Algorithms for Model Checking (2IW55)

Lecture 12

Timed Verification: Timed Automata

Background material:Chapter 16, 17 and handout R. Alur, "Timed Automata"

Tim Willemse
(timw@win.tue.nl)
http://www.win.tue.nl/∼timw
HG 6.81

Outline

Timed Automata

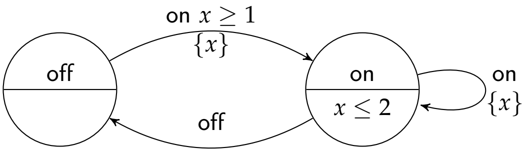
Recalling notation:

- A clock valuation ν for a set C of clocks is a function $\nu : C \to \mathbb{R}_{\geq 0}$
- We write $\nu \models \phi$ iff $[\phi]_\nu =$ true.
- Clock valuation update: $\nu + d$ is defined as: $(\nu + d)(x) = \nu(x) + d$ for all $d \in \mathbb{R}_{\geq 0}$.
- Clock valuation reset: $[\nu]_R$ is defined as: $[\nu]_R(x) = 0$ if $x \in R$, else $\nu(x)$.
- Let $\mathcal{C}(C)$ be the set of clock constraints over C.

Timed Automata

A timed automaton is a tuple $\mathcal{T} = \langle L, L_0, Act, C, \longrightarrow, \iota \rangle$

- L is a finite set of locations; $L_0 \subseteq L$ is a non-empty set of initial locations
- Act is the set of actions
- C is a finite set of clock variables
- $\longrightarrow \subseteq L \times \mathcal{C}(C) \times Act \times 2^C \times L$ is the set of switches
- $\iota : L \rightarrow \mathcal{C}(C)$ is the invariant assignment function



Timed Automata

Recalling intuition:

- A switch $l \xrightarrow{g\ a\ R} l'$ means that:
  - action a is enabled whenever guard g evaluates to true.
  - upon executing the switch, we move from location l to location l' and reset all clocks in R to zero.
  - only locations l' that can be reached with clock values that satisfy the location invariant.
- an invariant $\iota(l)$ limits the time that can be spent in location l.
  - staying in location l only is allowed as long as the invariant evaluates to true.
  - before the invariant becomes invalid location l must be left.
  - if no switch is enabled when the invariant becomes invalid no further progress is possible: timed deadlock, or time-lock.

Timed Automata

Let $\mathcal{T} = \langle L, L_0, Act, C, \longrightarrow, \iota \rangle$ be a Timed Automaton.
Its semantics is defined as a timed transition system: $[\mathcal{T}] = \langle S, S_0, Act, \rightarrow, \mapsto \rangle$

- $S = \{(l, \nu) \in L \times (C \rightarrow \mathbb{R}_{\geq 0}) \mid \nu \models \iota(l)\}$, i.e. all combinations of locations and clock valuations that do not violate the location invariant.
- $S_0 = \{(l, \nu) \in L_0 \times (C \rightarrow \mathbb{R}_{\geq 0}) \mid \nu \models \iota(l) \wedge \forall x \in C : \nu(x) = 0\}$.
- $\longrightarrow \subseteq S \times Act \times S$ is defined as follows:

$$\frac{l \xrightarrow{g\ a\ R} l' \qquad \nu \models g \wedge \iota(l) \qquad \nu' = [\nu]_R \qquad \nu' \models \iota(l')}{(l, \nu) \xrightarrow{a} (l', \nu')}$$

- $\mapsto \subseteq S \times \mathbb{R}_{\geq 0} \times S$ is defined as follows:

$$\frac{\nu \models \iota(l) \qquad \nu + d \models \iota(l)}{(l, \nu) \overset{d}{\mapsto} (l, \nu + d)}$$

Outline

Analysing Semantics

Let $\mathcal{T} = \langle L, L_0, Act, C, \longrightarrow, \iota \rangle$ be a Timed Automaton.

- Assume $\nu \models \iota(l)$ and $\nu + d \models \iota(l)$ for fixed $d \in \mathbb{R}_{\geq 0}$
- A possible execution fragment starting from the location l is:

$$(l, \nu) \overset{d_1}{\mapsto} (l, \nu + d_1) \overset{d_2}{\mapsto} (l, \nu + d_1 + d_2) \overset{d_3}{\mapsto} (l, \nu + d_1 + d_2 + d_3) \overset{d_4}{\mapsto} \ldots$$

  - where $d_i > 0$ and the infinite sequence $d_1 + d_2 + \ldots$ converges towards d
  - such path fragments are called time-convergent, i.e. time advances only up to a certain value.
- Time-convergent execution fragments are unrealistic and ignored
  - compare to unrealistic executions in Kripke Structures and fairness constraints that eliminate these

Analysing Semantics

Let $\mathcal{T} = \langle L, L_0, Act, C, \longrightarrow, \iota \rangle$ be a Timed Automaton.

- Infinite path π is time-divergent if $\Delta(\pi) = \infty$
- The function $\Delta : Act \cup \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is defined as follows:

$$\Delta(\tau) = \begin{cases} 0 & \text{if } \tau \in Act \\ d & \text{if } \tau = \tau \in \mathbb{R}_{\geq 0} \end{cases}$$

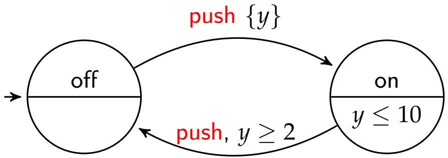
- For infinite execution fragments $\sigma = s_0 \xRightarrow{\tau_1} s_1 \xRightarrow{\tau_2} s_2 \ldots$ in $[\mathcal{T}]$ (with $\Longrightarrow \in \{\rightarrow, \mapsto\}$), let:

$$\Delta(\sigma) = \sum_{i=0}^{\infty} \Delta(\tau_i)$$

- for path fragment π in $[\mathcal{T}]$ induced by execution fragment σ: $\Delta(\pi) = \Delta(\sigma)$
- For a state $s \in [\mathcal{T}]$: $\text{Path}_{\text{div}}(s) = \{\pi \in \text{path}(s) \mid \pi \text{ is time-divergent}\}$

Analysing Semantics

Light automaton:



- The path $\pi \in [\![\text{Light}]\!]$ in which on-and off-periods of one/two time units alternate:

$$\pi = (\text{off}, 0)\ (\text{off}, 1)\ (\text{on}, 0)\ (\text{on}, 1)(\text{on}, 2)\ (\text{off}, 2)\ (\text{off}, 3)\ (\text{on}, 0)\ (\text{on}, 1) \ldots$$

  is time-divergent as $\Delta(\pi) = 1 + 2 + 1 + 2 + \ldots = \infty$

- The path:

$$\pi' = (\text{off}, 0)\ (\text{off}, \frac{1}{2})\ (\text{off}, \frac{3}{4})\ (\text{off}, \frac{7}{8})\ (\text{off}, \frac{15}{16})\ \ldots$$

  is time-convergent, since $\Delta(\pi') = \sum_{i \geq 1} (\frac{1}{2})^i = 1 < \infty$

Analysing Semantics

Let $\mathcal{T} = \langle L, L_0, \mathit{Act}, C, \longrightarrow, \iota \rangle$ be a Timed Automaton.

- State $s \in [\mathcal{T}]$ contains a timelock if $\mathsf{Path}_{\mathsf{div}}(s) = \emptyset$
  - there is no behaviour in s where time can progress *ad infinitum*
- $\mathcal{T}$ is timelock-free if no reachable state in $[\mathcal{T}]$ contains a timelock
- Thus, timelocks can only be detected by means of an analysis of the infinite semantics of $\mathcal{T}$
- Timelocks are usually modelling flaws that should be avoided
  - like deadlocks, we need mechanisms to check their presence

Analysing Semantics

Let $\mathcal{T} = \langle L, L_0, Act, C, \longrightarrow, \iota \rangle$ be a Timed Automaton.

- If $\mathcal{T}$ can perform infinitely many actions in finite time it is Zeno
- A path π in $[\mathcal{T}]$ is Zeno if:
  - it is time-convergent, and
  - infinitely many actions $a \in Act$ are executed along the execution fragment σ underlying path π
- $\mathcal{T}$ is non-Zeno if there is no initial Zeno path in $[\mathcal{T}]$, i.e., for all paths π:
  - $\pi \in \text{path}([\mathcal{T}])$ is time-divergent or
  - π is time-convergent, with nearly all (except for finitely many) transitions being delay transitions
- Zeno paths are considered modelling flaws that should be avoided
  - like deadlocks and timelocks, we need mechanisms to check for Zenoness
- Zenoness can be checked syntactically

Analysing Semantics

Let $\mathcal{T} = \langle L, L_0, Act, C, \longrightarrow, \iota \rangle$ be a Timed Automaton.

Non-Zenoness can be checked directly on the Timed Automaton:

Suppose that for every control cycle:

$$l_0 \xrightarrow{g_1 \;\; a_1 \;\; R_1} l_1 \xrightarrow{g_2 \;\; a_2 \;\; R_2} \dots \xrightarrow{g_n \;\; a_n \;\; R_n} l_n$$

with $l_0 = l_n$, there exists a clock $x \in C$ such that:

1. $x \in R_i$ for some $0 < i \leq n$, and
2. for all clock evaluations ν:

$$\nu(x) < 1 \text{ implies } (\nu \not\models g_j \text{ or } \nu \not\models \iota(l_j)) \text{ for some } 0 < j \leq n$$

Then $\mathcal{T}$ is non-Zeno

Outline

1. Timed Automata
2. Analysing Semantics
3. **Reachability Problem**
4. Clock Equivalence
5. Region Automata
6. Exercise

Reachability Problem

Problem Statement

Let $\mathcal{T} = \langle L, L_0, Act, C, \longrightarrow, \iota \rangle$ be a non-Zeno Timed Automaton and $L_F \subseteq L$.
The reachability problem $(\mathcal{T}, L^F)$ is defined as:

$$\exists \pi \in \mathsf{Path}_{\mathsf{div}}([\mathcal{T}]) : \; \pi(0) \in L^0 \times (C \rightarrow \mathbb{R}_{\geq 0}) \wedge \exists i \in \mathbb{N} : \pi(i) \in L^F \times (C \rightarrow \mathbb{R}_{\geq 0})$$

- Problem: $[\mathcal{T}]$ is infinite state, so it cannot be explored exhaustively.
- Solution: construct a finite quotient of $[\mathcal{T}]$ with respect to a bisimulation relation $\sim$

Reachability Problem

- $[\mathcal{T}]$ contains infinitely many states; only some information is important
- $[\mathcal{T}]$ contains infinitely many transitions; only some are of importance for the reachability problem.

Definition (Time-abstract Transition System)

Let $T = \langle S, S_0, Act, \mapsto, \rightarrow \rangle$ be a timed transition system.
The time-abstract transition system of T is defined as $T_a = \langle S, S_0, Act, \Longrightarrow \rangle$, where:

$$s \stackrel{a}{\Longrightarrow} s' \quad \text{iff} \quad \exists d \in \mathbb{R}_{\geq 0}, s'' \in S : \; s \stackrel{d}{\mapsto} s'' \stackrel{a}{\rightarrow} s'$$

Observation: (s, v) is reachable in T iff (s, v) is reachable in T_a.

Reachability Problem

Definition (Time-abstract bisimulation)

Let $T_a = \langle S, S_0, Act, \Longrightarrow \rangle$ be a time-abstract transition system. Two states s and u are time abstract bisimilar, denoted $s\ B\ u$ iff for all $a \in Act$:

- $s \stackrel{a}{\Longrightarrow} s'$ then there is a state u' such that $u \stackrel{a}{\Longrightarrow} u'$ and $s'\ B\ u'$,
- $u \stackrel{a}{\Longrightarrow} u'$ then there is a state s' such that $s \stackrel{a}{\Longrightarrow} s'$ and $s'\ B\ u'$.

- Informally: two states (l, v) and (l, v') have the same behaviour when:
  1. Any action transition enabled from v is also enabled from v' and the target states have the same behaviour
  2. For any delay transition d from v, there is a delay transition d', such that $(l, v + d)$ and $(l, v' + d')$ have the same behaviour
- A time-abstract bisimulation relation B is L^F sensitive iff whenever $(l, v)\ B\ (l', v')$ implies both l and l' in L^F or both are not in L^F.
- The reachability problem $(\mathcal{T}, L^F)$ can be solved by looking for an L^F sensitive time-abstract bisimulation relation with finitely many equivalence classes.

Outline

Clock Equivalence

Clock Equivalence (1)

- $\nu \models x < c$ whenever $\nu(x) < c$
- Equivalently: $\lfloor \nu(x) \rfloor < c$ (i.e. the greatest integer at most $\nu(x)$)
- $\nu \models x \leq c$ whenever $\nu(x) < c$ or $\nu(x) = c$
- Equivalently: $\lfloor \nu(x) \rfloor < c$ or $\lfloor \nu(x) \rfloor = c$ and $\mathrm{frac}(\nu(x)) = 0$

First proposal

Two clock valuations ν and ν' are equivalent, denoted $\nu \sim \nu'$ iff

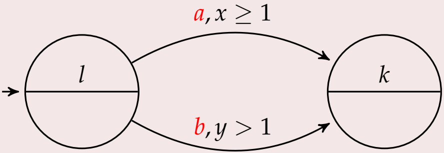
1. for any $x \in C$:
   $\lfloor \nu(x) \rfloor = \lfloor \nu'(x) \rfloor$ and $\mathrm{frac}(\nu(x)) = 0$ iff $\mathrm{frac}(\nu'(x)) = 0$

- Decidability of $\sim$ is guaranteed because clocks are compared to natural numbers.

Clock Equivalence

Example

Consider the following Timed Automaton:



- Assume $0 < v(x) < 1$ and $0 < v(y) < 1$
- Obviously: $(l, v) \overset{a}{\not\rightarrow}$ and $(l, v) \overset{b}{\not\rightarrow}$
- Invariant l is true, so time may elapse
- The transition that is first enabled depends on $x < y$ or $x \geq y$
- This is not covered in the clock equivalence
- Action a is enabled first if $\text{frac}(v(x)) \geq \text{frac}(v(y))$

Clock Equivalence

Clock Equivalence (2)

- Clustering clock valuations with equivalent integer bases is not sufficient.
- Ordering of fractional values of clocks is needed.

Second proposal

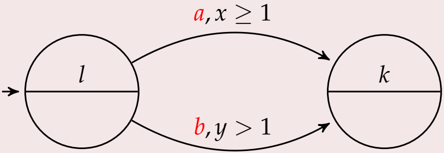
Two clock valuations ν and ν' are equivalent, denoted $\nu \sim \nu'$ iff

1. for any $x \in C$:
   $\lfloor \nu(x) \rfloor = \lfloor \nu'(x) \rfloor$, and $\mathrm{frac}(\nu(x)) = 0$ iff $\mathrm{frac}(\nu'(x)) = 0$
2. for all $x, y \in C$: $\mathrm{frac}(\nu(x)) \leq \mathrm{frac}(\nu(y))$ iff $\mathrm{frac}(\nu'(x)) \leq \mathrm{frac}(\nu'(y))$

Clock Equivalence

Example

Consider the following Timed Automaton:



- Problem second proposal: countable, but still infinite: $1 < x < 2$, $2 < x < 3$, $3 < x < 4$, ...
- Observation: only the clock constraints in $\mathcal{T}$ are relevant.

Clock Equivalence

Clock Equivalence (3)

- Let $c_x \in \mathbb{N}$ be the largest constant to which x is compared in $\mathcal{T}$
- If $\nu(x) > c_x$, then the exact value of x is of no importance (x only grows)

Final proposal

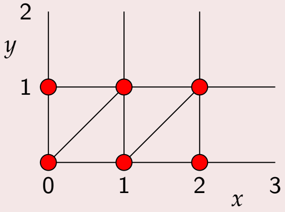
Two clock valuations ν and ν' are equivalent, denoted $\nu \sim \nu'$ iff

1. for any $x \in C$: $\nu(x), \nu'(x) > c_x$ or $\nu(x), \nu'(x) \leq c_x$
2. for any $x \in C$: if $\nu(x), \nu'(x) \leq c_x$ then: $\lfloor \nu(x) \rfloor = \lfloor \nu'(x) \rfloor$ and $\mathrm{frac}(\nu(x)) = 0$ iff $\mathrm{frac}(\nu'(x)) = 0$
3. for any $x, y \in C$: if $\nu(x), \nu'(x) \leq c_x$ and $\nu(y), \nu'(y) \leq c_y$, then: $\mathrm{frac}(\nu(x)) \leq \mathrm{frac}(\nu(y))$ iff $\mathrm{frac}(\nu'(x)) \leq \mathrm{frac}(\nu'(y))$

Clock Equivalence

Example

Consider a Timed Automaton with clocks x and y, with $c_x = 2$ and $c_y = 1$. The clock regions are shown below:



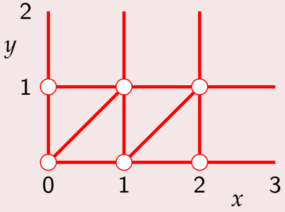
Regions:

- 6 Corner points, e.g. $[(0,0)]$

Clock Equivalence

Example

Consider a Timed Automaton with clocks x and y, with $c_x = 2$ and $c_y = 1$. The clock regions are shown below:



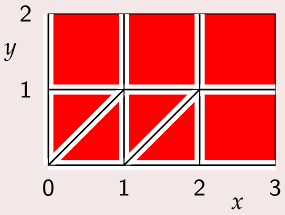
Regions:

- 14 Open line segments: e.g. $[0 < x = y < 1]$

Clock Equivalence

Example

Consider a Timed Automaton with clocks x and y, with $c_x = 2$ and $c_y = 1$. The clock regions are shown below:



Regions:

- 8 Open regions: e.g. $[0 < x < y < 1]$

Clock Equivalence

- The clock region of $\nu \in [C \to \mathbb{R}_{\geq 0}]$, denoted $[\nu]$ is defined by:

$$[\nu] := \{\nu' : C \to \mathbb{R}_{\geq 0} \mid \nu \sim \nu'\}$$

- The state region of a state (l, ν) in $[\mathcal{T}]$ is defined by:

$$[(l, \nu)] := (l, [\nu])$$

- The number of clock regions is bounded from below by:

$$\text{if for all } x \in C:\ c_x \geq 1 \text{ then } R_l := |C|! \times \prod_{x \in C} c_x$$

- The number of clock regions is bounded from above by:

$$\text{if for all } x \in C:\ c_x \geq 1 \text{ then } R_u := |C|! \times 2^{|C|-1} \times \prod_{x \in C} (2(c_x + 1))$$

- The number of state regions in $[\mathcal{T}]/\!\sim$ is finite:

$$|L| \times R_l \quad \leq \quad S/\!\sim \quad \leq \quad |L| \times R_u$$

Clock Equivalence

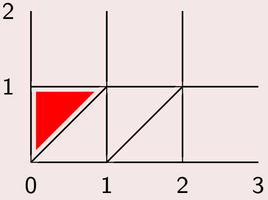
Property

Let C be a set of clocks and let $\phi \in \mathcal{C}(C)$. For $\nu, \nu' : C \to \mathbb{R}_{\geq 0}$ such that $[\nu] = [\nu']$

$$\nu \models \phi \qquad \text{iff} \qquad \nu' \models \phi$$

Example

Consider a Timed Automaton with clocks x and y, with $c_x = 2$ and $c_y = 1$.



$$\begin{cases} \nu(x) = 0.5 \\ \nu(y) = 0.75 \end{cases} \qquad \begin{cases} \nu'(x) = 0.5 \\ \nu'(y) = 0.95 \end{cases}$$

- $\nu, \nu' \in [0 < x < y < 1]$
- $\nu \models x < 2$ iff $\nu' \models x < 2$
- $\nu \models y > 1$ iff $\nu' \models y > 1$

Outline

Region Automata

Let $\mathcal{T} = \langle L, l_0, Act, C, \longrightarrow, \iota \rangle$ be a Timed Automaton.

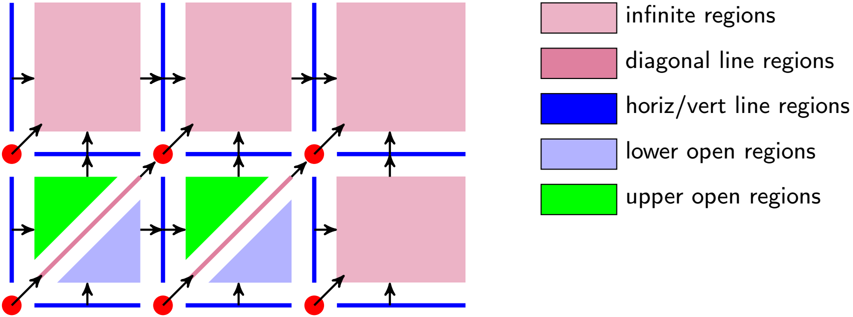
- Clock region $r_\infty = \{v \in [C \to \mathbb{R}_{\geq 0}] \mid \forall x \in C : \; v(x) > c_x\}$ is unbounded
- r' is the successor clock region of r, denoted $r' = \mathsf{succ}(r)$, if either:
  1. $r = r_\infty$ and $r = r'$, or
  2. $r \neq r_\infty, r \neq r'$ and for all $v \in r$:

$$\exists d \in \mathbb{R}_{\geq 0} : \; (v + d \in r' \qquad \text{and} \qquad \forall 0 \leq d' \leq d : \; v + d' \in r \cup r')$$

- The successor region: $\mathsf{succ}((l, v)) := (l, \mathsf{succ}(v))$
- Resetting a region: $r[R := 0] := \{v \in [C \to \mathbb{R}_{\geq 0}] \mid \exists v' \in r : v = [v']_R\}$

Region Automata

Clock regions and their successor regions.



Representation of regions:

1. for every clock x, one clock constraint from the set

$$\{x = c \mid c = 0, 1, \ldots, c_x\} \cup \{c - 1 < x < c \mid c = 1, \ldots, c_x\} \cup \{x > c_x\}$$

2. for every pair of clocks x and y for which $c - 1 < x < c$ and $d - 1 < y < d$ appear in (1), whether $\text{frac}(x)$ is less than, equal to, or greater than $\text{frac}(y)$.

Region Automata

Definition (Region Automaton)

The Region Automaton $R(\mathcal{T})$ of a non-Zeno $\mathcal{T} = \langle L, L_0, Act, C, \longrightarrow, \iota\rangle$ is defined as:

$$R(\mathcal{T}) = \langle S, S_0, Act \cup \{\tau\}, \rightarrow'\rangle$$

- $S = (L \times (C \rightarrow \mathbb{R}_{\geq 0}))/\!\sim\, = \{[s] \mid s \in S_{[\mathcal{T}]}\}$
- $S_0 = \{[s] \mid s \in S_{0\ [\mathcal{T}]}\}$
- $\rightarrow' \subseteq S \times (Act \cup \{\tau\}) \times S$ is defined as:

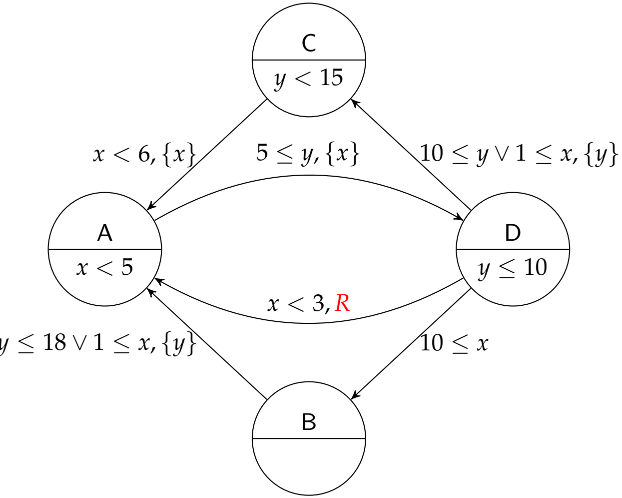
$$\frac{l \xrightarrow{g\ a\ R} l' \qquad r \models g \wedge \iota(l) \qquad r[R := 0] \models \iota(l')}{(l, r) \xrightarrow{a}{}' (l', r[R := 0])} \qquad\qquad \frac{r \models \iota(l) \qquad \mathsf{succ}(r) \models \iota(l)}{(l, r) \xrightarrow{\tau}{}' (l, \mathsf{succ}(r))}$$

Region Automata

- Location l in $\mathcal{T}$ is reachable iff a state region (l, r) is reachable in $R(\mathcal{T})$.
- Safety properties can be translated to reachability problems
- Absence of time-lock in TA $\mathcal{T}$ iff $R(\mathcal{T})$ does not deadlock.
- Extension to model checking for TCTL follows basically the same recipe.

Outline

Exercise



Is the Timed Automaton Non-Zeno when:

- $R = \{x\}$
- $R = \{y\}$
- $R = \{x, y\}$

Is the Timed Automaton Timelock-free when:

- $R = \{x\}$
- $R = \{y\}$
- $R = \{x, y\}$

Explain and motivate your answers.