

Model Checking

René Thiemann

Institute of Computer Science
University of Innsbruck

SS 2009

Outline

- Why Real-Time Systems are Important
- Timed Automata - A Way to Model Real-Time Systems
 - Syntax and Composition
 - Semantic
 - Undesired Behaviors
- Timed CTL
 - Model Checking for Timed CTL
 - Region Transition System
 - Model Checking via CTL
 - Summary

Outline

- Why Real-Time Systems are Important
- Timed Automata - A Way to Model Real-Time Systems
 - Syntax and Composition
 - Semantic
 - Undesired Behaviors
- Timed CTL
 - Model Checking for Timed CTL
 - Region Transition System
 - Model Checking via CTL
 - Summary

Examples

Outline

- Why Real-Time Systems are Important
- Timed Automata - A Way to Model Real-Time Systems
 - Syntax and Composition
 - Semantic
 - Undesired Behaviors
- Timed CTL
 - Model Checking for Timed CTL
 - Region Transition System
 - Model Checking via CTL
 - Summary

Clocks and Constraints

Clocks

- A **clock** can measure times $\in \mathbb{R}^{\geq 0}$
- Clocks are usually written by x, y, \dots , sets of clocks are C, D, \dots

Clock-Constraints

A **clock-constraint** over clocks C is $g \in CC(C)$:

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

where $x \in C, c \in \mathbb{N}$

- Extension to rational numbers possible (but simple to avoid)
- Constraints of form $x - y < c, \dots$ possible (but not considered)

Timed Automata

Main ideas:

- Global time
- Add clock constraints to states (**invariants**) and transitions (**guards**)
- Clocks can be reseted when performing transition
- Time can elapse in states
- Transitions are performed instantaneous
- Parallel composition of timed automata via hand-shaking actions

Timed Automata (formally)

A **timed automata** is octuple $TA = (Loc, Act, C, \rightarrow, Loc_0, Inv, AP, L)$

- Loc : set of locations
- Act : set of actions
- Loc_0 : set of initial locations
- AP : set of atomic propositions
- L : labeling function, $L : Loc \rightarrow 2^{AP}$
- C : set of clocks
- \rightarrow : transition relation, $\rightarrow \subseteq Loc \times CC(C) \times Act \times 2^C \times Loc$
- Inv : invariant assignment, $Inv : Loc \rightarrow CC(C)$

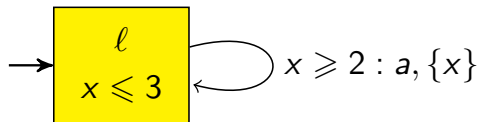
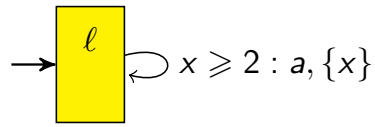
State of transition system for timed automaton consists of location and **clock-evaluation** α ($\alpha : C \rightarrow \mathbb{R}^{\geq 0}$)

Meaning of $s \xrightarrow{g:a,D} t$: If α satisfies g then one can perform a -step and all clocks in D will be reseted to 0.

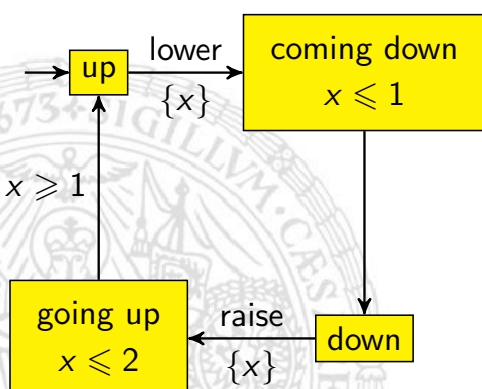
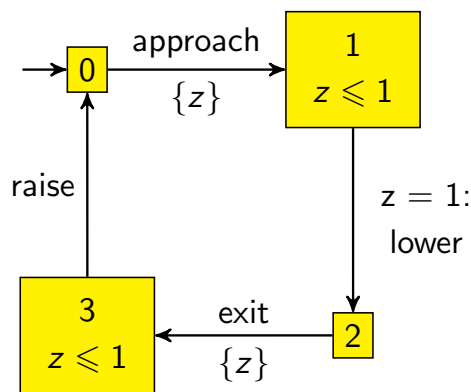
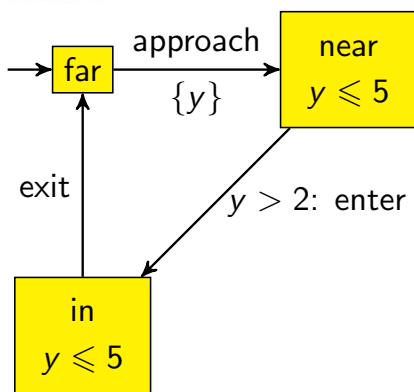
Meaning of $Inv(s) = g$: One can only stay in s if α satisfies g

Example: Guards versus Location Invariants

Convention: Omit constraint "true" and empty set of reseted clocks



Example: Train-Gate-Controller



Composing Timed Automata

Definition

Given $TA_i = (Loc_i, Act_i, C_i, \rightarrow_i, Loc_{0,i}, Inv_i, AP_i, L_i)$ where $AP_1 \cap AP_2 = C_1 \cap C_2 = \emptyset$. Let $H \subseteq Act_1 \cap Act_2$ be a set of **handshake-actions**. Then the timed automaton $TA_1 ||_H TA_2$ is defined as $(Loc_1 \times Loc_2, Act_1 \cup Act_2, C_1 \cup C_2, \rightarrow, Loc_{0,1} \times Loc_{0,2}, Inv, AP_1 \cup AP_2, L)$

- $L((l_1, l_2)) = L_1(l_1) \cup L_2(l_2)$
- $Inv((l_1, l_2)) = Inv_1(l_1) \wedge Inv_2(l_2)$
- \rightarrow is defined as follows:

$$\frac{l_1 \xrightarrow{g_1:a,D_1} l'_1 \quad l_2 \xrightarrow{g_2:a,D_2} l'_2 \quad \text{if } a \in H}{(l_1, l_2) \xrightarrow{g_1 \wedge g_2:a, D_1 \cup D_2} (l'_1, l'_2)}$$

$$\frac{l_1 \xrightarrow{g_1:a,D_1} l'_1}{(l_1, l_2) \xrightarrow{g_1:a,D_1} (l'_1, l_2)} \quad \text{if } a \notin H \quad \frac{l_2 \xrightarrow{g_2:a,D_2} l'_2}{(l_1, l_2) \xrightarrow{g_2:a,D_2} (l_1, l'_2)} \quad \text{if } a \notin H$$

Example

Semantic w.r.t. Clocks

- Given $g \in CC(C)$ and $\alpha : C \rightarrow \mathbb{R}^{\geq 0}$ the meaning of $\alpha \models g$ is obvious
- For $d \in \mathbb{R}^{\geq 0}$ and α define $\alpha + d$ as

$$(\alpha + d)(x) = \alpha(x) + d$$

- For $D \subseteq C$ and α define $\alpha[D := 0]$ as

$$\alpha[D := 0](x) = \begin{cases} 0 & \text{if } x \in D \\ \alpha(x) & \text{otherwise} \end{cases}$$

Transition System for a Timed Automaton

For $TA = (Loc, Act, C, \rightarrow, Loc_0, Inv, AP, L)$ obtain transition system:

- States = Locations + Clock-Evaluation (infinite number of states)
- Discrete Transitions: perform (classical) transition
- Delay Transitions: let time pass and stay in a location

Formally: $TS(TA)$ is the transition system $(S, Act', \rightarrow', I, AP, L')$

- $S = Loc \times (C \rightarrow \mathbb{R}^{\geq 0})$
- $Act' = Act \cup \mathbb{R}^{\geq 0}$
- $I = \{(\ell, \alpha) \mid \ell \in Loc_0, \alpha \models Inv(\ell)\}$ where $\alpha(x) = 0$ for all $x \in C$
- $L'((\ell, \alpha)) = L(\ell)$
- \rightarrow' is composed of two parts: discrete and delay transitions

Transitions of $TS(TA)$

Discrete Transition

$$(\ell, \alpha) \xrightarrow{a} (\ell', \alpha[D := 0]) \quad \text{iff}$$

- $\ell \xrightarrow{g:a,D} \ell'$ is transition in TA
- $\alpha \models g$
- $\alpha[D := 0] \models Inv(\ell')$

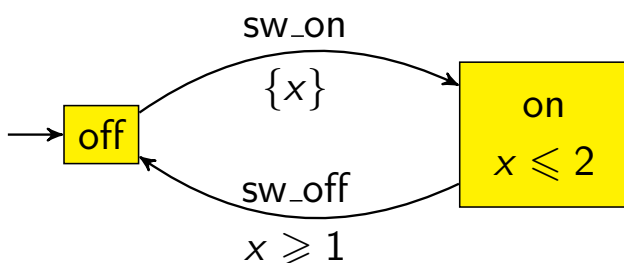
Delay Transition

$$(\ell, \alpha) \xrightarrow{d} (\ell, \alpha + d) \quad \text{iff}$$

- $\alpha + d \models Inv(\ell)$ and $d \in \mathbb{R}^{\geq 0}$

(This implies that $\alpha + d' \models Inv(\ell)$ for all $0 \leq d' \leq d$)

Example



Progress of Time

Essentially, the semantics of TA is obtained from paths in $TS(TA)$

For each path $\pi = s_0 \xrightarrow{\tau_0} s_1 \xrightarrow{\tau_1} s_2 \xrightarrow{\tau_2} \dots$ define its **execution time** as

$$ExecTime(\pi) = \sum_{\tau_i \in \mathbb{R}^{\geq 0}} \tau_i$$

For semantics of TA there are certain undesired paths / states in $TS(TA)$

- **Time convergent paths** (will be ignored)
- States which are **timelocks** (modeling flaw)
- **Zeno paths** (modeling flaw)

Time Convergent Paths

Race of Achilles (10m/s) versus some fast turtle (1m/s)

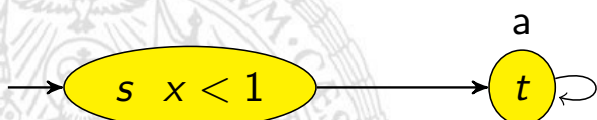
Turtle gets 100m in advance

Time elapsed	Achilles	Turtle
0s	0m	100m
10s	100m	110m
11s	110m	111m

Every time Achilles reaches previous point of turtle, the turtle is already a bit ahead. Thus, the turtle wins!?!

Problem: The above claim is only valid for time-points $< 11.111\dots s$

Similar problem: **time convergent paths** π which satisfy $ExecTime(\pi) < \omega$



Does TA satisfy formula $AF a$? Yes, time-convergent paths like

$s \xrightarrow{\frac{1}{2}} s \xrightarrow{\frac{1}{4}} s \xrightarrow{\frac{1}{8}} \dots$ will be ignored. Only consider **time divergent paths!**

Design Flaws

Usually, **time-convergent paths** cannot be avoided and will just be **ignored for model-checking**

The following kinds of phenomena are seen as **design flaws** and the user has to **modify the timed automata** to get rid of these phenomena

- A state s is a **time-lock** if there is no time-divergent path starting in s .
 TA has a time-lock if there is some reachable state s of $TS(TA)$ which is a time-lock.

Problem of time-locks: **Time cannot proceed** beyond certain point

- A path π is **zeno** if it is time-convergent and contains infinitely many actions $a \in Act$. TA is zeno if there is some (initial) path in $TS(TA)$ which is zeno.

Problem of zeno paths: **infinitely actions in finite time**, unrealistic

Dealing with Design Flaws

- First step: Apply algorithm to detect time-locks and zeno-paths
- Second step: Fix problem
 Example: one way to avoid zeno paths is to add $x \geq \epsilon$ "small value" as additional guard to actions where additionally it is ensured that x is reseted before

Outline

- Why Real-Time Systems are Important
- Timed Automata - A Way to Model Real-Time Systems
 - Syntax and Composition
 - Semantic
 - Undesired Behaviors
- Timed CTL
 - Model Checking for Timed CTL
 - Region Transition System
 - Model Checking via CTL
 - Summary

Timed Computational Tree Logic (TCTL)

A **TCTL-state formula** Φ has the following form:

$$\Phi ::= a \mid g \mid \Phi \wedge \Phi \mid \neg \Phi \mid E \psi \mid A \psi$$

where $a \in AP$ is atomic proposition and $g \in CC(C)$ is clock constraint, and ψ is a **TCTL-path formula**

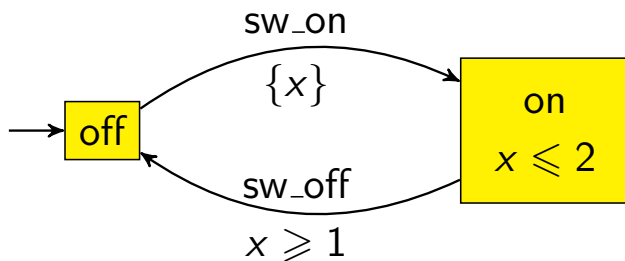
$$\psi ::= \Phi U^J \Phi$$

where $J \subseteq \mathbb{R}^{\geq 0}$ is an interval with bounds in $\mathbb{N} \cup \{\infty\}$

The connectives \vee, true, \dots are derived as usual. Moreover,

$$\begin{aligned} F^J \Phi &\equiv \text{true} U^J \Phi \\ EG^J \Phi &\equiv \neg AF^J \neg \Phi \\ AG^J \Phi &\equiv \neg EF^J \neg \Phi \end{aligned}$$

Examples



Towards the Semantics of TCTL

- Already stated: only time-divergent paths are considered
- Compare

$$s_1 \xrightarrow{\frac{1}{6}} s_1 + \frac{1}{6} \xrightarrow{\frac{7}{6}} s_1 + \frac{4}{3} \xrightarrow{a} s_2 \xrightarrow{4} s_2 \xrightarrow{b} s_3 \dots$$

with

$$s_1 \xrightarrow{\frac{4}{3}} s_1 + \frac{4}{3} \xrightarrow{a} s_2 \xrightarrow{4} s_2 + 4 \xrightarrow{b} s_3 \dots$$

Both paths are equivalent and we only consider the latter one where **consecutive delay-transitions are merged** into one delay-transition

- Afterwards **merge each delay-transition with the following discrete transition** to **compressed path**. Since actions are ignored by (T)CTL, only denote the consumed time:

$$s_1 \xrightarrow{\frac{4}{3}} s_2 \xrightarrow{4} s_3 \dots$$

- If compressed path contains only **finitely many discrete transitions** then use \rightarrow_1 -steps until infinity: $s_1 \xrightarrow{\frac{4}{3}} s_2 \rightarrow_1 s_2 + 1 \rightarrow_1 s_2 + 2 \dots$

Semantics of TCTL

Let $TA = (Loc, Act, C, \rightarrow, Loc_0, Inv, AP, L)$. Then a state s of $TS(TA)$ has the form (ℓ, α) where $\ell \in Loc$ and α is a clock-evaluation.

- $s \models a$ iff $a \in L(\ell)$
- $s \models g$ iff $\alpha \models g$
- $s \models \neg\Phi$ iff $s \not\models \Phi$
- $s \models \Phi \wedge \Psi$ iff $s \models \Phi$ and $s \models \Psi$
- $s \models E\varphi$ iff there is some time-divergent compressed path $\pi: \pi \models \varphi$
- $s \models A\varphi$ iff for all time-divergent compressed paths $\pi: \pi \models \varphi$

Semantics of TCTL (continued)

Let π be a time-divergent compressed path

$$\pi = s_0 \xrightarrow{d_0} s_1 \xrightarrow{d_1} s_2 \xrightarrow{d_2} s_3 \dots$$

Then $\pi \models \Phi U^J \Psi$ iff

- there is some i such that $s_i + d \models \Psi$ for some $d \in [0, d_i]$ with

$$d + \sum_{k=0}^{i-1} d_k \in J$$

and for all $j \leq i$ and all $d' \in [0, d_j]$ such that

$$d' + \sum_{k=0}^{j-1} d_k \leq d + \sum_{k=0}^{i-1} d_k$$

the relation $s_j + d' \models \Phi \vee \Psi$ is valid

As usual, $TA \models \Phi$ iff all initial states satisfy Φ

Some Notes on TCTL

- There is no next-operator (X) since it is unclear what the next point in time should be
- The intervals need not be hit by state of path, e.g.,

$$s_0 \xrightarrow{1} s_1 \xrightarrow{4} s_2 \dots \models F^{[2,3]} a$$

provided that $a \in L(s_1)$

- The semantics of until requires that the left formula is satisfied **from now on** until the right- formula is satisfied, and not only from the start of J onwards, e.g.,

$$s_0 \xrightarrow{1} s_1 \xrightarrow{4} s_2 \dots \not\models a U^{[2,3]} a$$

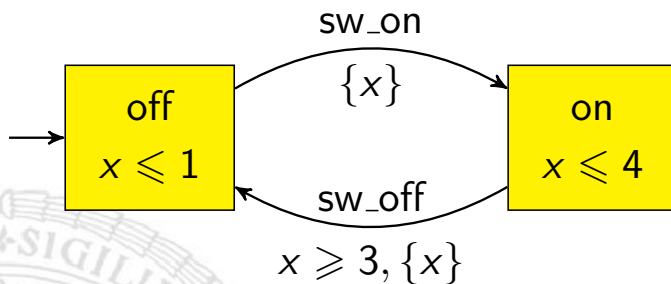
provided that $a \notin L(s_0), a \in L(s_1)$

More Notes on TCTL

In CTL: $\pi \models \Phi U \Psi \dots$ and for all $j \leq i : s_j \models \Phi$

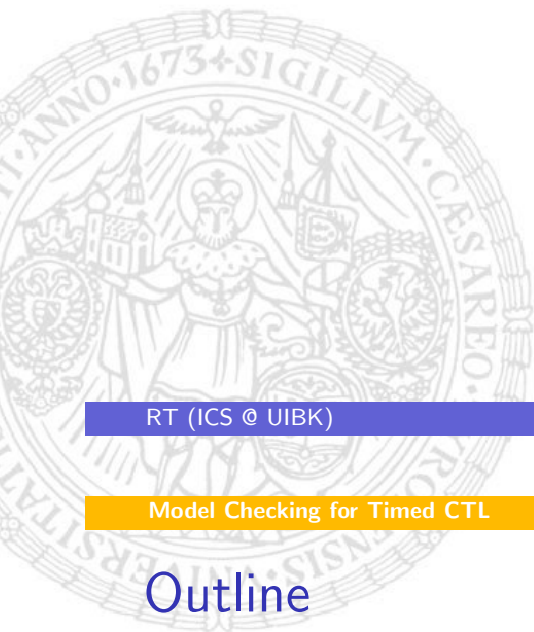
In TCTL: $\pi \models \Phi U^J \Psi \dots$ and for all $j \leq i : s_j + d' \models \Phi \vee \Psi$

- Not a real difference in CTL since $\Phi U \Psi \equiv_{CTL} (\Phi \vee \Psi) U \Psi$
- Allows early satisfaction of right formula:



$$TA \models A \text{ off } U^{[1,2]} \text{ on}$$

Example



Outline

- Why Real-Time Systems are Important
- Timed Automata - A Way to Model Real-Time Systems
 - Syntax and Composition
 - Semantic
 - Undesired Behaviors
- Timed CTL
 - Model Checking for Timed CTL
 - Region Transition System
 - Model Checking via CTL
 - Summary

Overview

Main question:

$$TA \models \Phi$$

for timed automata TA and TCTL-state-formula Φ

- Know: $TA \models \Phi$ iff $TS(TA) \models \Phi$
 - First Problem: $TS(TA)$ has infinitely many states
 - Solution: Construct **region transition system** RTS (quotient of $TS(TA)$) with finitely many states
 - Second Problem: How to deal with intervals J in $\Psi_1 U^J \Psi_2$
 - Solution: Add additional clock which allows to transform Φ into **CTL-formula** Ψ
- $\Rightarrow TA \models \Phi$ iff $RTS \models \Psi$
 \Rightarrow TCTL-model checking boils down to CTL-model checking
- Restriction: From now **only consider non-zero** timed automata

Idea of Region Transition System

Goal: Checking $TA \models \Phi$. Let x be some clock of TA

- First observation: All clock constraints consists of atoms $x < / \leq / \geq / > c$ for some $c \in \mathbb{N}$
- \Rightarrow It does not matter whether $x = 2.334$ or $x = 2.893$, both values of x satisfy the same clock constraints
- \Rightarrow Abstract from concrete value of x , only consider following **intervals**:

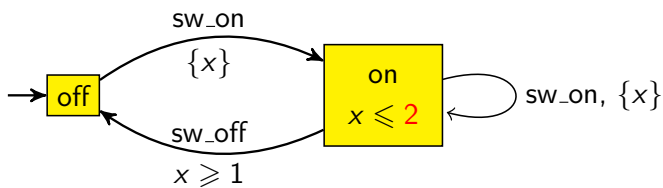
$$\{0\}, (0, 1), \{1\}, (1, 2), \{2\}, (2, 3), \dots$$
- \Rightarrow Far less values, but still infinitely many

- Second observation: There is some **largest constant** c_x which occurs in a clock constraints about x in TA and Φ
- \Rightarrow The following **finite set of intervals** suffices:

$$\{0\}, (0, 1), \{1\}, (1, 2), \dots, (c_x - 1, c_x), \{c_x\}, (c_x, \infty)$$

By just looking at these intervals one can still decide all clock constraints which occur in TA and Φ

Example



Clocks: $C = \{x, y\}$. Question: $TA \models AF(y = 3 \wedge \text{off})$?

- States in $TS(TA)$: (l, x, y) with $l \in \{\text{on}, \text{off}\}$, $x, y \in \mathbb{R}^{\geq 0}$
- From TA and Φ extract $c_x = 2$ and $c_y = 3$
- States in region transition system RTS : (l, x, y) with $l \in \{\text{on}, \text{off}\}$ and (x, y) is one of the following 48 **regions** (point, line segment, or white area):

Delay Transitions in Region Transition System

- Have: finite region transition system, fine enough to decide clock constraints
- ⇒ Possible to mimic **discrete transitions** within region transition system
 - Guard of transition can be checked by region
 - Resetting clocks can be done directly with regions
 - Invariant of locations can be checked by region
- Region transition system is **not fine enough** to mimic **delay transitions**:
 - Consider clocks x, y and region $R = "x \in (0, 1) \wedge y \in (2, 3)"$
 - Want to compute the next region. Candidates:

$$x = 1 \wedge y \in (2, 3) \quad \text{or} \quad x = 1 \wedge y = 3 \quad \text{or} \quad x \in (0, 1) \wedge y = 3$$
 - Problem: all three cases are possible when starting in R

$$x = 0.8, y = 2.3 \quad \text{or} \quad x = 0.7, y = 2.7 \quad \text{or} \quad x = 0.4, y = 2.9$$
 - Solution: construct finer regions where additionally the **fractional parts of clock values are compared** with \leq

Example

With refinement obtain 12 additional regions

Regions (Formally)

$\text{frac}(d)$ denotes the fractional part of d , $\lfloor d \rfloor$ denotes the integral part of d

Definition (Clock Equivalence, Region, Unbounded Region)

Let $\alpha, \beta \in C \rightarrow \mathbb{R}^{\geq 0}$ be clock valuations. Let c_x, c_y, \dots be the maximal occurring constants. Then α and β are **clock equivalent** ($\alpha \approx \beta$) iff one of the following two conditions are satisfied

- for all $x \in C$: $\alpha(x) > c_x$ and $\beta(x) > c_x$
- for all $x, y \in C$ where $\alpha(x), \beta(x) \leq c_x$ and $\alpha(y), \beta(y) \leq c_y$ the following two conditions are satisfied:
 - $\lfloor \alpha(x) \rfloor = \lfloor \beta(x) \rfloor$ and $\text{frac}(\alpha(x)) = 0$ iff $\text{frac}(\beta(x)) = 0$
 - $\text{frac}(\alpha(x)) \leq \text{frac}(\alpha(y))$ iff $\text{frac}(\beta(x)) \leq \text{frac}(\beta(y))$

The **regions** are the equivalence classes of \approx

The **unbounded region** R_∞ is the equivalence class of α (i.e., $R_\infty = [\alpha]_{\approx}$) where $\alpha(x) = c_x + 1$, for all $x \in C$

Number of Regions

Let α be a clock valuation. The corresponding region is identified by

- integral parts α , i.e., by $\lfloor \alpha(x) \rfloor, \lfloor \alpha(y) \rfloor, \dots$

$$\prod_{x \in C} c_x + 1 \text{ possibilities}$$

- being an natural number or not, i.e., by bits $\text{frac}(\alpha(x)) = 0, \dots$

$$2^{|C|} \text{ possibilities}$$

- order of fractional parts, e.g., $\text{frac}(\alpha(x)) = \text{frac}(\alpha(z)) < \text{frac}(\alpha(y))$

$$|C|! \cdot 2^{|C|-1} \text{ possibilities}$$

⇒ number of regions is bounded by

$$\left(\prod_{x \in C} c_x + 1 \right) \cdot 4^{|C|} \cdot |C|!$$

⇒ size of region transition system is **exponential in number of clocks**

Successor of a Region

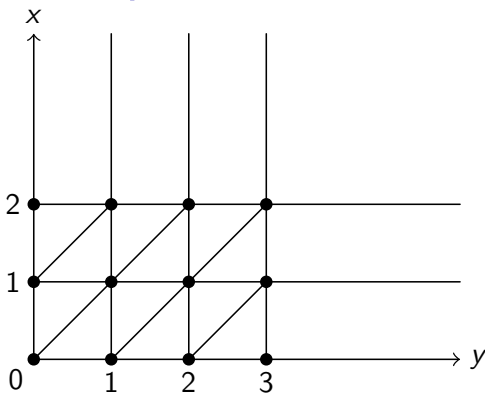
For each region R there is a unique **successor region** $\text{succ}(R)$:

- If $R = R_\infty$ then $\text{succ}(R) = R_\infty$
- If $R \neq R_\infty$ then $\text{succ}(R)$ is the unique region R' such that $R' \neq R$ and for all $\alpha \in R$:

$$\exists d > 0 : (\alpha + d \in R' \text{ and } \forall 0 \leq d' \leq d : \alpha + d' \in R \cup R')$$

So, $R' \neq R$ is the region that is visited next when starting in R

Example



$$x = 0 \wedge 1 < y < 2$$

$$\rightarrow_{succ} 0 < x < 1 \wedge 1 < y < 2 \wedge \text{frac}(x) < \text{frac}(y)$$

$$\rightarrow_{succ} 0 < x < 1 \wedge y = 2$$

$$\rightarrow_{succ} 0 < x < 1 \wedge 2 < y < 3 \wedge \text{frac}(x) > \text{frac}(y)$$

$$\rightarrow_{succ} x = 1 \wedge 2 < y < 3$$

$$\rightarrow_{succ} 1 < x < 2 \wedge 2 < y < 3 \wedge \text{frac}(x) < \text{frac}(y)$$

$$\rightarrow_{succ} 1 < x < 2 \wedge y = 3$$

$$\rightarrow_{succ} 1 < x < 2 \wedge y > 3$$

$$\rightarrow_{succ} x = 2 \wedge y > 3$$

$$\rightarrow_{succ} R_\infty : x > 2 \wedge y > 3$$

Region Transition System

Definition

Let $TA = (Loc, Act, C, \rightarrow, Loc_0, Inv, AP, L)$ and TCTL-formula Φ be given. Then $RTS(TA, \Phi)$ is the **region transition system**

$$RTS = (Loc \times (C \rightarrow \mathbb{R}^{\geq 0} / \cong), \rightarrow', I, AP \cup CC(\Phi), L')$$

- $C \rightarrow \mathbb{R}^{\geq 0} / \cong$ are the clock evaluations modulo \cong , i.e, the regions
- $I = \{(\ell, [\alpha]_{\cong}) \mid \ell \in Loc_0, \alpha \models Inv(\ell)\}$ where $\alpha(x) = 0$ for all $x \in C$
- $CC(\Phi)$ are the clock-constraints that are occurring in Φ
- $L'((\ell, R)) = L(\ell) \cup \{g \in CC(\Phi) \mid R \models g\}$
- $(\ell, R) \rightarrow' (\ell, R')$ if $succ(R) = R'$ and $R' \models Inv(\ell)$
- $(\ell, R) \rightarrow' (\ell', R[D := 0])$ if
 - $\ell \xrightarrow{g:a,D} \ell'$ is transition in TA
 - $R \models g$
 - $R[D := 0] \models Inv(\ell')$

Remarks on Region Transition System

- $R[D := 0] = \{\alpha[D := 0] \mid \alpha \in R\}$
 - $R \models g$ iff **for all** $\alpha \in R : \alpha \models g$ iff **there exists** $\alpha \in R : \alpha \models g$
- ⇒ There is no ambiguity in the labeling
- \cong needs values c_x, c_y, \dots . These are extracted from TA and Φ
 - **Clock constraints** of Φ seen as TCTL-formula **become atomic propositions** in $RTS(TA, \Phi)$

Example

Properties of Region Transition System

Recall: only timed automata are considered which are non-zeno

Theorem

TA has a time-lock iff $RTS(TA, true)$ has a reachable terminal state

⇒ Directly yields method to check for time-locks

Theorem

$$TA \models \Phi \text{ iff } RTS(TA, \Phi) \models \Phi$$

($TS(TA)$ is bisimilar to $RTS(TA, \Phi)$ w.r.t. AP' where AP' does not contain guards exceeding c_x, c_y, \dots)

⇒ Perform CTL-model checking on finite system to answer $TA \models \Phi$

One Remaining Problem

Now: standard CTL-model checking on RTS applicable to answer $TA \models \Phi$

With this approach cover

$$\Phi ::= a \mid g \mid \Phi \wedge \Phi \mid \neg \Phi \mid E \psi \mid A \psi$$

where

$$\psi ::= \Phi U \Phi$$

However, unclear how to handle $\Phi U^J \Phi$ where $J \neq [0, \infty)$

Elimination of Timing Parameters

Aim: Get rid of J in $\Phi U^J \Psi$

Solution:

- Add **fresh clock z** to TA , obtain $TA \uplus \{z\}$
(z is not reseted, neither contained in guards nor in invariants)

$\Rightarrow z$ counts global elapsed time

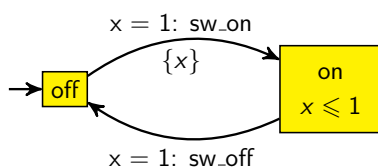
- Lift $Sat(\Phi)$ and $Sat(\Psi)$ from TA to $TA \uplus \{z\}$
- Replace $\Phi U^J \Psi$ by $\xi := (\Phi \vee \Psi) U (z \in J \wedge \Psi)$

Theorem (Elimination of Timing Parameters)

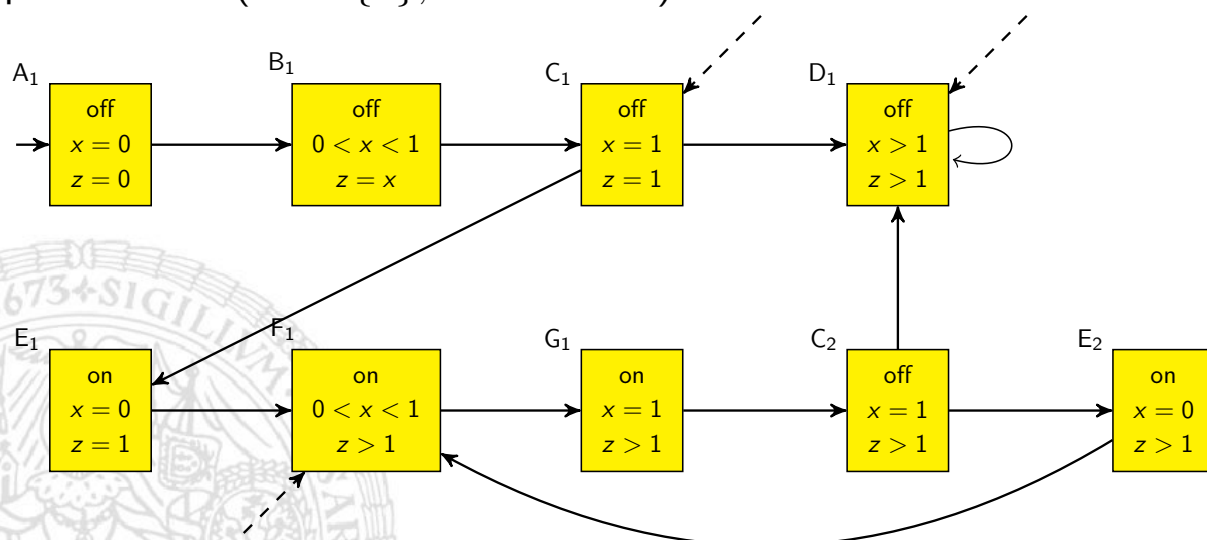
- $s \models E \Phi U^J \Psi$ iff $s[\{z\} := 0] \models E \xi$ (*pure CTL model-checking*)
- $s \models A \Phi U^J \Psi$ iff $s[\{z\} := 0] \models A \xi$ (*pure CTL model-checking*)

Here, s is state of $RTS(TA, \dots)$ and $s[\{z\} := 0]$ is state of $RTS(TA \uplus \{z\}, \dots)$. Note that for building $RTS(TA \uplus \{z\}, \dots)$ one also has to consider the clock constraint $z \in J$ which determines c_z

Example



parts of $RTS(TA \uplus \{z\}, \dots z < 1 \dots)$



Example

Outline

- Why Real-Time Systems are Important
- Timed Automata - A Way to Model Real-Time Systems
 - Syntax and Composition
 - Semantic
 - Undesired Behaviors
- Timed CTL
 - Model Checking for Timed CTL
 - Region Transition System
 - Model Checking via CTL
 - Summary

Summary

- Often modeling is only adequate if real-time aspects can be expressed
- Complex real-time systems can be modeled via composition of timed automata (containing clocks, guards, invariants)
- Timed CTL is extension of CTL where until-operator is equipped with intervals
- Model-checking for timed CTL possible via region transition system (but exponential in number of clocks)