Formal techniques for software and hardware verification

Lecturers:
Vladimir Zakharov
Vladislav Podymov

e-mail:
valdus@yandex.ru

2020, fall semester

Lecture 10

A model-checking algorithm for TCTL

Clock and state regions

Region transition systems

Networks of timed automata

Introduction/Reminder

Model checking problem for TCTL (MC-TCTL)
Given a **sound** timed automaton (*TA*) A and a tctl-formula φ,
check the relation $A \models \varphi$

Model checking problem for CTL (MC-CTL)
Given a Kripke structure (KS) M and a ctl-formula φ,
check the relation $M \models \varphi$

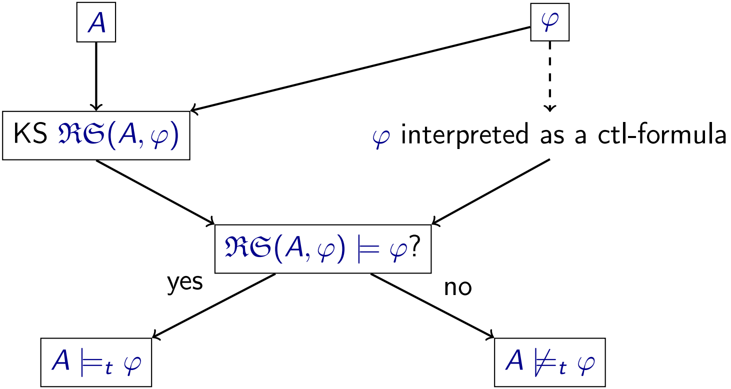
For clarity,
$\models_t$ denotes the satisfiability relation for TCTL in this lecture

AP is, as usual, a finite set of *atomic propositions* —
which is used by default for all TA, KSs and formulas

Solution scheme for MC-TCTL

Given: a TA A, a tctl-formula φ
Check: $A \models_t \varphi$

Solution scheme:



$\mathfrak{RS}(A,\varphi)$ is what will be called
a **region transition system (RTS)**

Preliminary discussion of an RTS

ACC_φ is the set of atomic clock constraints contained in φ

$\mathfrak{RS}(A, \varphi)$ is a KS over the set $AP \cup ACC_\varphi$
(*so that* φ *fits into CTL syntax*)

Each configuration of A is mapped to a state of $\mathfrak{RS}(A, \varphi)$:

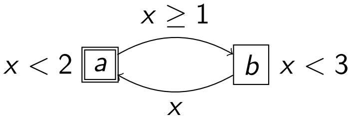
- The set of all *clock valuations* is partitioned into a finite number of equivalence classes (regions)
- Each clock valuation ν is mapped to its region $[\nu]$
- A configuration (ℓ, ν) is mapped to a state $(\ell, [\nu])$
- $[(0, 0, \ldots, 0)] = \{(0, 0, \ldots, 0)\}$

An execution step $(\ell, \nu) \rightarrow (\ell, \nu')$ of A is mapped to a sequence of transitions $(\ell, [\nu]) \rightarrow \cdots \rightarrow (\ell, [\nu'])$
(*so that explicit and implicit parts of all execution steps are "simulated" in a discrete way*)

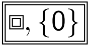
A state $(\ell, [\nu])$ is labeled by propositions from AP w.r.t. ℓ, and by atomic clock constraints w.r.t. ν

Preliminary discussion of an RTS

Example: let us try to construct a KS similar to an RTS for simple TA A and formula φ not knowing any definitions:



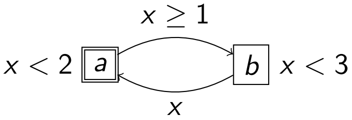
$\mathbf{AGAF}(x = 1)$



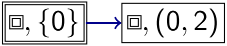
The only initial state is ⧈ + x set to 0

Preliminary discussion of an RTS

Example: let us try to construct a KS similar to an RTS for simple TA A and formula φ not knowing any definitions:



$\mathbf{AGAF}(x = 1)$

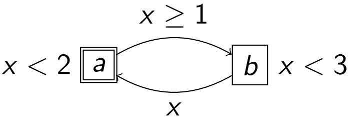


Starting an execution in (⧈, 0), A inevitably waits (*delays*) up until any clock value from $(0, 2)$

Let us "simulate" all such execution steps as a single KS transition

Preliminary discussion of an RTS

Example: let us try to construct a KS similar to an RTS for simple TA A and formula φ not knowing any definitions:



$\mathbf{AGAF}(x = 1)$

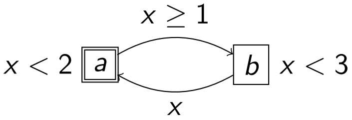
$\boxed{\boxed{⧈, \{0\}}}$ $\boxed{⧈, (0, 1)}$ $\boxed{⧈, [1, 2)}$

For clock values from $[1, 2)$ the top transition of A is enabled, and for values from $(0, 1)$ the transition is disabled

To "simulate" executions
of the top transition of A **deterministically**,
we should split the state $(⧈, (0, 2))$ into $(⧈, (0, 1))$ and $(⧈, [1, 2))$

Preliminary discussion of an RTS

Example: let us try to construct a KS similar to an RTS for simple TA A and formula φ not knowing any definitions:



$\mathbf{AGAF}(x = 1)$

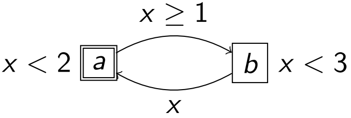
$$\boxed{⧈, \{0\}} \longrightarrow \boxed{⧈, (0, 1)} \longrightarrow \boxed{⧈, [1, 2)}$$

When A **continuously** waits starting in $(⧈, 0)$, the value of x crosses the intervals $\{0\}$, $(0, 1)$, and $[1, 2)$ consecutively

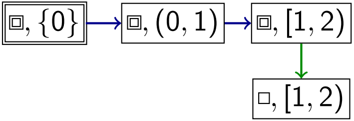
To "simulate" the wait,
let us connect the corresponding states in order

Preliminary discussion of an RTS

Example: let us try to construct a KS similar to an RTS for simple TA A and formula φ not knowing any definitions:



$\mathbf{AGAF}(x = 1)$



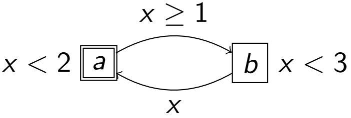
For any configuration of the form $(\boxed{\square}, d)$, where $1 \leq d < 2$,

the relation $(\boxed{\square}, d) \xhookrightarrow{\boxed{\square} \xrightarrow{x \geq 1} \square} (\square, d)$ holds

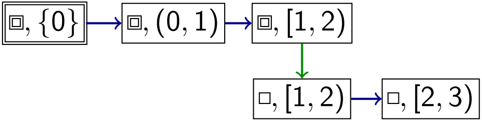
Let us add a state and a transition
to "simulate" all such execution steps

Preliminary discussion of an RTS

Example: let us try to construct a KS similar to an RTS for simple TA A and formula φ not knowing any definitions:



$\mathbf{AGAF}(x = 1)$

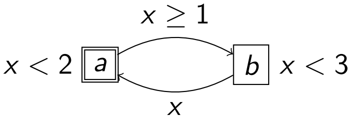


When A waits starting in $(\square, d)$, where $1 \leq d < 2$, the clock value might cross the rest of the interval $[1, 2)$, and then the interval $[2, 3)$

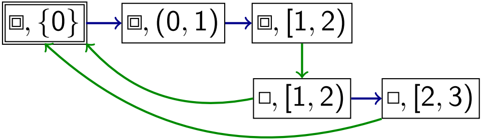
Let us add a state and a transition to "simulate" the wait

Preliminary discussion of an RTS

Example: let us try to construct a KS similar to an RTS for simple TA A and formula φ not knowing any definitions:



$\mathbf{AGAF}(x = 1)$



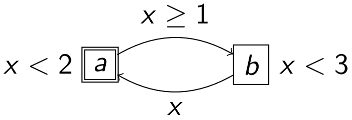
For each configuration $(\square, d)$, where $1 \leq d < 3$,

the relation $(\square, d) \xhookrightarrow{\square \xrightarrow{x} \boxed{\square}} (\boxed{\square}, 0)$ holds

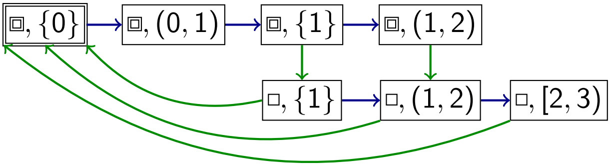
Let us add all transitions corresponding to these execution steps

Preliminary discussion of an RTS

Example: let us try to construct a KS similar to an RTS for simple TA A and formula φ not knowing any definitions:



AGAF$(x = 1)$

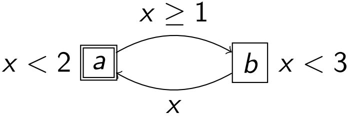


$(x = 1) \equiv (x \leq 1 \,\&\, \neg(x < 1))$:
The formula φ contains constraints $x \leq 1$ and $x < 1$

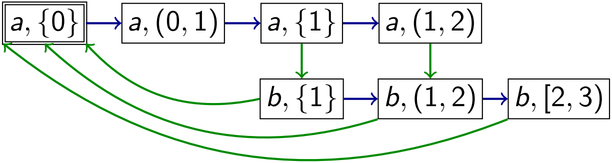
To label KS states with these constraints **deterministically**, we should split the interval $[1, 2)$ into $\{1\}$ and $(1, 2)$

Preliminary discussion of an RTS

Example: let us try to construct a KS similar to an RTS for simple TA A and formula φ not knowing any definitions:



$\mathbf{AGAF}(x = 1)$

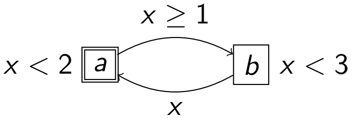


We constructed a KS which contains (*discretely and explicitly*) all execution steps of all runs of A

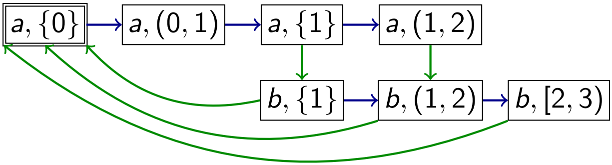
It is not hard to check that $A \models_t \varphi$ and $M \models \varphi$

Preliminary discussion of an RTS

Example: let us try to construct a KS similar to an RTS for simple TA A and formula φ not knowing any definitions:



$\mathbf{AGAF}(x = 1)$



Proceeding to the hard part: in general, ...

- ... how regions should be structured to provide required accuracy, determinism, and finiteness?
- ... how to combine the states of a TA and the regions to magically transform "$\models_t$" into "$\models$"?

Clock regions

Partitioning of clock valuations is based on
a regional equivalence relation ($\sim$) of the valuations,
which will be defined in details a bit later

A clock region is an equivalence class of $\sim$

Clock regions are used as second components
(sets of clock valuations) of RTS states

$\mathfrak{R}$ denotes the set of all clock regions

For technical simplicity,
hereafter in definitions and statements related to $\sim$ we assume
that atomic clock constraints of the form $x - y < k$ and $x - y \leq k$
are **completely forbidden** in A and φ

Clock regions

ACC_A is the set of atomic clock constraints contained in A

Due to mentioned accuracy, finiteness, and determinism features required in construction of an RTS,
$\sim$ should have (*at least*) the following properties:

- Finiteness: the number of equivalence classes of $\sim$ is finite
  - $\Rightarrow$ *the set of RTS states is finite*
- Indistinguishability by clock constraints:
  if $\nu_1 \sim \nu_2$ and $acc \in ACC_A \cup ACC_\varphi$, then $\nu_1 \models acc \Leftrightarrow \nu_2 \models acc$
  - $\Rightarrow$ *determinism w.r.t. guards and state labels*
- Sound reset: if r is a region, and X is a set of clocks, then $r[X] = \{\nu[X] \mid \nu \in r\}$ is a region
  - $\Rightarrow$ *each transition execution step corresponds to a single RTS transition*
- Sound delay: for each region r there is exactly one region r^+ which succeeds r when the TA continuously waits
  - $\Rightarrow$ *each delay execution step corresponds to a single RTS transition*

Clock regions

Definition of $\sim$, first try (unsuccessful)

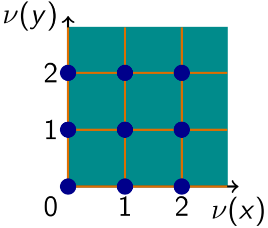
$\lfloor t \rfloor$ is an integer part of a real number t

$frac(t)$ is a fractional part of a real number t

$\nu_1 \sim_1 \nu_2 \quad \Leftrightarrow \quad$ for any clock x the following holds:

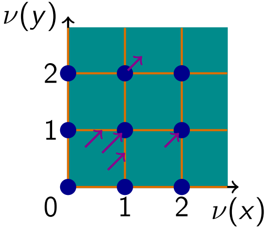
1. $\lfloor \nu_1(x) \rfloor = \lfloor \nu_2(x) \rfloor$
2. $frac(\nu_1(x)) = 0 \Leftrightarrow frac(\nu_2(x)) = 0$

Example: $\sim_1$-regions for two clocks (x, y) are illustrated as connected areas of a single color



Clock regions

Definition of $\sim$, first try (unsuccessful)



Good properties of $\sim_1$*:*

- Indistinguishability by clock constraints
- Sound reset

Remaining problems:

- $|\mathfrak{R}| = \infty$
- r^+ is not uniquely defined by r
  - some pairs (r, r^+) are illustrated with arrows above

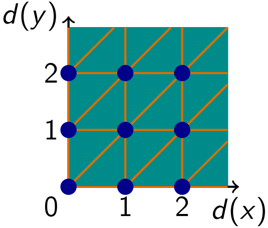
Clock regions

Definition of $\sim$, second try (unsuccessful)

$\nu_1 \sim_2 \nu_2 \quad \Leftrightarrow \quad$ for any clocks x, y the following holds:

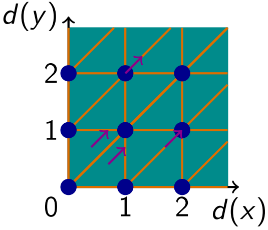
1. $\lfloor \nu_1(x) \rfloor = \lfloor \nu_2(x) \rfloor$
2. $\mathit{frac}(\nu_1(x)) = 0 \Leftrightarrow \mathit{frac}(\nu_2(x)) = 0$
3. $\mathit{frac}(\nu_1(x)) \leq \mathit{frac}(\nu_1(y)) \Leftrightarrow \mathit{frac}(\nu_2(x)) \leq \mathit{frac}(\nu_2(y))$

Example: $\sim_1$-regions for two clocks (x, y) are illustrated as connected areas of a single color



Clock regions

Definition of $\sim$, second try (unsuccessful)



Good properties of $\sim_2$:

- Indistinguishability by clock constraints
- Sound reset
- Sound delay

Remaining problems:

- $|\mathfrak{R}| = \infty$

Clock regions

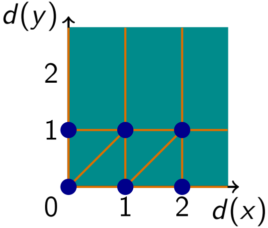
Definition of $\sim$ (*third try, <u>successful</u>*)

k_x is a maximal integer constant used in atomic clock constraints ($x < k$, $x \leq k$) from $ACC_A \cup ACC_\varphi$

$\nu_1 \sim \nu_2 \quad \Leftrightarrow \quad$ for any clocks x, y the following holds:

1. $\nu_1(x) > k_x \Leftrightarrow \nu_2(x) > k_x$
2. if $\nu_1(x) \leq k_x$ и $\nu_1(y) \leq k_y$, then
    - $\lfloor \nu_1(x) \rfloor = \lfloor \nu_2(x) \rfloor$
    - $frac(\nu_1(x)) = 0 \Leftrightarrow frac(\nu_2(x)) = 0$
    - $frac(\nu_1(x)) \leq frac(\nu_1(y)) \Leftrightarrow frac(\nu_2(x)) \leq frac(\nu_2(y))$

Example: regions for two clocks (x, y) and constants $k_x = 2$, $k_y = 1$ are illustrated as connected areas of a single color



Number of regions

Proposition

Let $\mathfrak{R}$ by the set of all regions constructed for a finite set $\mathcal{C}$ of clocks and given constants k_x, $x \in \mathcal{C}$. Then

$$|\mathcal{C}|! \cdot \prod_{x \in \mathcal{C}} k_x \leq |\mathfrak{R}| \leq |\mathcal{C}|! \cdot 2^{|\mathcal{C}|-1} \cdot \prod_{x \in \mathcal{C}} (2k_x + 2)$$

Proof.

Explanations for the expression ...

- $\prod_{x \in C} k_x$ is the number of unit $|\mathcal{C}|$-dimensional cubes covering the product of intervals $[0, k_x]$
- $|C|!$ is the number of orders of fractional parts of clock values
  - Lower bound: at least this much regions are contained in a unit-cube interior

Number of regions

Proposition

Let $\mathfrak{R}$ by the set of all regions constructed for a finite set $\mathcal{C}$ of clocks and given constants k_x, $x \in \mathcal{C}$. Then

$$|\mathcal{C}|! \cdot \prod_{x \in \mathcal{C}} k_x \leq |\mathfrak{R}| \leq |\mathcal{C}|! \cdot 2^{|\mathcal{C}|-1} \cdot \prod_{x \in \mathcal{C}} (2k_x + 2)$$

Proof.

Explanations for the expression ...

- $2k_x + 2$ is the number of possible value intervals for a clock x in a region ($\{0\}; (0,1); \{1\}; \ldots$)
- $2^{|C|-1}$ is the number of options
  to declare fractional parts for different clocks
  to be equal in a region for a given order of these parts ▼

Corollary (Finiteness)

The total number of regions is (*quite high, but*) finite

Other properties of regions

Proposition (Indistinguishability by atomic clock constraints)
If $\nu_1 \sim \nu_2$, x is a clock, and $k \in \{0, 1, \ldots, k_x\}$, then

$$\nu_1 \models x < k \quad \Leftrightarrow \quad \nu_2 \models x < k \quad \text{and}$$
$$\nu_1 \models x \leq k \quad \Leftrightarrow \quad \nu_2 \models x \leq k$$

A clock constraint g over atomic constraints $ACC_A \cup ACC_\varphi$
is satisfied by a region r ($r \models g$)
iff any clock valuation ν from r satisfies g

Proposition (Sound reset)
For any region r and any subset X of clocks
the set $r[X]$ is a region

Other properties of regions

A region is open for a clock x iff
it contains a clock valuation ν such that $\nu(x) > k_x$

A region is open iff it is open for all clocks,
and all other regions are called closed

A region r^+ is a successor of a region r iff the following holds:

- if r is open, then r^+ equals to r
- if r is closed, then r^+ is the region newline for which the following holds:
  - $r^+ \neq r$
  - if $\nu \in r$ and $(\nu + d) \in r^+$, where $d > 0$,
    then for any real number d' such that $0 \leq d' \leq d$
    the relation $(\nu + d') \in r \cup r^+$ holds

Proposition (Sound delay)
For any region r there exists exactly one successor r^+

[Bonus task]: prove the last 3 propositions

State regions, and region transition systems

Given a TA $A = (L, \ell_0, \mathcal{C}, \xi, I, T)$ and a tctl-formula φ, ...

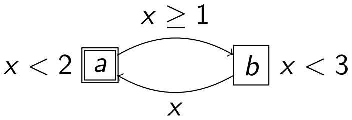
... a state region (ℓ, r) consists of a state ℓ of A and a clock region r

... a region transition system (RTS) $\mathfrak{RS}(A, \varphi)$ is a Kripke structure defined as a subgraph of the following graph G induced by the set of all vertices reachable from the initial one:

- Vertices of G are state regions
- $(\ell_0, \{(0, \dots, 0)\})$ is the initial vertex of G
- Each vertex (ℓ, r) of G is labeled by the set
$$\xi(\ell) \cup \{acc | acc \in ACC_\varphi,\ r \models acc\}$$
- An edge $(\ell, r) \to (\ell', r')$ belongs to G iff at least one of the following holds:
  - $r' = r^+$, $\ell' = \ell$ и $r^+ \models I(\ell)$
  - There exists a transition $\ell \xrightarrow{g,X} \ell'$ of A such that $r \models g$, $r' = r[X]$, and $r' \models I(\ell')$

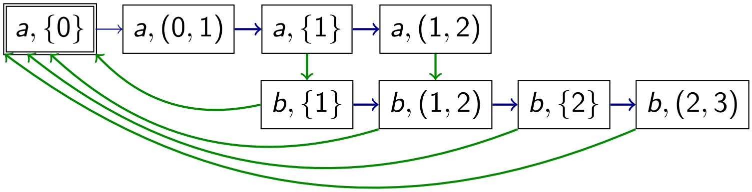
State regions, and region transition systems

Example:



$\mathbf{AGAF}(x = 1)$

A region transition system for these TA and formula:



State regions, and region transition systems

Theorem

For any *sound* timed automaton A and any tctl-formula φ:

$$A \models_t \varphi \quad \Leftrightarrow \quad \mathfrak{RS}(A, \varphi) \models \varphi$$

Proof.

For a clock valuation ν and a state ℓ of A,

- $[\nu]$ is a region of ν
- $[(\ell, \nu)] = (\ell, [\nu])$

It is sufficient to show (*by induction*) that
for any subformula ψ of φ and any configuration σ
generated by any divergent run of A the following holds:

$$A, \sigma \models_t \psi \quad \Leftrightarrow \quad \mathfrak{RS}(A, \varphi), [\sigma] \models \psi$$

For clarity, in the proof we assume that
$A = (L, \ell_0, \mathcal{C}, \xi, I, T)$ and $\mathfrak{RS}(A, \varphi) = (S, s_0, \Rightarrow, \mathcal{L})$

State regions, and region transition systems

Proof. $\quad (A, \sigma \models_t \psi \quad \Leftrightarrow \quad \mathfrak{RS}(A, \varphi), [\sigma] \models \psi)$

Base case (1): $\psi = p \in AP$

$A, (\ell, \nu) \models_t p \Leftrightarrow p \in \xi(\ell)$

$\Leftrightarrow p \in \mathcal{L}(\ell, [\nu]) \Leftrightarrow \mathfrak{RS}(A, \varphi), (\ell, [\nu]) \models p$

Base case (2): $\psi = acc \in ACC_\varphi$

$A, (\ell, \nu) \models_t acc \Leftrightarrow \nu \models_t acc \Leftrightarrow [\nu] \models_t acc$

$\Leftrightarrow acc \in \mathcal{L}(\ell, [\nu]) \Leftrightarrow \mathfrak{RS}(A, \varphi), (\ell, [\nu]) \models acc$

Induction step (1): $\psi = \neg\chi$

$A, \sigma \models_t \neg\chi \Leftrightarrow A, \sigma \not\models_t \chi$

$\Leftrightarrow \mathfrak{RS}(A, \varphi), [\sigma] \not\models \chi \Leftrightarrow \mathfrak{RS}(A, \varphi), [\sigma] \models \neg\chi$

Induction step (2): $\psi = \chi_1 \,\&\, \chi_2$

$A, \sigma \models_t \chi_1 \,\&\, \chi_2 \Leftrightarrow A, \sigma \models_t \chi_1$ and $A, \sigma \models_t \chi_2$

$\Leftrightarrow \mathfrak{RS}(A, \varphi), [\sigma] \models \chi_1$ and $\mathfrak{RS}(A, \varphi), [\sigma] \models \chi_2$

$\Leftrightarrow \mathfrak{RS}(A, \varphi), [\sigma] \models \chi_1 \,\&\, \chi_2$

State regions, and region transition systems

Proof. $(A, \sigma \models_t \psi \quad \Leftrightarrow \quad \mathfrak{RS}(A, \varphi), [\sigma] \models \psi)$

Induction step (3): $\psi = \mathbf{E}(\chi_1 \mathbf{U} \chi_2)$

$(\Leftarrow)$:

Assume that $\mathfrak{RS}(A, \varphi), [\sigma] \models \mathbf{E}(\chi_1 \mathbf{U} \chi_2)$

Then there exist a path $(\gamma_1 \Rightarrow \gamma_2 \Rightarrow \dots)$ from $[\sigma]$ in $\mathfrak{RS}(A, \varphi)$ and an index k such that:

- $\mathfrak{RS}(A, \varphi), \gamma_k \models \chi_2$
- for any state γ_i, $i < k$, the relation $\mathfrak{RS}(A, \varphi), \gamma_i \models \chi_1$ holds

By definition of $\mathfrak{RS}(A, \varphi)$, there exists a divergent σ-trace $(\sigma_1 \to \sigma_2 \to \dots)$ of A such that $[\sigma_i] = \gamma_i$, $i \geq 2$

By induction hypothesis and by definition of $\mathfrak{RS}(A, \varphi)$:

- $A, \sigma_k \models_t \chi_2$
- for any configuration δ generated by $\sigma_1 \to \dots \to \sigma_k$ the inclusion $[\delta] \in \{\gamma_1, \dots, \gamma_k\}$ holds, and therefore $A, \delta \models_t \chi_1$ or $A, \delta \models_t \chi_2$, which means $A, \delta \models_t \chi_1 \vee \chi_2$

State regions, and region transition systems

Proof. $(A, \sigma \models_t \psi \quad \Leftrightarrow \quad \mathfrak{RS}(A, \varphi), [\sigma] \models \psi)$

Induction step (3): $\psi = \mathbf{E}(\chi_1 \mathbf{U} \chi_2)$

($\Rightarrow$):

Assume that $A, \sigma \models_t \mathbf{E}(\chi_1 \mathbf{U} \chi_2)$

Then there exists a σ-trace

$$(\ell_1, \nu_1) \rightarrow (\ell_2, \nu_2) \rightarrow \dots$$

of A and an index k such that:

- $A, (\ell_k, \nu_k) \models_t \chi_2$
- for any configuration δgenerated by

  $$(\ell_1, \nu_1) \rightarrow \dots \rightarrow (\ell_k, \nu_k),$$

  at least one of the following holds: $A, \delta \models_t \chi_1$, or $A, \delta \models_t \chi_2$

State regions, and region transition systems

Proof. $(A, \sigma \models_t \psi \quad \Leftrightarrow \quad \mathfrak{RS}(A, \varphi), [\sigma] \models \psi)$

Induction step (3): $\psi = \mathbf{E}(\chi_1 \mathbf{U} \chi_2)$

$(\Rightarrow)$:

Consider the following path

$$\gamma_1 \Rightarrow \gamma_2 \Rightarrow \dots$$

in $\mathfrak{RS}(A, \varphi)$:

- $\gamma_1 = [(\ell_1, \nu_1)] = (\ell_1, [\nu_1])$
- a step $(\ell_i, \nu_i) \mapsto (\ell_{i+1}, \nu_{i+1})$ for a **closed** region $[\nu_i]$ corresponds to a subpath
  $$(\ell_i, [\nu_i]) \Rightarrow (\ell_i, [\nu_i]^+) \Rightarrow (\ell_i, [\nu_i]^{++}) \Rightarrow \dots \Rightarrow (\ell_i, [\nu_{i+1}])$$
- a step $(\ell_i, \nu_i) \mapsto (\ell_{i+1}, \nu_{i+1})$ for an open region $[\nu_i]$ and a step $(\ell_i, \nu_i) \hookrightarrow (\ell_{i+1}, \nu_{i+1})$ correspond to a subpath $(\ell_i, [\nu_i]) \Rightarrow (\ell_{i+1}, [\nu_{i+1}])$

State regions, and region transition systems

Proof. $(A, \sigma \models_t \psi \quad \Leftrightarrow \quad \mathfrak{RS}(A, \varphi), [\sigma] \models \psi)$

Induction step (3): $\psi = \mathbf{E}(\chi_1 \mathbf{U} \chi_2)$

$(\Rightarrow)$:

By definition of $\mathfrak{RS}(A, \varphi)$ and induction hypothesis, there exists an index m such that:

- $\mathfrak{RS}(A, \varphi), \gamma_m \models \chi_2$
- for each state γ_i, $i < m$, there exists a configuration δ generated by $(\ell_1, \nu_1) \to \cdots \to (\ell_k, \nu_k)$ such that $[\delta] = \gamma_i$, and therefore $\mathfrak{RS}(A, \varphi), \gamma_i \models \chi_1$ or $\mathfrak{RS}(A, \varphi), \gamma_i \models \chi_2$, which means $\mathfrak{RS}(A, \varphi), \gamma_i \models \chi_2 \vee \chi_2$

Thus, $\mathfrak{RS}(A, \varphi), \gamma_1 \models \mathbf{E}(\chi_1 \mathbf{U} \chi_2)$

Induction step (4): $\psi = \mathbf{A}(\chi_1 \mathbf{U} \chi_2)$ — is analogous to (3) ▼

State regions, and region transition systems

Theorem

For any *sound* timed automaton A and any tctl-formula φ:

$$A \models_t \varphi \quad \Leftrightarrow \quad \mathfrak{RS}(A, \varphi) \models \varphi$$

[Bonus task]:

Define the regional equivalence and prove the same theorem for the general case, in which constraints $x - y < k$ and $x - y \leq k$ are allowed in A and φ

Networks of timed automata

A real-time system usually contains several components
running **in parallel** and communicating with each other

A timed automaton is a **sequential** model

Attemps to describe a parallel (distributed) RTS
in sequential terms "by hand" usually lead to subtle errors
which negate all formal verification guarantees

To avoid such errors, it is sufficient to have means to design and
analyze parallel collections of communicating timed automata

Networks of timed automata

A synchronized timed automaton is defined over finite sets of *atomic propositions* (AP) and communication channels (CH)

The only syntactic difference between a syncronized TA and a "usual" one is: each transition of a synchronized TA is additionally marked with one of the expressions $c!$, $c?$, or λ, where $c \in CH$, which means that when the transition is executed, a signal is sent via c, or a signal is received via c, or no communication happens

$Sync(CH) = \{c! \mid c \in CH\} \cup \{c? \mid c \in CH\} \cup \{\lambda\}$

Thus, the set of all possible transitions of a synchronized TA $A = (L, \ell_0, \mathcal{C}, \xi, I, T)$ over AP and CH has the following form:

$$L \times Guard(\mathcal{C}) \times Sync(CH) \times 2^{\mathcal{C}} \times L$$

$\ell \xrightarrow{g,s,X} \ell'$ is an illustration of a transition (ℓ, g, s, X, ℓ')

Networks of timed automata

A network of timed automata (NTA) over AP is a tuple $(\mathcal{C}, CH, (A_1, \ldots, A_k))$, where

- $\mathcal{C}$ and CH are finite sets of **clocks** and **channels**, respectively
- $A_i = (L^i, \ell_0^i, \mathcal{C}, \xi^i, I^i, T^i)$, $1 \leq i \leq k$, is a synchronized TA over AP_i and CH
- $L^i \cap L^j = \emptyset$ for $1 \leq i < j \leq k$
- $AP_i \cap AP_j = \emptyset$ for $1 \leq i < j \leq k$
- $AP_1 \cup \cdots \cup AP_k = AP$

Networks of timed automata

A configuration of an NTA $(\mathcal{C}, CH, (A_1, \ldots, A_k))$ for $A_i = (L^i, \ell_0^i, \mathcal{C}, \xi^i, I^i, T^i)$ is a pair $(\vec{\ell}, \nu)$, where

- $\vec{\ell} \in L^1 \times L^2 \times \cdots \times L^k$
- ν is a clock valuation over C

An initial configuration of the NTA is $(\ell_0^1, \ldots, \ell_0^k, 0, \ldots, 0)$

All basic denotations for configurations ($\sigma + d$, $\sigma[X]$, $\sigma[\ell/\ell']$) are directly and naturally extended from TA to NTA

An execution step ($\rightarrow$) of an NTA is a union of **three** kinds of steps:

- A delay step: $\sigma \mapsto \sigma'$
- A transition step: $\sigma \hookrightarrow \sigma'$
- A (peer-to-peer) synchronization step: $\sigma \Rightarrow \sigma'$

Networks of timed automata

Let $N = (\mathcal{C}, CH, (A_1, \ldots, A_k))$ be an NTA, where $A_i = (L^i, \ell_0^i, \mathcal{C}, \xi^i, I^i, T^i)$, and $\sigma = (\ell_1, \ldots, \ell_k, \nu)$ — a configuration

Delay step

$\sigma \overset{d}{\mapsto} \sigma'$, where $d \in \mathbb{R}_{>0}$,
if $\sigma' = \sigma + d$ and $\nu + d \models I^1(\ell_1) \,\&\ldots\&\, I^k(\ell_k)$

$\sigma \mapsto \sigma'$ iff there exists d, $d \in \mathbb{R}_{>0}$, such that $\sigma \overset{d}{\mapsto} \sigma'$

Transition step

$\sigma \xhookrightarrow{\ell_i \xrightarrow{g,\lambda,X} \ell_i'} \sigma'$, where $\ell_i \xrightarrow{g,\lambda,X} \ell_i' \in T^i$, if:

- $\sigma' = \sigma[X][\ell_i/\ell_i']$
- $\nu \models g$
- $\nu[X] \models I^i(\ell_i')$

$\sigma \hookrightarrow \sigma'$ iff there exists a TA A_i in N
and a transition t of A_i such that $\sigma \overset{t}{\hookrightarrow} \sigma'$

Networks of timed automata

Let $N = (\mathcal{C}, CH, (A_1, \ldots, A_k))$ be an NTA, where $A_i = (L^i, \ell_0^i, \mathcal{C}, \xi^i, I^i, T^i)$, and $\sigma = (\ell_1, \ldots, \ell_k, \nu)$ — a configuration

Synchronization step

$\sigma \xRightarrow{t_1, t_2} \sigma'$, where $t_1 = (\ell_i \xrightarrow{g_1, c!, X_1} \ell'_i) \in T^i$, $t_2 = (\ell_j \xrightarrow{g_2, c?, X_2} \ell'_j) \in T^j$, and $i \neq j$, if:

- $\sigma' = \sigma[X_1][X_2][\ell_i/\ell'_i][\ell_j/\ell'_j]$
- $\nu \models g_1 \,\&\, g_2$
- $\nu[X_1][X_2] \models I^i(\ell'_i) \,\&\, I^j(\ell'_j)$

$\sigma \Rightarrow \sigma'$ iff there exist TA A_i and A_j in N, $i \neq j$, and transitions t_1, t_2 of these automata such that $\sigma \xRightarrow{t_1, t_2} \sigma'$

Sequentialization of NTA

An NTA N and a TA A are equivalent iff
the execution step relations for N and A are equal

Theorem

For any NTA N there exists an equivalent TA A

Proof.

Let $N = (\mathcal{C}, CH, (A_1, \dots, A_k))$, where $A_i = (L^i, \ell_0^i, \mathcal{C}, \xi^i, I^i, T^i)$

A required TA $A = (L, \ell_0, \mathcal{C}, \xi, I, T)$ may be constructed as follows:

- $L = L^1 \times \cdots \times L^k$
- $\ell_0 = (\ell_0^1, \dots, \ell_0^k)$
- $\xi(\ell_1, \dots, \ell_k) = \xi^1(\ell_1) \cup \cdots \cup \xi^k(\ell_k)$
- $I(\ell_1, \dots, \ell_k) = I^1(\ell_1) \,\&\, \dots \,\&\, I^k(\ell_k)$

Sequentialization of NTA

An NTA N and a TA A are equivalent iff
the execution step relations for N and A are equal

Theorem

For any NTA N there exists an equivalent TA A

Proof.

Let $N = (\mathcal{C}, CH, (A_1, \ldots, A_k))$, where $A_i = (L^i, \ell_0^i, \mathcal{C}, \xi^i, I^i, T^i)$

A required TA $A = (L, \ell_0, \mathcal{C}, \xi, I, T)$ may be constructed as follows:

- T consists of the following transitions:
  - $(\ell_1, \ldots, \ell_m) \xrightarrow{g, X} (\ell_1, \ldots, \ell_{i-1}, \ell'_i, \ell_{i+1}, \ldots, \ell_m)$,
    if $\ell_i \xrightarrow{g, \lambda, X} \ell'_i$
  - $(\ell_1, \ldots, \ell_m) \xrightarrow{g_1 \,\&\, g_2, X_1 \cup X_2}$
    $(\ell_1, \ldots, \ell_{i-1}, \ell'_i, \ell_{i+1}, \ldots, \ell_{j-1}, \ell'_j, \ell_{j+1}, \ldots, \ell_m)$, if
    - $\ell_i \xrightarrow{g_1, c!, X_1} \ell'_i$ and $\ell_j \xrightarrow{g_2, c?, X_2} \ell'_j$, or
    - $\ell_i \xrightarrow{g_1, c?, X_1} \ell'_i$ and $\ell_j \xrightarrow{g_2, c!, X_2} \ell'_j$