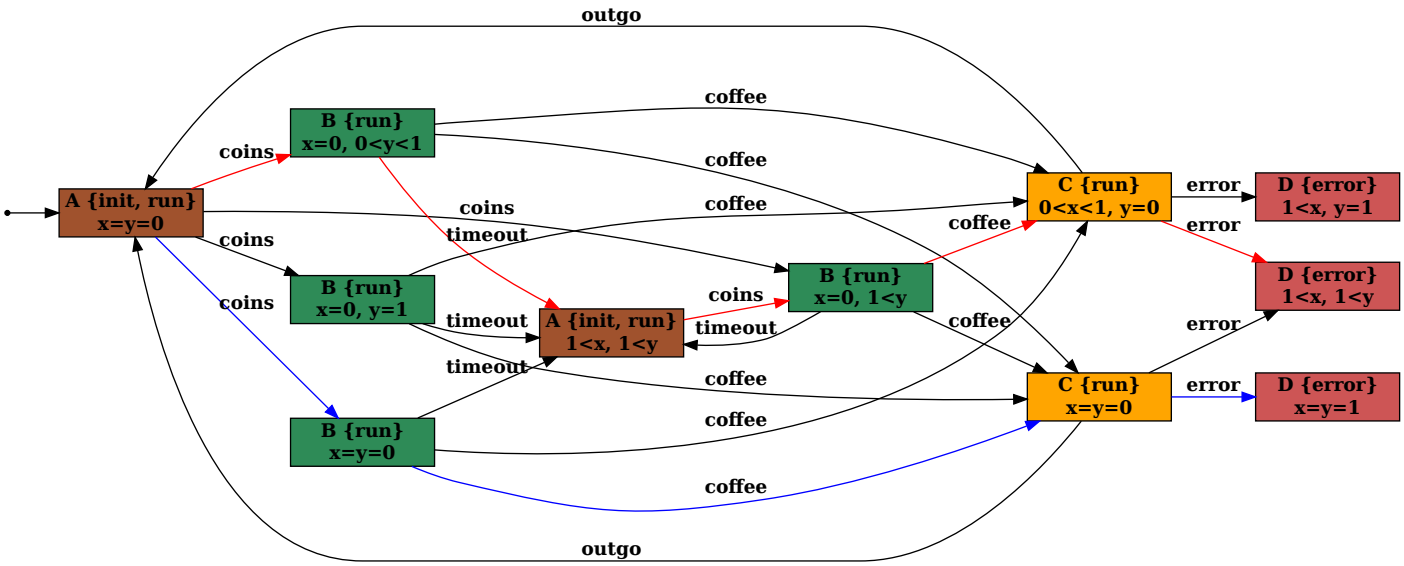

Figure 6. Region abstraction $R(\mathbb{B})$ of the Timed Automaton $\mathbb{B}$.

Decide, whether is reachable the state in which is valid the predicate *error*?

Reachability problem. The input of such problem is the timed automaton **A** and the subset of atomic propositions $\mathbf{p} \in \mathbf{2^{AP}}$. The task is verify whether exists the path $\rho = (\mathbf{l_0}, \mathbf{0^C} \xrightarrow{\mathbf{s_1}} \ldots \xrightarrow{\mathbf{s_i}} (\mathbf{l}, \mathbf{v})$, where $\mathbf{l_0} \in \mathbf{Loc_0}$ and $\mathbf{p} \subseteq \mathbf{L}(\mathbf{l})$. In general, this problem is decidable for timed automaton and it is *P-SPACE* complete problem.

Proof. Now we proceed to establish a correspondence between the paths of $\mathbb{B}$ and the paths of $R(\mathbb{B})$. For a run $\mathbf{r} = (\bar{\mathbf{s}}, \bar{\mathbf{v}})$ of $\mathbb{B}$ of the form:

$$r:\ (l_0,v_0) \xrightarrow{t_1,s_1} (l_1,v_1) \xrightarrow{t_2,s_2} (s_2,v_2) \xrightarrow{t_3,s_3} \ldots$$

define its projection $[\mathbf{r}] = (\bar{\mathbf{s}}, [\bar{\mathbf{v}}])$ to be the sequence

$$[r]:\ (l_0,[v_0]) \xrightarrow{s_1} (l_1,[v_1]) \xrightarrow{s_2} (s_2,v_2) \xrightarrow{s_3} \ldots$$

From the definition of the edge relation for $\mathbf{R}(\mathbb{B})$, it follows that $[\mathbf{r}]$ is a path of $\mathbf{R}(\mathbb{B})$ over $\mathbf{s}$. Now we have to find the path of $\mathbf{R}(\mathbb{B})$ where we reach the state in which is valid the predicate `error` and such we prove the given problem. Let us consider the following paths $[\mathbf{r_1}]$ and $[\mathbf{r_2}]$ of $\mathbf{R}(\mathbb{B})$, which are drawn in Figure 6 by the blue, respectively red colour:

$$[r_1]:\ (A,[x=y=0]) \xrightarrow{coins} (B,[x=y=0]) \xrightarrow{coffee} (C,[x=y=0]) \xrightarrow{error} (D,[x=y=1])$$

$$[r_2]:\ (A,[x=y=0]) \xrightarrow{coins} (B,[x=0,0<y<1]) \xrightarrow{timeout} (A,[1<x,1<y]) \xrightarrow{coins} (B,[x=0,1<y])$$
$$\xrightarrow{coffee} (C,[0<x<1,y=0]) \xrightarrow{error} (D,[1<x,1<y])$$

And now we show the example projections of these paths to paths $\mathbf{r_1}$ and $\mathbf{r_2}$ of the Timed Automaton $\mathbb{B}$:

$$r_1 : (A,0,0) \xrightarrow{0,\,coins} (B,0,0) \xrightarrow{0,\,coffee} (C,0,0) \xrightarrow{1,\,error} (D,1,1)$$

$$r_2 : (A,0,0) \xrightarrow{0.5,\,coins} (B,0,0.5) \xrightarrow{2,\,timeout} (A,2,2.5) \xrightarrow{1,\,coins} (B,0,3.5) \xrightarrow{0.9,\,coffee} (C,0.9,0) \xrightarrow{2,error} (D,2.9,2)$$

We showed two example paths, that leads to the state **D** where is valid the predicate `error`. As we can see in the constructed *region abstraction* in Figure 6, there are exists also other paths that can lead to this state **D**. Thus, we proved that the state in which is valid the predicate `error` **is reachable** in the Timed Automaton $\mathbb{B}$ from Figure 4. □

Decide, whether is valid $\mathbb{B} \vDash \exists(\textbf{run}\ U^{<2}\ \textbf{error})$

Satisfaction Relation for TCTL. Le *TA* be a timed automaton and $\mathbf{J} \subseteq \mathbb{R}_{\geq 0}$. For state $\mathbf{s} = (\mathbf{l}, \eta)$ in $\mathbf{TS}(\mathbf{TA})$ and *TCTL* path formula ϕ, the relevant part of the satisfaction relation $\vDash$ is defines for state formulae by: $s \vDash \exists\phi \iff \exists\pi \in Paths_{div}(s) : \pi \vDash \phi$. For *time-divergent* path $\pi \in s_0 \xrightarrow{d_0} s_1 \xrightarrow{d_1} \ldots$, the satisfaction relation $\vDash$ for path formulae is defined by:

$$\pi \vDash \Phi U^J \psi \iff \exists i \geq 0 : s_i + d \vDash \psi \text{ for some } d \in [0,d_i] \text{ with } \sum_{k=0}^{j-1} d_k + d \in J \text{ and}$$

$$\forall j \leq i : s_j + d' \vDash \Phi \vee \psi \text{ for any } d' \in [0,d_j] \text{ with } \sum_{k=0}^{j-1} d_k + d' \leq \sum_{k=0}^{i-1} d_k + d,$$

where for $s_i = (l_i, \eta_i)$ and $d \geq 0$ we have $s_i + d = (l_i, \eta_i + d)$.

TCTL Semantics for Timed Automaton. Let *TA* be a timed automaton with clocks **C** and locations **Loc**. For *TCTL* state formula Φ, the *satisfaction* set $\mathbf{Sat}(\Phi)$ is defined by: $\mathbf{Sat}(\Phi) = \{s \in Loc \times Eval(C) \mid s \vDash \Phi\}$. The timed automaton *TA* satisfies TCTL state formula Φ if and only if Φ holds in all initial states of *TA*: $TA \vDash \Phi \iff \forall l_0 \in Loc_0 : (l_0, \eta_0) \vDash \Phi$ where $\eta_0(x) = 0$ for all $x \in C$.

Proof. State formula $\exists\phi$ is true in state **s** if and only if there exists *some time-divergent* path starting in **s** that satisfies ϕ. As stated before, path quantification is over time-divergent paths. Time-divergent path $\pi \in s_0 \xrightarrow{d_O} s_1 \xrightarrow{d_1} \ldots$ satisfies $\Phi U^J \psi$ whenever at some time point in **J**, a state is reached satisfying ψ and at all previous time instants $\Phi \vee \psi$ holds. Firstly, we construct the set of the initial configuration of times automaton $\mathbb{B}$ from Figure 4: $Init_{\mathbb{B}} = \{(l, O^{|C|}) \mid l \in Loc_0\} = \{(l,0,0)\}$. We have to prove that the timed automaton $\mathbb{B}$ satisfies the given TCTL formula $\mathbb{B} \vDash \exists(\text{run}\ U^{<2}\ \text{error})$, thus have to be valid $Init_{\mathbb{B}} \subseteq \mathbf{Sat}(\Phi)$. Therefore, we have to show, that the initial configuration satisfies the given formula $(A,0,0) \vDash \exists(\text{run}\ U^{<2}\ \text{error})$, and thus belongs to $\mathbf{Sat}(\Phi)$, where Φ marks the given formula. Let us consider the following *time-divergent* path:

$$(A,0,0) \xrightarrow{0.5,coins} (B,0,0.5) \xrightarrow{0.25,coffee} (C,0.25,0) \xrightarrow{1.20,error} (D,1.45,1.00) \xrightarrow{10} \ldots$$

According to the defined labelling function **L**, we can conclude that in the states $(\mathbf{A},\mathbf{B},\mathbf{C})$ are satisfied the atomic proposition `run`. Subsequently, in the time $\mathbf{1.95}$ $(< \mathbf{2})$ will switch to the state **D**, where is satisfy the atomic proposition `error`. Since this state has no outgoing transitions, this atomic proposition will satisfy forever with any progress over time. Therefore, we can conclude that formula $\mathbb{B} \vDash \exists(\text{run}\ U^{<2}\ \text{error})$ **is satisfied** because of validity the condition $Init_{\mathbb{B}} \subseteq \mathbf{Sat}(\Phi)$, respectively the validity of the formula $(A,0,0) \vDash \exists(\text{run}\ U^{<2}\ \text{error})$. □

Decide, whether is valid $(B, x = 0, y = 0) \vDash \forall\ (\textbf{run}\ U^{<2}\ \textbf{init})$

Proof. $\forall\phi$ holds in state **s** whenever all *time-divergent* paths starting in **s** satisfy this formula ϕ. As we defined above, the *time-divergent* path satisfy the formula $\Phi U^J \psi$, when exists the time moment **t** from the interval *J*, in which is valid the formula ψ. Further have to be valid, that for arbitrary time moment which is less than **t** is valid the formula $\Phi \vee \psi$. In other words, for each reachable configuration of each *time-divergent* path is valid the required formula. In our case, have to be valid that in each *time-divergent* path of the state $(\mathbf{B},\mathbf{0},\mathbf{0})$ will satisfy the formula $\forall$ (run $\mathbf{U^{<2}}$ init): $\forall \mathbf{p} \in \mathbf{Paths_{div}}((\mathbf{B},\mathbf{0},\mathbf{0})) : \mathbf{p} \vDash$ (run $\mathbf{U^{<2}}$ init). However, let us consider the following *time-divergent* path from the state $(\mathbf{B},\mathbf{0},\mathbf{0})$ of the Timed Automaton from Figure 4:

$$p_1 : (B,0,0) \xrightarrow{0.5,coffee} (C,0.5,0) \xrightarrow{1.25,error} (D,1.75,1.25) \xrightarrow{10} \ldots$$

According to the given labelling function **L**, we can conclude, that in the first two states of this path $\mathbf{B}, \mathbf{C}$ are satisfied the atomic proposition `run`. Subsequently, in the time $\mathbf{1.75}$ $(< \mathbf{2})$ will switch to the state **D** where is satisfy the atomic proposition `error`. Since this state has no outgoing transitions, this atomic proposition will satisfy forever with any progress over time. Clearly, in this state are satisfied none of the required atomic propositions (`run`, `init`) and therefore this path not satisfy the given formula $(\text{run } U^{<2} \text{ init})$. Now, we can conclude that formula $(B, x = 0, y = 0) \vDash \forall\, (\text{run } U^{<2} \text{ init})$ **is not satisfied**, because we found the *time-divergent* path $\mathbf{p_1} \in \mathbf{Paths_{div}}((\mathbf{B}, \mathbf{0}, \mathbf{0}))$ which not satisfied this formula. □
