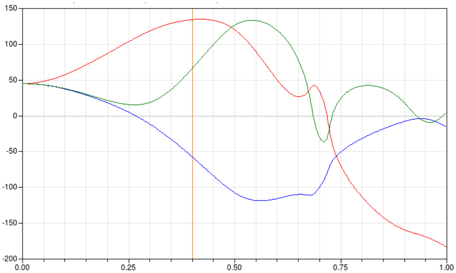
Time Domains in Hybrid Systems Modeling

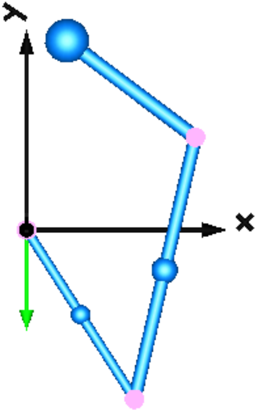
Albert Benveniste[1] Timothy Bourke[1] Benoît Caillaud[1]
Marc Pouzet[3]

1. INRIA
3. École normale supérieure

Lund, May 2015

From Dynamical to Hybrid Systems, informally





Dynamical system: smooth dynamics

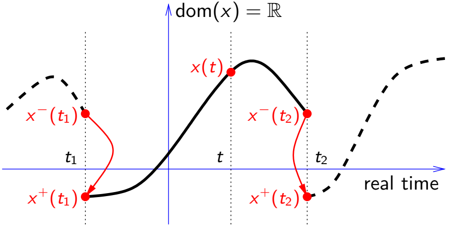
$$x : \mathbb{R} \to \mathbb{R}^n$$

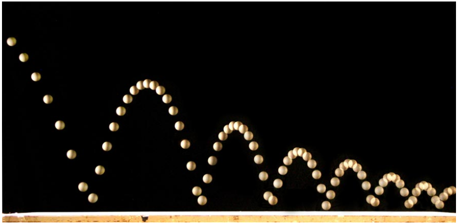
solution of the IVP

$$\begin{cases} f(\dot{x}, x, t) = 0 \\ x(t_0) = x_0 \end{cases}$$

Can we capture Hybrid Systems trajectories as $x : \mathbb{R} \to \mathbb{R}^n$?

From Dynamical to Hybrid Systems, informally



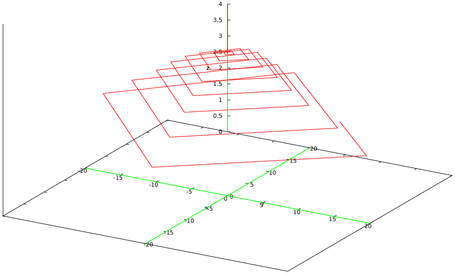


Simple Hybrid Systems: smooth dynamics almost all the time, except for state jumps $x^+ = g(x^-)$ at some discrete t.

$x : \mathbb{R} \to \mathbb{R}^n$ still works.

How general is this?

From Dynamical to Hybrid Systems, informally



$$\begin{cases} \dot{x} &= -\mathrm{sgn}(x) + 2\mathrm{sgn}(y) \\ \dot{y} &= -2\mathrm{sgn}(x) - \mathrm{sgn}(y) \\ \dot{z} &= \mathrm{sgn}(x) + \mathrm{sgn}(y) \end{cases}$$

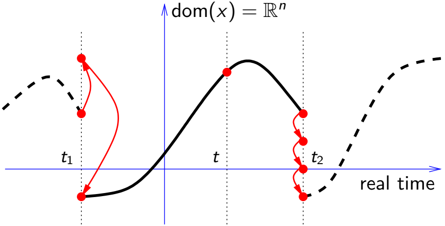
Non-Smooth Dynamical Systems: right-hand of differential equations is non-smooth.

- Filippov Differential Inclusions
- Complementarity Systems

$x : \mathbb{R} \to \mathbb{R}^n$ still works.

However...

From Dynamical to Hybrid Systems, informally



In general, Hybrid Systems trajectory may have:

- Instantaneous cascades of state jumps
- Chattering

Can not be captured as:

$$x : \mathbb{R} \to \mathbb{R}^n$$

Need a Time Domain "denser" than $\mathbb{R}$

Semantics of Hybrid Systems Modelers

Instrumental to design:

1. Static analyzers / model-checkers / theories for interactive provers
2. Compile-time analysis / simulation code generation
3. Numerical simulation environments (run-time)

Need for a precise mathematical semantics

Focus of this talk:

- Comparison of Time Domains used to the define the semantics of hybrid systems modelers
- Emphasis on compile-time analysis / simulation code generation

Background: Synchronous Languages

Syntax of a simple synchronous language ($\approx$ Lustre)

$$
\begin{aligned}
d &::= \texttt{let}\ x = e \mid \texttt{let}\ f(p) = e\ \texttt{where}\ E \mid d; d \\
e &::= x \mid v \mid op(e) \mid e\ \texttt{fby}\ e \mid \texttt{pre}(e) \mid f(e) \mid (e, e) \\
p &::= (p, p) \mid x \\
E &::= () \mid E\ \texttt{and}\ E \mid x\ =\ e \mid \\
&\quad\ \mid \texttt{if}\ e\ \texttt{then}\ E\ \texttt{else}\ E
\end{aligned}
$$

Examples

```
let min_max(x,y) = (a,b) where
  if x<y
  then a = x and b = y
  else a = y and b = x
```

```
let sum(x) = cpt where
  cpt = (0 fby pre(cpt)) + x
```

Background: Semantics of Synhronous Languages

Chronograms

time	=	0	1	2	3	4	5
x	=	2	4	2	1	2	3
y	=	3	6	5	1	1	9
$min\_max(x, y)$	=	$(2, 3)$	$(4, 6)$	$(2, 5)$	$(1, 1)$	$(1, 2)$	$(3, 9)$
$\texttt{pre}(x)$	=	*nil*	2	4	2	1	2
$x\,\texttt{fby}\,y$	=	2	6	5	1	1	9
$sum(x)$	=	2	6	8	9	11	14

Examples

```
let min_max(x,y) = (a,b) where
  if x<y
  then a = x and b = y
  else a = y and b = x
```

```
let sum(x) = cpt where
  cpt = (0 fby pre(cpt)) + x
```

Background: Synchronous Languages

Chronograms

time	=	0	1	2	3	4	5
x	=	2	4	2	1	2	3
y	=	3	6	5	1	1	9
$min\_max(x, y)$	=	$(2, 3)$	$(4, 6)$	$(2, 5)$	$(1, 1)$	$(1, 2)$	$(3, 9)$
$\texttt{pre}(x)$	=	*nil*	2	4	2	1	2
$x\ \texttt{fby}\ y$	=	2	6	5	1	1	9
$sum(x)$	=	2	6	8	9	11	14

Main features

- A signal is a sequence of values or stream
- A system is function from streams to streams.
- Operations apply pointwise to their arguments.
- All streams progress synchronously.

Background: Constructive Fixpoint Semantics

Define semantics as mutual least fixpoint of set of monotonous operators (one for each equation) [Berry 1999]

Step-by-step execution

time	=	0	1	2	3	4	5
x	=	2	4	2	$\perp$	$\perp$	$\perp$
y	=	*nil*	2	6	$\perp$	$\perp$	$\perp$
z	=	0	2	6	$\perp$	$\perp$	$\perp$
cpt	=	2	6	8	$\perp$	$\perp$	$\perp$

Program

```
let sum(x) = cpt where
   y = pre(cpt)
and z = 0 fby y
and cpt = z + x
```

Extended domains and streams

- $t \in \mathbb{N}$ dicrete time
- $v \in V \uplus \{\perp\}$: $\perp$ if undefined, $\perp < v \in V$
- $S(V) = \mathbb{N} \to (V \uplus \{\perp\})$

Background: Constructive Fixpoint Semantics

Define semantics as mutual least fixpoint of set of monotonous operators (one for each equation) [Berry 1999]

Step-by-step execution

$time$	$=$	0	1	2	3	4	5
x	$=$	2	4	2	1	$\perp$	$\perp$
y	$=$	nil	2	6	8	$\perp$	$\perp$
z	$=$	0	2	6	$\perp$	$\perp$	$\perp$
cpt	$=$	2	6	8	$\perp$	$\perp$	$\perp$

Program

```
let sum(x) = cpt where
    y = pre(cpt)
and z = 0 fby y
and cpt = z + x
```

Extended domains and streams

- $t \in \mathbb{N}$ dicrete time
- $v \in V \uplus \{\perp\}$: $\perp$ if undefined, $\perp < v \in V$
- $S(V) = \mathbb{N} \to (V \uplus \{\perp\})$

Background: Constructive Fixpoint Semantics

Define semantics as mutual least fixpoint of set of monotonous operators (one for each equation) [Berry 1999]

Step-by-step execution

time	=	0	1	2	3	4	5
x	=	2	4	2	1	$\perp$	$\perp$
y	=	*nil*	2	6	8	$\perp$	$\perp$
z	=	0	2	6	8	$\perp$	$\perp$
cpt	=	2	6	8	$\perp$	$\perp$	$\perp$

Program

```
let sum(x) = cpt where
    y = pre(cpt)
and z = 0 fby y
and cpt = z + x
```

Extended domains and streams

- $t \in \mathbb{N}$ dicrete time
- $v \in V \uplus \{\perp\}$: $\perp$ if undefined, $\perp < v \in V$
- $S(V) = \mathbb{N} \to (V \uplus \{\perp\})$

Background: Constructive Fixpoint Semantics

Define semantics as mutual least fixpoint of set of monotonous operators (one for each equation) [Berry 1999]

Step-by-step execution

$time$	$=$	0	1	2	3	4	5
x	$=$	2	4	2	1	$\perp$	$\perp$
y	$=$	nil	2	6	8	$\perp$	$\perp$
z	$=$	0	2	6	8	$\perp$	$\perp$
cpt	$=$	2	6	8	9	$\perp$	$\perp$

Program

```
let sum(x) = cpt where
    y = pre(cpt)
and z = 0 fby y
and cpt = z + x
```

Extended domains and streams

- $t \in \mathbb{N}$ dicrete time
- $v \in V \uplus \{\perp\}$: $\perp$ if undefined, $\perp < v \in V$
- $S(V) = \mathbb{N} \to (V \uplus \{\perp\})$

Background: Constructive Fixpoint Semantics

Define semantics as mutual least fixpoint of set of monotonous operators (one for each equation) [Berry 1999]

Step-by-step execution

time	=	0	1	2	3	4	5
x	=	2	4	2	1	2	$\perp$
y	=	*nil*	2	6	8	9	$\perp$
z	=	0	2	6	8	$\perp$	$\perp$
cpt	=	2	6	8	9	$\perp$	$\perp$

Program

```
let sum(x) = cpt where
    y = pre(cpt)
and z = 0 fby y
and cpt = z + x
```

Extended domains and streams

- $t \in \mathbb{N}$ dicrete time
- $v \in V \uplus \{\perp\}$: $\perp$ if undefined, $\perp < v \in V$
- $S(V) = \mathbb{N} \to (V \uplus \{\perp\})$

Background: Constructive Fixpoint Semantics

Define semantics as mutual least fixpoint of set of monotonous operators (one for each equation) [Berry 1999]

Step-by-step execution

$time$	$=$	0	1	2	3	4	5
x	$=$	2	4	2	1	2	$\perp$
y	$=$	nil	2	6	8	9	$\perp$
z	$=$	0	2	6	8	9	$\perp$
cpt	$=$	2	6	8	9	$\perp$	$\perp$

Program

```
let sum(x) = cpt where
    y = pre(cpt)
and z = 0 fby y
and cpt = z + x
```

Extended domains and streams

- $t \in \mathbb{N}$ dicrete time
- $v \in V \uplus \{\perp\}$: $\perp$ if undefined, $\perp < v \in V$
- $S(V) = \mathbb{N} \to (V \uplus \{\perp\})$

Background: Constructive Fixpoint Semantics

Define semantics as mutual least fixpoint of set of monotonous operators (one for each equation) [Berry 1999]

Step-by-step execution

$time$	$=$	0	1	2	3	4	5
x	$=$	2	4	2	1	2	$\perp$
y	$=$	nil	2	6	8	9	$\perp$
z	$=$	0	2	6	8	9	$\perp$
cpt	$=$	2	6	8	9	11	$\perp$

Program

```
let sum(x) = cpt where
    y = pre(cpt)
and z = 0 fby y
and cpt = z + x
```

Extended domains and streams

- $t \in \mathbb{N}$ dicrete time
- $v \in V \uplus \{\perp\}$: $\perp$ if undefined, $\perp < v \in V$
- $S(V) = \mathbb{N} \rightarrow (V \uplus \{\perp\})$

Background: Constructive Fixpoint Semantics

Define semantics as mutual least fixpoint of set of monotonous operators (one for each equation) [Berry 1999]

Step-by-step execution

$time$	$=$	0	1	2	3	4	5
x	$=$	2	4	2	1	2	3
y	$=$	nil	2	6	8	9	11
z	$=$	0	2	6	8	9	$\perp$
cpt	$=$	2	6	8	9	11	$\perp$

Program

```
let sum(x) = cpt where
    y = pre(cpt)
and z = 0 fby y
and cpt = z + x
```

Extended domains and streams

- $t \in \mathbb{N}$ dicrete time
- $v \in V \uplus \{\perp\}$: $\perp$ if undefined, $\perp < v \in V$
- $S(V) = \mathbb{N} \to (V \uplus \{\perp\})$

Background: Constructive Fixpoint Semantics

Define semantics as mutual least fixpoint of set of monotonous operators (one for each equation) [Berry 1999]

Step-by-step execution

$time$	$=$	0	1	2	3	4	5
x	$=$	2	4	2	1	2	3
y	$=$	nil	2	6	8	9	11
z	$=$	0	2	6	8	9	11
cpt	$=$	2	6	8	9	11	$\perp$

Program

```
let sum(x) = cpt where
    y = pre(cpt)
and z = 0 fby y
and cpt = z + x
```

Extended domains and streams

- $t \in \mathbb{N}$ dicrete time
- $v \in V \uplus \{\perp\}$: $\perp$ if undefined, $\perp < v \in V$
- $S(V) = \mathbb{N} \to (V \uplus \{\perp\})$

Background: Constructive Fixpoint Semantics

Define semantics as mutual least fixpoint of set of monotonous operators (one for each equation) [Berry 1999]

Step-by-step execution

$time$	$=$	0	1	2	3	4	5
x	$=$	2	4	2	1	2	3
y	$=$	nil	2	6	8	9	11
z	$=$	0	2	6	8	9	11
cpt	$=$	2	6	8	9	11	14

Program

```
let sum(x) = cpt where
    y = pre(cpt)
and z = 0 fby y
and cpt = z + x
```

Extended domains and streams

- $t \in \mathbb{N}$ dicrete time
- $v \in V \uplus \{\bot\}$: $\bot$ if undefined, $\bot < v \in V$
- $S(V) = \mathbb{N} \to (V \uplus \{\bot\})$

Requirements on Semantics

Recall, semantics to help designing:

1. Static analyzers / model-checkers / theories for interactive provers
2. Compile-time analysis / simulation code generation
3. Numerical simulation environments (run-time)

Therefore:

- Every well-typed program E should have a semantics $[\![E]\!]$
- The semantics should be structural, i.e., roughly speaking:

$$
\begin{aligned}
[\![E_1 \; \mathtt{and} \; E_2]\!] &= \{[\![E_1]\!]; [\![E_2]\!]\} \\
[\![\mathtt{if}\; e \;\mathtt{then}\; E_1 \;\mathtt{else}\; E_2]\!] &= \mathtt{if}\; [\![e]\!] \;\mathtt{then}\; [\![E_1]\!] \;\mathtt{else}\; [\![E_2]\!], \text{ etc.}
\end{aligned}
$$

- The alternative is informal “mytool” semantics

Requirements on Semantics

Recall, semantics to help designing:

1. Static analyzers / model-checkers / theories for interactive provers
2. Compile-time analysis / simulation code generation
3. Numerical simulation environments (run-time)

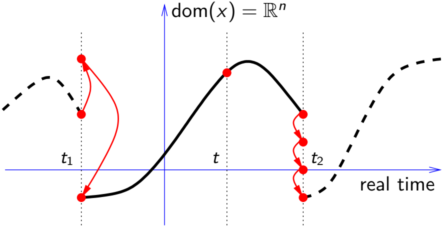
Therefore:

- Every well-typed program E should have a semantics $[\![E]\!]$
- The semantics should be structural, i.e., roughly speaking:

$$\begin{array}{rcl} [\![E_1 \ \mathtt{and} \ E_2]\!] & = & \{[\![E_1]\!];[\![E_2]\!]\} \\ [\![\mathtt{if}\, e\, \mathtt{then}\, E_1\, \mathtt{else}\, E_2]\!] & = & \mathtt{if}\, [\![e]\!]\, \mathtt{then}\, [\![E_1]\!]\, \mathtt{else}\, [\![E_2]\!], \ \text{etc.} \end{array}$$

- The alternative is informal "mytool" semantics

Time Domains

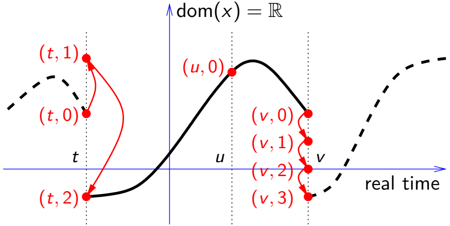


Phases of continuous dynamics interleaved with cascades of instantaneous state-jumps

However:

- Cascades may be complex or even unbounded
- The Time Domain should be such that time may progress during cascades of state-jumps

Time Domains



Superdense Model of Time:
$\mathbb{T} = \mathbb{R}_+ \times \mathbb{N}$
[Pnueli et al. 1992]
[Lee et al. 2005]

$\mathbb{T}$ is equipped with lexicographic order (as shown on the figure).

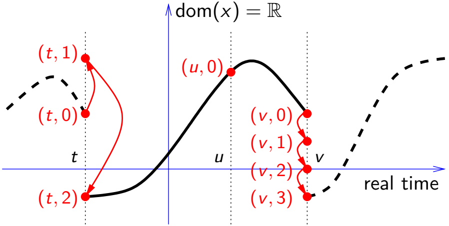
Two approaches for capturing signals with finite cascades of changes:

1. $x(t, n)$ defined for $0 \leq n \leq m_t$ and undefined for $n > m_t$ [the figure]
2. $x(t, n)$ defined for every n but $x(t, n) = x(t, m_t)$ for $n > m_t$ [Lee]

where m_t is the number of changes at time t.

In the figure: $m_t = 2, m_u = 0, m_v = 3$.

Time Domains

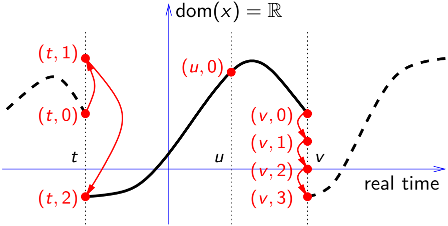


Superdense Model of Time:
$\mathbb{T} = \mathbb{R}_+ \times \mathbb{N}$
[Pnueli et al. 1992]
[Lee et al. 2005]

[Lee 2014]:

> *Such piecewise-continuous signals coexist nicely with standard ODE solvers. At the time of discontinuity or discrete event, the final value signal provides the initial boundary condition for the solver. [. . .]*

Time Domains

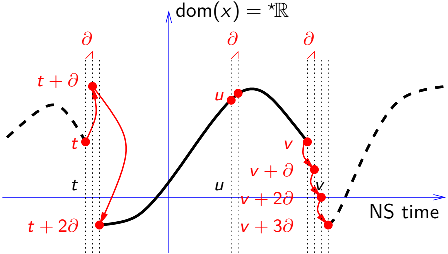


Superdense Model of Time

$\mathbb{T} = \mathbb{R}_+ \times \mathbb{N}$

[Pnueli et al. 1992]

[Lee et al. 2005]



Nonstandard Model of Time

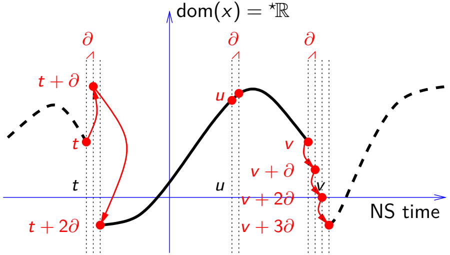
$\mathbb{T} = \{n\partial \mid n \in {}^\star\mathbb{N}\}$

[Benveniste et al. 2012]

Time Domains

Aim:

- getting rid of the burden of smoothness assumptions
- making hybrid systems discrete
- getting the semantics by reusing techniques from discrete systems



Nonstandard Model of Time

$\mathbb{T} = \{n\partial \mid n \in {}^\star\mathbb{N}\}$

[Benveniste et al. 2012]

A Toy Hybrid Systems Language

Syntax ≈ Zélus [Bourke et al. 2013]

$$
\begin{aligned}
d &::= \texttt{let}\ x = e \mid \texttt{let}\ f(p) = e\ \texttt{where}\ E \mid d; d \\
e &::= x \mid v \mid op(e) \mid e\ \texttt{fby}\ e \mid \texttt{pre}(e) \mid f(e) \mid (e, e) \\
p &::= (p, p) \mid x \\
E &::= () \mid E\ \texttt{and}\ E \mid x\ =\ e \mid \\
&\quad \mid \texttt{init}\ x\ =\ e \mid \texttt{der}\ x\ =\ e \mid \\
&\quad \mid \texttt{if}\ e\ \texttt{then}\ E\ \texttt{else}\ E \\
&\quad \mid \texttt{der}\ x\ =\ e \mid \\
&\quad \mid \texttt{init}\ x\ =\ e \mid \texttt{reinit}\ x\ =\ e \mid \\
&\quad \mid \texttt{when}\ e\ \texttt{do}\ E
\end{aligned}
$$

The Superdense Model of Time as a semantic domain

- $\mathbb{T} =_{\text{def}} \mathbb{R}_+ \times \mathbb{N}$; we identify $(t, 0) \in \mathbb{T}$ with $t \in \mathbb{R}_+$
- $x_{(t,n)}$ remains constant for $n \geq m_t^x$

equation	semantics
$\operatorname{der} x = f(x,u)$; $\operatorname{init} x = e$	$m_t^u = m_t^x = 0$ and $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$ and $x_0 = [\![e]\!]_0$ (1)
$\operatorname{der} x = f(x,u)$; $\operatorname{init} x = a$; when $x \geq 1$ do reinit $x = b$	$t_0 = 0$ and $t_{n+1} =$ $\inf\{s > t_n \mid \forall r \in (s-\varepsilon; s), x_r < 1 \wedge x_s \geq 1\}$ reset effective at $(t_n, 1)$, hence $m_{t_n}^x = 1$ $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$, for $t_n < t \leq t_{n+1}$, (2) $x_{t_0} = [\![a]\!]_{t_0}$, $x_{(t_n,1)} = [\![b]\!]_{(t_n,1)}$, $n \geq 1$ $\neq$ (1)
when z do reinit $x = b$	reuse of (2) not possible since $m_{t_n}^z \neq 0$

The Superdense Model of Time as a semantic domain

- $\mathbb{T} =_{\text{def}} \mathbb{R}_+ \times \mathbb{N}$; we identify $(t, 0) \in \mathbb{T}$ with $t \in \mathbb{R}_+$
- $x_{(t,n)}$ remains constant for $n \geq m^x_t$

equation	semantics
$\text{der}\, x = f(x, u)$; $\text{init}\, x = e$	$m^u_t = m^x_t = 0$ and $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$ and $x_0 = [\![e]\!]_0$ (1)
$\text{der}\, x = f(x, u)$; $\text{init}\, x = a$; $\text{when}\, x \geq 1\, \text{do}$ $\text{reinit}\, x = b$	$t_0 = 0$ and $t_{n+1} =$ $\inf\{s > t_n \mid \forall r \in (s - \varepsilon; s), x_r < 1 \wedge x_s \geq 1\}$ reset effective at $(t_n, 1)$, hence $m^x_{t_n} = 1$ $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$, for $t_n < t \leq t_{n+1}$, (2) $x_{t_0} = [\![a]\!]_{t_0}$, $x_{(t_n,1)} = [\![b]\!]_{(t_n,1)}$, $n \geq 1$ $\neq$ (1)
$\text{when}\, z\, \text{do}$ $\text{reinit}\, x = b$	reuse of (2) not possible since $m^z_{t_n} \neq 0$

The Superdense Model of Time as a semantic domain

- $\mathbb{T} =_{\mathrm{def}} \mathbb{R}_+ \times \mathbb{N}$; we identify $(t, 0) \in \mathbb{T}$ with $t \in \mathbb{R}_+$
- $x_{(t,n)}$ remains constant for $n \geq m_t^x$

equation	semantics
$\mathrm{der}\, x = f(x, u)$; $\mathrm{init}\, x = e$	$m_t^u = m_t^x = 0$ and $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$ and $x_0 = [\![e]\!]_0$ (1)
$\mathrm{der}\, x = f(x, u)$; $\mathrm{init}\, x = a$; $\mathrm{when}\, x \geq 1\, \mathrm{do}$ $\mathrm{reinit}\, x = b$	$t_0 = 0$ and $t_{n+1} =$ $\inf\{s > t_n \mid \forall r \in (s - \varepsilon; s), x_r < 1 \wedge x_s \geq 1\}$ reset effective at $(t_n, 1)$, hence $m_{t_n}^x = 1$ $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$, for $t_n < t \leq t_{n+1}$, (2) $x_{t_0} = [\![a]\!]_{t_0}$, $x_{(t_n,1)} = [\![b]\!]_{(t_n,1)}$, $n \geq 1$ $\neq$ (1)
$\mathrm{when}\, z\, \mathrm{do}$ $\mathrm{reinit}\, x = b$	reuse of (2) not possible since $m_{t_n}^z \neq 0$

The Superdense Model of Time as a semantic domain

- $\mathbb{T} =_{\text{def}} \mathbb{R}_+ \times \mathbb{N}$; we identify $(t, 0) \in \mathbb{T}$ with $t \in \mathbb{R}_+$
- $x_{(t,n)}$ remains constant for $n \geq m^x_t$

equation	semantics
$\text{der}\, x = f(x, u)$; $\text{init}\, x = e$	$m^u_t = m^x_t = 0$ and $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$ and $x_0 = [\![e]\!]_0$ (1)
$\text{der}\, x = f(x, u)$; $\text{init}\, x = a$; $\text{when}\, x \geq 1\, \text{do}$ $\text{reinit}\, x = b$	$t_0 = 0$ and $t_{n+1} =$ $\inf\{s > t_n \mid \forall r \in (s - \varepsilon; s), x_r < 1 \wedge x_s \geq 1\}$ reset effective at $(t_n, 1)$, hence $m^x_{t_n} = 1$ $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$, for $t_n < t \leq t_{n+1}$, (2) $x_{t_0} = [\![a]\!]_{t_0}$, $x_{(t_n,1)} = [\![b]\!]_{(t_n,1)}$, $n \geq 1$ $\neq$ (1)
$\text{when}\, z\, \text{do}$ $\text{reinit}\, x = b$	reuse of (2) not possible since $m^z_{t_n} \neq 0$

The Superdense Model of Time as a semantic domain

- $\mathbb{T} =_{\text{def}} \mathbb{R}_+ \times \mathbb{N}$; we identify $(t, 0) \in \mathbb{T}$ with $t \in \mathbb{R}_+$
- $x_{(t,n)}$ remains constant for $n \geq m_t^x$

equation	semantics
$\text{der}\, x = f(x, u)$; $\text{init}\, x = e$	$m_t^u = m_t^x = 0$ and $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$ and $x_0 = [\![e]\!]_0$ (1)
$\text{der}\, x = f(x, u)$; $\text{init}\, x = a$; $\text{when}\, x \geq 1\, \text{do}$ $\text{reinit}\, x = b$	$t_0 = 0$ and $t_{n+1} =$ $\inf\{s > t_n \mid \forall r \in (s - \varepsilon; s), x_r < 1 \wedge x_s \geq 1\}$ reset effective at $(t_n, 1)$, hence $m_{t_n}^x = 1$ $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$, for $t_n < t \leq t_{n+1}$, (2) $x_{t_0} = [\![a]\!]_{t_0}$, $x_{(t_n,1)} = [\![b]\!]_{(t_n,1)}$, $n \geq 1$ $\neq$ (1)
$\text{when}\, z\, \text{do}$ $\text{reinit}\, x = b$	reuse of (2) not possible since $m_{t_n}^z \neq 0$

The Superdense Model of Time as a semantic domain

- $\mathbb{T} =_{\text{def}} \mathbb{R}_+ \times \mathbb{N}$; we identify $(t, 0) \in \mathbb{T}$ with $t \in \mathbb{R}_+$
- $x_{(t,n)}$ remains constant for $n \geq m_t^x$

equation	semantics
$\text{der}\, x = f(x, u);$ $\text{init}\, x = e$	$m_t^u = m_t^x = 0$ and $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$ and $x_0 = [\![e]\!]_0$ (1)
$\text{der}\, x = f(x, u);$ $\text{init}\, x = a;$ $\text{when}\, x \geq 1\, \text{do}$ $\text{reinit}\, x = b$	$t_0 = 0$ and $t_{n+1} =$ $\inf\{s > t_n \mid \forall r \in (s - \varepsilon; s), x_r < 1 \wedge x_s \geq 1\}$ reset effective at $(t_n, 1)$, hence $m_{t_n}^x = 1$ $\dot{x}_t = [\![f]\!]_t(x_t, u_t)$, for $t_n < t \leq t_{n+1}$, (2) $x_{t_0} = [\![a]\!]_{t_0}$, $x_{(t_n,1)} = [\![b]\!]_{(t_n,1)}$, $n \geq 1$ $\neq$ (1)
$\text{when}\, z\, \text{do}$ $\text{reinit}\, x = b$	reuse of (2) not possible since $m_{t_n}^z \neq 0$

The Superdense Model of Time as a semantic domain

> *Such piecewise-continuous signals coexist nicely with standard ODE solvers. At the time of discontinuity or discrete event, the final value signal provides the initial boundary condition for the solver. [. . .]*

Lessons:

- Superdense time semantics seems simple as long as you keep it informal
- Actually, it is hard to formalize
- In addition to the problems shown:
  - Smoothness assumptions are needed, and
  - Must be stated on the global system
  - Can not capture chattering (sliding modes).
- [Lee 2014]: getting rid of the above difficulties by moving to constructive semantics?

The Superdense Model of Time as a semantic domain

Moving to constructive semantics

- [Berry 1999] The *constructive semantics* gives a meaning to fixpoint problems specified via sets of equations
  - does not rely on arguments of numerical analysis (convergence of approximation schemes)
  - uses instead fixpoint theorems where the distance between signals is defined as the largest prefix of time in which the two signals coincide
  - constructive $\Rightarrow$ helps understanding causality issues

- No constructive semantics exists for continuous-time systems ($\mathbb{T} = \mathbb{R}_+$) [Matsikoudis and Lee 2014]

- [Lee 2014] invokes constructive semantics as given by the solver (which works by steps)
  - Non compositional, not structural
  - Depends on munerical convergence properties of discretization scheme

The Superdense Model of Time as a semantic domain

Moving to constructive semantics

- [Berry 1999] The *constructive semantics* gives a meaning to fixpoint problems specified via sets of equations
  - does not rely on arguments of numerical analysis (convergence of approximation schemes)
  - uses instead fixpoint theorems where the distance between signals is defined as the largest prefix of time in which the two signals coincide
  - constructive $\Rightarrow$ helps understanding causality issues
- No constructive semantics exists for continuous-time systems ($\mathbb{T} = \mathbb{R}_+$) [Matsikoudis and Lee 2014]
- [Lee 2014] invokes constructive semantics as given by the solver (which works by steps)
  - Non compositional, not structural
  - Depends on munerical convergence properties of discretization scheme

The Nonstandard Time Domain

$$
\begin{aligned}
{}^\star\mathbb{N}, {}^\star\mathbb{R} &=_{\text{def}} \text{non-standard extensions of } \mathbb{N}, \mathbb{R} \\
{}^\star\mathbb{R} \supseteq \mathbb{T} &=_{\text{def}} \{t_n = n\partial \mid n \in {}^\star\mathbb{N}\} \text{ where } \partial \text{ is an \textit{infinitesimal} time step} \\
{}^\bullet t &=_{\text{def}} \max\{s \mid s \in \mathbb{T}, s < t\} = t - \partial \\
t^\bullet &=_{\text{def}} \min\{s \mid s \in \mathbb{T}, s > t\} = t + \partial \\
\dot{x}_t &=_{\text{def}} \frac{x_{t^\bullet} - x_t}{\partial} \text{ (explicit scheme) or } \frac{x_t - x_{{}^\bullet t}}{\partial} \text{ (implicit scheme)}
\end{aligned}
$$

- with the non-standard interpretation, hybrid systems become "discrete time" and inherit a non-standard semantics
  - no more difficult than Lustre semantics
  - every syntactically correct program has a semantics
  - the non-standard semantics is structural and compositional
- does not depend on the particular choice for the time base ∂

Nonstandard Semantics

Set $\;{}^{\bullet}x_t = x_{{}^{\bullet}t}\;,\quad x_t^{\bullet} = x_{t^{\bullet}}\;$, and $\dot{x}_t = \dfrac{x_t^{\bullet} - x_t}{\partial}\;$ in:

equation	semantics
der $x = e$; init $x = f$	$x_{t_0} = [\![f]\!]_{t_0}$ and $x_t^{\bullet} = x_t + \partial[\![e]\!]_t$ forall $t \in \mathbb{T}, t \geq t_0$
der $x = e$; init $x = a$; when $x \geq 1$ do reinit $x = b$	$z = {}^{\bullet}x_t{<}1 \wedge x_t{\geq}1$ $x_{t_0} = [\![a]\!]_{t_0}$ $x_t^{\bullet} =$ if z then $[\![b]\!]_{t^{\bullet}}$ else $x_t + \partial[\![e]\!]_t,\; t \geq t_0$

- Just as for Lustre
- Since the non-standard semantics is step-based, constructive semantics exists [Benveniste et al. 2012]
  - Having ${}^{*}\mathbb{N}$ many steps instead of $\mathbb{N}$ many ones is not an issue
  - Of course, this semantics can not be used for simulation ($\neq$ programming languages)

Nonstandard Semantics

Set ${}^\bullet x_t = x_{{}^\bullet t}$, $x_t^\bullet = x_{t^\bullet}$, and $\dot{x}_t = \dfrac{x_t^\bullet - x_t}{\partial}$ in:

equation	semantics
$\mathsf{der}\, x = e;$ $\mathsf{init}\, x = f$	$x_{t_0} = [\![f]\!]_{t_0}$ and $x_t^\bullet = x_t + \partial[\![e]\!]_t$ forall $t \in \mathbb{T}, t \geq t_0$
$\mathsf{der}\, x = e;$ $\mathsf{init}\, x = a;$ $\mathsf{when}\, x \geq 1\, \mathsf{do}$ $\quad \mathsf{reinit}\, x = b$	$z = {}^\bullet x_t{<}1 \wedge x_t{\geq}1$ $x_{t_0} = [\![a]\!]_{t_0}$ $x_t^\bullet = \mathsf{if}\, z\, \mathsf{then}\, [\![b]\!]_{t^\bullet}\, \mathsf{else}\, x_t + \partial[\![e]\!]_t,\ t \geq t_0$

- Just as for Lustre
- Since the non-standard semantics is step-based, constructive semantics exists [Benveniste et al. 2012]
  - Having ${}^\star\mathbb{N}$ many steps instead of $\mathbb{N}$ many ones is not an issue
  - Of course, this semantics can not be used for simulation ($\neq$ programming languages)

There is no free lunch

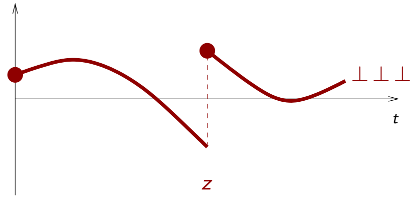
Theorem [Benveniste et al. 2014]

The nonstandard semantics of every causally-correct program is:

1. standardizable,
2. independent of ∂,
3. continuous

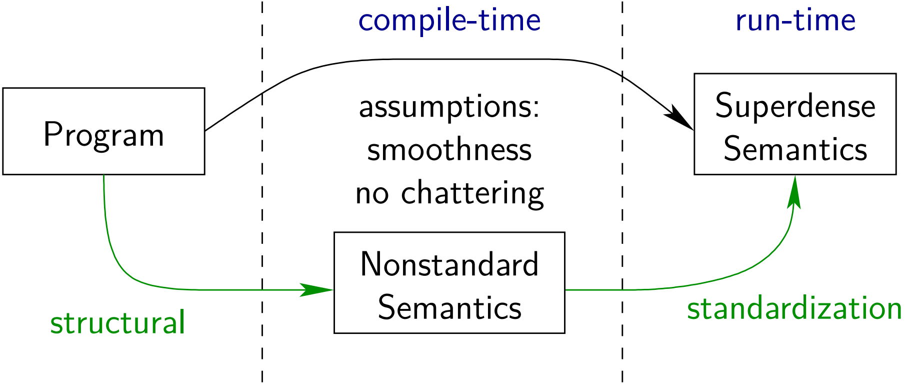
on every compact set of dates not containing:

1. an event, or
2. an undefined value ($\perp$)



- When defined, the superdense semantics coincides with the nonstandard semantics
- The nonstandard semantics is not effective (cannot be executed)

There is no free lunch



DAE Hybrid Systems: index theory & reduction

- With non-standard semantics, DAE become *dAE* (*difference* Algebraic Equations); define $x^\bullet$ = next x
- dAE may involve more equations than specified

$$\left\{\begin{array}{lcl} x^\bullet & = & f(x,u) \\ 0 & = & g(x) \end{array}\right. \quad \overset{\text{shifting}}{\Longrightarrow} \quad \left\{\begin{array}{lcl} x^\bullet & = & f(x,u) \\ 0 & = & g(x) \\ 0 & = & g(x^\bullet) \end{array}\right.$$

$$\overset{\text{substituting}}{\Longrightarrow} \quad \left\{\begin{array}{lcll} x^\bullet & = & f(x,u) & (1) \\ 0 & = & g(x) & (2) \\ 0 & = & g(f(x,u)) & (3) \end{array}\right.$$

Whence the constructive semantics ($\sim$ execution scheme):

1. Given x such that $g(x) = 0$
2. Use (3) to evaluate u (constraint solver needed)
3. Use (1) to evaluate $x^\bullet$, which satisfies $g(x^\bullet) = 0$, and repeat

DAE Hybrid Systems: index theory & reduction

- With non-standard semantics, DAE become *dAE* (*difference* Algebraic Equations); define $x^\bullet =$ next x
- dAE may involve more equations than specified

$$\left\{\begin{array}{lcl} x^\bullet & = & f(x,u) \\ 0 & = & g(x) \end{array}\right. \quad \stackrel{\text{shifting}}{\Longrightarrow} \quad \left\{\begin{array}{lcl} x^\bullet & = & f(x,u) \\ 0 & = & g(x) \\ 0 & = & g(x^\bullet) \end{array}\right.$$

$$\stackrel{\text{substituting}}{\Longrightarrow} \quad \left\{\begin{array}{lcll} x^\bullet & = & f(x,u) & (1) \\ 0 & = & g(x) & (2) \\ 0 & = & g(f(x,u)) & (3) \end{array}\right.$$

Thm: the diff. index of a DAE coincides with the index of the dAE obtained with the non-standard semantics

Cor: Defining the index of DAE Hybrid Systems as the index of its non-standard semantics yields a conservative extension of DAE and dAE indexes

Conclusion

- The superdense model of time is useful as a simulation semantics:
  - Even from this point of view it has limits
  - No support for nonsmooth dynamical systems simulation (with possible chattering)
- More is needed for supporting compilation:
  - Structural semantics
  - Getting rid of smoothness assumptions
- The nonstandard model of time is a good candidate:
  - Yields a structural semantics
  - No smoothness assumption
  - Coincides with superdense semantics, when defined
  - Supports the slicing of execution engine into
    - an event handler and
    - a ODE/DAE/nonsmooth solver