Bâtiment IMAG, Université Grenoble Alpes
700, avenue centrale
38401 Saint Martin d’Hères, France
http://www-verimag.imag.fr

Signal Temporal Logic Specifications

José-Ignacio Requeno
Alexey Bakhirkin
Nicolas Basset
Oded Maler*

Introduction

CYBER-PHYSICAL SYSTEMS

Cyber-Physical Systems:

- *hardware/software interacting with the physical environment*
- computerized control systems
- uncertain / changing environment (timing constraints)
- large scale, distributed / networked

Criticality:

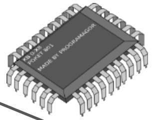
- safety (absence of errors)
- security (resistance to attacks)
- certification

Validation & Verification:

- **hybrid: discrete + real-time systems**

```
void wg_wait(WaitGroup *wg) {
  pthread_mutex_lock (& (m));
  while (wg->cpt > 0) {
    pthread_cond_wait (&(ADDegDONE), & (m));
  }
  pthread_cond_signal (&(ADDegDONE));
  pthread_mutex_unlock (& (m));
}
```

+



© Antoine Heusse

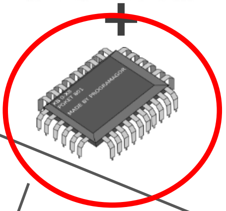
© Sancio83

CYBER-PHYSICAL SYSTEMS

- **Cyber-Physical Systems:**
  - *hardware/software interacting with the physical environment*
  - computerized control systems
  - uncertain / changing environment (timing constraints)
  - large scale, distributed / networked
- **Criticality:**
  - safety (absence of errors)
  - security (resistance to attacks)
  - certification
- **Validation & Verification:**
  - **hybrid: discrete + real-time systems**

```
void wg_wait(WaitGroup *wg) {
   pthread_mutex_lock (& (m));
   while (wg->cpt > 0) {
     pthread_cond_wait (&(ADDegDONE), & (m));
   }
   pthread_cond_signal (&(ADDegDONE));
   pthread_mutex_unlock (& (m));
}
```

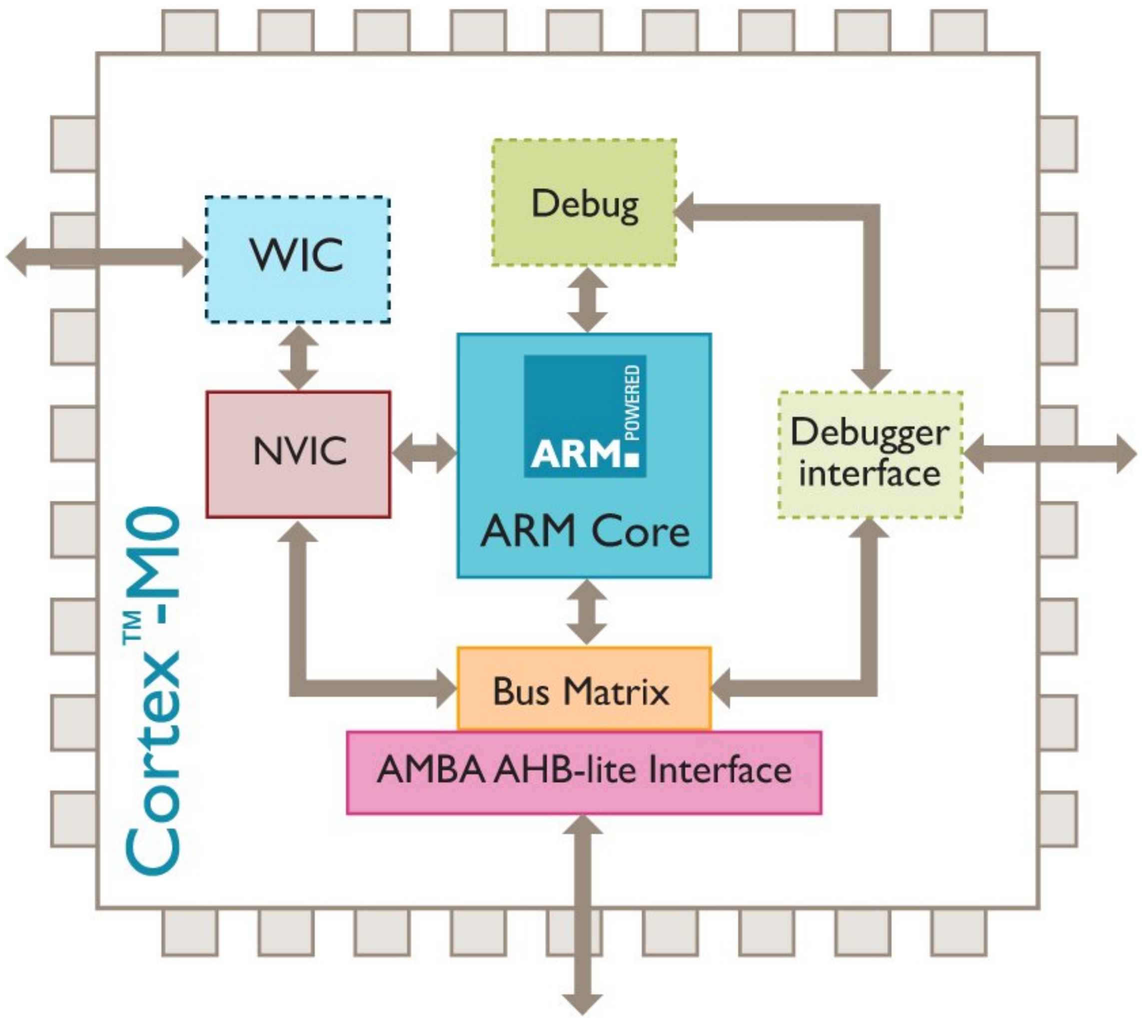
+



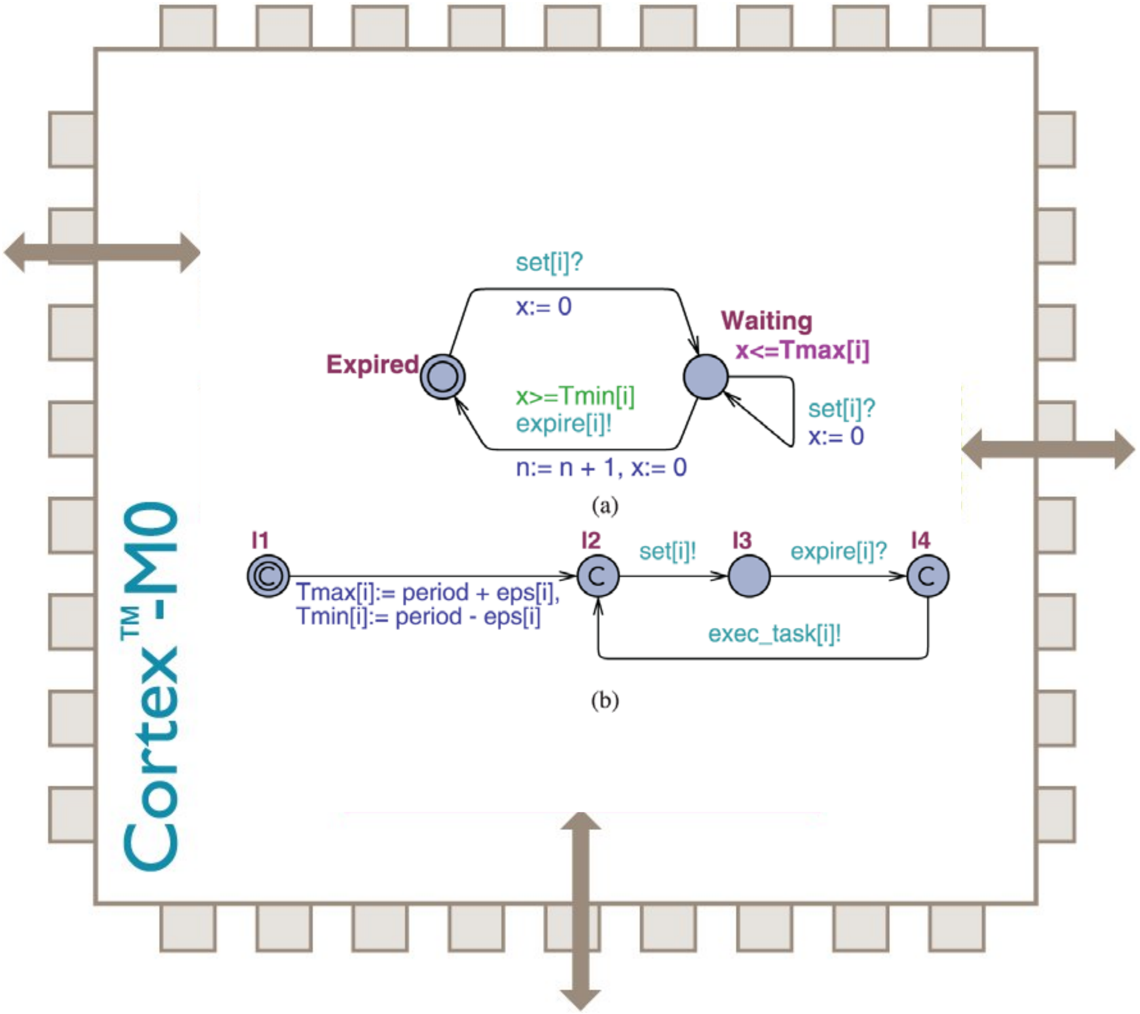
© Antoine Heusse

© Sancio83

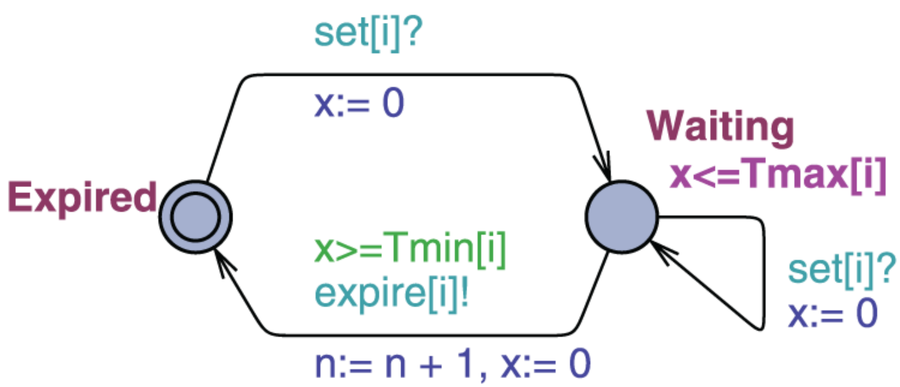
HYBRID SYSTEM



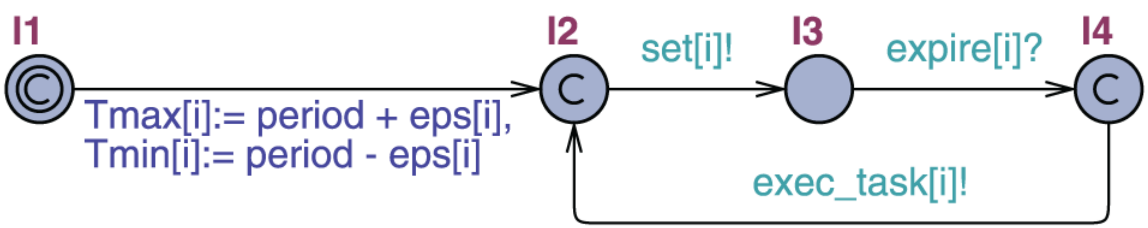
HYBRID SYSTEM



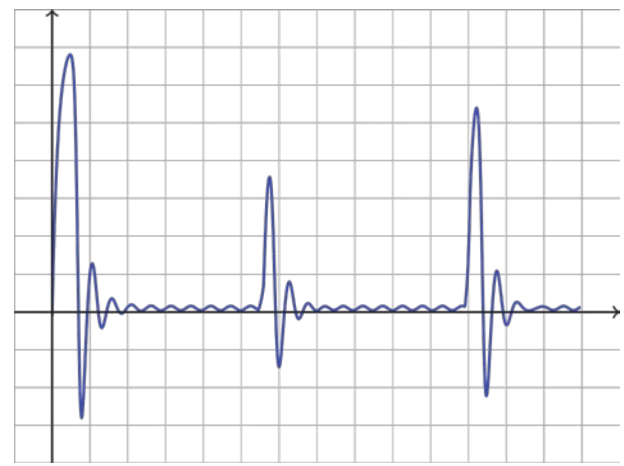
HYBRID SYSTEM



(a)



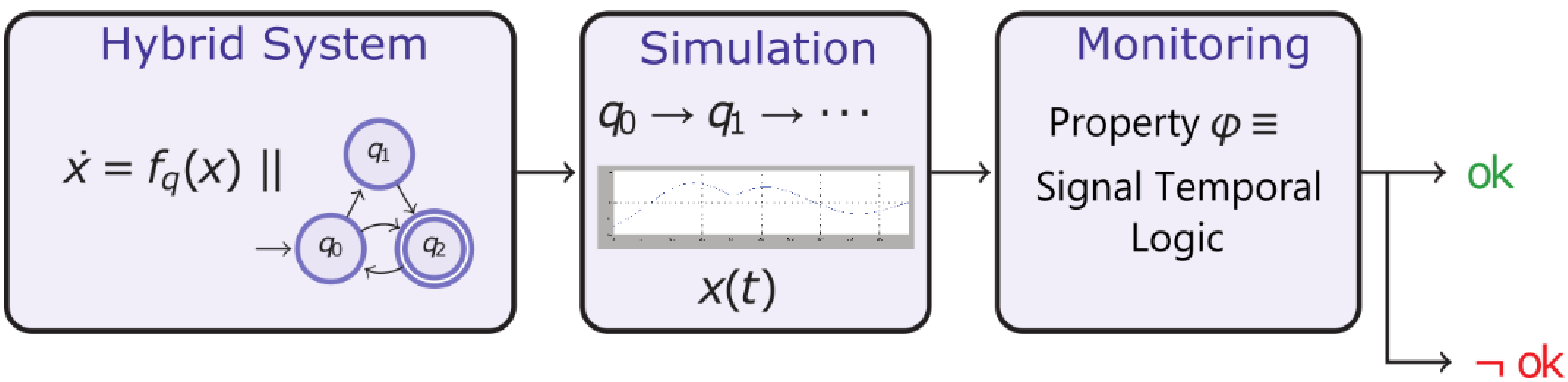
(b)



*Rodriguez-Navas, G., & Proenza, J. (2013). Using timed automata for modeling distributed systems with clocks: Challenges and solutions. IEEE Transactions on Software Engineering, 39(6), 857-868.

MONITORING

- Models are generally hybrid systems producing hybrid **traces**
- Runtime verification (monitoring) analyses a **single** behavior at a time
  - Online: consumes the trace **while** the program still runs
  - Offline: consumes the trace **after** the execution is finished

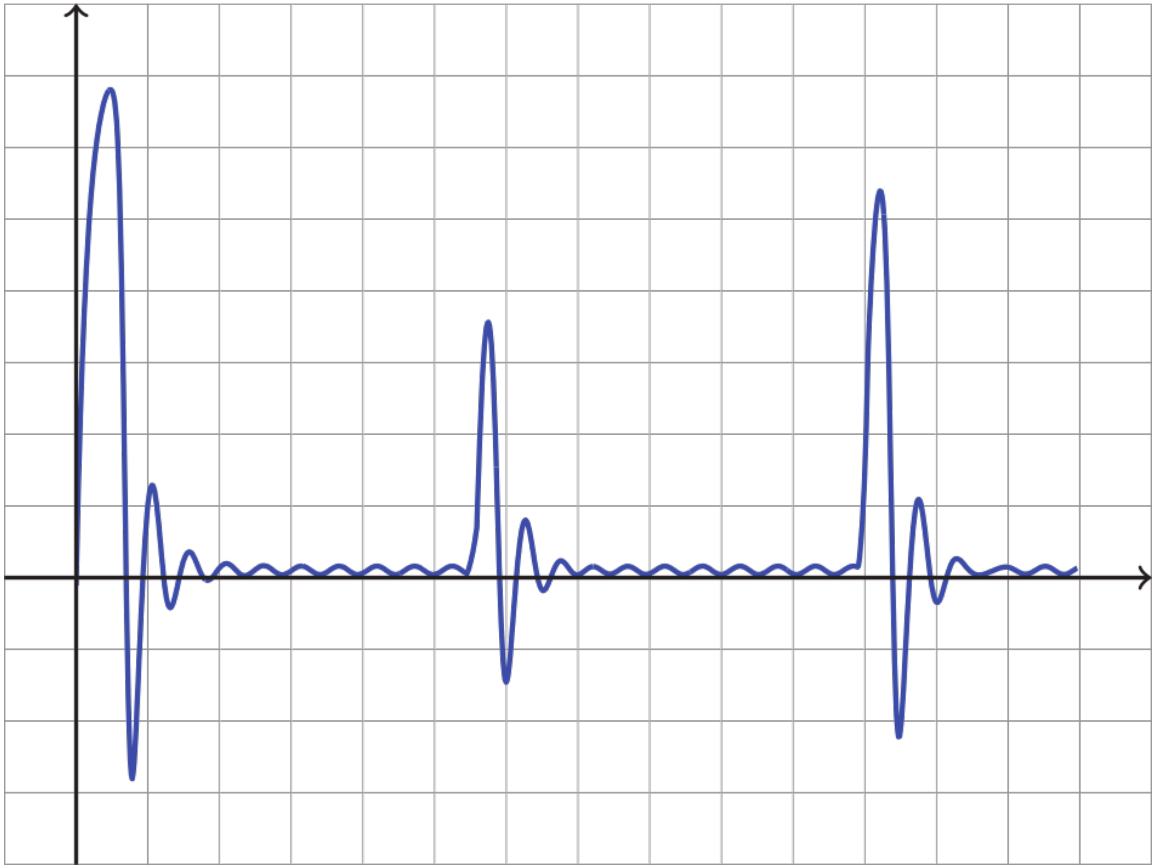


MONITORING

- Online monitoring is useful for:
  - **Infinite** traces
  - **Reacting** against bad events
- Online monitoring requires **efficient** monitors in **time** and **space**
- Runtime verification must be **impartial** and **not anticipate**
  - No monitor gives a verdict based on **incomplete information**
  - The monitor gives a verdict **as soon as possible**

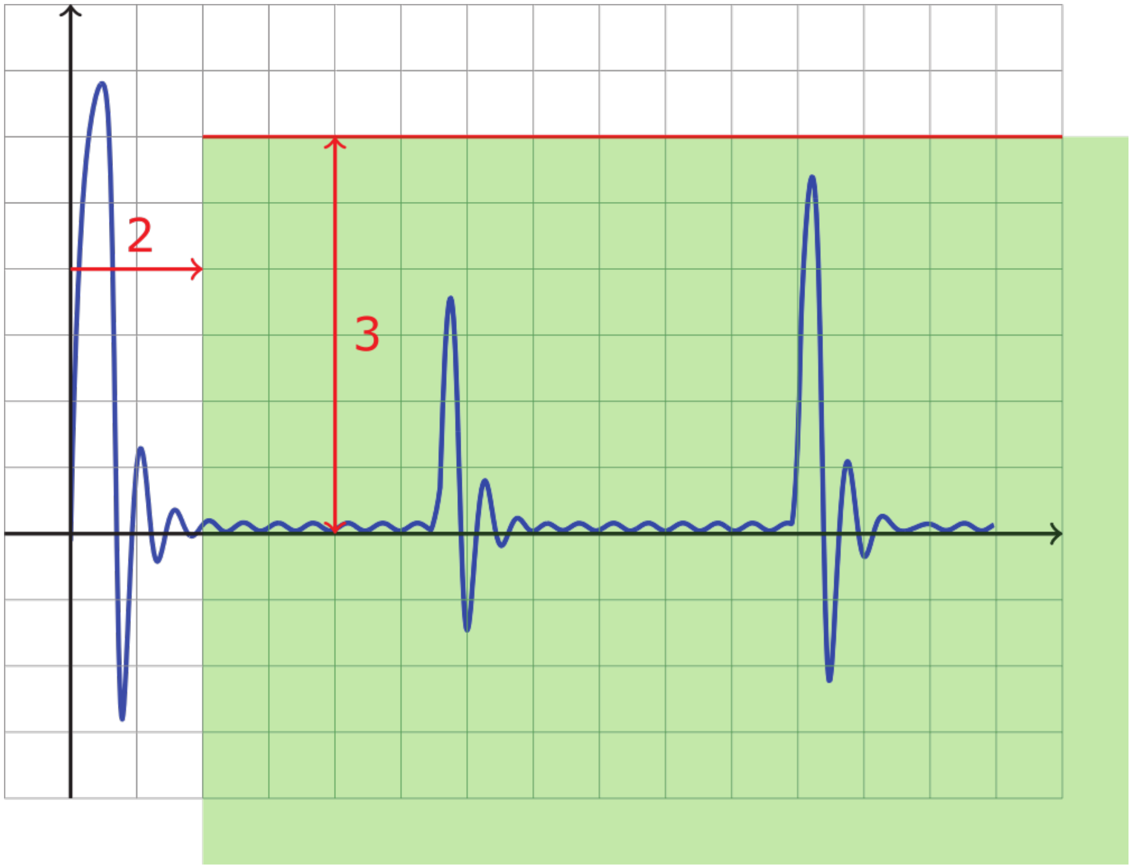
Temporal Logic for Signals

MOTIVATION



MOTIVATION

After 2s, the signal is never above 3



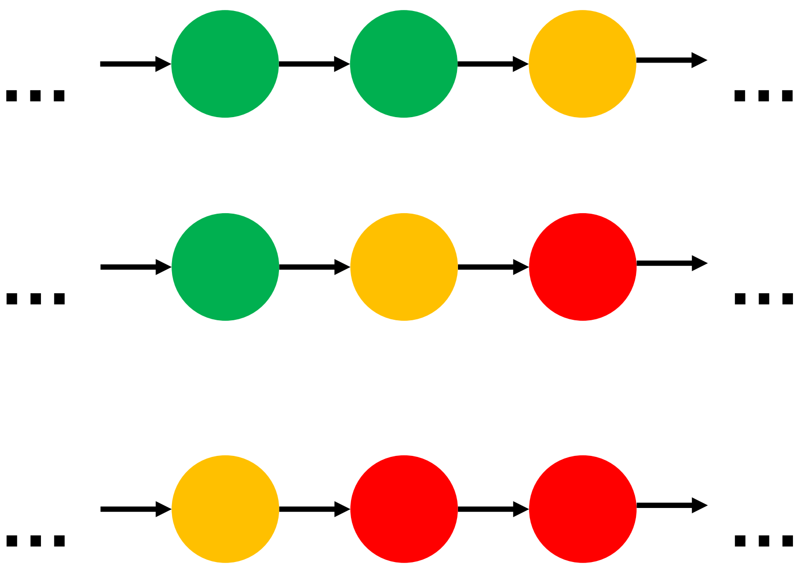
*Donzé, A.: On signal temporal logic. In: Legay, A., Bensalem, S. (eds.) RV 2013. LNCS, vol. 8174, pp. 382–383. Springer, Heidelberg (2013).

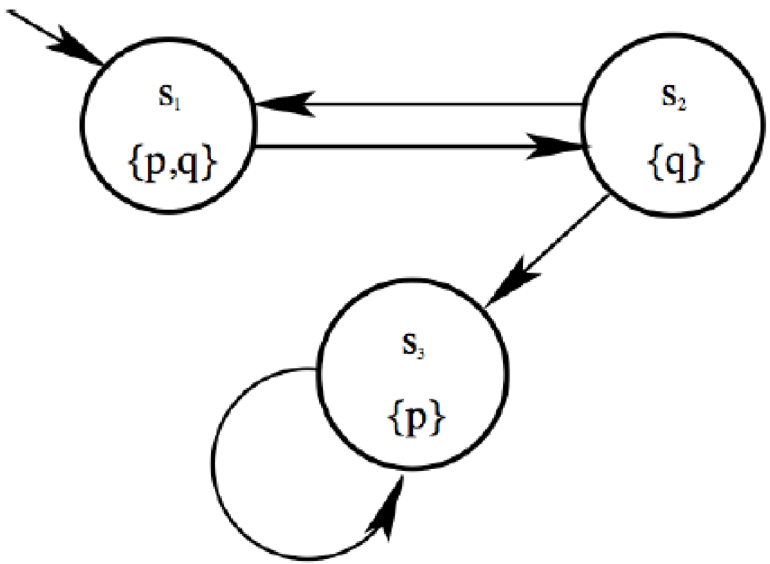
SIGNAL TEMPORAL LOGIC

Linear Temporal Logic (LTL) A. Pnueli, 1977

← Discrete-time semantics

SIGNAL TEMPORAL LOGIC

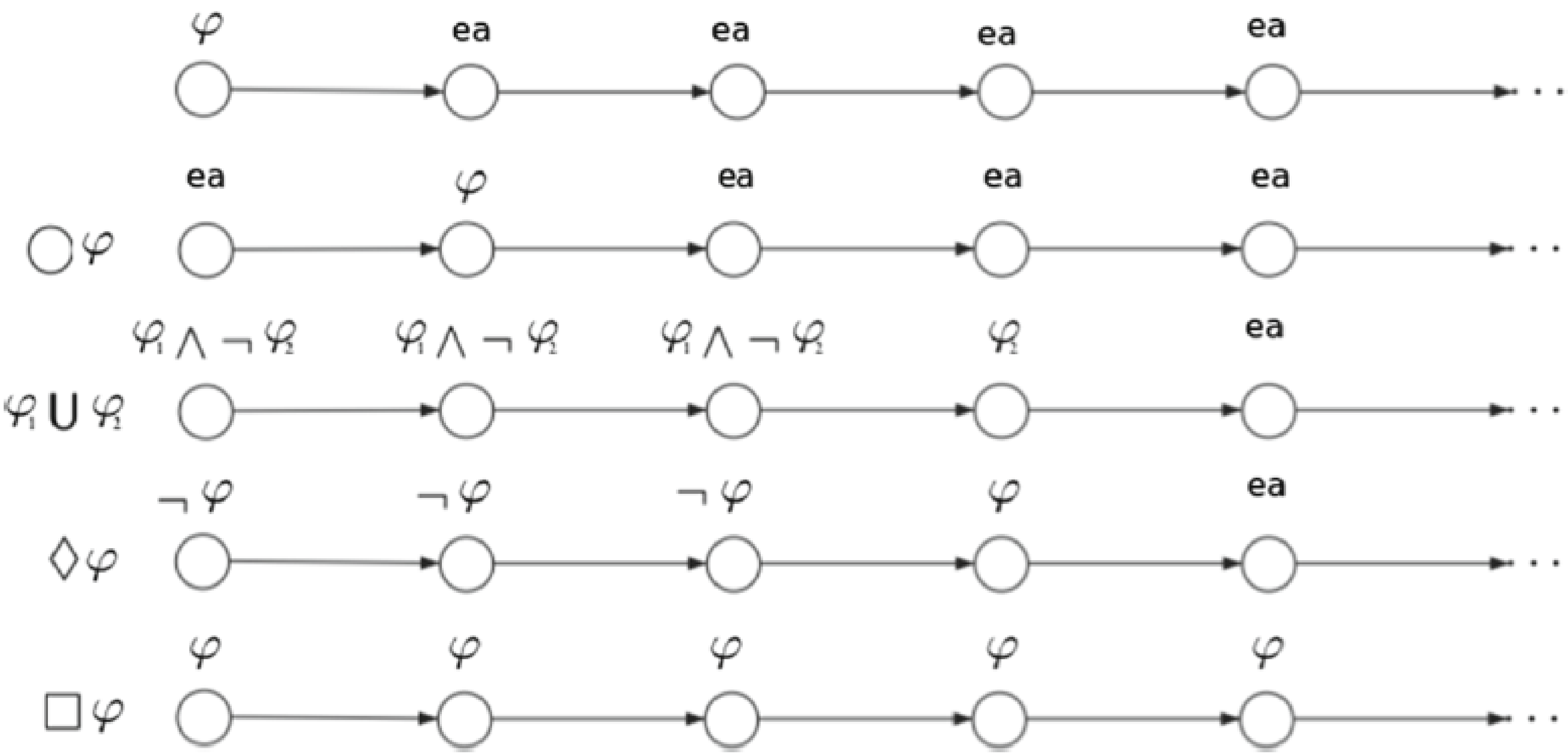




Kripke Structure

SIGNAL TEMPORAL LOGIC

State sequence starting with atomic property φ



SIGNAL TEMPORAL LOGIC

Syntax:

$$\varphi := p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \bigcirc\, \varphi \mid \varphi_1 \mathcal{U} \varphi_2$$

Syntactic sugar for Eventually and Globally:

$$\Diamond\varphi = \top\, \mathcal{U}\varphi \qquad \Box\varphi = \neg\Diamond\neg\varphi$$

Semantics:

$$\begin{array}{lcl}
(\xi, t) \models p & \leftrightarrow & p[t] = 1 \\
(\xi, t) \models \neg\varphi & \leftrightarrow & (\xi, t) \not\models \varphi \\
(\xi, t) \models \varphi_1 \vee \varphi_2 & \leftrightarrow & (\xi, t) \models \varphi_1 \text{ or } (\xi, t) \models \varphi_2 \\
(\xi, t) \models \bigcirc\varphi & \leftrightarrow & (\xi, t+1) \models \varphi \\
(\xi, t) \models \varphi_1 \mathcal{U} \varphi_2 & \leftrightarrow & \exists t' \geq t\ (\xi, t') \models \varphi_2 \text{ and } \forall t'' \in [t, t'), (\xi, t'') \models \varphi_1
\end{array}$$

SIGNAL TEMPORAL LOGIC

Linear Temporal Logic (LTL)

A. Pnueli, 1977

Metric Interval Temporal Logic (MITL)

R. Alur, T. Feder, T. A. Henzinger, 1996

Time Propositional Temporal Logic (TPTL)

R. Alur, T. A. Henzinger, 1994

They extend LTL with continuous-time semantics

SIGNAL TEMPORAL LOGIC

Syntax:

$$\varphi := p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 \mid \varphi_1 \mathcal{U} \varphi_2 \qquad b > a \geq 0$$

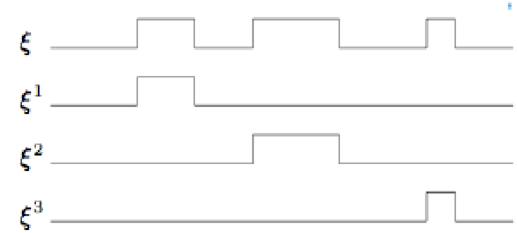
Syntactic sugar for Eventually and Globally:

$$\Diamond_{[a,b]}\varphi = \mathrm{T}\,\mathcal{U}_{[a,b]}\varphi \qquad \Box_{[a,b]}\varphi = \neg\Diamond_{[a,b]}\neg\varphi$$

Semantics:

$$\begin{array}{lll}
(\xi,t) \models p & \leftrightarrow & p[t] = \mathrm{T} \\
(\xi,t) \models \neg\varphi & \leftrightarrow & (\xi,t) \not\models \varphi \\
(\xi,t) \models \varphi_1 \vee \varphi_2 & \leftrightarrow & (\xi,t) \models \varphi_1 \text{ or } (\xi,t) \models \varphi_2 \\
(\xi,t) \models \varphi_1 \mathcal{U} \varphi_2 & \leftrightarrow & \exists t' \geq t\ (\xi,t') \models \varphi_2 \text{ and} \\
 & & \forall t'' \in [t,t'], (\xi,t'') \models \varphi_1 \\
(\xi,t) \models \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 & \leftrightarrow & \exists t' \in t \oplus [a,b]\ (\xi,t') \models \varphi_2 \text{ and} \\
 & & \forall t'' \in [t,t'], (\xi,t'') \models \varphi_1
\end{array}$$

Boolean Signals: $\xi : T \rightarrow \mathbb{B}^n$



Minkowski sum and difference:

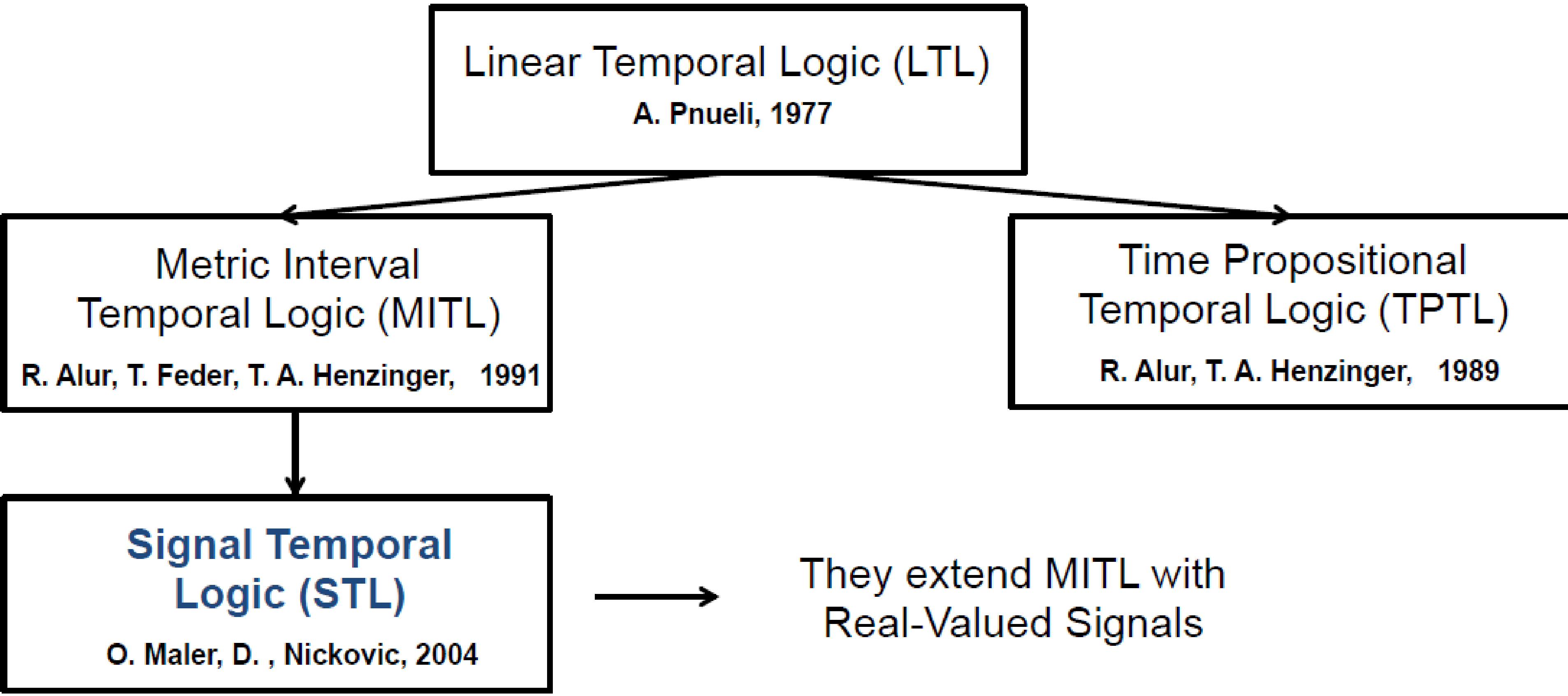
$$\{t\} \oplus [a,b] = [t+a, t+b]$$

$$\{t\} \ominus [a,b] = [t-b, t-a]$$

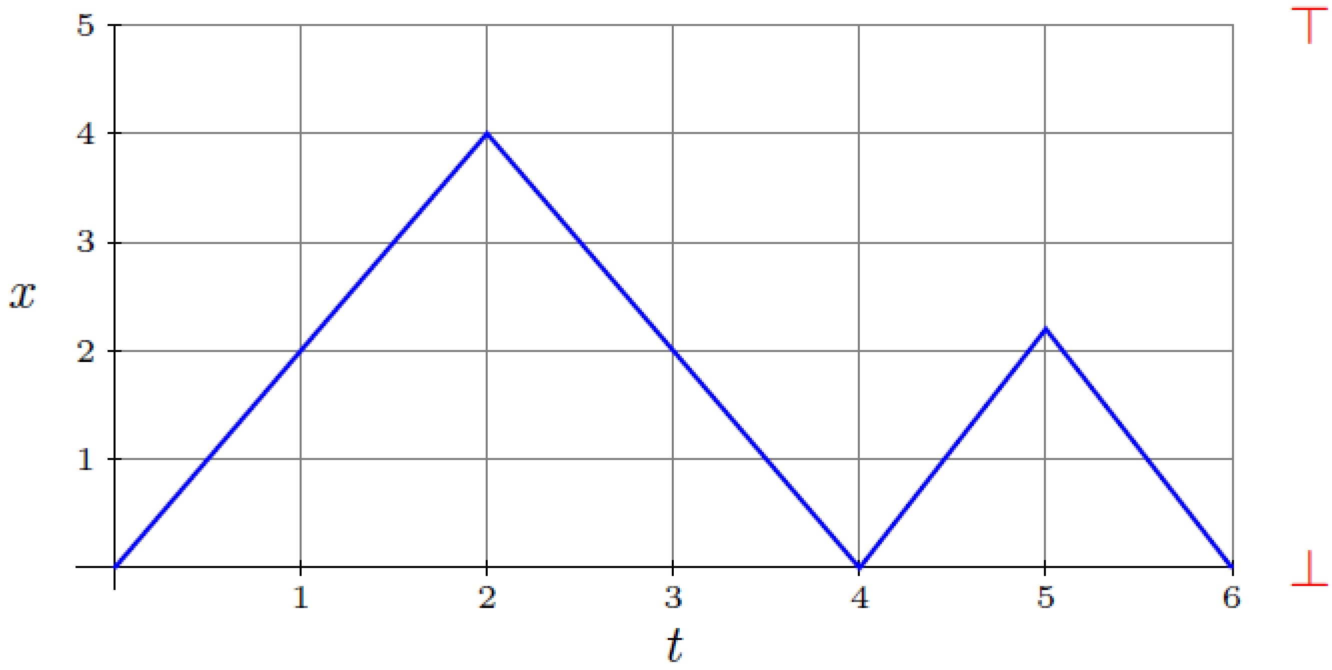
$$[m,n) \oplus [a,b] = [m+a, n+b)$$

$$[m,n) \ominus [a,b] - [m-b, n-a)$$

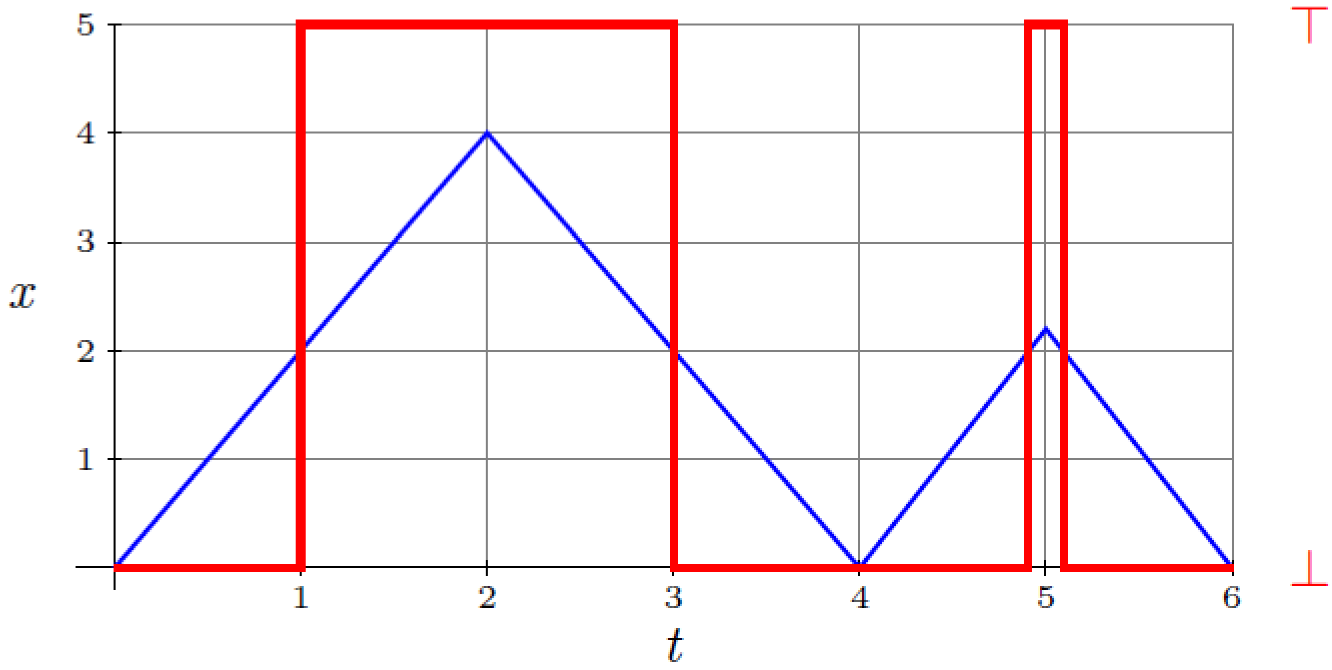
SIGNAL TEMPORAL LOGIC



SIGNAL TEMPORAL LOGIC



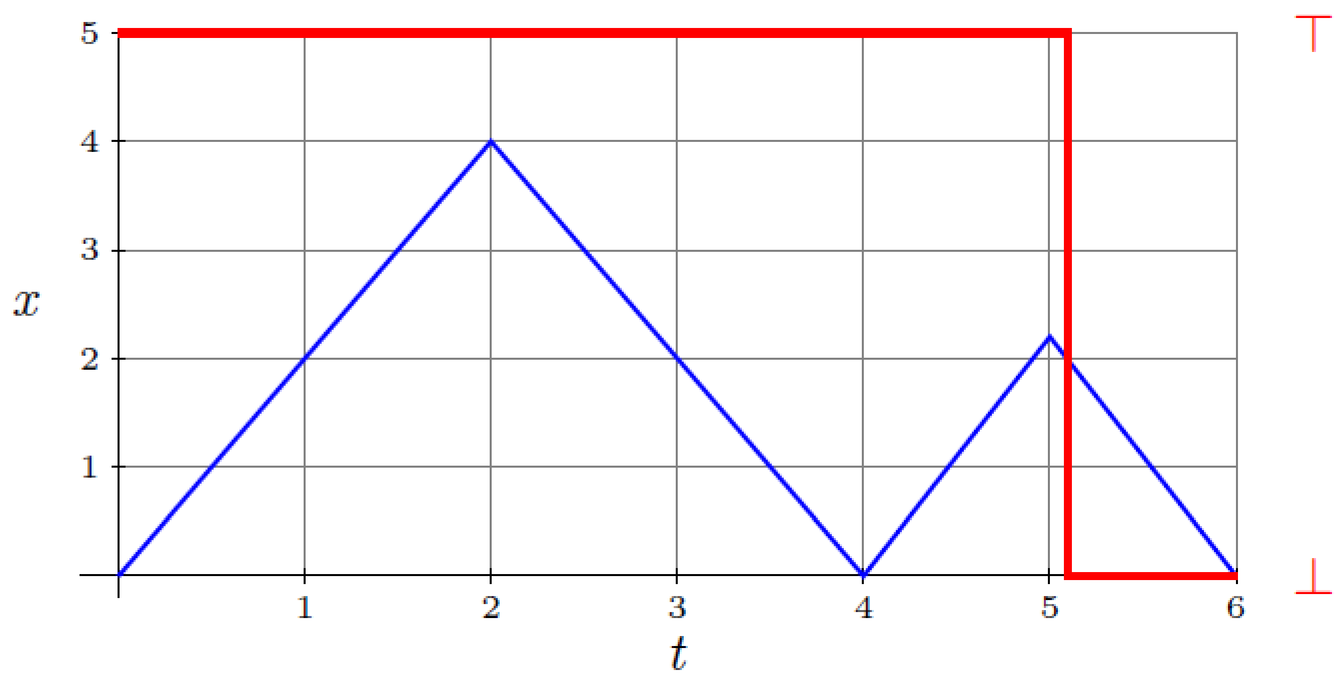
SIGNAL TEMPORAL LOGIC



Satisfaction signal of :

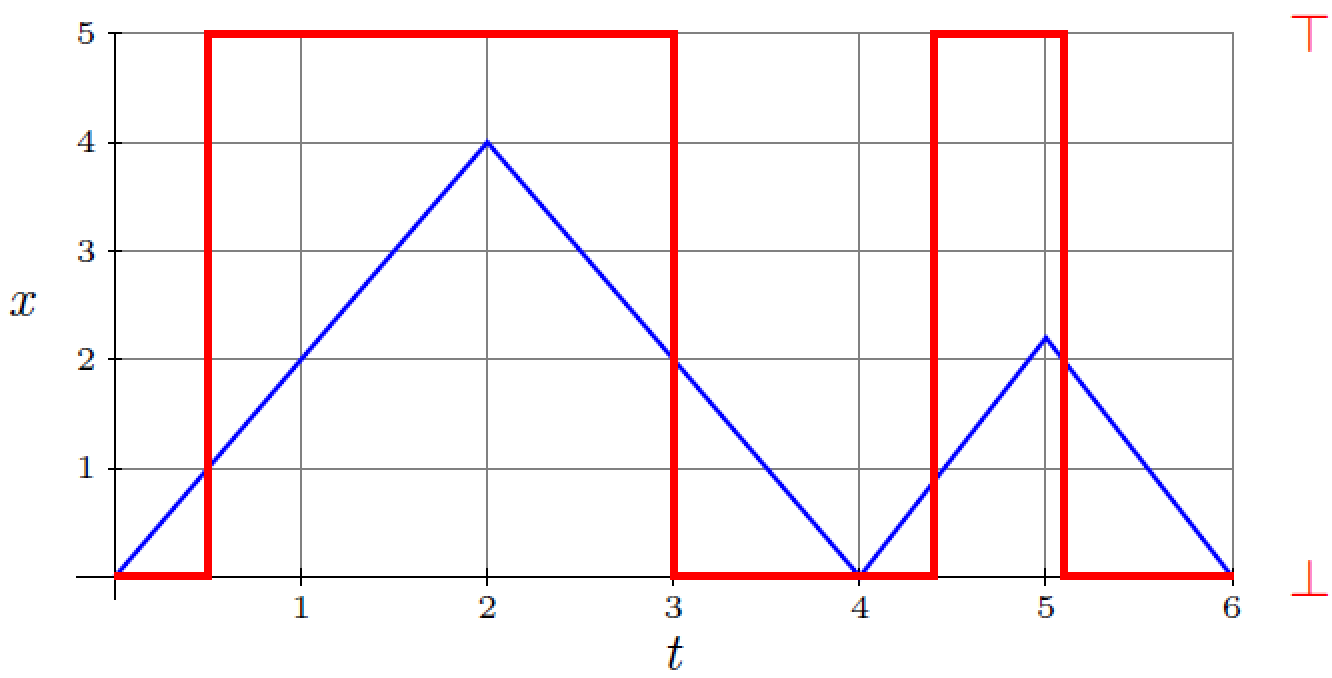
- $\varphi = x \geq 2$

SIGNAL TEMPORAL LOGIC



Satisfaction signal of :

- $\varphi = \mathbf{F}(x \geq 2)$



Satisfaction signal of :

- $\varphi = \mathbf{F}_{[0,0.5]}(x \geq 2)$

STL SYNTAX

$$\varphi := \top \mid \mu \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \, \mathbf{U}_{[a,b]} \, \psi$$

- Assume signals x1[t], x2[t], …, xn[t], then atomic predicates are of the form

$$\mu = f(x_1[t], \ldots, x_n[t]) > 0$$

STL SEMANTICS

The satisfaction of a formula φ by a signal $\mathbf{x} = (x_1, \ldots, x_n)$ at time t is

$$
\begin{array}{lcl}
(\mathbf{x}, t) \models \mu & \Leftrightarrow & f(x_1[t], \ldots, x_n[t]) > 0 \\
(\mathbf{x}, t) \models \varphi \wedge \psi & \Leftrightarrow & (x, t) \models \varphi \wedge (x, t) \models \psi \\
(\mathbf{x}, t) \models \neg\varphi & \Leftrightarrow & \neg((x, t) \models \varphi) \\
(\mathbf{x}, t) \models \varphi\, \mathcal{U}_{[a,b]}\; \psi & \Leftrightarrow & \exists t' \in [t + a, t + b] \text{ such that } (x, t') \models \psi \wedge \\
 & & \qquad \forall t'' \in [t, t'],\ (x, t'') \models \varphi\}
\end{array}
$$

STL SEMANTICS

- Eventually is $\mathsf{F}_{[a,b]}\ \varphi = \top\ \mathcal{U}_{[a,b]}\ \varphi$

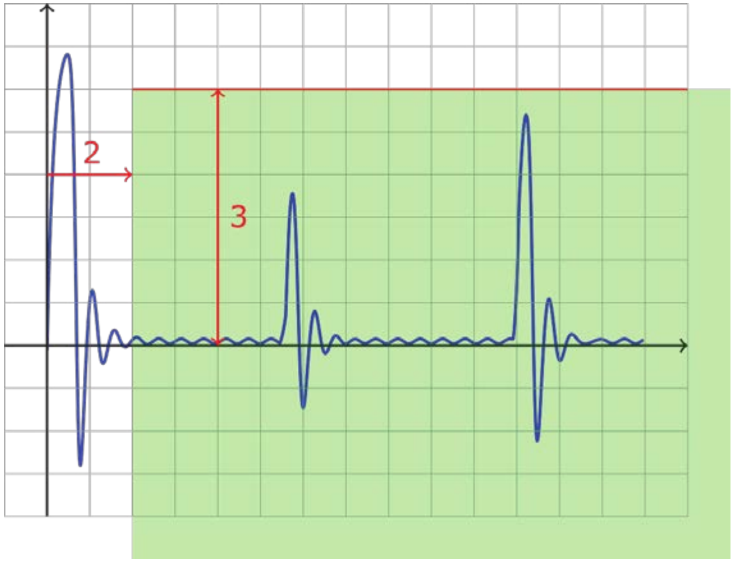
$$(\mathbf{x}, t) \models\ \mathsf{F}_{[a,b]}\ \psi \Leftrightarrow \exists t' \in [t+a, t+b] \text{ such that } (x, t') \models \psi$$

- Always is $\mathsf{G}_{[a,b]}\varphi = \neg(\ \mathsf{F}_{[a,b]}\ \neg\varphi)$

$$(\mathbf{x}, t) \models \mathsf{G}_{[a,b]}\psi \Leftrightarrow \forall t' \in [t+a, t+b] \text{ such that } (x, t') \models \psi$$

SIGNAL TEMPORAL LOGIC

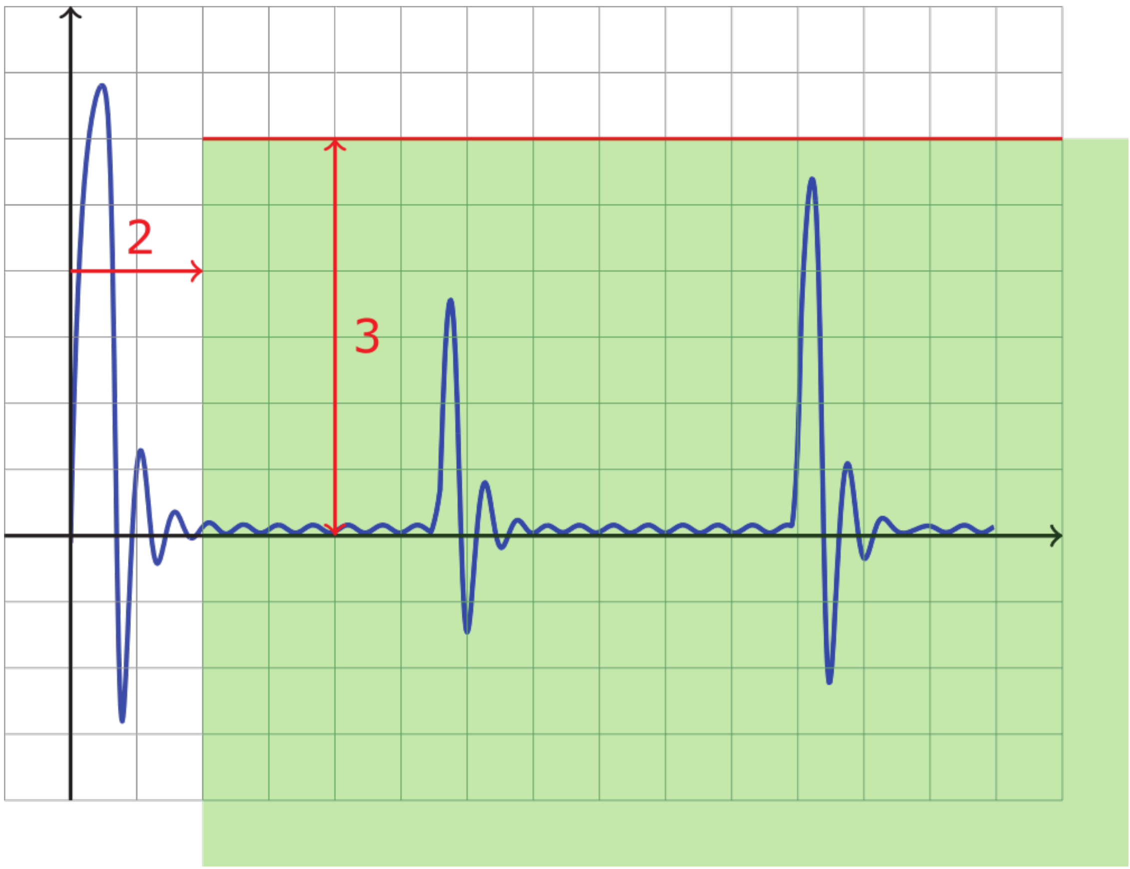
- **LTL:**
  - ▶ G(r => F g)
  - ▶ Boolean predicates, discrete-time
- **MTL:**
  - ▶ G(r => F [0,.5s] g)
  - ▶ Boolean predicates, real-time
- **STL:**
  - ▶ G(x[t] > 0 => F [0,.5s] y[t] > 0)
  - ▶ Predicates over **real values**, **real-time**



STL EXAMPLE

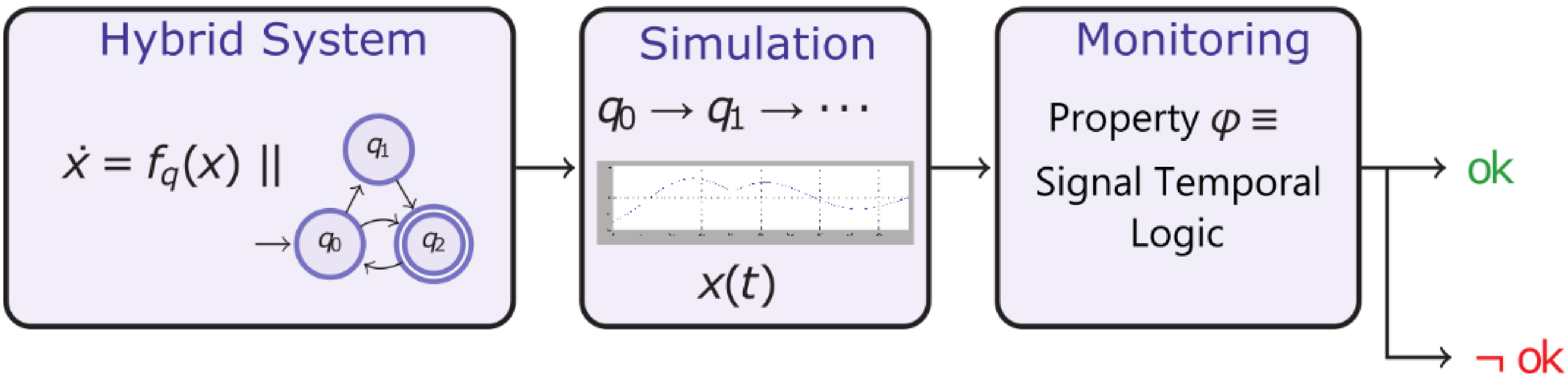
After 2s, the signal is never above 3

$$\varphi := \ \mathsf{F}_{[2,\infty]} \ \ (x[t] < 3)$$

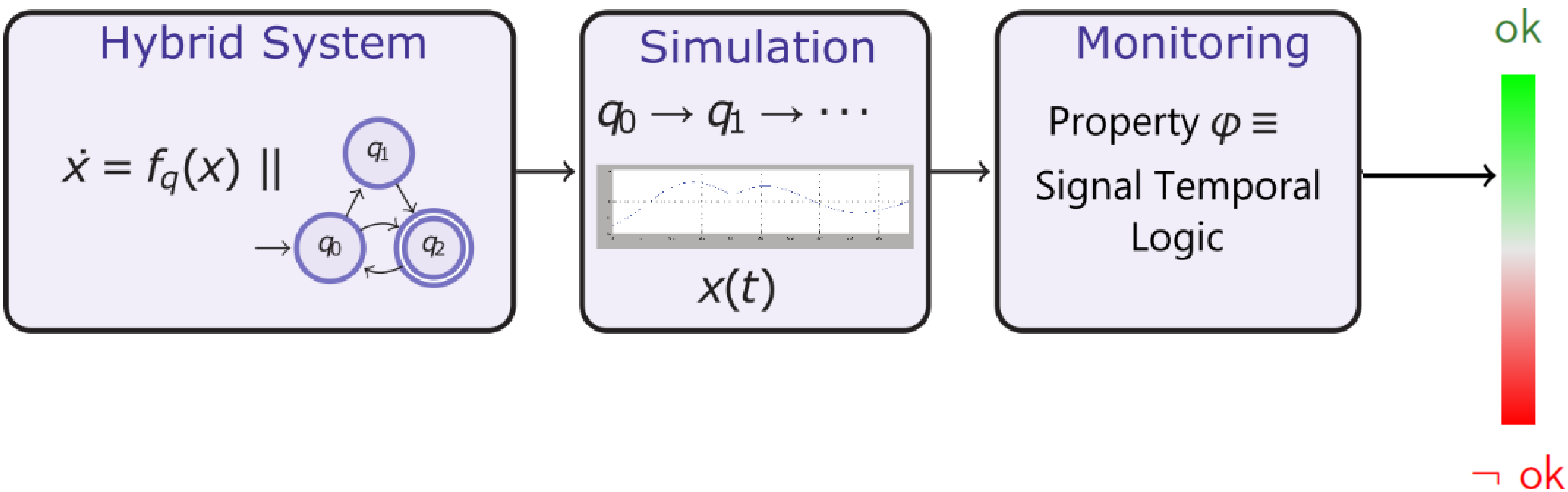


Quantitative Semantics for STL

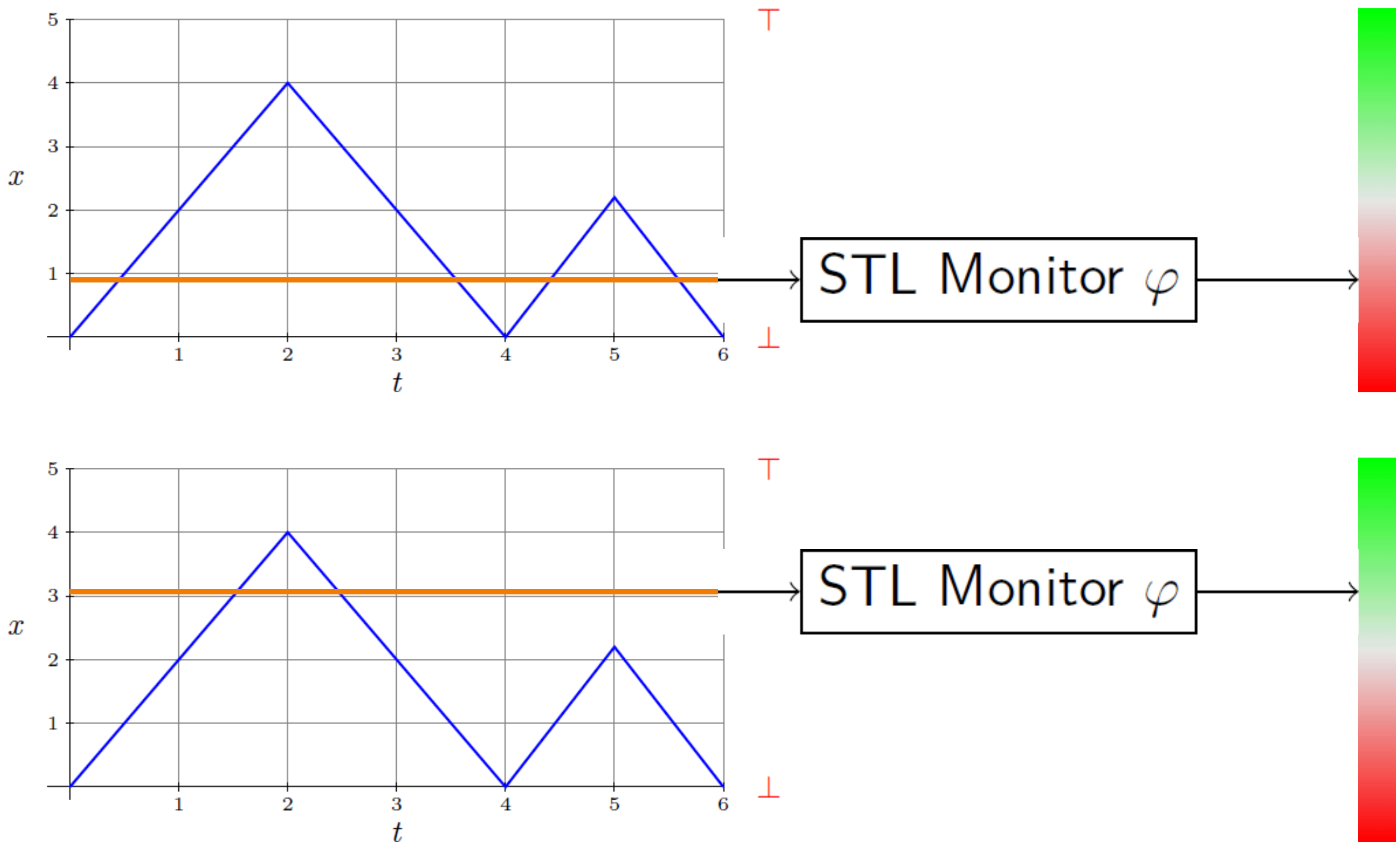
QUALITATIVE SEMANTICS



QUANTITATIVE SEMANTICS

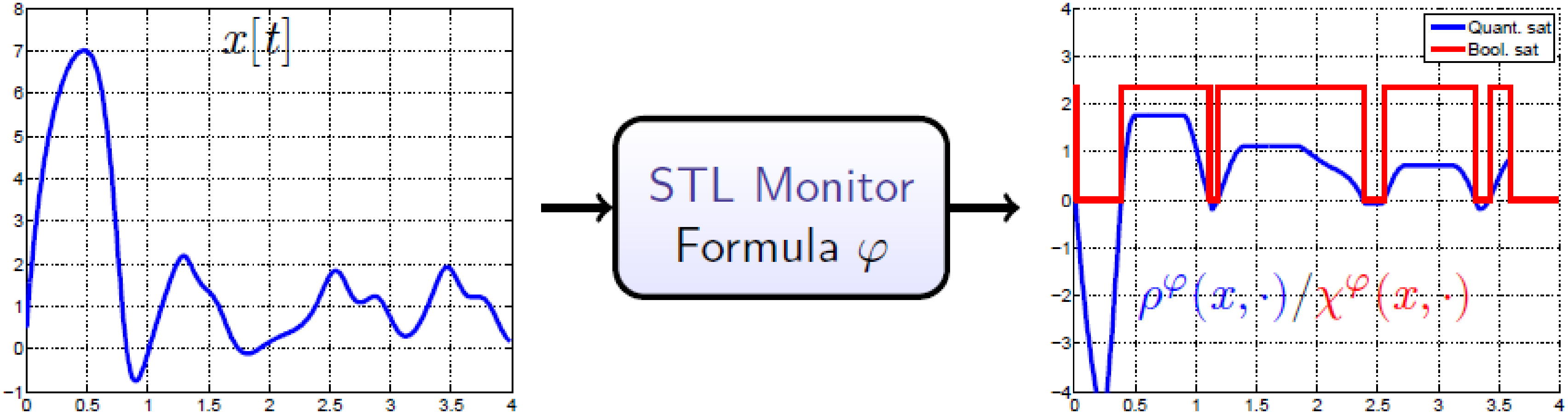


QUANTITATIVE SEMANTICS



QUANTITATIVE STL

A robust STL monitor is a *transducer* that transform x into $\rho^{\varphi}(x,.)$



In practice

- Trace: time words over alphabet $\mathbb{R}$, linear interpolation

  Input: $x(\cdot) \triangleq (t_i, x(t_i))_{i\in\mathbb{N}}$ Output: $\rho^{\varphi}(x,\cdot) \triangleq (r_j, z(r_j))_{j\in\mathbb{N}}$
- Continuity and piecewise affine property preserved

QUANTITATIVE STL

Given a formula φ, a signal x and a time t, recall that we have:

$$\rho^{\varphi}(x,t) > 0 \Rightarrow x, t \models \varphi$$
$$\rho^{\varphi}(x,t) < 0 \Rightarrow x, t \not\models \varphi$$

ok

$$p_u \longrightarrow \boxed{\text{System } \mathcal{S}} \longrightarrow x(t) \longrightarrow \boxed{\text{STL Monitor } \varphi} \longrightarrow \rho^{\varphi}(x,t)$$

$\neg$ ok

As x is obtained by simulation using input parameters p_u, the falsification problem can be reduced to solving

$$\rho^* = \min_{p_u \in \mathcal{P}_u} \rho^{\varphi}(x, 0)$$

If $\rho^* < 0$, we found a counterexample.

BOOLEAN OPERATORS

Negation

- Input signal: $(t_i, x(t_i))_{i \leq n_x}$
- Output signal: $(t_i, -x(t_i))_{i \leq n_x}$

Conjunction

- Input signals: $(t_i, x(t_i))_{i \leq n_x}$, $(t'_i, x'(t'_i))_{i \leq n_{x'}}$
- Output signal: $(r_i, z(r_i))_{i \leq n_z}$
  Time sequence r contains t, t', and punctual intersections $x \cap x'$
  Value $z(r_i) = \min\{x(r_i), x'(r_i)\}$

QUANTITATIVE OPERATORS

$$
\begin{aligned}
\rho^{\mu}(x, t) &= f(x_1[t], \ldots, x_n[t]) \\
\rho^{\neg\varphi}(x, t) &= -\rho^{\varphi}(x, t) \\
\rho^{\varphi_1 \wedge \varphi_2}(x, t) &= \min(\rho^{\varphi_1}(x, t), \rho^{\varphi_2}(w, t)) \\
\rho^{\varphi_1 \, \mathcal{U}_{[a,b]} \, \varphi_2}(x, t) &= \sup_{\tau \in t+[a,b]} (\min(\rho^{\varphi_2}(x, \tau), \inf_{s \in [t,\tau]} \rho^{\varphi_1}(x, s))
\end{aligned}
$$

UNTIL OPERATOR

Rewrite Property

- Boolean Semantics
  $\varphi \mathbf{U}_{[a,b]} \psi \sim \mathbf{G}_{[0,a]} \varphi \wedge \mathbf{F}_{[a,b]} \psi \wedge \mathbf{F}_{\{a\}} (\varphi \mathbf{U} \psi)$

- Quantitative Semantics
  $\rho^{\varphi \mathbf{U}[a,b] \psi}(x,t) = \rho^{\mathbf{G}_{[0,a]} \varphi \wedge \mathbf{F}_{[a,b]} \psi \wedge \mathbf{F}_{\{a\}} (\varphi \mathbf{U} \psi)}(x,t)$

Combines *untimed until* and *timed eventually*

UNTIMED UNTIL

Computed by *backward induction*:

For all $s < t$, we note $x_{\upharpoonright[s,t)}$ the restriction of x to $[s,t)$.

- Boolean Semantics $\quad x, s \models \varphi \mathbf{U} \psi \quad$ iff
  $x_{\upharpoonright[s,t)}, s \models \varphi \mathbf{U} \psi$ or $(x_{\upharpoonright[s,t)}, s \models \mathbf{G}\varphi$ and $x, t \models \varphi \mathbf{U} \psi)$

- Quantitative Semantics $\quad \rho^{\varphi \mathbf{U} \psi}(x, s) =$
  $\max\left\{\rho^{\varphi \mathbf{U} \psi}(x_{\upharpoonright[s,t)}, s), \min\{\rho(\mathbf{G}\varphi, x_{\upharpoonright[s,t)}, s), \rho(\varphi \mathbf{U} \psi, x, t)\}\right\}$

TIMED EVENTUALLY

Definition: $\rho^{\mathbf{F}_{[a,b]}\varphi}(x,t) = \sup_{t' \in [t+a,t+b]} \rho^{\varphi}(x,t) = \sup_{[t+a,t+b]} x$

Computation:

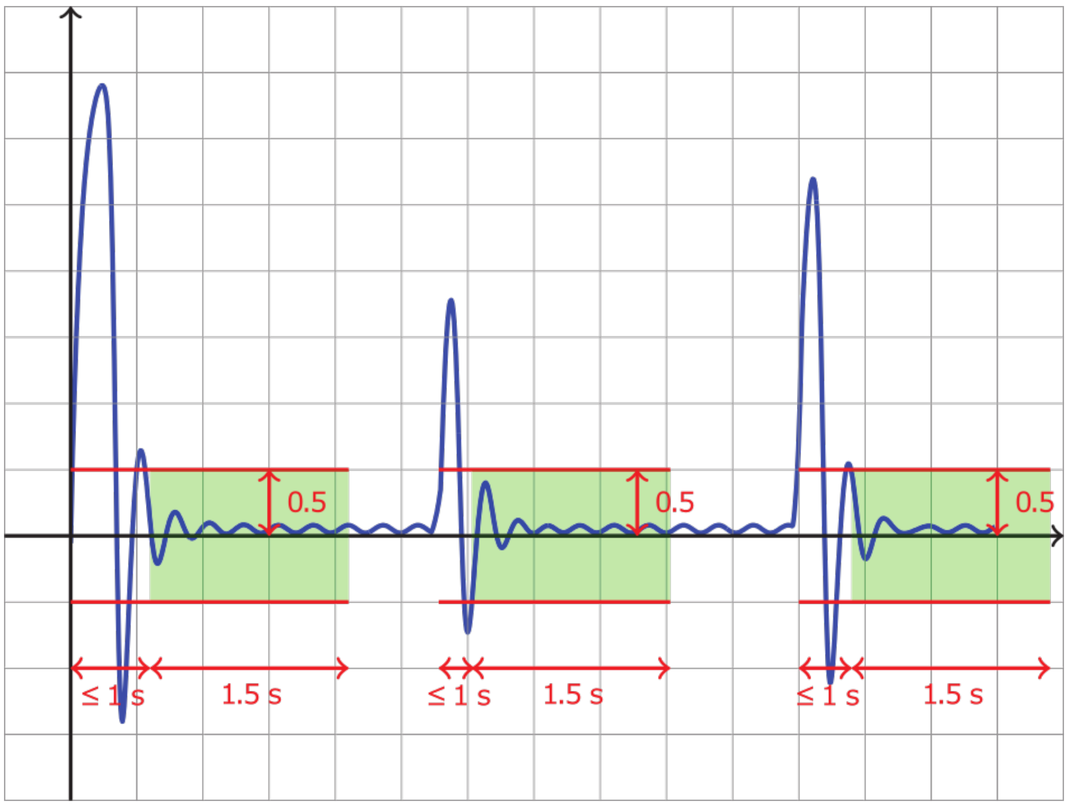
- the maximum is reached at $t+a$, $t+b$, or at sample point in $\{t_i \mid t_i \in (t+a, t+b]\}$
- $\max\{x(t_i) \mid t_i \in (t+a, t+b]\}$ computed by Lemire's algorithm:

  we maintain an ordered set M such that $\max\{x(t_i) | i \in M\} = \max\{x(t_i) \mid t_i \in (t+a, t+b]\}$

SIGNAL STABILIZATION

Always |x| > 0.5 then after 1s, |x| settles under 0.5 for 1.5 s

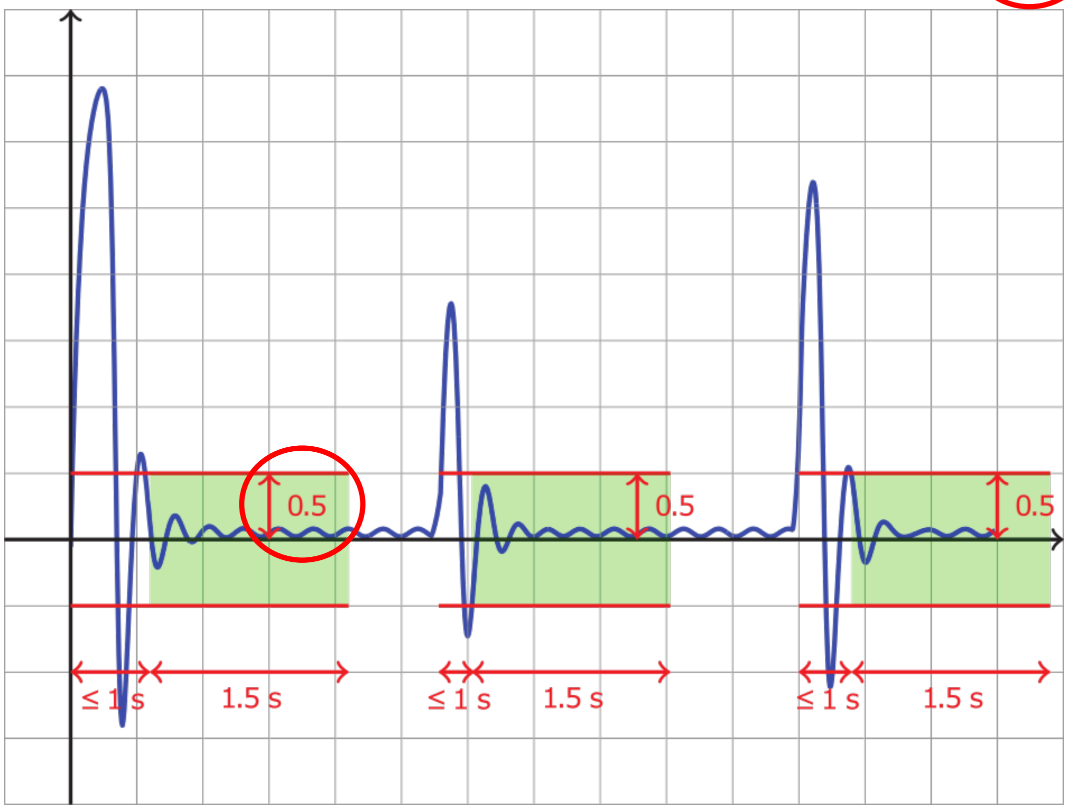
$$\varphi := \mathsf{G}(x[t] > .5 \rightarrow \ \mathsf{F}_{[0,.6]} \ \ (\ \mathsf{G}_{[0,1.5]} \ x[t] < 0.5))$$



SIGNAL STABILIZATION

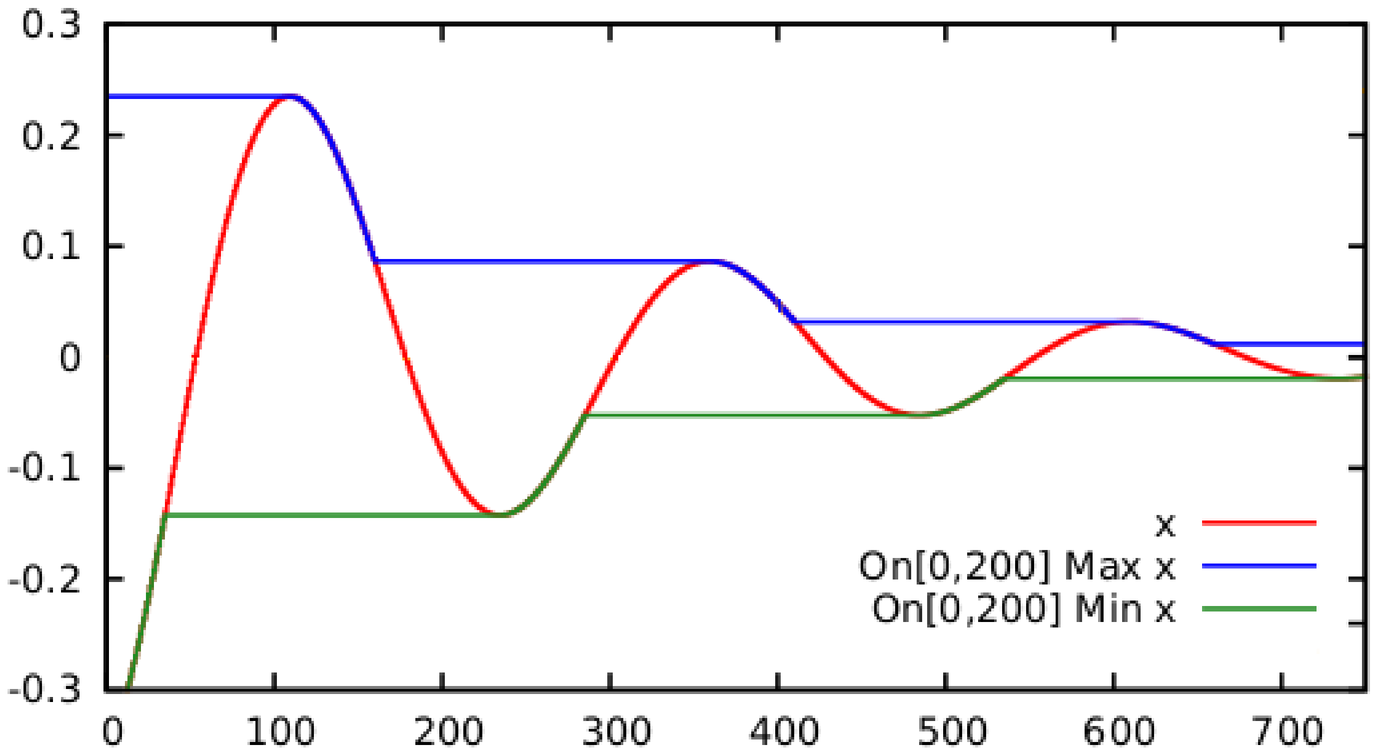
Always |x| > 0.5 then after 1s, |x| settles under 0.5 for 1.5 s

$$\varphi := \mathsf{G}(x[t] > .5 \rightarrow \ \mathsf{F}_{[0,.6]} \ \ (\ \mathsf{G}_{[0,1.5]} \ x[t] < 0.5))$$



EXTENDED STL

Amplitude |x| ≤ 0.1 for 200 s



*Bakhirkin, A., & Basset, N.: Specification and Efficient Monitoring Beyond STL. In TACAS 2019. To appear.

EXTENDED STL SYNTAX

$$\varphi ::= c \mid x \mid f(\varphi_1 \cdots \varphi_n) \mid \mathrm{On}_{[a,b]}\,\psi \mid \psi\,\mathrm{U}^d_{[a,b]}\,\varphi \mid \varphi_1 \downarrow \mathrm{U}^d_{[a,b]} \varphi_2$$

EXTENDED STL SYNTAX

- STL + min/max operators

$$\varphi ::= c \mid x \mid f(\varphi_1 \cdots \varphi_n) \mid \mathrm{On}_{[a,b]}\, \psi \mid \psi\, \mathrm{U}^{d}_{[a,b]}\, \varphi \mid \varphi_1 \downarrow \mathrm{U}^{d}_{[a,b]} \varphi_2$$

$$\psi ::= \mathrm{Min}\, \varphi \mid \mathrm{Max}\, \varphi$$

- **Efficient monitoring method for evaluating min/max operators over a signal**
  - Linear cost with respect to the length of the signal

$$[\![\mathrm{On}_{[a,b]}\,\psi]\!](t) = [\![\psi]\!]([t+a, t+b])$$

$$[\![\mathrm{Min}\,\varphi]\!][a,b] = \min_{[a,b]} [\![\varphi]\!]$$

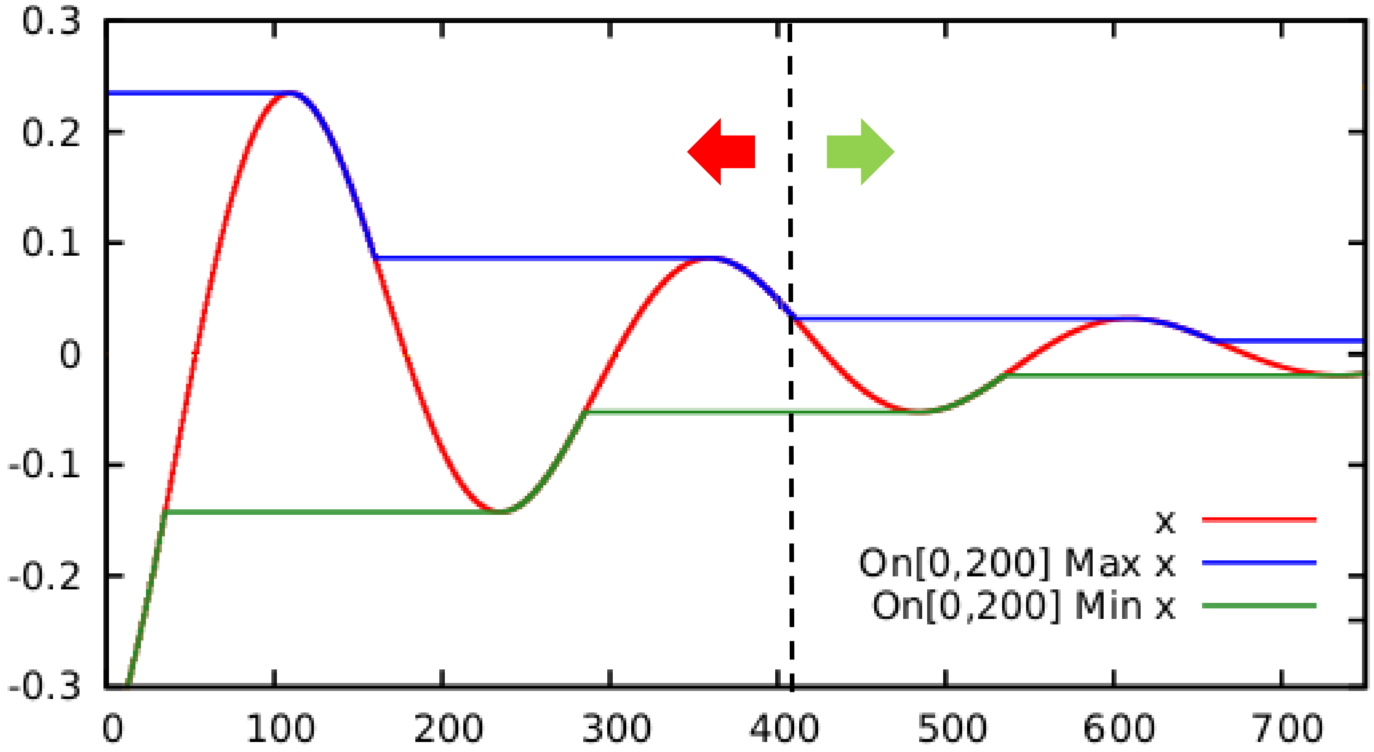
$$[\![\mathrm{Max}\,\varphi]\!][a,b] = \max_{[a,b]} [\![\varphi]\!]$$

  - New aggregation functions beyond min/max (e.g., derivative, integral)

SIGNAL STABILIZATION

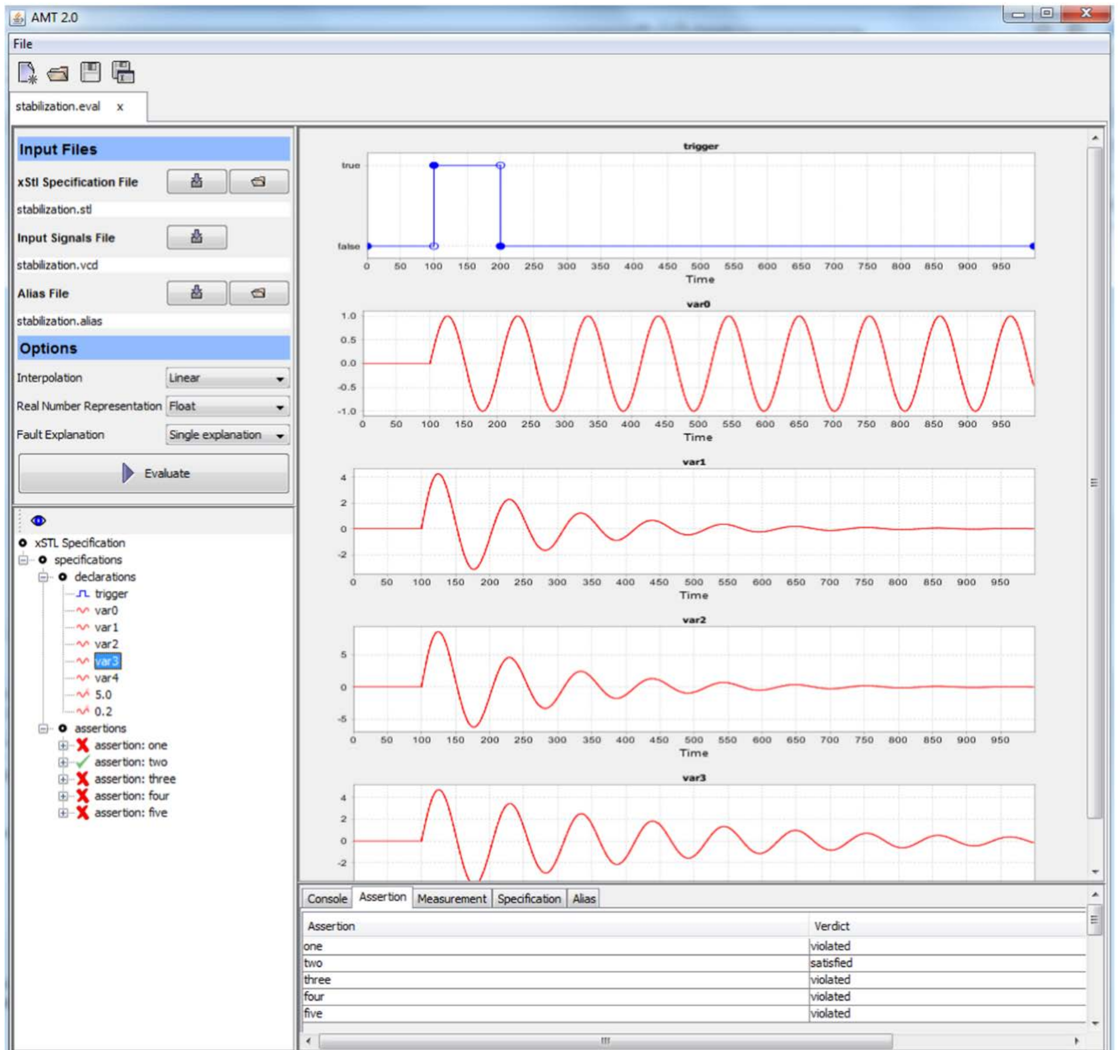
Amplitude $|x| \leq 0.1$ for 200 s

$$\text{On}_{[0,200]}\ \text{Max}\ x - \text{On}_{[0,200]}\ \text{Min}\ x \leq 0.1$$



Tools

AMT 2.0



AMT 2.0
File
stabilization.eval
Input Files
xStl Specification File
stabilization.stl
Input Signals File
stabilization.vcd
Alias File
stabilization.alias
Options
Interpolation
Linear
Real Number Representation
Float
Fault Explanation
Single explanation
Evaluate
xSTL Specification
specifications
declarations
trigger
var0
var1
var2
var3
var4
5.0
0.2
assertions
assertion: one
assertion: two
assertion: three
assertion: four
assertion: five
trigger
true
false
Time
var0
var1
var2
var3
Console
Assertion
Measurement
Specification
Alias
Assertion
Verdict
one
violated
two
satisfied
three
violated
four
violated
five
violated

AMT 2.0

AMT 2.0

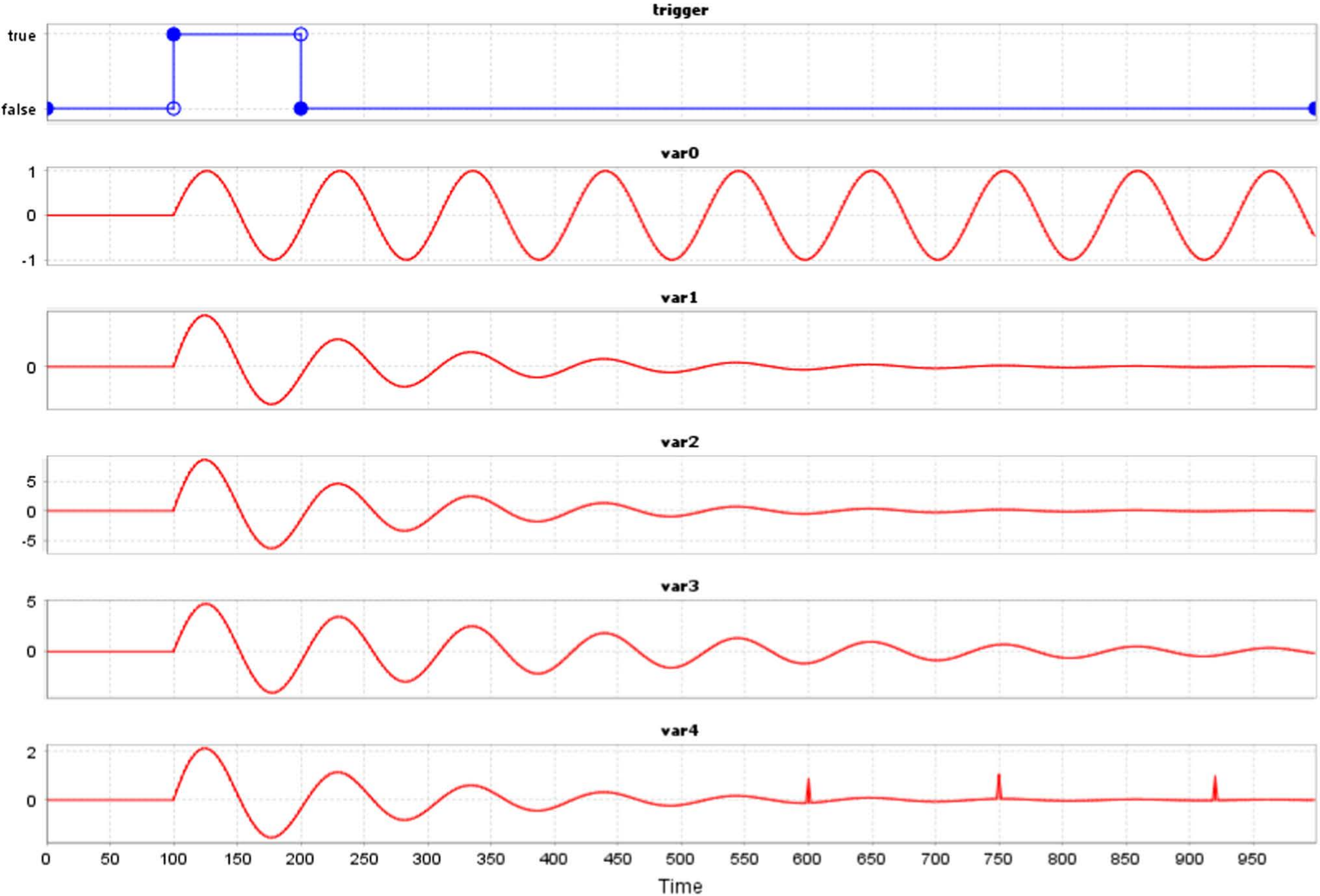
File

stabilization.stl x

```
bool trigger;
real vara;
real varb;
real varc;
real vard;
real vare;

const real vh = 5.0;
const real vl = 0.2;

assertion one:
	always((vara <= vh) and (rise(trigger) -> (eventually[0:600] always[0:300] vara <= vl)));

assertion two:
	always ((varb <= vh) and (rise(trigger) -> (eventually[0:600] always[0:300] varb <= vl)));

assertion three:
	always ((varc <= vh) and (rise(trigger) -> (eventually[0:600] always[0:300] varc <= vl)));

assertion four:
	always ((vard <= vh) and (rise(trigger) -> (eventually[0:600] always[0:300] vard <= vl)));

assertion five:
	always ((vare <= vh) and (rise(trigger) -> (eventually[0:600] always[0:300] vare <= vl)));

```

AMT 2.0

```
bool trigger;
real var0;
...
real var4;
const real vh = 5;
const real vl = 0.2;

template bool stabilization (bool tg, real x, real vhigh, real vlow) {
  bool result = ((x <= vhigh) and (rise(tg) -> (eventually[0:600] always[0:300]
   x <= vlow)));
  return result;
}

assertion one:
  always(stabilization(trigger, var0, vh, vl));
...
assertion five:
  always(stabilization(trigger, var4, vh, vl));
```



trigger
true
false
var0
1
0
-1
var1
0
var2
5
0
-5
var3
5
0
var4
2
0
0
50
100
150
200
250
300
350
400
450
500
550
600
650
700
750
800
850
900
950
Time

AMT 2.0

- **Additional features:**
  - Error diagnosis and trace counterexample generator
  - xSTL = STL + Timed Regular Expressions (TRE)
  - Quantitative analysis via pattern matching
- **STLEval:**
  - Quantitative analysis: min/max operators, ε-count

THE END