Robust Satisfaction of Temporal Logic over Real-Valued Signals

FORMATS'10

Alexandre Donzé, Oded Maler

September 8$^{\text{th}}$, 2010

Context

Model-based design and analysis of complex systems:

- Continuous and hybrid systems
- High dimensional, non-linear dynamics
- Uncertain parameters

Real life applications

- Systems biology: ODE models of complex biochemical reactions
- Emebedded system design involving continuous dynamics and discrete control logics
- Mixed-signal circuits: complex designs integrating analog and digital components

Approach

Previous work: simulation-based verification of safety properties

Numerical simulation:

- general and scalable

$+$ Sensitivity analysis:

- Influence of a parameter variation on a simulation trace
- Used for efficient approximate reachability analysis

Approach

Previous work: simulation-based verification of safety properties

Numerical simulation:

- general and scalable

+ Sensitivity analysis:

- Influence of a parameter variation on a simulation trace
- Used for efficient approximate reachability analysis

Contribution of this work: Extension to general temporal specifications

- Temporal logic formulas adapted to continuous time and space
- Quantitative (robust) monitoring
- Sensitivity of the quantitative satisfaction

Temporal logics in the wild

Model checking temporal logics have enjoyed success in formal verification and synthesis for hardware digital circuits

Most ongoing research efforts in model checking and formal verification in general aims at software

But growing interest in even scarier fields such as analog/mixed-signal circuits and systems biology

⇒ Tendency to move from discrete-time discrete systems to continuous or mixed-continuous, i.e., hybrid systems

From discrete to continuous and hybrid

Natural approach: discretize the continuous domain to obtain discrete transition system and apply favorite advanced model checker.

This ignores however the fundamental nature of the continuous domain:

There is no *a priori* ideal discretization

(except maybe for toy examples used to illustrate such techniques)

$\Rightarrow$ It is better to use temporal logics adapted to the continuous (time and space) domain

Temporal logic formulas: atomic predicates

Assume some signal, multi-dimensional function of time $w[t] = (x_0[t], \cdots, x_n[t])$

A predicate is defined as a general inequality constraint on the variables at time t with the canonical form:

$$\mu \equiv (\mu(t, w, \mathbf{p}) \geq 0) \text{ where } \mathbf{p} \text{ is some parameter vector}$$

Examples (actually valid expressions in our prototype):

```
% distance to (p0,p1) is more than 2.
 sqrt((x0[t]-p0)^2 + (x1[t]-p1)^2) >= 2.

% the system reached steady state (very slow evolution)
abs(ddt{x0}[t])+abs(ddt{x1}[t])) <= 1e-3

% x0 is sensitive to parameter p3
abs(d{x0}{p3}[t]) >= 100*x0[t]/p3
```

Temporal operators

Metric Interval Temporal Logic (MITL) syntax:

$$\varphi := \mu \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi\, \mathcal{U}_I\, \varphi$$

$+$ usual syntactic sugars for disjunction, eventually and always.

```
% x0 becomes more than -.9 whithin .5 s
ev_[0,.5] (x0[t] >= -.9)

% x0 remains low until x1 stabilizes before 10 seconds
(x0[t] < 0.1) until_[0, 10] always ((abs(ddt{x1}[t]) <= 1e-6))
```

Note: signals are finite traces interpreted as infinite signals by constant extrapolation

Outline

1. Continuous time and space syntax
2. Quantitative semantics
   - Space-robustness semantics
   - Time-robustness semantics
3. Computing the robust satisfaction
   - Sensitivity of the robustness operator

STL semantics, usual and functional notations

$$
\begin{array}{lcl}
(w,t) \models \mu & \Leftrightarrow & w \text{ satisfies } \mu \text{ at time } t \\
(w,t) \models \neg\varphi & \Leftrightarrow & (w,t) \not\models \varphi \\
(w,t) \models \varphi_1 \wedge \varphi_2 & \Leftrightarrow & (w,t) \models \varphi_1 \text{ and } (w,t) \models \varphi_2 \\
(w,t) \models \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 & \Leftrightarrow & \exists t' \in [t+a, t+b] \text{ s.t. } (w,t') \models \varphi_2 \\
 & & \text{and } \forall t'' \in [t,t'], (w,t'') \models \varphi_1
\end{array}
$$

STL semantics, usual and functional notations

$$
\begin{array}{lcl}
(w,t) \models \mu & \Leftrightarrow & w \text{ satisfies } \mu \text{ at time } t \\
(w,t) \models \neg\varphi & \Leftrightarrow & (w,t) \not\models \varphi \\
(w,t) \models \varphi_1 \wedge \varphi_2 & \Leftrightarrow & (w,t) \models \varphi_1 \text{ and } (w,t) \models \varphi_2 \\
(w,t) \models \varphi_1 \mathcal{U}_{[a,b]} \varphi_2 & \Leftrightarrow & \exists t' \in [t+a, t+b] \text{ s.t. } (w,t') \models \varphi_2 \\
 & & \text{and } \forall t'' \in [t,t'], (w,t'') \models \varphi_1
\end{array}
$$

Same semantics using satisfaction function χ

Map $\{\text{false}, \text{true}\}$ to $\{-\infty, \infty\}$ and define the function $\chi : (t,w) \to \{-\infty, \infty\}$:

$$
\begin{array}{lcl}
\chi(\mu, w, t) & = & \text{sign}(\mu(t, w, \mathbf{p})) \times \infty \\
\chi(\neg\varphi, w, t) & = & -\chi(\varphi, w, t) \\
\chi(\varphi_1 \wedge \varphi_2, w, t) & = & \min(\chi(\varphi_1, w, t), \chi(\varphi_2, w, t)) \\
\chi(\varphi_1 \mathcal{U}_{[a,b]} \varphi_2, w, t) & = & \max\limits_{\tau \in t+[a,b]} (\min(\chi(\varphi_2, w, \tau), \min\limits_{s \in [t,\tau]} \chi(\varphi_1, w, s))
\end{array}
$$

It is easy to verify that $w, t \models \varphi \Leftrightarrow \chi(\varphi, w, t) = +\infty$

Space-robustness semantics

Let us look at the base case:

$$\chi(\mu, w, t) = \mathsf{sign}(\mu(t, w, \mathbf{p})) \times \infty$$

The sign function abstracts away all the good quantitative information provided by the μ function to keep only a boolean signals.

Space-robustness semantics

Let us look at the base case:

$$\rho_{\cancel{\chi}}(\mu, w, t) = \cancel{\mathsf{sign}}(\mu(t, w, \mathbf{p})) \cancel{\times \infty}$$

The sign function abstracts away all the good quantitative information provided by the μ function to keep only a boolean signals.

Our main contribution (slight extension of G. Fainekos and G. Pappas work): get rid of the sign function to get a quantitative satisfaction function ρ !

Space-robustness semantics

Let us look at the base case:

$$^{\rho}\cancel{\chi}(\mu, w, t) = \cancel{\mathsf{sign}}(\mu(t, w, \mathbf{p})) \cancel{\times \infty}$$

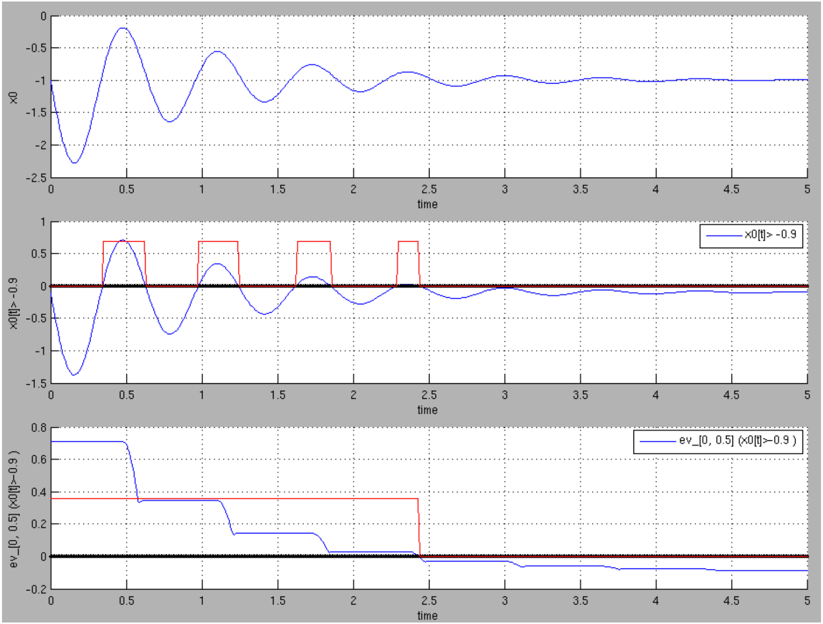
The sign function abstracts away all the good quantitative information provided by the μ function to keep only a boolean signals.

Our main contribution (slight extension of G. Fainekos and G. Pappas work): get rid of the sign function to get a quantitative satisfaction function ρ !

We keep the same inductive rules for the definition of the semantics:

$$\begin{array}{lcl}
\rho(\mu, w, t) & = & \mu(t, w, \mathbf{p}) \\
\rho(\neg\varphi, w, t) & = & -\rho(\varphi, w, t) \\
\rho(\varphi_1 \wedge \varphi_2, w, t) & = & \min(\rho(\varphi_1, w, t), \rho(\varphi_2, w, t)) \\
\rho(\varphi_1 \mathcal{U}_{[a,b]} \varphi_2, w, t) & = & \max\limits_{\tau \in t+[a,b]} (\min(\rho(\varphi_2, w, \tau), \min\limits_{s \in [t,\tau]} \rho(\varphi_1, w, s))
\end{array}$$

A simple first example



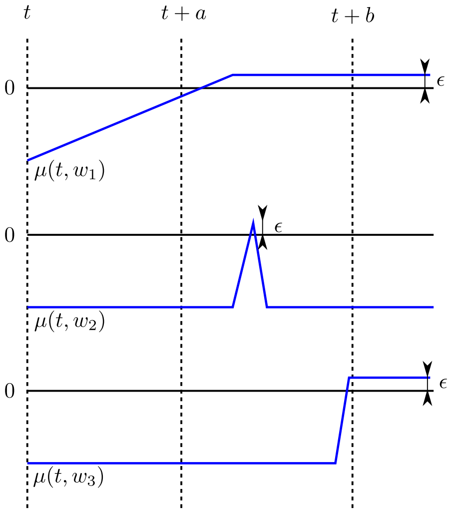
Limitations of space-robustness semantics

Let $\varphi = \mathsf{ev}_{[a,b]}(\mu)$ and consider the three signals w_1, w_2 and w_3.

We have $\rho(\varphi, w_1, t) = \rho(\varphi, w_2, t) = \rho(\varphi, w_3, t) = \epsilon$

whereas somehow intuitively:

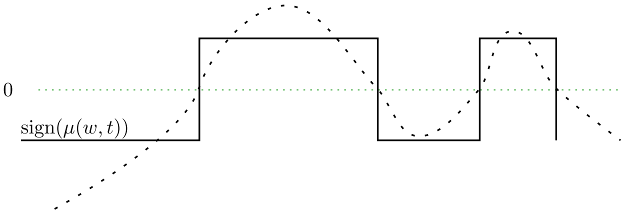
Robustness$((w_1, t) \models \varphi)$
$>$ Robustness$((w_2, t) \models \varphi)$
$>$ Robustness$((w_3, t) \models \varphi)$.



Time-robustness semantics

We define left and right time robustness θ^+ and θ^- of a predicate μ as

1. $\text{sign}(\theta^-(\mu, w, t)) = \text{sign}(\theta^+(\mu, w, t)) = \text{sign}(\rho(\mu, w, t))$ (same bool. sem.)

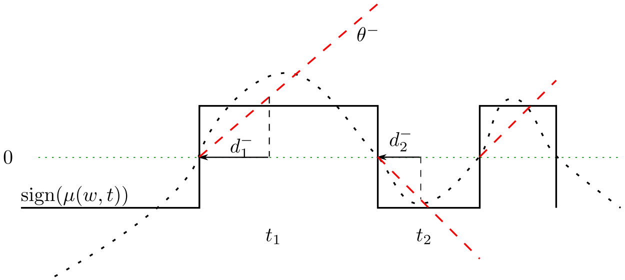


Example

Time-robustness semantics

We define left and right time robustness θ^+ and θ^- of a predicate μ as

1. $\mathsf{sign}(\theta^-(\mu, w, t)) = \mathsf{sign}(\theta^+(\mu, w, t)) = \mathsf{sign}(\rho(\mu, w, t))$ (same bool. sem.)
2. $|\theta^-(\mu, w, t)| =$ minimum d such that $\mathsf{sign}(\rho(\mu, w, t - d)) \neq \mathsf{sign}(\rho(\mu, w, t))$



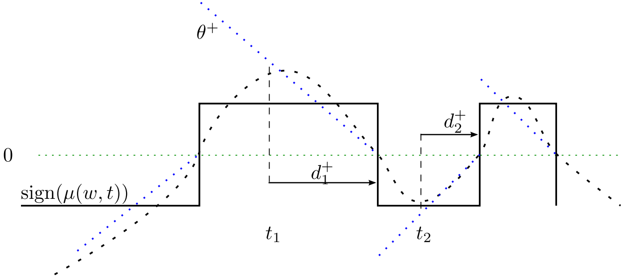
Example

$\theta^-(\mu, w, t_1) = d_1^-$
$\theta^-(\mu, w, t_2) = -d_2^-$

Time-robustness semantics

We define left and right time robustness θ^+ and θ^- of a predicate μ as

1. $\text{sign}(\theta^-(\mu, w, t)) = \text{sign}(\theta^+(\mu, w, t)) = \text{sign}(\rho(\mu, w, t))$ (same bool. sem.)
2. $|\theta^-(\mu, w, t)| =$ minimum d such that $\text{sign}(\rho(\mu, w, t - d)) \neq \text{sign}(\rho(\mu, w, t))$
3. $|\theta^+(\mu, w, t)| =$ minimum d such that $\text{sign}(\rho(\mu, w, t + d)) \neq \text{sign}(\rho(\mu, w, t))$



Example

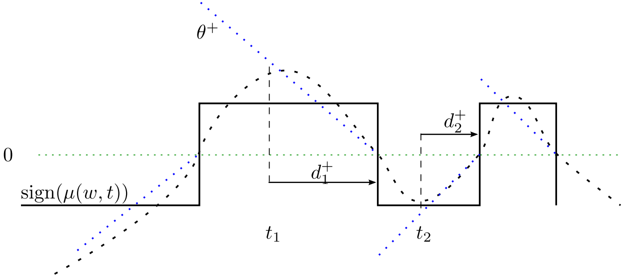
$$\theta^-(\mu, w, t_1) = d_1^-$$
$$\theta^-(\mu, w, t_2) = -d_2^-$$

$$\theta^+(\mu, w, t_1) = d_1^+$$
$$\theta^+(\mu, w, t_2) = -d_2^+$$

Time-robustness semantics

We define left and right time robustness θ^+ and θ^- of a predicate μ as

1. $\text{sign}(\theta^-(\mu, w, t)) = \text{sign}(\theta^+(\mu, w, t)) = \text{sign}(\rho(\mu, w, t))$ (same bool. sem.)
2. $|\theta^-(\mu, w, t)| =$ minimum d such that $\text{sign}(\rho(\mu, w, t - d)) \neq \text{sign}(\rho(\mu, w, t))$
3. $|\theta^+(\mu, w, t)| =$ minimum d such that $\text{sign}(\rho(\mu, w, t + d)) \neq \text{sign}(\rho(\mu, w, t))$



Example

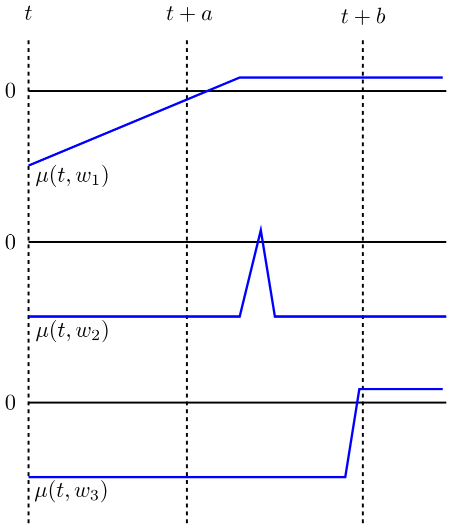
$$\theta^-(\mu, w, t_1) = d_1^-$$
$$\theta^-(\mu, w, t_2) = -d_2^-$$

$$\theta^+(\mu, w, t_1) = d_1^+$$
$$\theta^+(\mu, w, t_2) = -d_2^+$$

For a formula φ, θ^- and θ^+ are constructed using the same $\min - \max$ inductive rules as for ρ.

Time-robustness semantics example

Consider again $\varphi = \mathsf{ev}_{[a,b]}(\mu)$

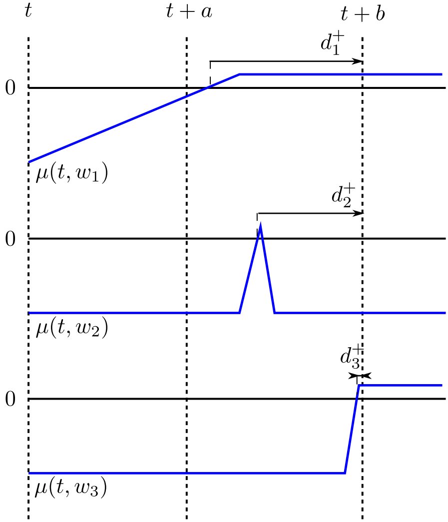


Time-robustness semantics example

Consider again $\varphi = \mathsf{ev}_{[a,b]}(\mu)$

We can now find that

$$\begin{aligned}\theta^+(\varphi, w_1, 0) = d_1^+ > \theta^+(\varphi, w_2, 0) = d_2^+ \\ > \ \theta^+(\varphi, w_3, 0) = d_3^+\end{aligned}$$



Time-robustness semantics example

Consider again $\varphi = \mathsf{ev}_{[a,b]}(\mu)$

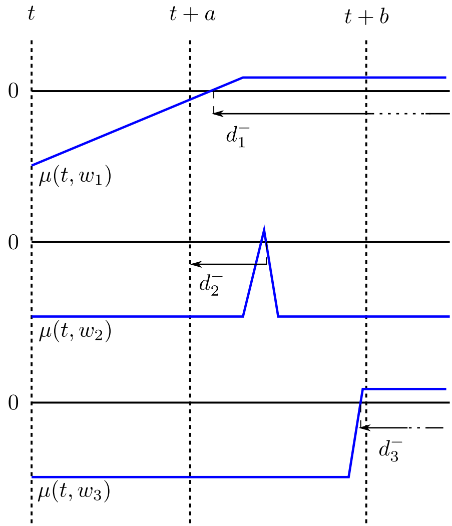
We can now find that

$$\theta^+(\varphi, w_1, 0) = d_1^+ > \theta^+(\varphi, w_2, 0) = d_2^+$$
$$> \theta^+(\varphi, w_3, 0) = d_3^+$$

Note that the left and right robustness can give different results:

Depending on the (hidden) future of the signals we might as well have:

$$d_3^+ > d_1^+ > d_2^-$$



Properties of the quantitative semantics

1. Robust semantics are sound with the boolean (STL) semantics

$$(w,t) \models \varphi \Leftrightarrow \rho(w,t,\varphi) \geq 0 \Leftrightarrow \theta^{-}(w,t,\varphi) \geq 0 \Leftrightarrow \theta^{+}(w,t,\varphi) \geq 0$$

Properties of the quantitative semantics

1. Robust semantics are sound with the boolean (STL) semantics

$$(w,t) \models \varphi \Leftrightarrow \rho(w,t,\varphi) \geq 0 \Leftrightarrow \theta^{-}(w,t,\varphi) \geq 0 \Leftrightarrow \theta^{+}(w,t,\varphi) \geq 0$$

2. Robust semantics provide neighborhoods of w such that any signal w' in it share the same boolean satisfaction of φ as w,i.e.:

Robustness interpretation

Let $d_\chi = \chi(\varphi, w, t)$, $d_\rho = \rho(\varphi, w, t)$, $d_{\theta^+} = \theta^+(\varphi, w, t)$ and w' a signal. If one of these conditions hold:

1. $|\mu(t, w') - \mu(t, w)| < d_\rho$
2. $|\mu(t, w) = \mu(t - d, w)|$ with $d < d_{\theta^+}$

for all t and all μ appearing in φ, then $\chi(\varphi, w, t) = d_\chi$.

Properties of the quantitative semantics

1. Robust semantics are sound with the boolean (STL) semantics

$$(w,t) \models \varphi \Leftrightarrow \rho(w,t,\varphi) \geq 0 \Leftrightarrow \theta^-(w,t,\varphi) \geq 0 \Leftrightarrow \theta^+(w,t,\varphi) \geq 0$$

2. Robust semantics provide neighborhoods of w such that any signal w' in it share the same boolean satisfaction of φ as w,i.e.:

Robustness interpretation

Let $d_\chi = \chi(\varphi, w, t)$, $d_\rho = \rho(\varphi, w, t)$, $d_{\theta+} = \theta^+(\varphi, w, t)$ and w' a signal. If one of these conditions hold:

1. $|\mu(t, w') - \mu(t, w)| < d_\rho$
2. $|\mu(t, w) = \mu(t - d, w)|$ with $d < d_{\theta+}$

for all t and all μ appearing in φ, then $\chi(\varphi, w, t) = d_\chi$.

Space and time semantics can be combined to get custom notions of robustness.

Other derived semantic can also be invented...

Difference with other approaches

Main difference is the definition of atomic predicates μ:

- In the framework of Fainekos and Pappas, μ is identified to $\mathcal{O}(\mu)$,the region where μ holds and robustness is given by the signed distance of w to $\mathcal{O}(\mu)$.
- In our framework, we more directly identify μ to its (quantitative) satisfaction function.

Difference with other approaches

Main difference is the definition of atomic predicates μ:

- In the framework of Fainekos and Pappas, μ is identified to $\mathcal{O}(\mu)$,the region where μ holds and robustness is given by the signed distance of w to $\mathcal{O}(\mu)$.
- In our framework, we more directly identify μ to its (quantitative) satisfaction function.

The result is more flexibility:

- The satisfaction function can be a signed distance (but doesn't have to be, and doesn't have to be a distance at all)
- Predicates do not all depend on the choice of one distance.

  E.g., they can be scaled w.r.t. one another: Changing $\mu(t, w)$ to $\lambda\mu(t, w)$ in φ may change the relative importance of μ in $\rho(\varphi, w, t)$ w.r.t. other predicates without changing the boolean satisfaction.

Outline

1. Continuous time and space syntax
2. Quantitative semantics
   - Space-robustness semantics
   - Time-robustness semantics
3. Computing the robust satisfaction
   - Sensitivity of the robustness operator

Computing the robust satisfaction

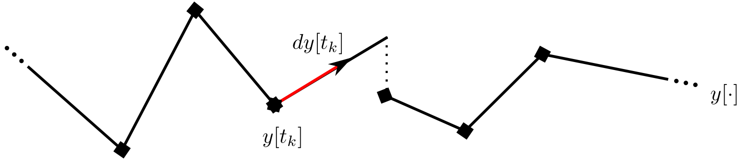
We compute the function $\rho(\varphi, w, \cdot)$ by induction on the structure of φ.

This reduces to three subproblems. Given two functions $y, y' : \mathbb{T} \to \mathbb{R}$,

1. (operator $\neg$) compute $z : \mathbb{T} \to \mathbb{R}$ such that $\forall t \in \mathbb{T},\ z[t] = -y[t]$;
2. (operator $\wedge$) compute $z : \mathbb{T} \to \mathbb{R}$ such that
$$\forall t \in \mathbb{T},\ z[t] = \min(y[t], y'[t])$$
3. (operator $\mathcal{U}$) given an interval $[a, b]$, compute $z : \mathbb{T} \to \mathbb{R}$ such that
$$\forall\ t \in \mathbb{T},\ z[t] = \max_{\tau \in t+[a,b]} (\min(y'[\tau], \min_{s \in [t,\tau]} y[s])))$$

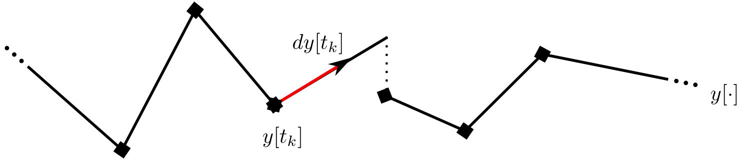
1. and 2. are reasonably trivials.3. is not (maybe for a $\max - \min$ guru).
We consider piecewise-affine signals.

Computational scheme for piecewise-affine signals



y is entirely described by the sequence $(t_k, y[t_k], dy[t_k])$. An event is a time instant where (y, dy) is discontinuous.

Computational scheme for piecewise-affine signals

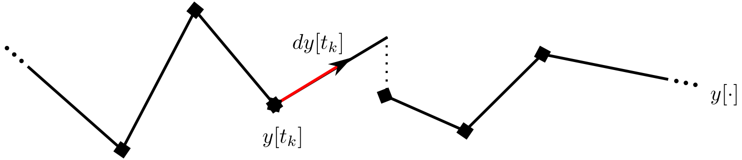


y is entirely described by the sequence $(t_k, y[t_k], dy[t_k])$. An event is a time instant where (y, dy) is discontinuous.

We compute this sequence forward in time using an event detection mechanism:

1: **Init** r_1, $k = 1$
2: **Repeat**
3: Compute $(z[r_k], dz[r_k])$ from (y, y')
4: Compute $r_{k+1} = \mathsf{NextEvent}(z, r_k)$ from (y, y')
5: Let $k = k + 1$
6: **Until** $r_k = \infty$

Computational scheme for piecewise-affine signals



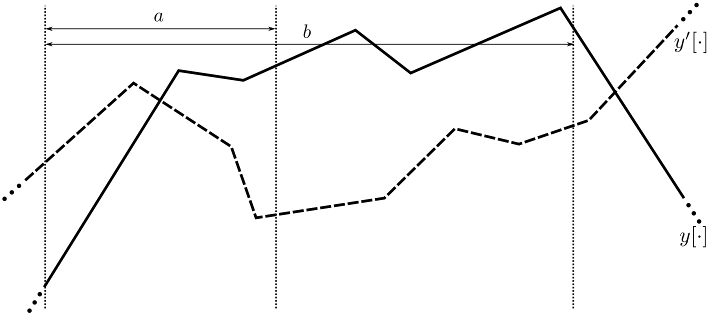
y is entirely described by the sequence $(t_k, y[t_k], dy[t_k])$. An event is a time instant where (y, dy) is discontinuous.

We compute this sequence forward in time using an event detection mechanism:

1: **Init** r_1, $k = 1$
2: **Repeat**
3: Compute $(z[r_k], dz[r_k])$ from (y, y')
4: Compute $r_{k+1} = \text{NextEvent}(z, r_k)$ from (y, y')
5: Let $k = k + 1$
6: **Until** $r_k = \infty$

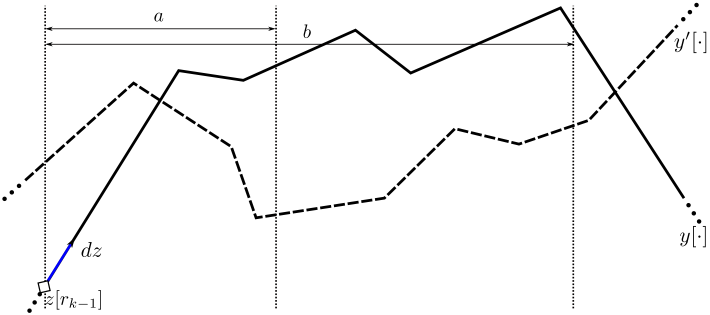
Computing the NextEvent function efficiently is tricky, we don't go into the details.(Illustration follows)

Until computation illustration



Recall that $z[t] = \max_{\tau \in [t+a, t+b]} \min(y'[\tau], \min_{s \in [t,\tau]} y[s])$

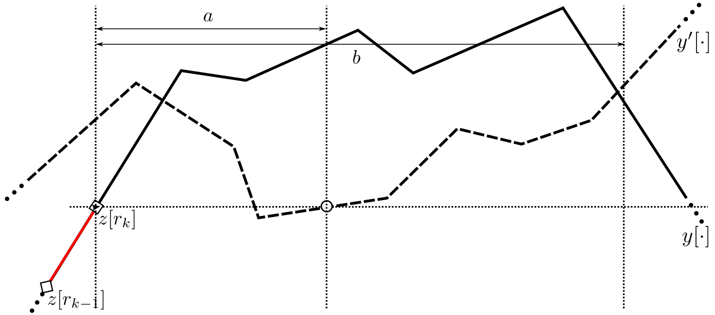
Until computation illustration



Recall that $z[t] = \max_{\tau \in [t+a, t+b]} \min(y'[\tau], \min_{s \in [t,\tau]} y[s])$

- $z[r_{k-1}]$ and $dz[r_{k-1}]$ are given by y

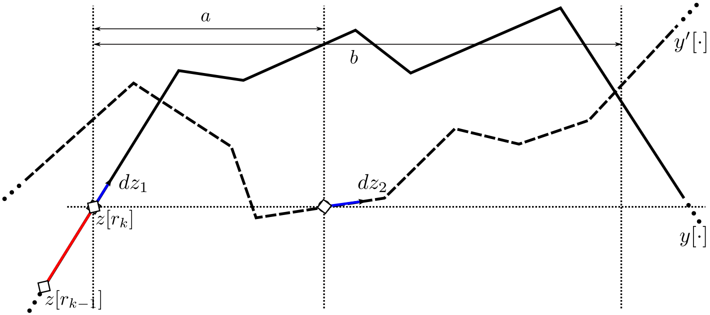
Until computation illustration



Recall that $z[t] = \max_{\tau \in [t+a, t+b]} \min(y'[\tau], \min_{s \in [t,\tau]} y[s])$

- $z[r_{k-1}]$ and $dz[r_{k-1}]$ are given by y
- at this point a value of y' can become the new value for z

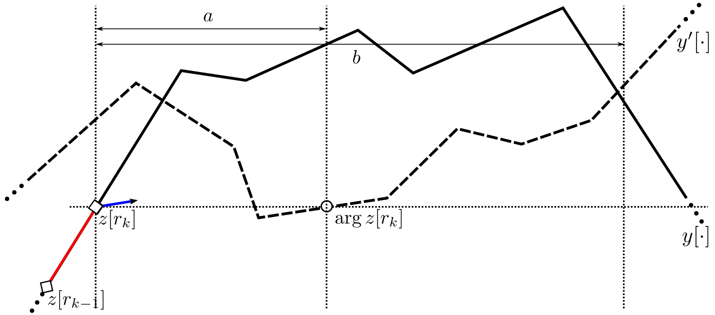
Until computation illustration



Recall that $z[t] = \max_{\tau \in [t+a, t+b]} \min(y'[\tau], \min_{s \in [t,\tau]} y[s])$

- $z[r_{k-1}]$ and $dz[r_{k-1}]$ are given by y
- at this point a value of y' can become the new value for z
- we proceed with the smallest slope according to the $\min$

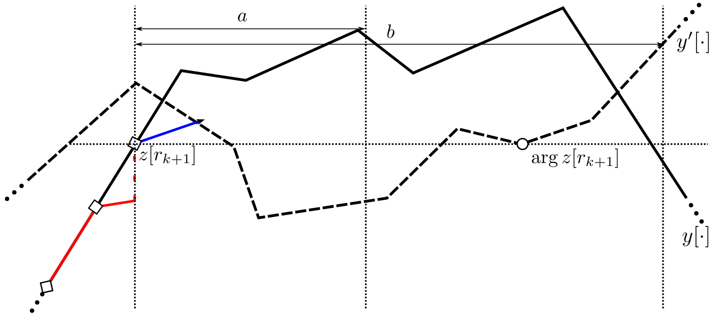
Until computation illustration



Recall that $z[t] = \max_{\tau \in [t+a, t+b]} \min(y'[\tau], \min_{s \in [t,\tau]} y[s])$

- $z[r_{k-1}]$ and $dz[r_{k-1}]$ are given by y
- at this point a value of y' can become the new value for z
- we proceed with the smallest slope according to the $\min$

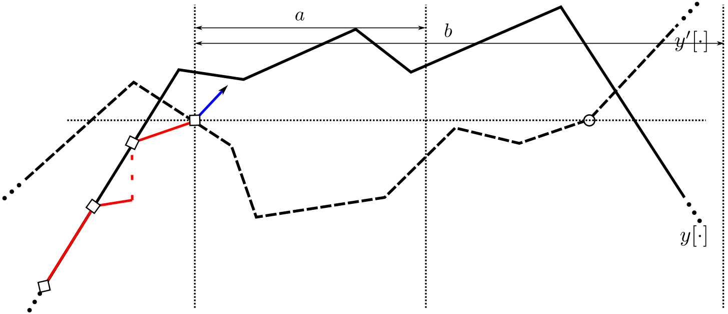
Until computation illustration



Recall that $z[t] = \max_{\tau \in [t+a, t+b]} \min(y'[\tau], \min_{s \in [t,\tau]} y[s])$

- $z[r_{k-1}]$ and $dz[r_{k-1}]$ are given by y
- at this point a value of y' can become the new value for z
- we proceed with the smallest slope according to the $\min$
- there, another value of y' can pass the current z value because of the $\max$

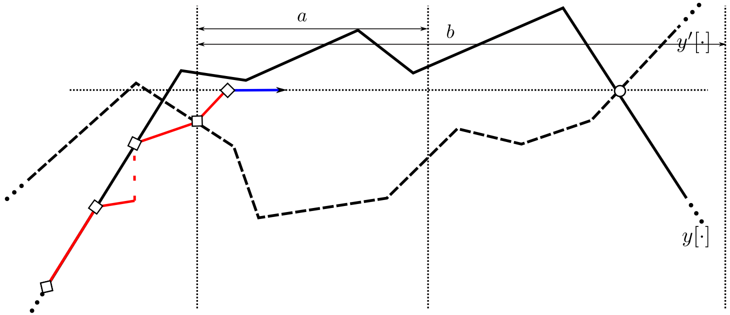
Until computation illustration



Recall that $z[t] = \max_{\tau \in [t+a, t+b]} \min(y'[\tau], \min_{s \in [t,\tau]} y[s])$

- $z[r_{k-1}]$ and $dz[r_{k-1}]$ are given by y
- at this point a value of y' can become the new value for z
- we proceed with the smallest slope according to the $\min$
- there, another value of y' can pass the current z value because of the $\max$

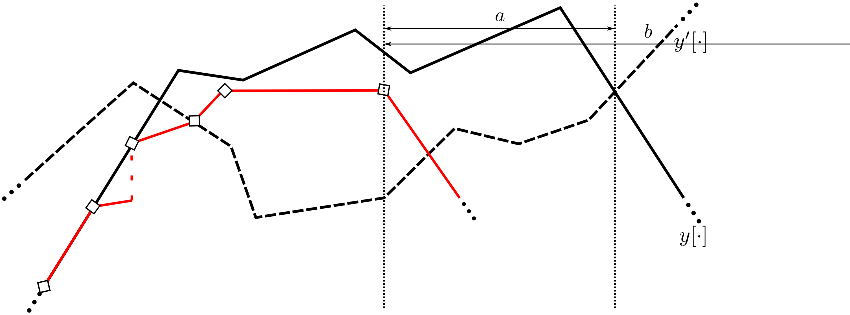
Until computation illustration



Recall that $z[t] = \max_{\tau \in [t+a, t+b]} \min(y'[\tau], \min_{s \in [t,\tau]} y[s])$

- $z[r_{k-1}]$ and $dz[r_{k-1}]$ are given by y
- at this point a value of y' can become the new value for z
- we proceed with the smallest slope according to the $\min$
- there, another value of y' can pass the current z value because of the $\max$

Until computation illustration

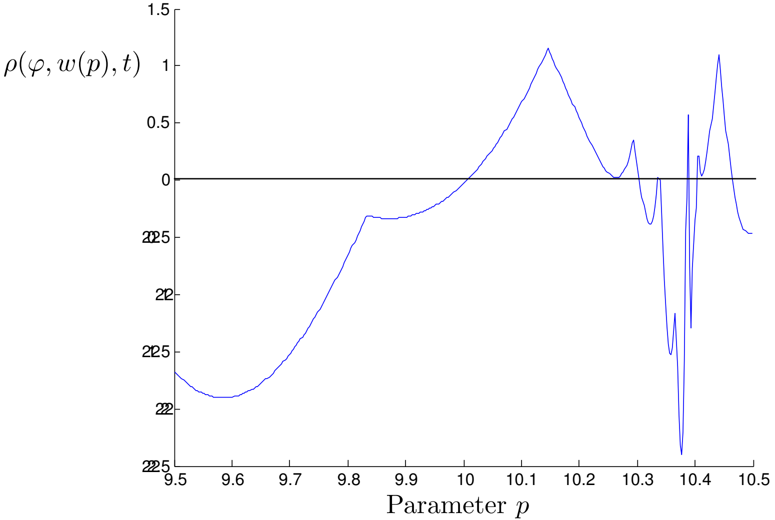


Recall that $z[t] = \max_{\tau \in [t+a, t+b]} \min(y'[\tau], \min_{s \in [t,\tau]} y[s])$

- $z[r_{k-1}]$ and $dz[r_{k-1}]$ are given by y
- at this point a value of y' can become the new value for z
- we proceed with the smallest slope according to the $\min$
- there, another value of y' can pass the current z value because of the $\max$

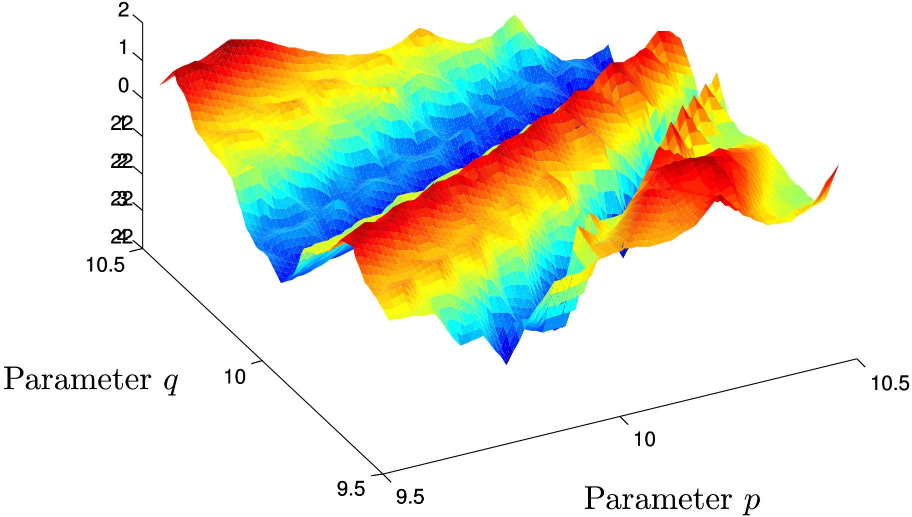
Motivations

$$\varphi := \text{ev} \; (\; \text{alw} \; (\; (|x_0[t,p]| < 1 \;) \wedge (|x_1[t,p]| < 1) \wedge (|x_2[t,p]| < 1) \;)$$

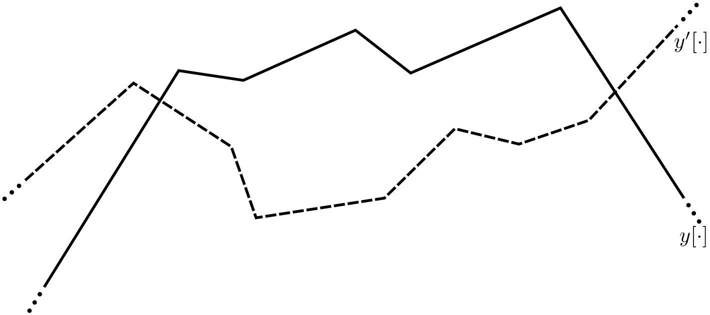


Motivations

$\rho(\varphi, w(p,q), t)$

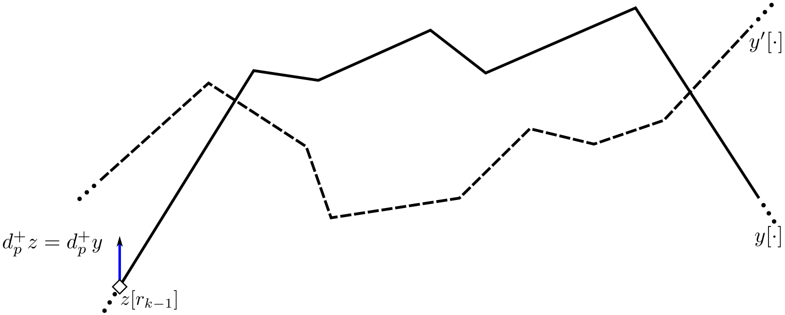


Going sensitive



The mechanism to compute dz can be used to compute the (right-)derivative of z w.r.t. any parameter p.
Assume that y and y' have such a derivative $d^+_{\mathbf{p}}y$ and $d^+_{\mathbf{p}}y'$

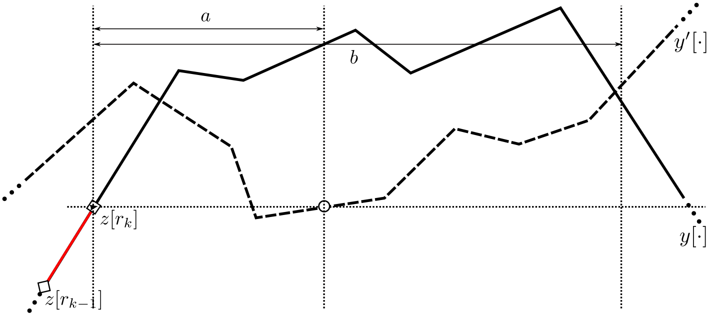
Going sensitive



The mechanism to compute dz can be used to compute the (right-)derivative of z w.r.t. any parameter p.
Assume that y and y' have such a derivative $d^+_{\mathbf{p}}\, y$ and $d^+_{\mathbf{p}}\, y'$

- when z has a unique origin, the derivative is inherited from it

Going sensitive

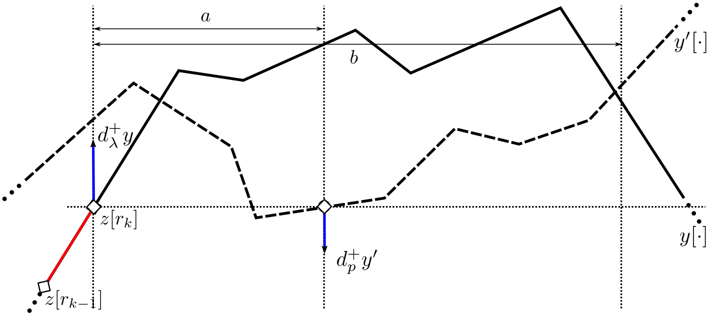


The mechanism to compute dz can be used to compute the (right-)derivative of z w.r.t. any parameter p.

Assume that y and y' have such a derivative $d^+_{\mathbf{p}}y$ and $d^+_{\mathbf{p}}y'$

- when z has a unique origin, the derivative is inherited from it
- otherwise, we observe that the operators $\min$ and $\max$ are distributive w.r.t. the right derivative: $d^+_{\mathbf{p}}(\min(y, y')) = \min(d^+_{\mathbf{p}}(y), d^+_{\mathbf{p}}(y'))$

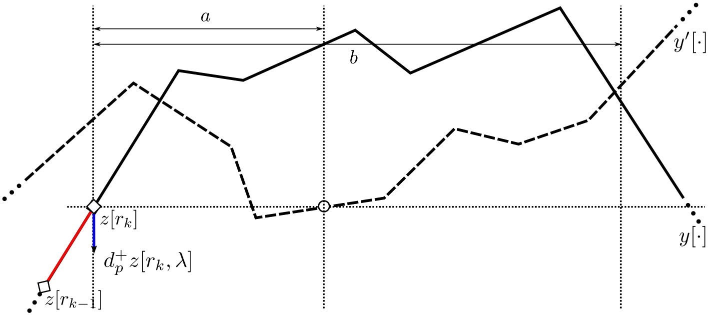
Going sensitive



The mechanism to compute dz can be used to compute the (right-)derivative of z w.r.t. any parameter p.
Assume that y and y' have such a derivative $d^+_{\mathbf{p}} y$ and $d^+_{\mathbf{p}} y'$

- when z has a unique origin, the derivative is inherited from it
- otherwise, we observe that the operators $\min$ and $\max$ are distributive w.r.t. the right derivative: $d^+_{\mathbf{p}}(\min(y, y')) = \min(d^+_{\mathbf{p}}(y), d^+_{\mathbf{p}}(y'))$

Going sensitive



The mechanism to compute dz can be used to compute the (right-)derivative of z w.r.t. any parameter p.
Assume that y and y' have such a derivative $d_{\mathbf{p}}^+ y$ and $d_{\mathbf{p}}^+ y'$

- when z has a unique origin, the derivative is inherited from it
- otherwise, we observe that the operators $\min$ and $\max$ are distributive w.r.t. the right derivative: $d_{\mathbf{p}}^+(\min(y, y')) = \min(d_{\mathbf{p}}^+(y), d_{\mathbf{p}}^+(y'))$

Conclusion

We presented

- a language to specify arbitrarily complex temporal properties adapted to continuous-time and real-valued signals and
- a tool to explore the space of possible behaviors of a system against such properties

Future work