Modal Logics for Timed Control

Patricia Bouyer[1], Franck Cassez[2] and François Laroussinie[1]

[1]LSV, ENS-Cachan [2]IRCCyN, Nantes

France

CONCUR'05

San Francisco, CA

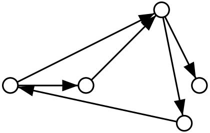
Outline of the talk

- Control of Timed Systems
- Controllability with L_ν

Outline

- **Control of Timed Systems**
- **Controllability with L_ν**

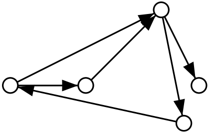
Model Checking and Control Problems



$\Box$ (not bad)

S ϕ

Model Checking and Control Problems



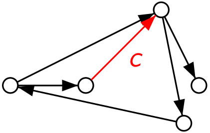
$\Box$ (not bad)

$S \qquad \models \qquad \phi$

Model Checking Problem

Does S satisfy ϕ ?

Model Checking and Control Problems



$\Box$ (not bad)

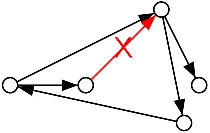
S

ϕ

Model Checking Problem

Does S satisfy ϕ ?

Model Checking and Control Problems



$\square$ (not bad)

S ϕ

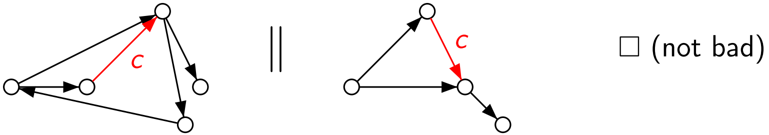
Model Checking Problem

Does S satisfy ϕ ?

Control Problem

Can S be restricted to satisfy ϕ ?

Model Checking and Control Problems



$$S \quad \| \quad C \quad \models \quad \phi$$

Model Checking Problem

Does S satisfy ϕ ?

Control Problem

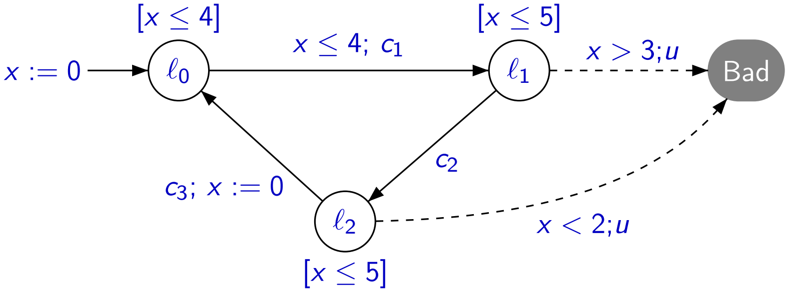
Can S be restricted to satisfy ϕ ?

Is there a Controller C s.t. $(S \parallel C) \models \phi$?

Model for Timed Systems: Timed Automata

- TA = Finite Automata + clocks

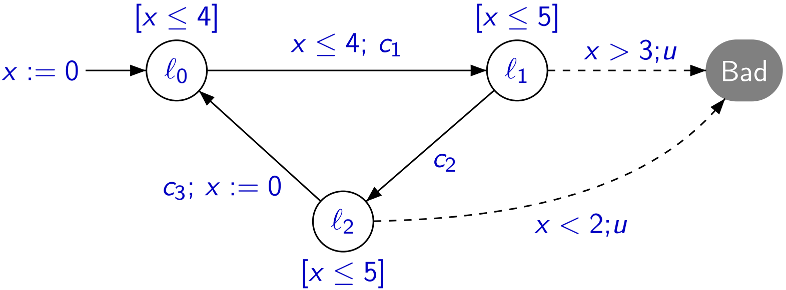
▸ Timed Automata



Model for Timed Systems: Timed Automata

- TA = Finite Automata + clocks

▸ Timed Automata



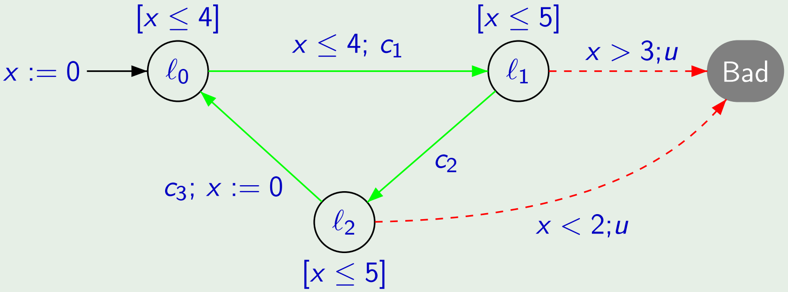
- Semantics = runs = sequences of dense-time and discrete steps

$$\rho: \quad (\ell_0, 0) \xrightarrow{1.1} (\ell_0, 1.1) \xrightarrow{c_1} (\ell_1, 1.1) \xrightarrow{2.1} (\ell_1, 3.2) \xrightarrow{c_2} (\ell_2, 3.2) \xrightarrow{0.1} (\ell_2, 3.3) \xrightarrow{u} (\ell_0, 0) \cdots$$

Model for Control: Timed Game Automata

- TGA = TA + controllable and uncontrollable actions

Actions partitioned as $\mathrm{Act}_c = \{c_1, c_2, c_3\}$ $\mathrm{Act}_u = \{u\}$

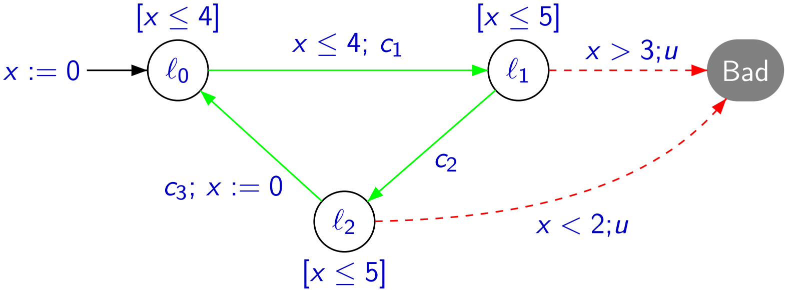


- Control Objective = subset of the runs of a TGA

Safety objective

"Avoid the Bad state"

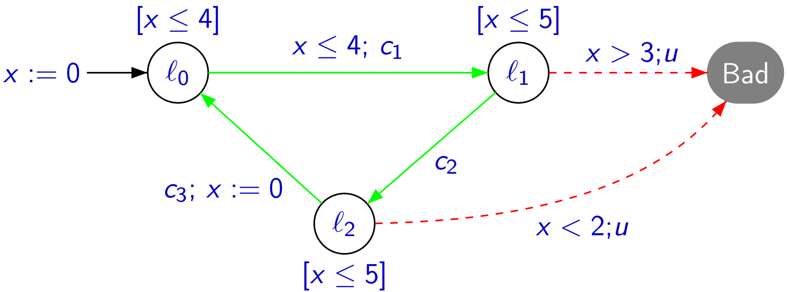
Solving Timed Games (1/2)



- A general controller is defined by a strategy f if ρ is a run from the initial state:

$$f(\rho) = \textit{do a controllable action or do nothing}$$

Solving Timed Games (1/2)

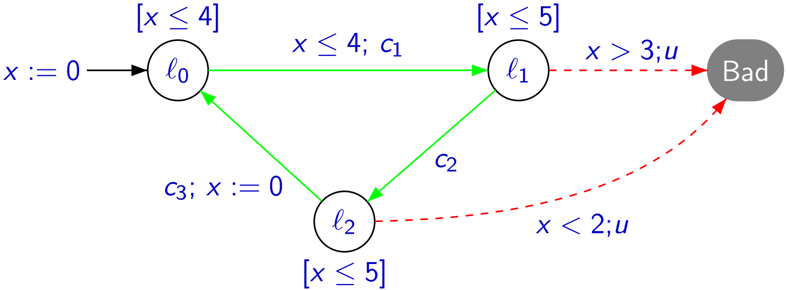


- A general controller is defined by a strategy f

A Partial Strategy f

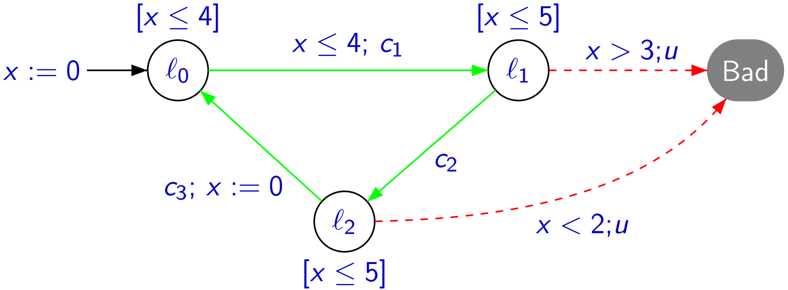
$f(\textit{each run ending in } \ell_0, x < 2) = \textit{do nothing}$
$f(\textit{each run ending in } \ell_0, x = 2) = c_1$

Solving Timed Games (1/2)



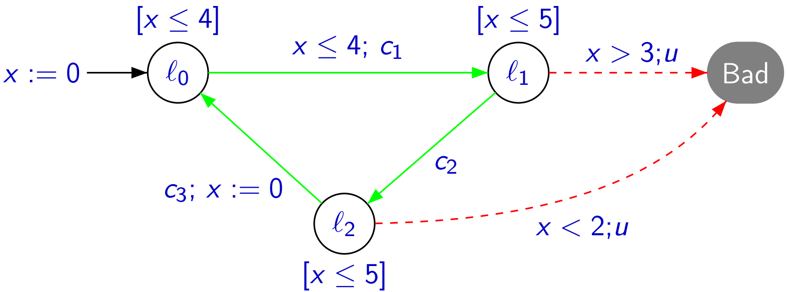
- A general controller is defined by a strategy f
- A strategy restricts the set of runs of the TGA

Solving Timed Games (1/2)



- A general controller is defined by a strategy f
- A strategy restricts the set of runs of the TGA
- $(G \parallel f) \quad = \quad G$ controlled by strategy f

Solving Timed Games (1/2)



- A general controller is defined by a strategy f
- A strategy restricts the set of runs of the TGA
- $(G \parallel f) \quad = \quad G$ controlled by strategy f
- Given ϕ a control objective, s a state,
  The strategy f is winning from s if $s \models \phi$ in $(G \parallel f)$
  The state s is winning if there is a winning strategy f_s from s

Solving Timed Games (2/2)

- Input: a TGA G and a control objective ϕ
- Problem: is there a strategy f s.t. $(G \parallel f) \models \phi$?
- Solution: compute the set of winning states
  1. define a controllable predecessors operator
  2. compute a fixed point that gives the set of winning states
  3. check whether the initial state is winning

Fundamental Results for Timed Control

[Maler et al., 95, De Alfaro et al., 01]

- Control Problem is EXPTIME-Complete for TA and reachability objectives
- Controller Synthesis is effective
- Memoryless strategies are sufficient to win

Our Contribution

- Control objective in L_ν (safety and bounded liveness)

Our Contribution

- Control objective in L_ν (safety and bounded liveness)
- Reduction of the Control Problem for (TA, L_ν) to a Model-Checking Problem for (TA, L_ν^c) :

$$\textit{there is a strategy } f \textit{ s.t. } (G \parallel f) \models \phi \iff G \models \overline{\phi}$$

Our Contribution

- Control objective in L_ν (safety and bounded liveness)
- Reduction of the Control Problem for (TA, L_ν) to a Model-Checking Problem for (TA, L_ν^c) :

$$\textit{there is a strategy } f \textit{ s.t. } (G \parallel f) \models \phi \iff G \models \overline{\phi}$$

- Properties of the new logic L_ν^c
  - Expressiveness
  - Model Checking over TA
  - Compositionality

Our Contribution

- Control objective in L_ν (safety and bounded liveness)
- Reduction of the Control Problem for (TA, L_ν) to a Model-Checking Problem for (TA, L_ν^c) :

$$\textit{there is a strategy } f \textit{ s.t. } (G \parallel f) \models \phi \iff G \models \overline{\phi}$$

- Properties of the new logic L_ν^c
  - Expressiveness
  - Model Checking over TA
  - Compositionality
- Implementation
  The tool CMC [Laroussinie et al., 98]

Outline

- Control of Timed Systems
- **Controllability with L_ν**

The Timed Modal Logic L_ν

- Atomic propositions + and, or

The Timed Modal Logic L_ν

- Atomic propositions + and, or
- Discrete step properties: $\langle a \rangle \varphi$, $[a]\, \varphi$, a an action

The Timed Modal Logic L_ν

- Atomic propositions + and, or
- Discrete step properties: $\langle a \rangle\, \varphi$, $[a]\, \varphi$, a an action
- Time step properties: $\langle \delta \rangle\, \varphi$, $[\delta]\, \varphi$

The Timed Modal Logic L_ν

- Atomic propositions + and, or + Clock Constraints $x \leq c$
- Discrete step properties: $\langle a \rangle\, \varphi$, $[a]\, \varphi$, a an action
- Time step properties: $\langle \delta \rangle\, \varphi$, $[\delta]\, \varphi$
- Time guarded properties: $x\ \underline{\text{in}}\ \varphi$ with x a formula clock

The Timed Modal Logic L_ν

- Atomic propositions + and, or + Clock Constraints $x \leq c$
- Discrete step properties: $\langle a \rangle\, \varphi$, $[a]\, \varphi$, a an action
- Time step properties: $\langle \delta \rangle\, \varphi$, $[\delta]\, \varphi$
- Time guarded properties: x $\underline{\text{in}}$ φ with x a formula clock
- Greatest fixed point properties: $Z =_\nu \varphi(Z)$

The Timed Modal Logic L_ν

- Atomic propositions + and, or + Clock Constraints $x \leq c$
- Discrete step properties: $\langle a \rangle\ \varphi$, $[a]\ \varphi$, a an action
- Time step properties: $\langle \delta \rangle\ \varphi$, $[\delta]\ \varphi$
- Time guarded properties: $x\ \underline{\text{in}}\ \varphi$ with x a formula clock
- Greatest fixed point properties: $Z =_\nu \varphi(Z)$

Syntax of L_ν

$$\begin{aligned} L_\nu \ni \varphi \quad ::= \quad & p \mid tt \mid ff \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \\ & x\ \underline{\text{in}}\ \varphi \mid x \bowtie c \\ & \mid [a]\ \varphi \mid \langle a \rangle\ \varphi \mid [\delta]\ \varphi \mid \langle \delta \rangle\ \varphi \mid \\ & Z =_\nu \phi \end{aligned}$$

where p an atomic prop., $a \in \text{Act}$, x a formula clock, $\bowtie \in \{<, \leq, =, \geq, >\}$, $c \in Q_{\geq 0}$, Z an identifier.

The Timed Modal Logic L_ν

- Atomic propositions + and, or + Clock Constraints $x \leq c$
- Discrete step properties: $\langle a \rangle\, \varphi$, $[a]\, \varphi$, a an action
- Time step properties: $\langle \delta \rangle\, \varphi$, $[\delta]\, \varphi$
- Time guarded properties: x in φ with x a formula clock
- Greatest fixed point properties: $Z =_\nu \varphi(Z)$

Some L_ν formulas

Σ the alphabet of all actions, x a formula clock, s a state of a TA

The Timed Modal Logic L_ν

- Atomic propositions + and, or + Clock Constraints $x \leq c$
- Discrete step properties: $\langle a \rangle\, \varphi$, $[a]\, \varphi$, a an action
- Time step properties: $\langle \delta \rangle\, \varphi$, $[\delta]\, \varphi$
- Time guarded properties: x in φ with x a formula clock
- Greatest fixed point properties: $Z =_\nu \varphi(Z)$

Some L_ν formulas

Σ the alphabet of all actions, x a formula clock, s a state of a TA

- "No a is enabled in s": $(s, x) \models [a]\, \mathit{ff}$

The Timed Modal Logic L_ν

- Atomic propositions + and, or + Clock Constraints $x \leq c$
- Discrete step properties: $\langle a \rangle\, \varphi$, $[a]\, \varphi$, a an action
- Time step properties: $\langle \delta \rangle\, \varphi$, $[\delta]\, \varphi$
- Time guarded properties: $x \ \underline{\text{in}}\ \varphi$ with x a formula clock
- Greatest fixed point properties: $Z =_\nu \varphi(Z)$

Some L_ν formulas

Σ the alphabet of all actions, x a formula clock, s a state of a TA

- "No a is enabled in s": $(s, x) \models [a]\, \mathit{ff}$
- "At most 5 t.u. can elapse from s": $(s, x) \models x \ \underline{\text{in}}\ [\delta]\, (x \leq 5)$

The Timed Modal Logic L_ν

- Atomic propositions + and, or + Clock Constraints $x \leq c$
- Discrete step properties: $\langle a \rangle\, \varphi$, $[a]\, \varphi$, a an action
- Time step properties: $\langle \delta \rangle\, \varphi$, $[\delta]\, \varphi$
- Time guarded properties: $x \; \underline{\text{in}} \; \varphi$ with x a formula clock
- Greatest fixed point properties: $Z =_\nu \varphi(Z)$

Some L_ν formulas

Σ the alphabet of all actions, x a formula clock, s a state of a TA

- "No a is enabled in s": $(s, x) \models [a]\, f\!f$
- "At most 5 t.u. can elapse from s": $(s, x) \models x \; \underline{\text{in}} \; [\delta]\, (x \leq 5)$
- "The states that avoid Bad": $(s, x) \in Z$, $Z =_\nu \overline{\text{BAD}} \wedge [\Sigma]\, Z \wedge [\delta]\, Z$

Semantics of L_ν

Given A a TA, ϕ an L_ν formula, ρ an assignment for identifiers (Z)
Interpretation of ϕ in context ρ is a set of extended states (s, w) with:

- $s = (\ell, v)$ a state of A and w a valuation of the formula clocks
- ρ assigns to each identifier a set of extended states

Semantics of L_ν

Given A a TA, ϕ an L_ν formula, ρ an assignment for identifiers (Z)
Interpretation of ϕ in context ρ is a set of extended states (s, w) with:

- $s = (\ell, v)$ a state of A and w a valuation of the formula clocks
- ρ assigns to each identifier a set of extended states

$$\llbracket x \bowtie c \rrbracket \rho \stackrel{\text{def}}{=} \{(s,u) \mid u(x) \bowtie c\}$$
$$\llbracket \varphi_1 \vee \varphi_2 \rrbracket \rho \stackrel{\text{def}}{=} \llbracket \varphi_1 \rrbracket \rho \cup \llbracket \varphi_2 \rrbracket \rho \quad (\cap \textit{ for } \wedge)$$
$$\llbracket \langle a \rangle\, \varphi \rrbracket \rho \stackrel{\text{def}}{=} \{(s,u) \mid \exists\, s \xrightarrow{a} s' \textit{ and } (s',u) \in \llbracket \varphi \rrbracket \rho\}$$
$$\llbracket [a]\, \varphi \rrbracket \rho \stackrel{\text{def}}{=} \{(s,u) \mid \forall\, s \xrightarrow{a} s'\ , (s',u) \in \llbracket \varphi \rrbracket \rho\}$$

Semantics of L_ν

Given A a TA, ϕ an L_ν formula, ρ an assignment for identifiers (Z)
Interpretation of ϕ in context ρ is a set of extended states (s, w) with:

- $s = (\ell, v)$ a state of A and w a valuation of the formula clocks
- ρ assigns to each identifier a set of extended states

$$\begin{aligned}
[\![\langle\delta\rangle\ \varphi]\!]\,\rho &\stackrel{\text{def}}{=} \{(s,u) \mid \exists s \xrightarrow{d} s' \textit{ and } (s', u+d) \in [\![\varphi]\!]\,\rho\} \\
[\![[\delta]\ \varphi]\!]\,\rho &\stackrel{\text{def}}{=} \{(s,u) \mid \forall s \xrightarrow{d} s',\ (s', u+d) \in [\![\varphi]\!]\,\rho\} \\
[\![x\ \underline{\text{in}}\ \varphi]\!]\,\rho &\stackrel{\text{def}}{=} \{(s,u) \mid (s, u[x \leftarrow 0]) \in [\![\varphi]\!]\,\rho\} \\
[\![X]\!]\,\rho &\stackrel{\text{def}}{=} \rho(X) \\
[\![X =_\nu \varphi]\!]\,\rho &\stackrel{\text{def}}{=} \bigcup\{S \mid S \subseteq [\![\varphi]\!]\,(\rho[X \mapsto S])\}
\end{aligned}$$

Semantics of L_ν

Given A a TA, ϕ an L_ν formula, ρ an assignment for identifiers (Z)
Interpretation of ϕ in context ρ is a set of extended states (s, w) with:

- $s = (\ell, v)$ a state of A and w a valuation of the formula clocks
- ρ assigns to each identifier a set of extended states

$$
\begin{aligned}
[\![\langle\delta\rangle\ \varphi]\!]\,\rho &\stackrel{\text{def}}{=} \{(s,u) \mid \exists s \xrightarrow{d} s' \textit{ and } (s', u+d) \in [\![\varphi]\!]\,\rho\} \\
[\![[\delta]\ \varphi]\!]\,\rho &\stackrel{\text{def}}{=} \{(s,u) \mid \forall s \xrightarrow{d} s',\ (s', u+d) \in [\![\varphi]\!]\,\rho\} \\
[\![x\ \underline{\text{in}}\ \varphi]\!]\,\rho &\stackrel{\text{def}}{=} \{(s,u) \mid (s, u[x \leftarrow 0]) \in [\![\varphi]\!]\,\rho\} \\
[\![X]\!]\,\rho &\stackrel{\text{def}}{=} \rho(X) \\
[\![X =_\nu \varphi]\!]\,\rho &\stackrel{\text{def}}{=} \bigcup\{S \mid S \subseteq [\![\varphi]\!]\,(\rho[X \mapsto S])\}
\end{aligned}
$$

- For closed formula, $[\![\phi]\!]$ does not depend on ρ
- $A \models \phi \iff ((\ell_0, 0), 0) \in [\![\phi]\!]$

Results for L_ν

[Laroussinie et al., 95a, Laroussinie et al.,95b]

Results for L_ν

[Laroussinie et al., 95a, Laroussinie et al.,95b]

- Model Checking over TA is EXPTIME-Complete

Results for L_ν

[Laroussinie et al., 95a, Laroussinie et al.,95b]

- Model Checking over TA is EXPTIME-Complete
- L_ν is compositional for TA: if $\phi \in L_\nu$ then

$$(A \parallel B) \models \phi \iff A \models \phi/B$$

with the quotient formula $\phi/B \in L_\nu$

Results for L_ν

[Laroussinie et al., 95a, Laroussinie et al.,95b]

- Model Checking over TA is EXPTIME-Complete
- L_ν is compositional for TA: if $\phi \in L_\nu$ then

$$(A \parallel B) \models \phi \iff A \models \phi/B$$

  with the quotient formula $\phi/B \in L_\nu$
- L_ν allows to express timed bisimilarity via characteristic formula

Results for L_ν

[Laroussinie et al., 95a, Laroussinie et al.,95b]

- Model Checking over TA is EXPTIME-Complete
- L_ν is compositional for TA: if $\phi \in L_\nu$ then

$$(A \parallel B) \models \phi \iff A \models \phi/B$$

  with the quotient formula $\phi/B \in L_\nu$
- L_ν allows to express timed bisimilarity via characteristic formula
- Model Checker for L_ν: CMC [Laroussinie et al., 98] compute quotient formula ϕ and check nil $\models \phi$

Results for L_ν

[Laroussinie et al., 95a, Laroussinie et al.,95b]

- Model Checking over TA is EXPTIME-Complete
- L_ν is compositional for TA: if $\phi \in L_\nu$ then

$$(A \parallel B) \models \phi \iff A \models \phi/B$$

  with the quotient formula $\phi/B \in L_\nu$
- L_ν allows to express timed bisimilarity
  via characteristic formula
- Model Checker for L_ν: CMC [Laroussinie et al., 98]
  compute quotient formula ϕ and check nil $\models \phi$

Open Problem for L_ν

Satisfiability for Timed Automata

Sampling Control with L_ν

$G(\Delta) = G$ with all controllable actions separated by $k \cdot \Delta$ t.u., $k \in \mathbb{N}$

Sampling Control with L_ν

$G(\Delta) = G$ with all controllable actions separated by $k \cdot \Delta$ t.u., $k \in \mathbb{N}$

Sampling Control Problem (SCP)

Input: G a TGA, ϕ an L_ν objective, $\Delta \in \mathbb{Q}_{\geq 0}$ a sampling rate
SCP: "Is there a controller f s.t. $G(\Delta) \parallel f \models \phi$?"

Sampling Control with L_ν

$G(\Delta) = G$ with all controllable actions separated by $k \cdot \Delta$ t.u., $k \in \mathbb{N}$

Sampling Control Problem (SCP)

Input: G a TGA, ϕ an L_ν objective, $\Delta \in \mathbb{Q}_{\geq 0}$ a sampling rate
SCP: "Is there a controller f s.t. $G(\Delta) \parallel f \models \phi$?"

Model for $G(\Delta)$

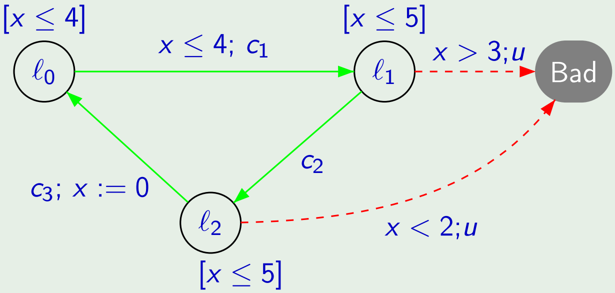
Sampling Control with L_ν

$G(\Delta) = G$ with all controllable actions separated by $k \cdot \Delta$ t.u., $k \in \mathbb{N}$

Sampling Control Problem (SCP)

Input: G a TGA, ϕ an L_ν objective, $\Delta \in \mathbb{Q}_{\geq 0}$ a sampling rate
SCP: "Is there a controller f s.t. $G(\Delta) \parallel f \models \phi$?"

Model for $G(\Delta)$



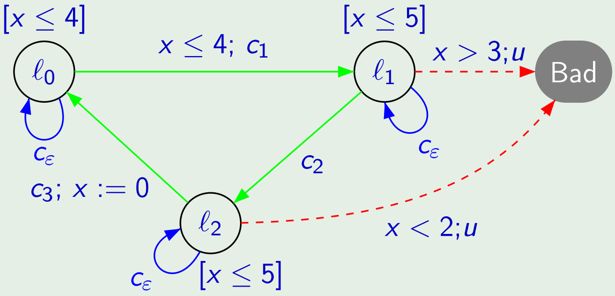
Sampling Control with L_ν

$G(\Delta) = G$ with all controllable actions separated by $k \cdot \Delta$ t.u., $k \in \mathbb{N}$

Sampling Control Problem (SCP)

Input: G a TGA, ϕ an L_ν objective, $\Delta \in \mathbb{Q}_{\geq 0}$ a sampling rate
SCP: "Is there a controller f s.t. $G(\Delta) \parallel f \models \phi$?"

Model for $G(\Delta)$



Sampling Control with L_ν

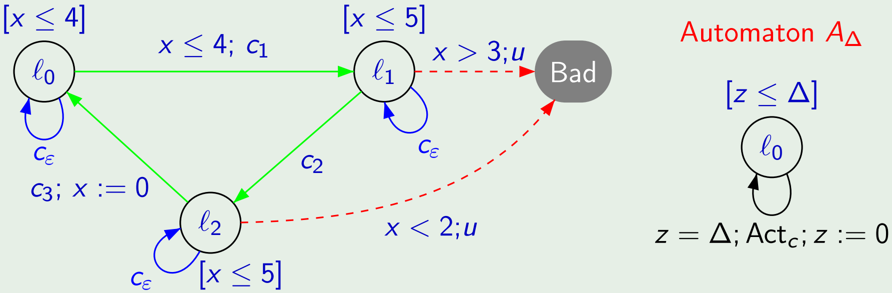
$G(\Delta) = G$ with all controllable actions separated by $k \cdot \Delta$ t.u., $k \in \mathbb{N}$

Sampling Control Problem (SCP)

Input: G a TGA, ϕ an L_ν objective, $\Delta \in \mathbb{Q}_{\geq 0}$ a sampling rate

SCP: "Is there a controller f s.t. $G(\Delta) \parallel f \models \phi$?"

Model for $G(\Delta)$



Reduction of Sampling Control Problem to a Model Checking Problem

Avoid Bad: $Z =_\nu \overline{\text{Bad}} \wedge [\Sigma]\, Z \wedge [\delta]\, Z$

"Is there a controller f s.t. $(G(\Delta) \parallel f) \models \phi$?"
with $\phi = Z$ and $Z =_\nu \overline{\text{Bad}} \wedge [\Sigma]\, Z \wedge [\delta]\, Z$

Reduction of Sampling Control Problem to a Model Checking Problem

Avoid Bad: $Z =_\nu \overline{\mathsf{Bad}} \wedge [\Sigma]\, Z \wedge [\delta]\, Z$

"Is there a controller f s.t. $(G(\Delta) \parallel f) \models \phi$?"
with $\phi = Z$ and $Z =_\nu \overline{\mathsf{Bad}} \wedge [\Sigma]\, Z \wedge [\delta]\, Z$
amounts to checking $G(\Delta) \models \overline{\phi}$ with $\overline{\phi} = Y$ and

$$Y =_\nu \overline{\mathsf{Bad}} \wedge [\mathsf{Act}_u]\, Y \wedge [\delta]\, Y \wedge ([\mathsf{Act}_c]\, \mathit{ff} \vee \langle \mathsf{Act}_c \rangle\, Y)$$

Reduction of Sampling Control Problem to a Model Checking Problem

Avoid Bad: $Z =_\nu \overline{\mathsf{Bad}} \wedge [\Sigma]\, Z \wedge [\delta]\, Z$

"Is there a controller f s.t. $(G(\Delta) \parallel f) \models \phi$?"
with $\phi = Z$ and $Z =_\nu \overline{\mathsf{Bad}} \wedge [\Sigma]\, Z \wedge [\delta]\, Z$
amounts to checking $G(\Delta) \models \overline{\phi}$ with $\overline{\phi} = Y$ and

$$Y =_\nu \overline{\mathsf{Bad}} \wedge [\mathsf{Act}_u]\, Y \wedge [\delta]\, Y \wedge ([\mathsf{Act}_c]\, \mathit{ff} \vee \langle \mathsf{Act}_c \rangle\, Y)$$

Theorem

Given G a TGA, ϕ a control objective in $L_\nu^{det} \subseteq L_\nu$, $\Delta \in \mathbb{Q}_{\geq 0}$

$$\exists f \text{ s.t. } G(\Delta) \parallel f \models \phi \iff G(\Delta) \models \overline{\phi} \iff G \parallel A_\Delta \models \overline{\phi}$$

- $\overline{\phi}$ *can be built automatically (syntactic translation of ϕ),*
- $\overline{\phi}$ *is in L_ν.*

Δ-Dense-Time Control

$G([\Delta, +\infty[)$ all controllable actions separated by at least Δ t.u.

Δ-Dense-Time Control

$G([\Delta, +\infty[)$ all controllable actions separated by at least Δ t.u.

Δ-Dense-Time Control Problem

Input: G a TGA, ϕ an L_ν objective, $\Delta \in \mathbb{Q}_{\geq 0}$ a minimum delay
Δ-CP: "Is there a controller f s.t. $G([\Delta, +\infty[) \parallel f \models \phi$?"

Δ-Dense-Time Control

$G([\Delta, +\infty[)$ all controllable actions separated by at least Δ t.u.

Δ-Dense-Time Control Problem

Input: G a TGA, ϕ an L_ν objective, $\Delta \in \mathbb{Q}_{\geq 0}$ a minimum delay
Δ-CP: "Is there a controller f s.t. $G([\Delta, +\infty[) \parallel f \models \phi$?"

Model for $G([\Delta, +\infty[)$

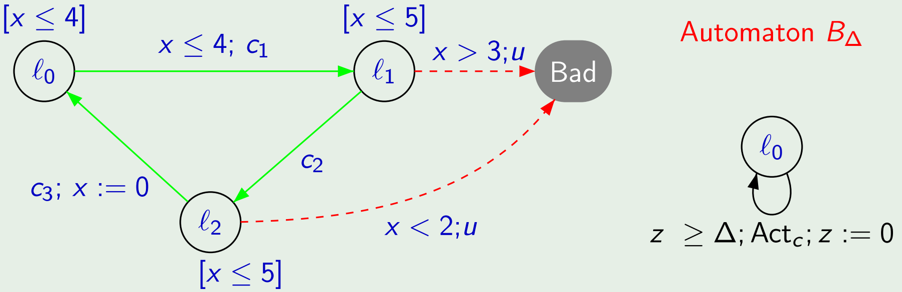
Δ-Dense-Time Control

$G([\Delta, +\infty[)$ all controllable actions separated by at least Δ t.u.

Δ-Dense-Time Control Problem

Input: G a TGA, ϕ an L_ν objective, $\Delta \in \mathbb{Q}_{\geq 0}$ a minimum delay
Δ-CP: "Is there a controller f s.t. $G([\Delta, +\infty[) \parallel f \models \phi$?"

Model for $G([\Delta, +\infty[)$



Dense-Time Control with L_ν

Reduction of Δ-Control Problem to Model-Checking

Aim: Given ϕ in L_ν, prove the following reduction:

$$\exists f \text{ s.t. } G([\Delta, +\infty[) \parallel f \models \phi \iff G([\Delta, +\infty[) \models \overline{\phi}$$

with $\overline{\phi}$ built syntactically.

Dense-Time Control with L_ν

Reduction of Δ-Control Problem to Model-Checking

Aim: Given ϕ in L_ν, prove the following reduction:

$$\exists f \; s.t. \; G([\Delta, +\infty[) \parallel f \models \phi \iff G([\Delta, +\infty[) \models \overline{\phi}$$

with $\overline{\phi}$ built syntactically.

- L_ν is not expressive enough for $\overline{\phi}$
  Objective ϕ given by $Z =_\nu \overline{\text{Bad}} \wedge [u]Z \wedge [c]Z \wedge [\delta]\, Z$

Dense-Time Control with L_ν

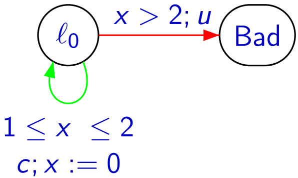
Reduction of Δ-Control Problem to Model-Checking

Aim: Given ϕ in L_ν, prove the following reduction:

$$\exists f \text{ s.t. } G([\Delta, +\infty[) \parallel f \models \phi \iff G([\Delta, +\infty[) \models \overline{\phi}$$

with $\overline{\phi}$ built syntactically.

- L_ν is not expressive enough for $\overline{\phi}$
  Objective ϕ given by $Z =_\nu \overline{\text{Bad}} \wedge [u]Z \wedge [c]Z \wedge [\delta]\, Z$



Intuition: $\overline{\phi}$ will contain $[\delta]$
$[\delta] \implies$ "after all delays"
$(\ell_0, x = 0)$ will not sat. $[\delta]\,[u]\,\overline{\text{Bad}}$

Dense-Time Control with L_ν

Reduction of Δ-Control Problem to Model-Checking

Aim: Given ϕ in L_ν, prove the following reduction:

$$\exists f \ s.t. \ G([\Delta, +\infty[) \parallel f \models \phi \iff G([\Delta, +\infty[) \models \overline{\phi}$$

with $\overline{\phi}$ built syntactically.

- L_ν is not expressive enough for $\overline{\phi}$
  Need some until operator: $[\delta\rangle$

Dense-Time Control with L_ν

Reduction of Δ-Control Problem to Model-Checking

Aim: Given ϕ in L_ν, prove the following reduction:

$$\exists f \ s.t. \ G([\Delta, +\infty[) \parallel f \models \phi \iff G([\Delta, +\infty[) \models \overline{\phi}$$

with $\overline{\phi}$ built syntactically.

- L_ν is not expressive enough for $\overline{\phi}$
  Need some until operator: $[\delta\rangle$
- We need to restrict the set of control objectives (ϕ)

Dense-Time Control with L_ν

Reduction of Δ-Control Problem to Model-Checking

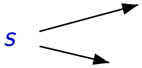
Aim: Given ϕ in L_ν, prove the following reduction:

$$\exists f \ s.t. \ G([\Delta, +\infty[) \parallel f \models \phi \iff G([\Delta, +\infty[) \models \overline{\phi}$$

with $\overline{\phi}$ built syntactically.

- L_ν is not expressive enough for $\overline{\phi}$
  Need some until operator: $[\delta\rangle$
- We need to restrict the set of control objectives (ϕ)

$\exists f_1 \ s.t. \ s \models \phi_1$



$\exists f_2 \ s.t. \ s \models \langle c \rangle \, \phi_2$

Objective: $\phi_1 \wedge \langle c \rangle \, \phi_2$
Build a strategy f from f_1 and f_2 to ensure $\phi_1 \wedge \langle c \rangle \, \phi_2$
$f_2(s) = c$, but $f_1(s)$?

Dense-Time Control with L_ν

Reduction of Δ-Control Problem to Model-Checking

Aim: Given ϕ in L_ν, prove the following reduction:

$$\exists f \text{ s.t. } G([\Delta, +\infty[) \parallel f \models \phi \iff G([\Delta, +\infty[) \models \overline{\phi}$$

with $\overline{\phi}$ built syntactically.

- L_ν is not expressive enough for $\overline{\phi}$
  Need some until operator: $[\delta\rangle$

- We need to restrict the set of control objectives (ϕ)
  Define a sublogic $L_\nu^{det} \subset L_\nu$ s.t. strategies can be merged

The logics L_ν^c and L_ν^{det}

- Extension of L_ν for Timed Control

The logics L_ν^c and L_ν^{det}

- Extension of L_ν for Timed Control
  - $L_\nu^c = L_\nu$ + new modality $[\delta\rangle$

The logics L_ν^c and L_ν^{det}

- Extension of L_ν for Timed Control
  - $L_\nu^c = L_\nu$ + new modality $[\delta\rangle$
  - $\varphi\ [\delta\rangle\ \psi \ \sim\ \varphi$ *Weak Until* ψ

The logics L_ν^c and L_ν^{det}

- Extension of L_ν for Timed Control
  - $L_\nu^c = L_\nu$ + new modality $[\delta\rangle$
  - $\varphi\ [\delta\rangle\ \psi \quad \sim \quad \varphi$ *Weak Until* ψ

Semantics of $\varphi\,[\delta\rangle\,\psi$

$(s, u) \models \varphi\,[\delta\rangle\,\psi \quad \Longleftrightarrow$

- either $\forall t \in \mathbb{R}_{\geq 0},\ s \xrightarrow{t} s' \Longrightarrow (s', u+t) \models \varphi$
- or $\exists t \in \mathbb{R}_{\geq 0}$ s.t. $s \xrightarrow{t} s'$ and $(s', v+t) \models \psi$ *and* $\forall 0 \leq t' < t,\ s \xrightarrow{t'} s''$ we have $(s'', v+t') \models \varphi$

Allows to express prevention of time-elapsing

The logics L_ν^c and L_ν^{det}

- Extension of L_ν for Timed Control
  - $L_\nu^c = L_\nu +$ new modality $[\delta\rangle$
  - $\varphi\ [\delta\rangle\ \psi \ \sim\ \varphi$ *Weak Until* ψ

  Allows to express prevention of time-elapsing
- Restriction of control objectives to L_ν^{det}

▸ Syntax

The logics L_ν^c and L_ν^{det}

- Extension of L_ν for Timed Control
  - $L_\nu^c = L_\nu$ + new modality $[\delta\rangle$
  - $\varphi\ [\delta\rangle\ \psi \ \sim\ \varphi$ *Weak Until* ψ

  Allows to express prevention of time-elapsing
- Restriction of control objectives to L_ν^{det} ▸ Syntax
  - rule out conjunctions of the type $(\langle c\rangle\ \psi) \wedge \phi$ for arbitrary ϕ

The logics L_ν^c and L_ν^{det}

- Extension of L_ν for Timed Control
  - $L_\nu^c = L_\nu$ + new modality $[\delta\rangle$
  - $\varphi\ [\delta\rangle\ \psi \ \sim\ \varphi$ *Weak Until* ψ

  Allows to express prevention of time-elapsing
- Restriction of control objectives to L_ν^{det} ▸ Syntax
  - rule out conjunctions of the type $(\langle c\rangle\ \psi) \wedge \phi$ for arbitrary ϕ
  - Allow only conjunctions like $\langle c_1\rangle\ \phi_1 \wedge \langle c_2\rangle\ \phi_2$

  Allows to merge strategies

The logics L_ν^c and L_ν^{det}

- Extension of L_ν for Timed Control
  - $L_\nu^c = L_\nu$ + new modality $[\delta\rangle$
  - $\varphi \ [\delta\rangle \ \psi \ \sim \ \varphi$ *Weak Until* ψ

  Allows to express prevention of time-elapsing
- Restriction of control objectives to L_ν^{det} ▸ Syntax
  - rule out conjunctions of the type $(\langle c\rangle\ \psi) \wedge \phi$ for arbitrary ϕ
  - Allow only conjunctions like $\langle c_1\rangle\ \phi_1 \wedge \langle c_2\rangle\ \phi_2$

  Allows to merge strategies

Theorem

Given G a TGA, ϕ a control objective in $L_\nu^{det} \subseteq L_\nu$, $\Delta \in \mathbb{Q}_{\geq 0}$

$$\exists f \text{ s.t. } G([\Delta, +\infty[) \parallel f \models \phi \iff G([\Delta, +\infty[) \models \overline{\phi} \iff G \parallel B_\Delta \models \overline{\phi}$$

- $\overline{\phi}$ *can be built automatically (syntactic translation of ϕ),*
- $\overline{\phi}$ *is in L_ν^c.*

How to build the control formula $\overline{\phi}$?

$\overline{\varphi} = \bigvee_{\sigma \in \mathrm{Act}_c \cup \{\lambda\}} \overline{\varphi}^{\,\sigma}$

$\overline{\varphi}^{\,\sigma}$ holds in s if there is a strategy prescribing σ in s which can enforce φ.

- $\overline{\bigwedge_{\alpha \in A} \alpha}^{\,\sigma} \stackrel{\text{def}}{=} \bigwedge_{\alpha \in A} \overline{\alpha}^{\,\sigma}$
- $\overline{\bigvee_{\alpha \in A} \alpha}^{\,\sigma} \stackrel{\text{def}}{=} \bigvee_{\alpha \in A} \overline{\alpha}^{\,\sigma}$
- $\overline{\langle a \rangle\, \varphi}^{\,\sigma} \stackrel{\text{def}}{=} \begin{cases} \mathit{ff} & \text{if } \sigma, a \in \mathrm{Act}_c \wedge \sigma \neq a \\ \langle a \rangle\, \overline{\varphi} \wedge \langle \sigma \rangle\, \mathit{tt} & \text{if } a \in \mathrm{Act}_u \\ \langle a \rangle\, \overline{\varphi} & \text{otherwise} \end{cases}$
- $\overline{\langle \delta \rangle\, \varphi}^{\,\sigma} \stackrel{\text{def}}{=} \begin{cases} \langle \delta \rangle\, \overline{\varphi} & \text{if } \sigma = \lambda \\ \overline{\varphi}^{\,\sigma} & \text{if } \sigma \in \mathrm{Act}_c \end{cases}$

How to build the control formula $\overline{\phi}$? (cont.)

- $\overline{[a_c]\,\varphi}^{\,\sigma} \stackrel{\text{def}}{=} \begin{cases} \langle\sigma\rangle\, tt & \text{if } a_c \neq \sigma \\ \langle a_c\rangle\, \overline{\varphi} & \text{if } a_c = \sigma \end{cases}$
- $\overline{[a_u]\,\varphi}^{\,\sigma} \stackrel{\text{def}}{=} [a_u]\,\overline{\varphi} \wedge \langle\sigma\rangle\, tt$
- $\overline{[\delta]\,\varphi}^{\,\sigma} \stackrel{\text{def}}{=} \begin{cases} \overline{\varphi}^{\,\sigma} & \text{if } \sigma \in \mathrm{Act}_c \\ \overline{\varphi}^{\,\lambda}\, [\delta\rangle \left(\bigvee_{a_c \in \mathrm{Act}_c} \overline{\varphi}^{\,a_c} \right) & \text{otherwise} \end{cases}$
- $\overline{x \sim c}^{\,\sigma} \stackrel{\text{def}}{=} x \sim c \wedge \langle\sigma\rangle\, tt$
- $\overline{r \ \underline{\text{in}}\ \varphi}^{\,\sigma} \stackrel{\text{def}}{=} r \ \underline{\text{in}}\ \overline{\varphi}^{\,\sigma}$
- $\overline{X}^{\,\sigma} \stackrel{\text{def}}{=} X_\sigma \wedge \langle\sigma\rangle\, tt$

Outline

- Control of Timed Systems
- Controllability with L_ν

Properties of the new operator $[\delta\rangle$

Expressivity

The logic L_ν^c is strictly more expressive than L_ν over timed automata.

Properties of the new operator $[\delta\rangle$

Expressivity

The logic L_ν^c is strictly more expressive than L_ν over timed automata.

$([a]\,\mathit{ff})\ [\delta\rangle\ (\langle b\rangle\ \mathit{tt})$ cannot be expressed with L_ν

Properties of the new operator $[\delta\rangle$

Expressivity

The logic L_ν^c is strictly more expressive than L_ν over timed automata.

$([a]\,ff)\ [\delta\rangle\ (\langle b\rangle\, tt)$ cannot be expressed with L_ν

Model checking

▸ Computation

The model-checking of L_ν^c over timed automata is EXPTIME-complete.

Properties of the new operator $[\delta\rangle$

Expressivity

The logic L_ν^c is strictly more expressive than L_ν over timed automata.

$([a]\ \mathit{ff})\ [\delta\rangle\ (\langle b\rangle\ \mathit{tt})$ cannot be expressed with L_ν

Model checking

▸ Computation

The model-checking of L_ν^c over timed automata is EXPTIME-complete.

Compositionality

▸ Quotient

The logic L_ν^c is compositional for the class of timed automata.

Properties of the new operator $[\delta\rangle$

Expressivity

The logic L_ν^c is strictly more expressive than L_ν over timed automata.

$([a]\,\mathit{ff})\ [\delta\rangle\ (\langle b\rangle\,\mathit{tt})$ cannot be expressed with L_ν

Model checking

▸ Computation

The model-checking of L_ν^c over timed automata is EXPTIME-complete.

Compositionality

▸ Quotient

The logic L_ν^c is compositional for the class of timed automata.

$$(A_1 \parallel A_2) \models \varphi \iff A_1 \models \varphi / A_2$$

Outline

- Control of Timed Systems
- Controllability with L_ν

Conclusion & Further Work

- Results
  - Control Objectives in L_ν^{det}
  - Reduction of Control Problem for (TA, L_ν^{det}) to a Model Checking Problem for (TA, L_ν^c)

$$\exists f \ s.t. \ (G \parallel f) \models \phi \iff G \models \overline{\phi}$$

  - Properties of the new logic L_ν^c
    - Strictly more expressive than L_ν
    - Model-Checking is EXPTIME-Complete
    - L_ν^c is compositional for TA
  - Implementation: The tool CMC [Laroussinie et al., 98]

Conclusion & Further Work

- Results
  - Control Objectives in L_ν^{det}
  - Reduction of Control Problem for (TA, L_ν^{det}) to a Model Checking Problem for (TA, L_ν^c)

$$\exists f \text{ s.t. } (G \parallel f) \models \phi \iff G \models \overline{\phi}$$

  - Properties of the new logic L_ν^c
    - Strictly more expressive than L_ν
    - Model-Checking is EXPTIME-Complete
    - L_ν^c is compositional for TA
  - Implementation: The tool CMC [Laroussinie et al., 98]
- Further Work
  - Extend L_ν^{det}
  - Synthesize Controllers
  - Extend to Partial Observation
  - Use More general notion of strategies

Related Work

- Discrete Time Case
  - ATL [Alur et al., 02]
  - Reduction of CP to MC Problem with μ-calculus:
    loop μ-calculus [Arnold et al., 03]
    Quantified μ-calculus [Riedweg et al., 03]
- Timed Case
  External specifications = TA [D'Souza et al., 02]
  TCTL control objective [Faella et al., 02]

References

Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman.
Alternating-time temporal logic.
Journal of the ACM, 49:672–713, 2002.

Luca De Alfaro, Thomas A. Henzinger, and Rupak Majumdar.
Symbolic algorithms for infinite-state games.
In *Proc. 12th International Conference on Concurrency Theory (CONCUR'01)*, volume 2154 of *LNCS*, pages 536–550. Springer, 2001.

Eugene Asarin, Oded Maler, Amir Pnueli, and Joseph Sifakis.
Controller synthesis for timed automata.
In *Proc. IFAC Symposium on System Structure and Control*, pages 469–474. Elsevier Science, 1998.

André Arnold, Aymeric Vincent, and Igor Walukiewicz.
Games for synthesis of controllers with partial observation.
Theoretical Computer Science, 303(1):7–34,2003.

References (cont.)

- Luca Aceto, Patricia Bouyer, Augusto Burgueño, and Kim G. Larsen.
  The power of reachability testing for timed automata.
  Theoretical Computer Science (TCS), 300(1–3):411–475, 2003.

- Patricia Bouyer, Franck Cassez, and François Laroussinie.
  Modal logics for timed control.
  Research Report LSV-05-04, Laboratoire Spécification & Vérification, ENS de Cachan, France, 2005.

- Deepak D'Souza and P. Madhusudan.
  Timed control synthesis for external specifications.
  In *Proc. 19th Int. Symp. Theoretical Aspects of Computer Science (STACS'2002)*, volume 2285 of LNCS, pages 571–582, Springer, 2002.

References (cont.)

Marco Faella, Salvatore La Torre, and Aniello Murano.
Dense real-time games.
In *Proc. 17th IEEE Symposium on Logic in Computer Science (LICS'02)*, pages 167–176. IEEE Computer Society Press, 2002.

François Laroussinie and Kim G. Larsen.
Compositional model-checking of real-time systems.
In *Proc. 6th International Conference on Concurrency Theory (CONCUR'95)*, volume 962 of *Lecture Notes in Computer Science*, pages 27–41. Springer, 1995.

François Laroussinie and Kim G. Larsen.
CMC: A tool for compositional model-checking of real-time systems.
In *Proc. IFIP Joint International Conference on Formal Description Techniques & Protocol Specification, Testing, and Verification (FORTE-PSTV'98)*, pages 439–456. Kluwer Academic, 1998.

References (cont.)

François Laroussinie, Kim G. Larsen, and Carsten Weise.
From timed automata to logic – and back.
In *Proc. 20th International Symposium on Mathematical Foundations of Computer Science (MFCS'95)*, volume 969 of *Lecture Notes in Computer Science*, pages 529–539. Springer, 1995.

Oded Maler, Amir Pnueli, and Joseph Sifakis.
On the synthesis of discrete controllers for timed systems.
In *Proc. 12th Annual Symposium on Theoretical Aspects of Computer Science (STACS'95)*, volume 900, pages 229–242. Springer, 1995.

Stéphane Riedweg and Sophie Pinchinat.
Quantified μ-calculus for control synthesis.
In *Proc. 28th International Symposium on Mathematical Foundations of Computer Science (MFCS'03)*, volume 2747 of *Lecture Notes in Computer Science*, pages 642–651. Springer, 2003.

Timed Automata

A Timed Automaton $\mathcal{A}$ is a tuple $(L, \ell_0, \mathrm{Act}, X, \mathrm{inv}, \longrightarrow)$ where:

- L is a finite set of locations
- ℓ_0 is the initial location
- X is a finite set of clocks
- Act is a finite set of actions
- $\longrightarrow$ is a set of transitions of the form $\ell \xrightarrow{g,a,R} \ell'$ with:
  - $\ell, \ell' \in L$,
  - $a \in \mathrm{Act}$
  - a guard g which is a clock constraint over X
  - a reset set R which is the set of clocks to be reset to 0

Clock constraints are boolean combinations of $x \sim k$ with $x \in C$ and $k \in \mathbb{Z}$ and $\sim \in \{\leq, <\}$.

Back

Semantics of Timed Automata

Let $\mathcal{A} = (L, \ell_0, \mathrm{Act}, X, \mathrm{inv}, \longrightarrow)$ be a Timed Automaton.

A state (ℓ, v) of $\mathcal{A}$ is in $L \times \mathbb{R}_{\geq 0}^X$

The semantics of $\mathcal{A}$ is a Timed Transition System $S_\mathcal{A} = (Q, q_0, \mathrm{Act} \cup \mathbb{R}_{\geq 0}, \longrightarrow)$ with:

- $Q = L \times \mathbb{R}_{\geq 0}^X$
- $q_0 = (\ell_0, \overline{0})$
- $\longrightarrow$ consists in:

(discrete transition): $(\ell, v) \xrightarrow{a} (\ell', v') \iff \begin{cases} \exists\, \ell \xrightarrow{g,a,r} \ell' \in \mathcal{A} \\ v \models g \\ v' = v[r \leftarrow 0] \\ v' \models \mathrm{inv}(\ell') \end{cases}$

(delay transition): $(\ell, v) \xrightarrow{d} (\ell, v + d) \iff d \in \mathbb{R}_{\geq 0} \wedge v + d \models \mathrm{inv}(\ell)$

Back

Syntax of L_ν^{det}

$$L_\nu^{det} \ni \quad \varphi, \psi ::= \quad X \mid \varphi \vee \psi \mid \bigwedge_{\alpha \in A} \alpha \mid Z =_\nu \phi$$

where A denotes a deterministic set of basic terms $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$:

- Basic terms:

$$\alpha ::= \quad tt \mid ff \mid x \bowtie c \mid r \ \underline{\text{in}} \ \langle\sigma\rangle \varphi \mid r \ \underline{\text{in}} \ [\sigma] \varphi$$

  with $\sigma \in \text{Act} \cup \{\lambda\}$

- Deterministic set of basic terms:
  for all $\sigma \in \text{Act} \cup \{\lambda\}$ there is at most one i s.t. $\alpha_i = r \ \underline{\text{in}} \ \langle\sigma\rangle \varphi$ or $\alpha_i = r \ \underline{\text{in}} \ [\sigma] \varphi$.

Back

The following results are taken from [Aceto et al., 03].

Test Automaton

Let T be a timed automaton with a set of rejecting locations N.
T is a test automaton for the property ϕ if for all timed automata B:

$$B \models \phi \iff \text{ReachableStatesOf}(B \parallel T) \cap N = \emptyset$$

A property ϕ can be tested if there is a test automaton T_ϕ for ϕ.

The following results are taken from [Aceto et al., 03].

Test Automaton

Let T be a timed automaton with a set of rejecting locations N.
T is a test automaton for the property ϕ if for all timed automata B:

$$B \models \phi \iff \text{ReachableStatesOf}(B \parallel T) \cap N = \emptyset$$

A property ϕ can be tested if there is a test automaton T_ϕ for ϕ.

The Logic $L_{\forall S}$

$L_{\forall S}$ is a strict subset of L_ν: no $\langle \delta \rangle$, restricted or, restricted $\langle a \rangle$.

The following results are taken from [Aceto et al., 03].

Test Automaton

Let T be a timed automaton with a set of rejecting locations N.
T is a test automaton for the property ϕ if for all timed automata B:

$$B \models \phi \iff \text{ReachableStatesOf}(B \parallel T) \cap N = \emptyset$$

A property ϕ can be tested if there is a test automaton T_ϕ for ϕ.

The Logic $L_{\forall S}$

$L_{\forall S}$ is a strict subset of L_ν: no $\langle \delta \rangle$, restricted or, restricted $\langle a \rangle$.

Test Automata and Logics

A property ϕ can be tested iff ϕ is a formula of $L_{\forall S}$.

The following results are taken from [Aceto et al., 03].

Test Automaton

Let T be a timed automaton with a set of rejecting locations N.
T is a test automaton for the property ϕ if for all timed automata B:

$$B \models \phi \iff \text{ReachableStatesOf}(B \parallel T) \cap N = \emptyset$$

A property ϕ can be tested if there is a test automaton T_ϕ for ϕ.

Test Automata and Logics

A property ϕ can be tested iff ϕ is a formula of $L_{\forall S}$.

L_ν^{det} is more expressive than $L_{\forall S}$

The formula

$$X =_\nu [\delta]\, X \wedge [a]\, X \wedge \langle\delta\rangle\, \langle b\rangle\, tt$$

cannot be expressed in $L_{\forall S}$.

Model-Checking and Compositionality for L_ν^{det}

Computation of $[\![\varphi\ [\delta\rangle\ \psi]\!]$ given $[\![\varphi]\!]$ and $[\![\psi]\!]$:

$$\left(\overleftarrow{[\![\varphi]\!]^c}\right)^c \cup \left[\left(\overleftarrow{\left(\overrightarrow{[\![\psi]\!]} \cup [\![\varphi]\!]\right)^c}\right)^c \cap \left([\![\psi]\!] \cup \left([\![\varphi]\!] \cap \left(\overleftarrow{[\![\varphi]\!]^+ \cap [\![\psi]\!]}\right)\right)\right)\right]$$

Back

Compositionality

$$\left(\varphi_1\ [\delta\rangle\ \varphi_2\right)/\ell \stackrel{\text{def}}{=} \left(\text{inv}(\ell) \Longrightarrow (\varphi_1/\ell)\right)\ [\delta\rangle\ \left(\text{inv}(\ell) \wedge (\varphi_2/\ell)\right)$$

This is what is implemented in CMC.

Back