ULB UNIVERSITÉ LIBRE DE BRUXELLES

Almost ASAP Semantics : From Timed Models to Timed Implementations

M. De Wulf, L. Doyen,J.-F. Raskin

University of Brussels
Centre Fédéré en Vérification

Motivations

- Embedded Controllers
  - … are difficult to develop (concurrency, real-time, continuous environment, …);
  - … are safety critical.

➽Use model–based development : Hybrid Automata and Reachability Analysis

Model-based Development

- Make a model of the environment

  Environment

- Make clear the control objective:

  Bad

- Make a model of your control strategy:

  ControllerMod

- Verify :

  Does Environment || ControllerMod avoid Bad ?

- Good, but after ?

From Correct Models to Correct Implementations

- Should we verify code ?
  - this may be difficult (too much details)
- Can we translate model into code ?

  … there are tools for that …
- … and preserve properties ?

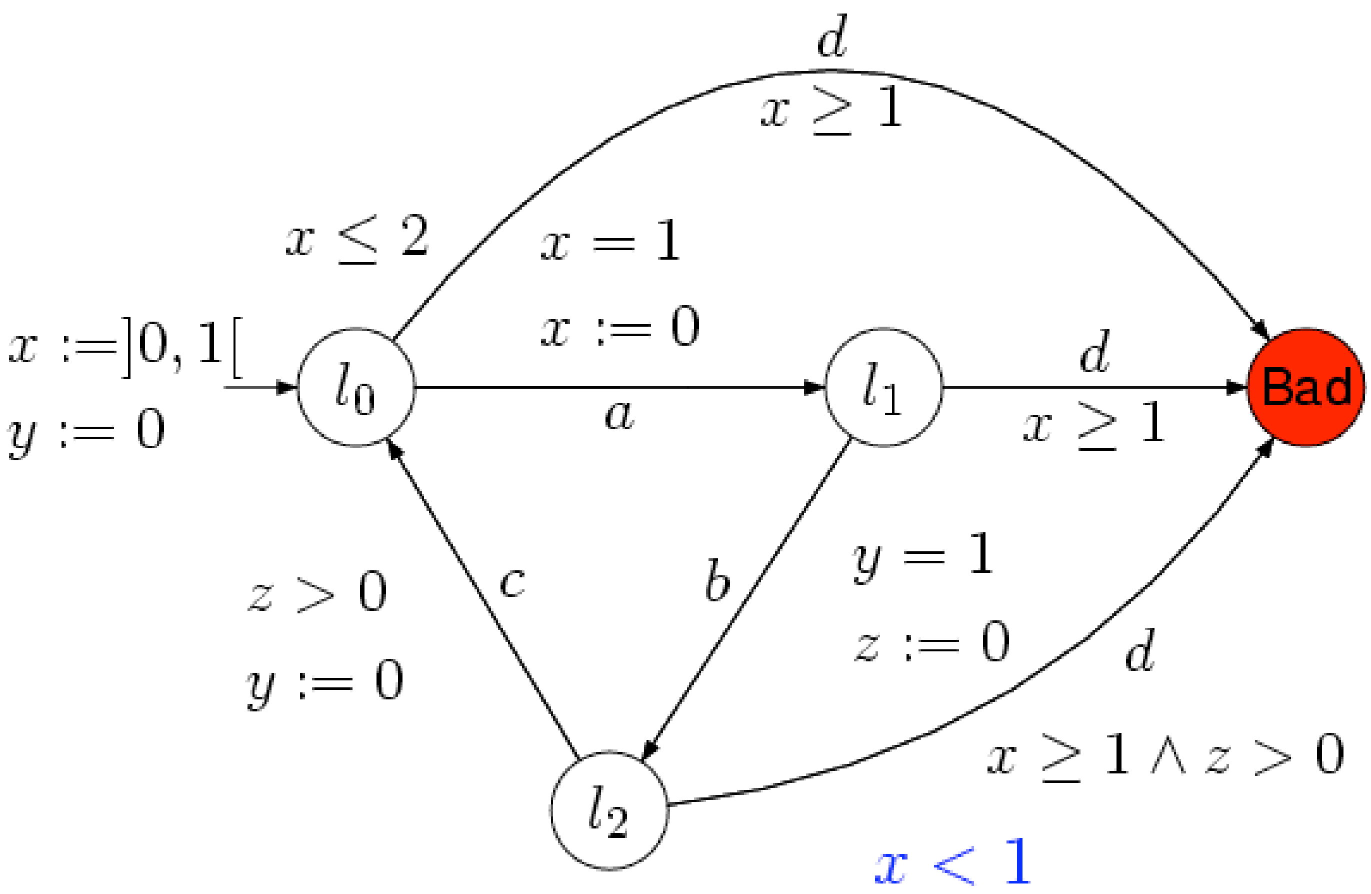
  … good question…

Problem

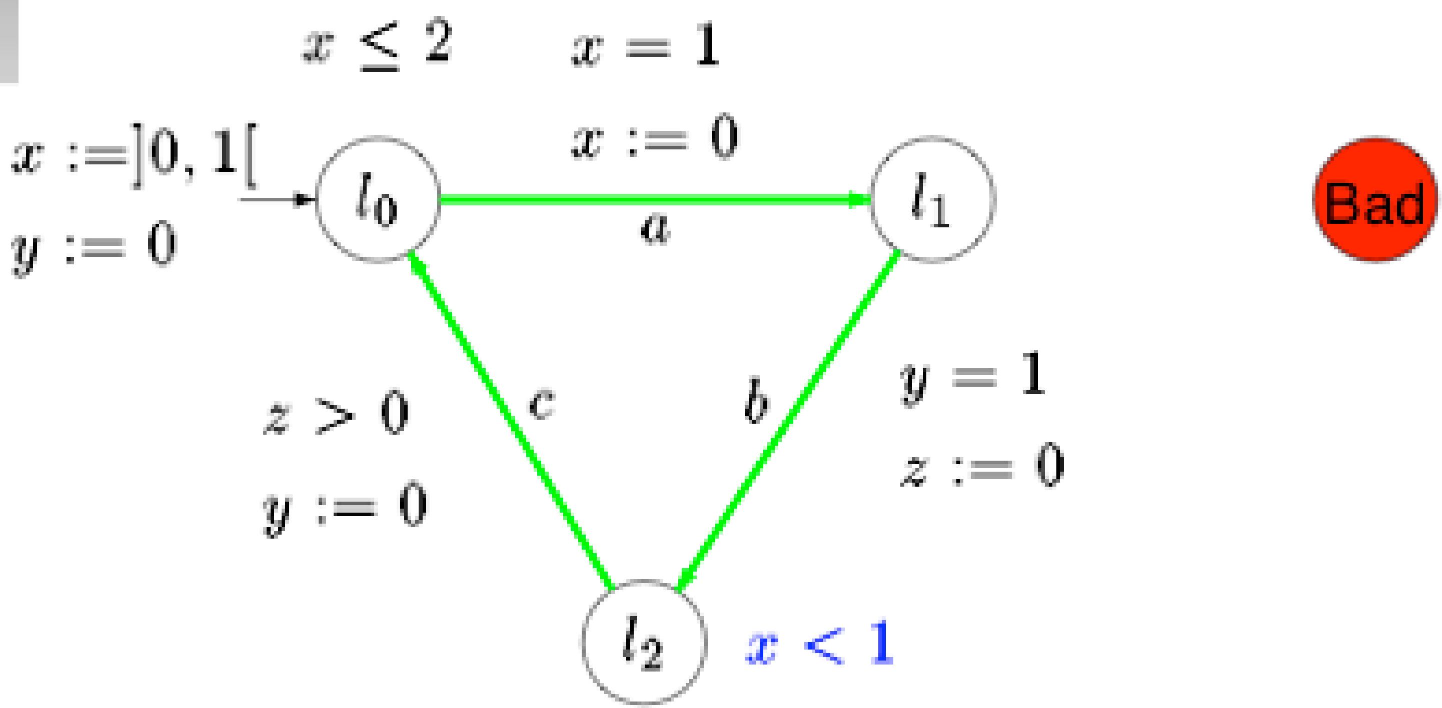
- Timed automata are (in general) not implementable (in a formal sense)…

  Why ?
  - Zenoness : 0, 0.5, 0.75, 0.875, …
  - No minimal bound between two transitions : 0,0.5,1,1.75,2,2.875,3,…
  - And more … (robustness)

No Minimal Bound between Two Transitions



It can be controlled

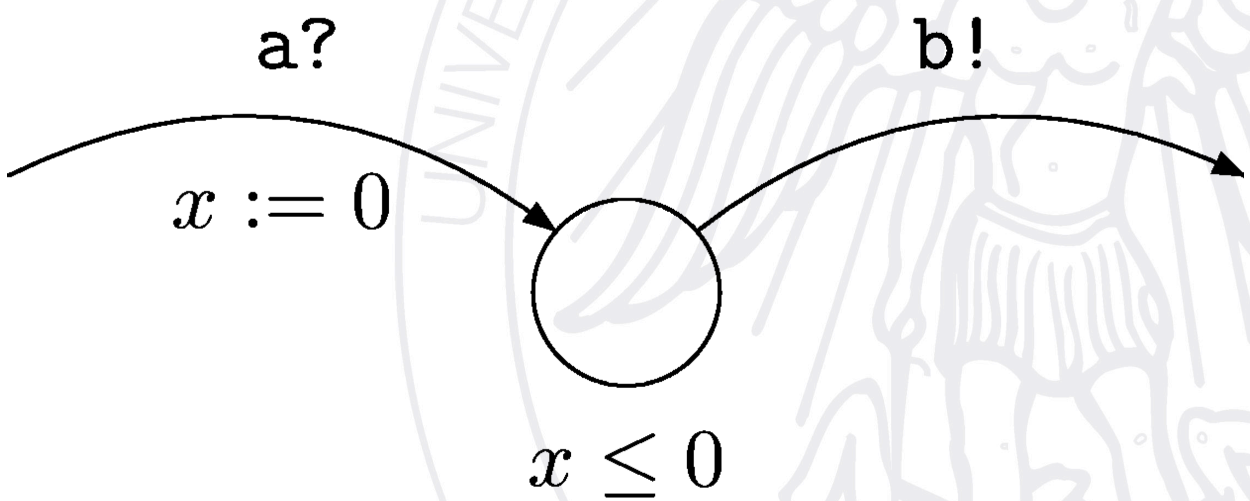


- δ_i : time in l_2 during loop i
- the controller must ensure : $\sum_{i=0}^{i=+\infty} \delta_i < x_0 - y_0$

More...

- One can specify instantaneous responses but not implement them.

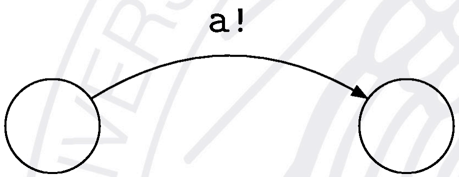
Not implementable



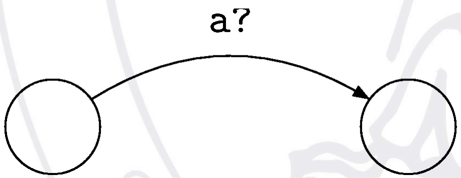
More...

- Instantaneous synchronisations between environment and controller are not implementable.

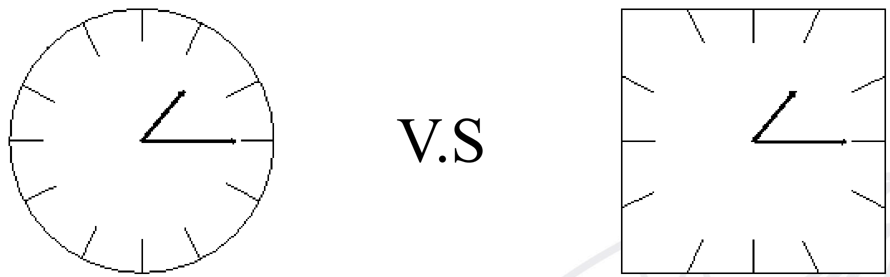
Environment



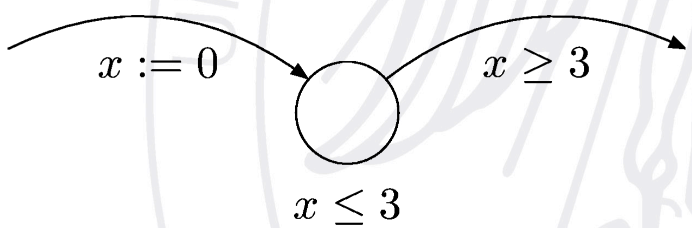
Classical controller
Not implementable



More…



- Models use continuous clocks and implementations use digital clocks with finite precision



Classical controller
Not implementable

Problems : Summary

- My controller stragegy may be correct because of
  - ... it is zeno...
  - ... it acts faster and faster?
  - ... it reacts instanteously to events, timeouts,...? (synchrony hypothesis)
  - ... it uses infinitely precise clocks?

A possible solution...

- Give an alternative semantics to timed automata : Almost ASAP semantics.
  - enabled transitions of the controller become urgent only after Δ time units;
  - events from the environment are received by the controller within Δ time units;
  - truth values of guards are enlarged by **f(Δ)**.

where Δ is a parameter

Definition of the AASAP semantics

Definition 13 [AASAP semantics] Given an ELASTIC controller

$$A = \langle \mathsf{Loc}, l_0, \mathsf{Var}, \mathsf{Lab}, \mathsf{Edg} \rangle$$

and $\Delta \in \mathbb{Q}^{\geq 0}$, the AASAP semantics of A, noted $[\![A]\!]_\Delta^{\mathsf{AAsap}}$ is the STTS

$$\mathcal{T} = \langle S, \iota, \Sigma_{\mathsf{in}}, \Sigma_{\mathsf{out}}, \Sigma_\tau, \rightarrow \rangle$$

where:

(A1) S is the set of tuples (l, v, I, d) where $l \in \mathsf{Loc}$, $v \in [\mathsf{Var} \rightarrow \mathbb{R}^{\geq 0}]$, $I \in [\Sigma_{\mathsf{in}} \rightarrow \mathbb{R}^{\geq 0} \cup \{\bot\}]$ and $d \in \mathbb{R}^{\geq 0}$;

(A2) $\iota = (l_0, v, I, 0)$ where v is such that for any $x \in \mathsf{Var} : v(x) = 0$, and I is such that for any $\sigma \in \Sigma_{\mathsf{in}}$, $I(\sigma) = \bot$;

(A3) $\Sigma_{\mathsf{in}} = \mathsf{Lab_{in}}$, $\Sigma_{\mathsf{out}} = \mathsf{Lab_{out}}$, and $\Sigma_\tau = \mathsf{Lab}_\tau \cup \mathsf{Lab_{in}} \cup \{\epsilon\}$;

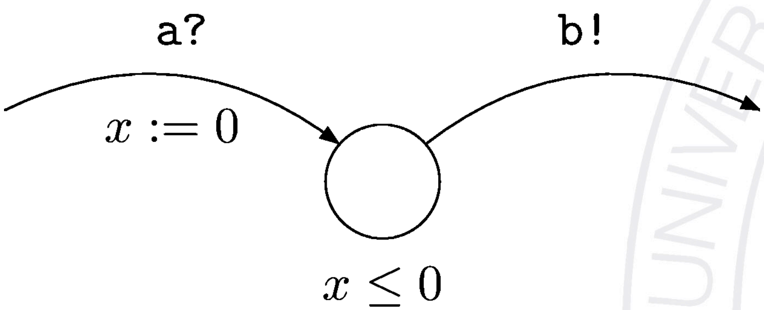
(A4) The transition relation is defined as follows:

- for the discrete transitions, we distinguish five cases:
  - (A4.1) let $\sigma \in \mathsf{Lab_{out}}$. We have $((l, v, I, d), \sigma, (l', v', I, 0)) \in \rightarrow$ iff there exists $(l, l', g, \sigma, R) \in \mathsf{Edg}$ such that $v \models {}_\Delta g_\Delta$ and $v' = v[R := 0]$;
  - (A4.2) let $\sigma \in \mathsf{Lab_{in}}$. We have $((l, v, I, d), \sigma, (l, v, I', d)) \in \rightarrow$ iff $I(\sigma) = \bot$ and $I' = I[\sigma := 0]$;
  - (A4.3) let $\bar{\sigma} \in \mathsf{Lab_{in}}$. We have $((l, v, I, d), \bar{\sigma}, (l', v', I', 0)) \in \rightarrow$ iff there exists $(l, l', g, \sigma, R) \in \mathsf{Edg}$, $v \models {}_\Delta g_\Delta$, $I(\sigma) \neq \bot$, $v' = v[R := 0]$ and $I' = I[\sigma := \bot]$;
  - (A4.4) let $\sigma \in \mathsf{Lab}_\tau$. We have $((l, v, I, d), \sigma, (l', v', I, 0)) \in \rightarrow$ iff there exists $(l, l', g, \sigma, R) \in \mathsf{Edg}$, $v \models {}_\Delta g_\Delta$, and $v' = v[R := 0]$;
  - (A4.5) let $\sigma = \epsilon$. We have for any $(l, v, I, d) \in S : ((l, v, I, d), \epsilon, (l, v, I, d)) \in \rightarrow$.
- for the continuous transitions:
  - (A4.6) for any $t \in \mathbb{R}^{\geq 0}$, we have $((l, v, I, d), t, (l, v + t, I + t, d + t)) \in \rightarrow$ iff the two following conditions are satisfied:
    - for any edge $(l, l', g, \sigma, R) \in \mathsf{Edg}$ with $\sigma \in \mathsf{Lab_{out}} \cup \mathsf{Lab}_\tau$, we have that:
      $$\forall t' : 0 \leq t' \leq t : (d + t' \leq \Delta \vee \mathsf{TS}(v + t', g) \leq \Delta)$$
    - for any edge $(l, l', g, \sigma, R) \in \mathsf{Edg}$ with $\sigma \in \mathsf{Lab_{in}}$, we have that:
      $$\forall t' : 0 \leq t' \leq t : (d + t' \leq \Delta \vee \mathsf{TS}(v + t', g) \leq \Delta \vee (I + t')(\sigma) \leq \Delta)$$

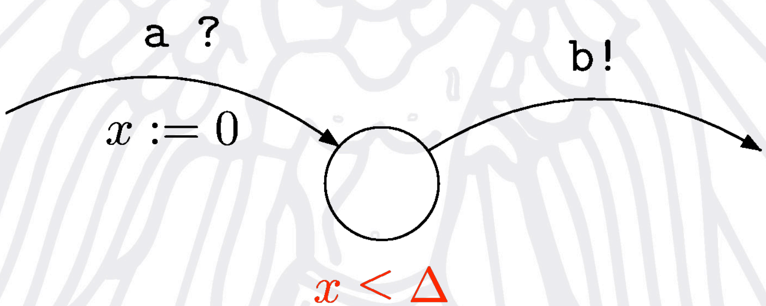
Intuition...

One can specify instantaneous responses but not implement them.

Not implementable



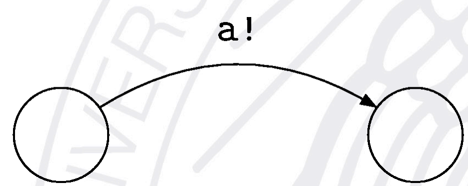
Solution : allow some delay



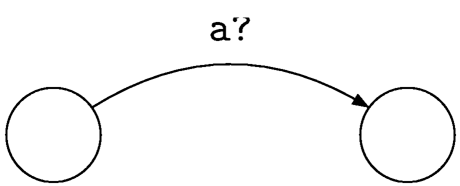
Intuition…

Instantaneous synchronisations between environment and controller are not implementable.

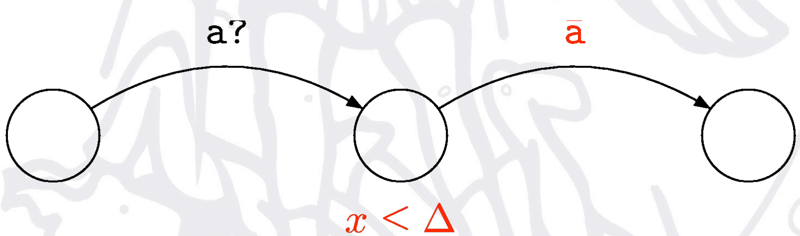
Environment



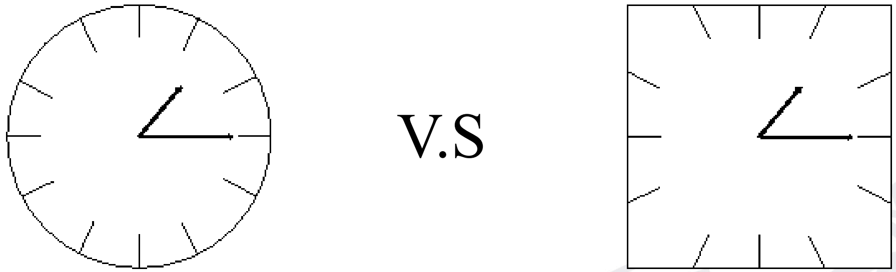
Classical controller
Not implementable



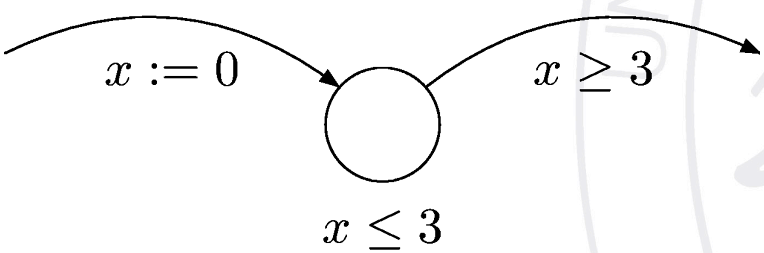
Solution :
Uncouple event from perception by the controller



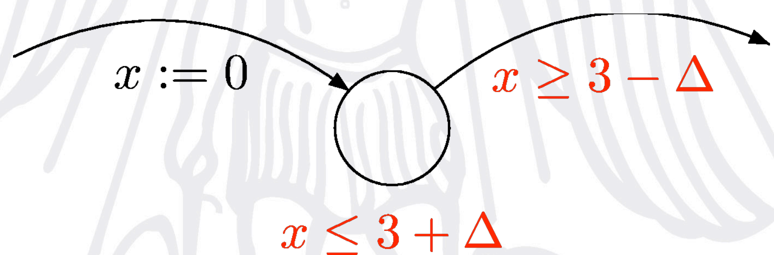
Intuition...



Models use continuous clocks and implementations use digital clocks with finite precision



Classical controller
Not implementable



Solution :
Slightly relax the constraints

Verification

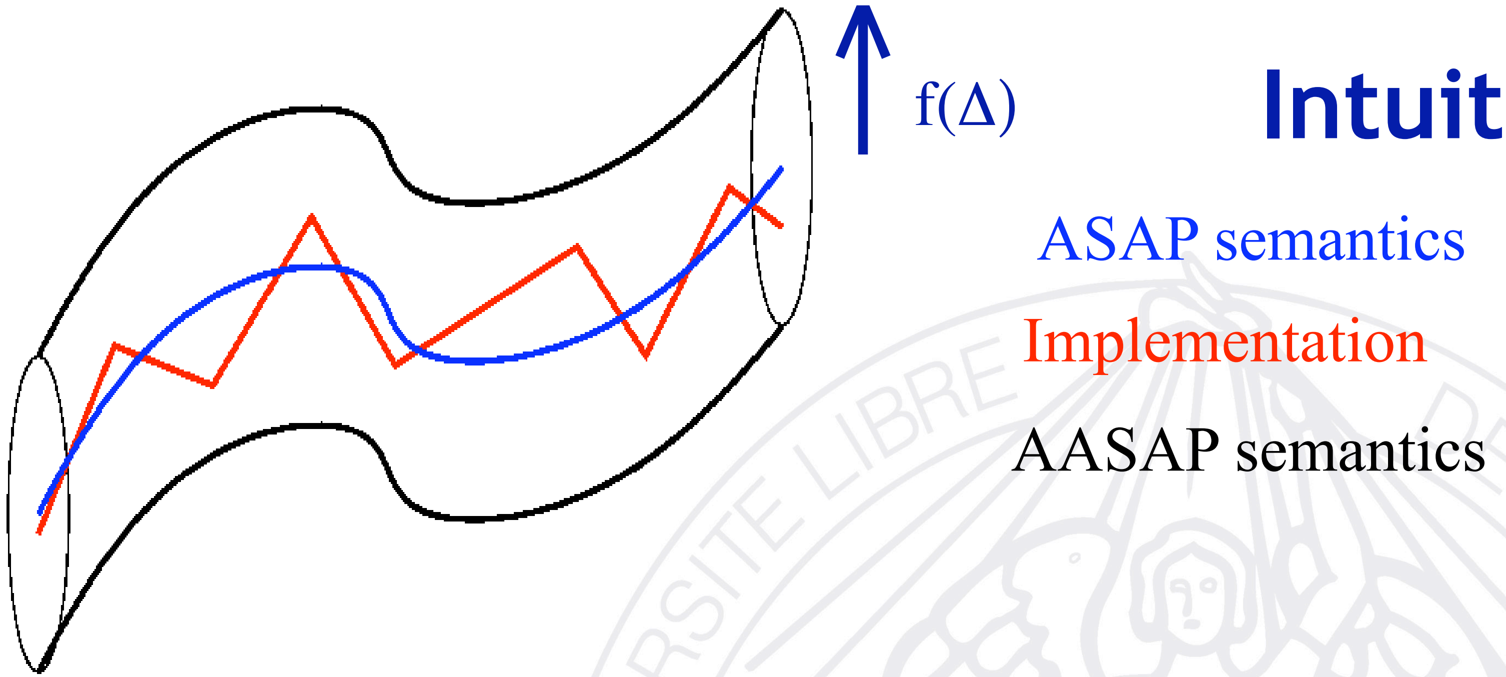
- The question that we ask when we make verification is no more:

Does Environment || ControllerMod avoid Bad ?

- But:

for which values of Δ,
does Environment || ControllerMod(Δ) avoid Bad ?

Intuition



- AASAP semantics defines a "tube" of strategies instead of a unique strategy in the ASAP semantics.
- This tube can be refined into an implementation while preserving safety properties verified on the AASAP-sem

Proof of "implementability" ?

- We define an "implementation semantics" based on:

Read System Clock
Update Sensor Values
Check all transitions and fire one if possible

- The timed behaviour of this scheme is determined by two values :
  - Time length of a loop : Δ_L
  - Time between two clock ticks : Δ_P

Program semantics

Definition 15 [Program Semantics] Let A be an ELASTIC controller and Δ_L, $\Delta_P \in \mathbb{Q}^{>0}$. We define $\Delta_S = \Delta_L + 2\Delta_P$. The (Δ_L, Δ_P) program semantics of A, noted $[\![A]\!]^{\mathsf{Prg}}_{\Delta_L,\Delta_P}$ is the structured timed transition system $\mathcal{T} = \langle S, \iota, \Sigma_{\mathsf{in}}, \Sigma_{\mathsf{out}}, \Sigma_\tau, \rightarrow \rangle$ where:

$(P1)$ S is the set of tuples (l, r, T, I, u, d, f) such that $l \in \mathsf{Loc}$, r is a function from Var into $\mathbb{R}^{\geq 0}$, $T \in \mathbb{R}^{\geq 0}$, I is a function from $\mathsf{Lab}_{\mathsf{in}}$ into $\mathbb{R}^{\geq 0} \cup \{\bot\}$, $u \in \mathbb{R}^{\geq 0}$, $d \in \mathbb{R}^{\geq 0}$, and $f \in \{\top, \bot\}$;

$(P2)$ $\iota = (l_0, r, 0, I, 0, 0, \bot)$ where r is such that for any $x \in \mathsf{Var}$, $r(x) = 0$, I is such that for any $\sigma \in \mathsf{Lab}_{\mathsf{in}}$, $I(\sigma) = \bot$;

$(P3)$ $\Sigma_{\mathsf{in}} = \mathsf{Lab}_{\mathsf{in}}$, $\Sigma_{\mathsf{out}} = \mathsf{Lab}_{\mathsf{out}}$, $\Sigma_\tau = \mathsf{Lab}_\tau \cup \overline{\mathsf{Lab}_{\mathsf{in}}} \cup \{\epsilon\}$;

$(P4)$ the transition relation $\rightarrow$ is defined as follows:

- for the discrete transitions:

  $(P4.1)$ let $\sigma \in \mathsf{Lab}_{\mathsf{out}}$. $((l, r, T, I, u, d, \bot), \sigma, (l', r', T, I, u, 0, \top)) \in \rightarrow$ iff there exists $(l, l', g, \sigma, R) \in \mathsf{Edg}$ such that $\lfloor T \rfloor_{\Delta_P} - r \models {}_{\Delta_S}g_{\Delta_S}$ and $r' = r[R := \lfloor T \rfloor_{\Delta_P}]$.

  $(P4.2)$ let $\sigma \in \mathsf{Lab}_{\mathsf{in}}$. $((l, r, T, I, u, d, f), \sigma, (l, r, T, I', u, d, f)) \in \rightarrow$ iff $I(\sigma) = \bot$ and $I' = I[\sigma := 0]$;

  $(P4.3)$ let $\bar{\sigma} \in \overline{\mathsf{Lab}_{\mathsf{in}}}$. $((l, r, T, I, u, d, \bot), \bar{\sigma}, (l', r', T, I', u, 0, \top)) \in \rightarrow$ iff there exists $(l, l', g, \sigma, R) \in \mathsf{Edg}$ such that $\lfloor T \rfloor_{\Delta_P} - r \models {}_{\Delta_S}g_{\Delta_S}$, $I(\sigma) > u$, $r' = r[R := \lfloor T \rfloor_{\Delta_P}]$ and $I' = I[\sigma := \bot]$;

  $(P4.4)$ let $\sigma \in \mathsf{Lab}_\tau$. $((l, r, T, I, u, d, \bot), \sigma, (l', r', T, I, u, 0, \top)) \in \rightarrow$ iff there exists $(l, l', g, \sigma, R) \in \mathsf{Edg}$ such that $\lfloor T \rfloor_{\Delta_P} - r \models {}_{\Delta_S}g_{\Delta_S}$ and $r' = r[R := \lfloor T \rfloor_{\Delta_P}]$.

  $(P4.5)$ let $\sigma = \epsilon$. $((l, r, T, I, u, d, f), \sigma, (l, r, T + u, I, 0, d, \bot)) \in \rightarrow$ iff either $f = \top$ or the two following conditions hold:

  · for any $\bar{\sigma}$ such that $\sigma \in \mathsf{Lab}_{\mathsf{in}}$, for any $(l, l', g, \sigma, R) \in \mathsf{Edg}$, we have that either $\lfloor T \rfloor_{\Delta_P} - r \not\models {}_{\Delta_S}g_{\Delta_S}$ or $I(\sigma) \leq u$

  · for any $\sigma \in \mathsf{Lab}_{\mathsf{out}} \cup \mathsf{Lab}_\tau$, for any $(l, l', g, \sigma, R) \in \mathsf{Edg}$, we have that $\lfloor T \rfloor_{\Delta_P} - r \not\models {}_{\Delta_S}g_{\Delta_S}$

- for the continuous transitions:

  $(P4.6)$ $((l, r, T, I, u, d, f), t, (l, r, T, I + t, u + t, d + t, f)) \in \rightarrow$ iff $u + t \leq \Delta_L$.

Proof of "implementability" ?

Theorem :

For any timed controller, its AASAP semantics simulates (in the formal sense) its implementation semantics, provided that :

$$\Delta > 2\Delta_L + 4\Delta_P$$

In this case, the implementation is guaranteed to preserve verified properties of the model, that is:

Environment || ControllerMod(Δ) avoid Bad

implies

Environment || ControllerImpl(Δ_L,Δ_P) avoid Bad

Properties of the AASAP Semantics

- Faster is better !

For any Δ_1, Δ_2 such that $\Delta_1 < \Delta_2$:

if

Environment || ControllerMod(Δ_2) avoid Bad

then

Environment || ControllerMod(Δ_1) avoid Bad

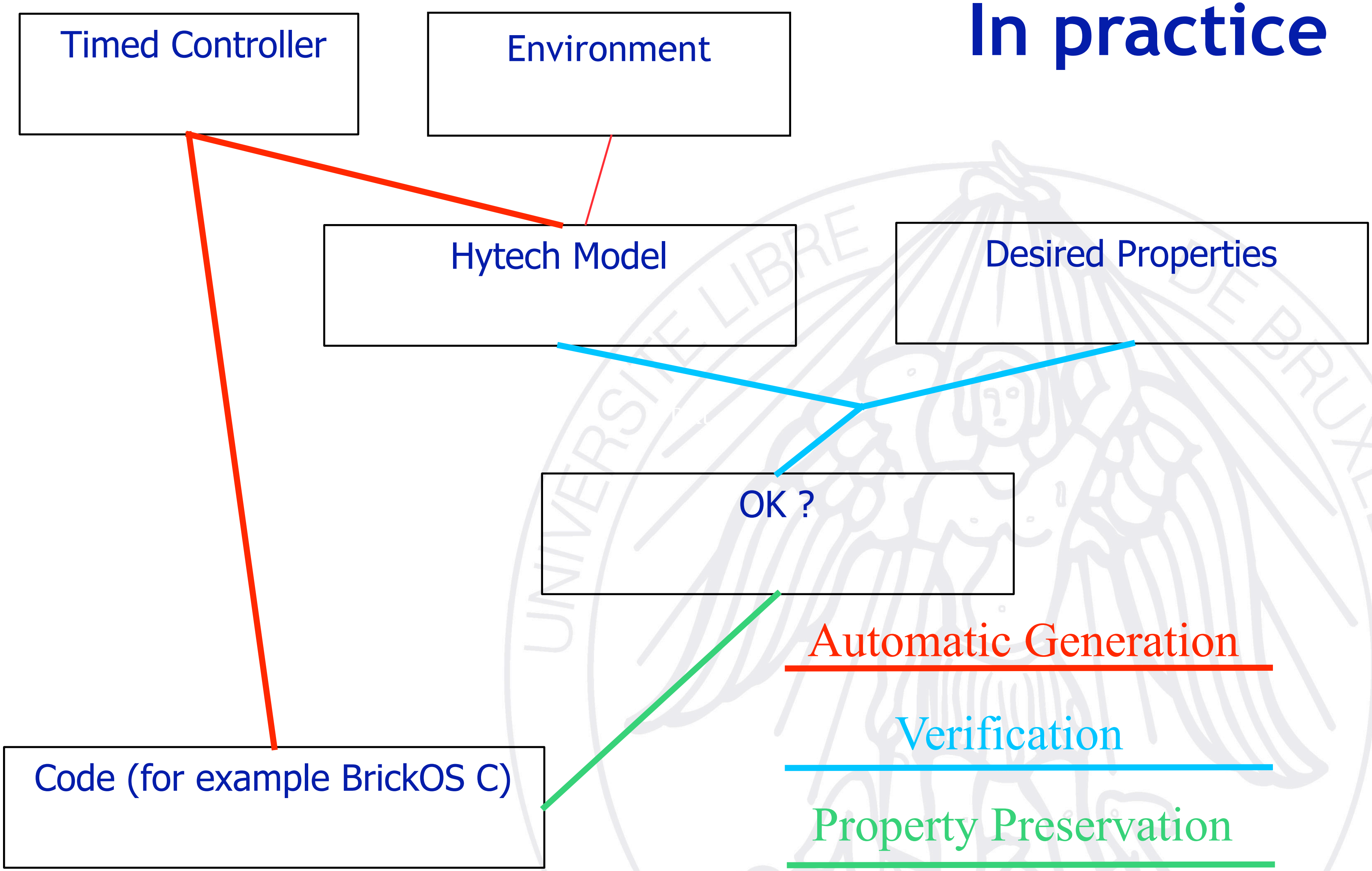
Properties of the AASAP Semantics

- If $\Delta>0$, we get for free a proof that strategies:
  - are nonzeno
  - are such that transitions does not need to be taken faster and faster
- If only $\Delta=0$ guarantees some reachability property, then the control strategy is not implementable

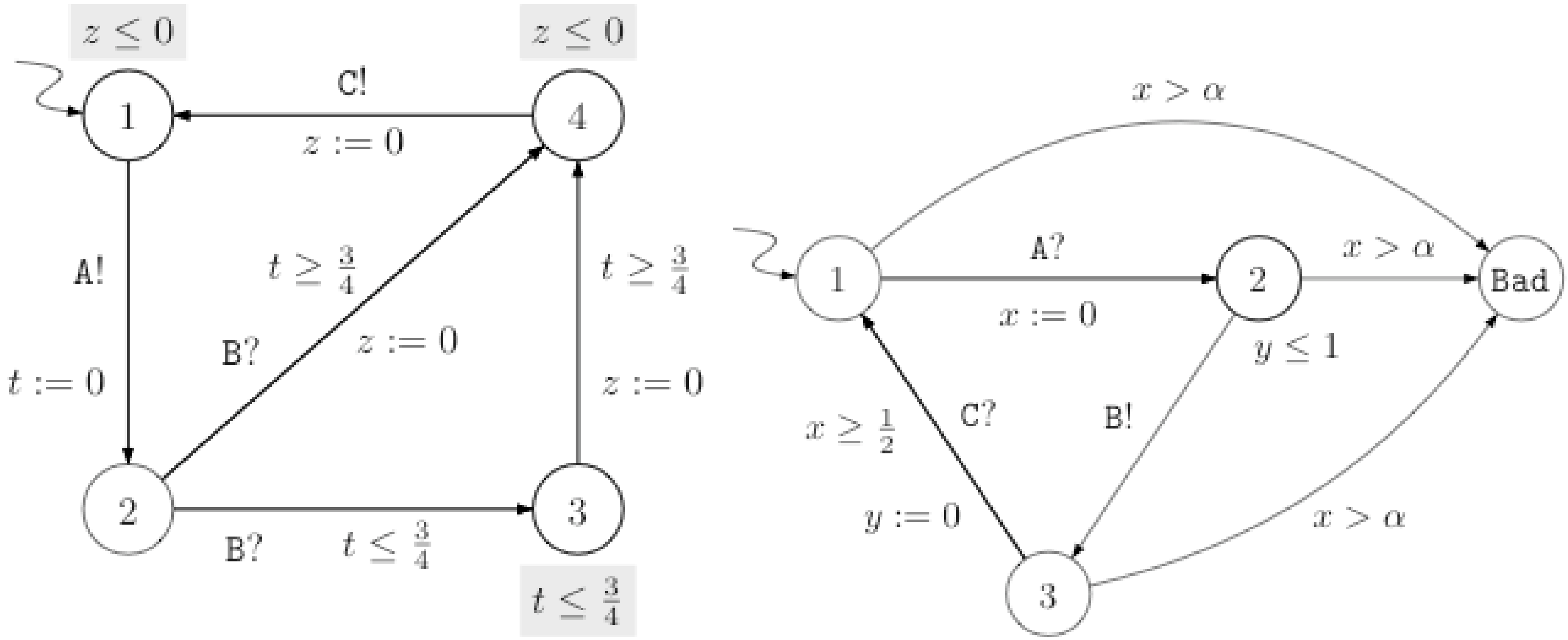
In practice ?

- The AASAP semantics can be coded into a parametric timed automata with only one clock compared to the parameter $\Delta \in \mathbb{Q}$.
- Unfortunately, the reachability problem for that class of timed automata is undecidable... Direct corollary of [CHR02].
- Hytech implements a semi-decision procedure for that problem.

In practice


Timed Controller
Environment
Hytech Model
Desired Properties
OK ?
Code (for example BrickOS C)
Automatic Generation
Verification
Property Preservation

An example



(a) The ASAP controller (b) The environment

If $\alpha=1$ then the system is safe if and only if $\Delta=0$

If $\alpha=2$ then the system is safe if and only if $\Delta<0.25$

A decidable sufficient condition

- A control strategy is structurally implementable if there exists $\Delta_1 > 0$ such that
  - the actions used by the controller in

    Environment || ControllerMod(Δ_1)

    are identical to the actions used by the controller in

    Environment || ControllerMod(0)
  - Environment || ControllerMod(Δ_1) avoid Bad
- The largest such Δ_1 can be expressed in the theory $T(\mathbb{R},+,0,1,\leq)$ from

  Environment || ControllerMod(0)

Methodology to develop controllers

❶ Models using synchrony hypothesis
Environment || ControllerMod

❷ Check
Does Environment || ControllerMod(0) avoid Bad ?

❸ Compute the largest Δ_1 such that
Environment || ControllerMod(Δ_1) avoid Bad
and
ControllerMod(Δ_1) is a structural implementation of ControllerMod(0) in Environment

❹ if $\Delta_1 > 2\,\Delta_L + 4\,\Delta_P$

❺ Generate code
This code will enforce the safety property

Conclusion

- Almost ASAP semantics is implementable!
- If we go for the sufficient condition, the verification overhead should (on going work) be OK
- Almost ASAP semantics guarantees correct code and not only correct idealized model !