PROGRAMMING WITH INFINITESIMALS

A WHILE-LANGUAGE FOR HYBRID SYSTEM MODELING

In: Proc. *ICALP Track B*, 2011

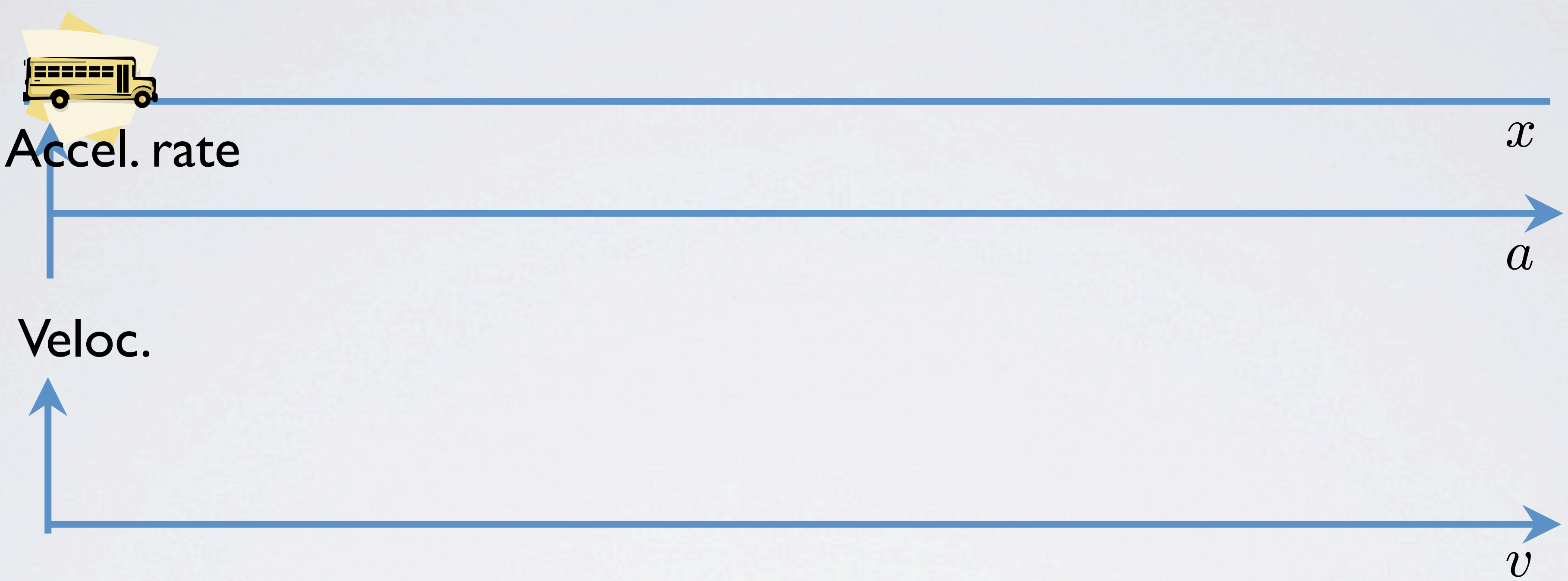
Kohei Suenaga
Kyoto University (JP)

Ichiro Hasuo
University of Tokyo (JP)

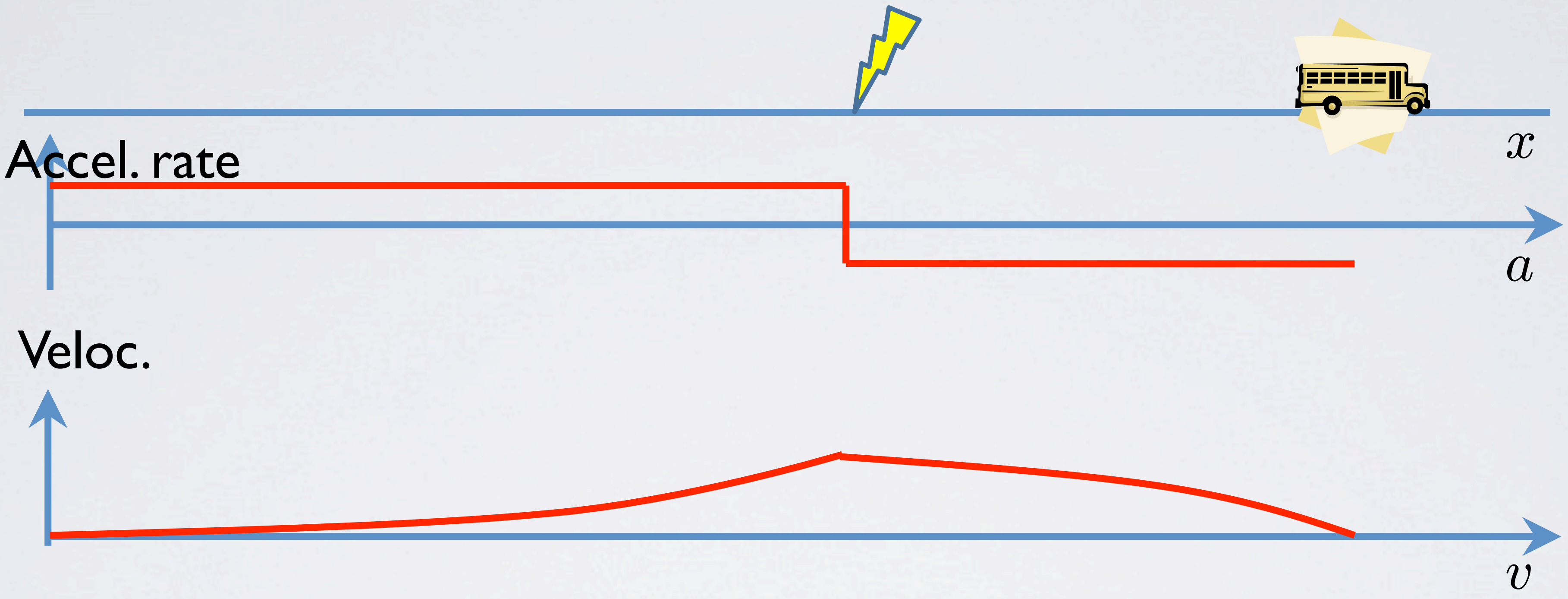
京都大学
KYOTO UNIVERSITY

東京大学
THE UNIVERSITY OF TOKYO

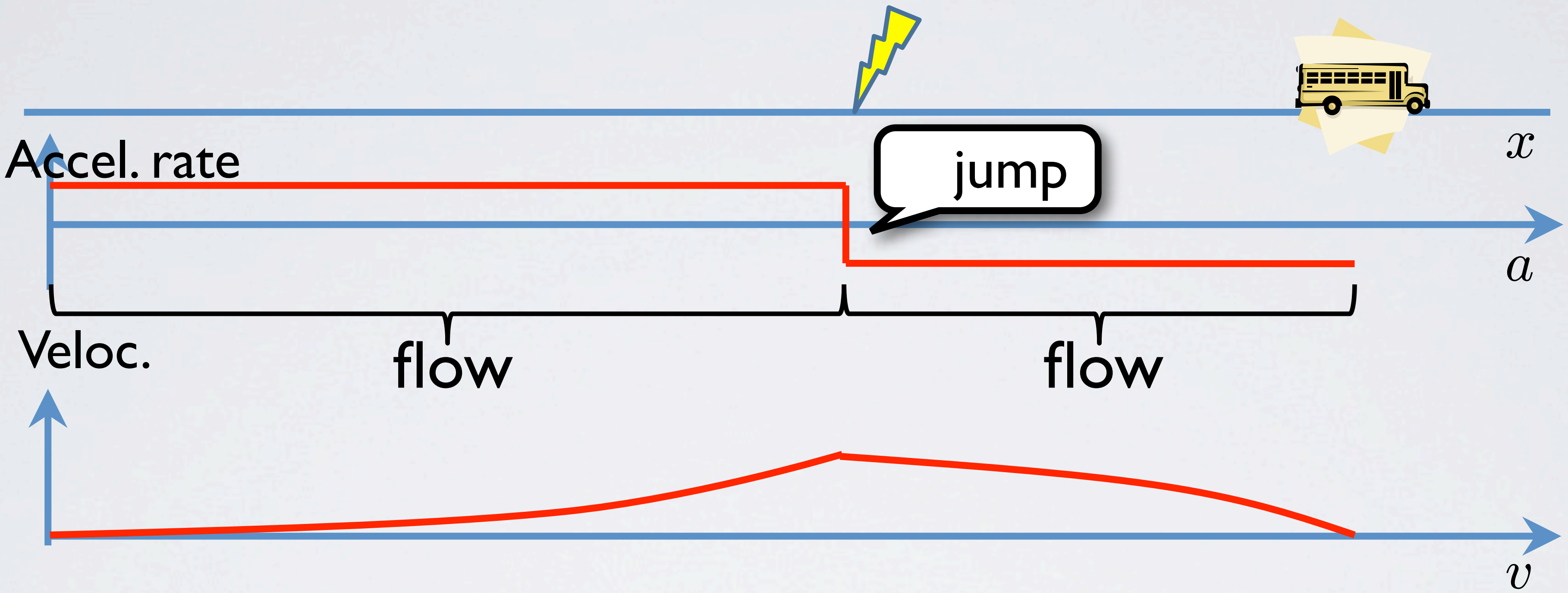
Hybrid System



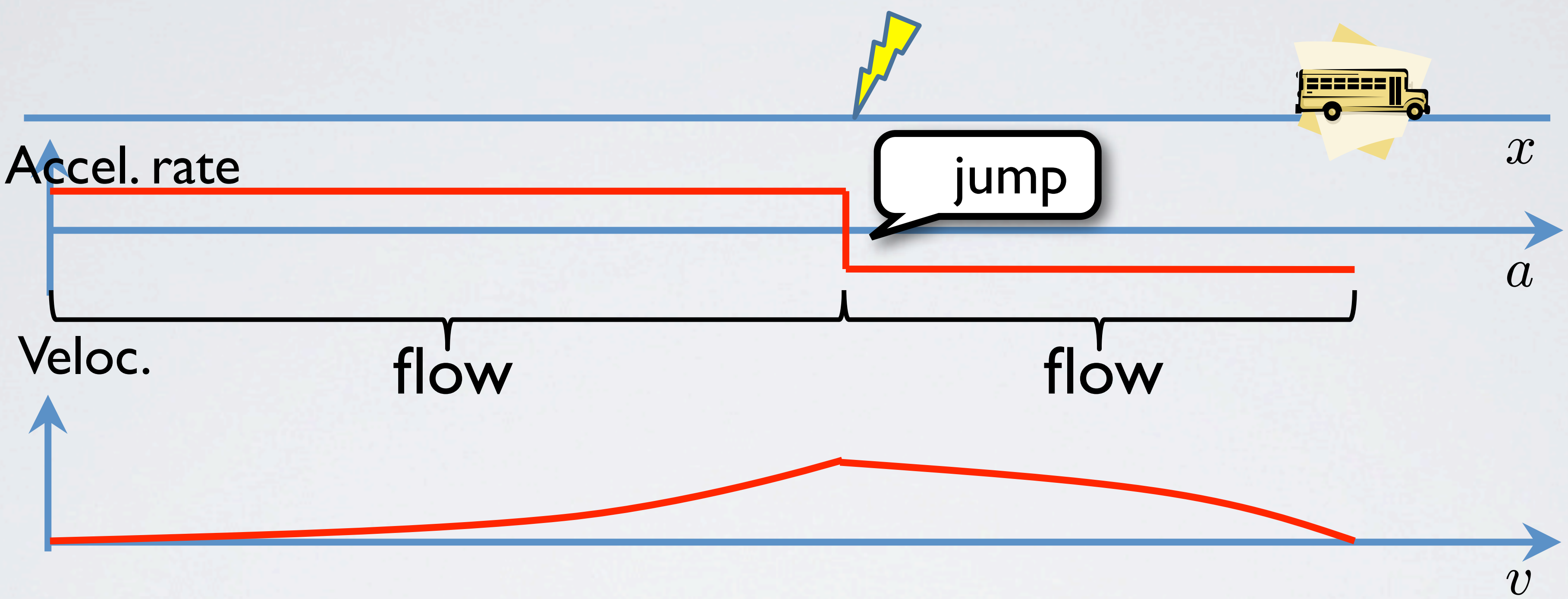
Hybrid System



Hybrid System



Hybrid System



- Flow & jump
  - Digital control in a physical environment
  - Component of cyber-physical systems

Hybrid System

Discrete
"jump"

and

Continuous
"flow"

Hybrid System

Discrete
"jump"

and

Continuous
"flow"

Hybrid System

Formal verification
(computer science)

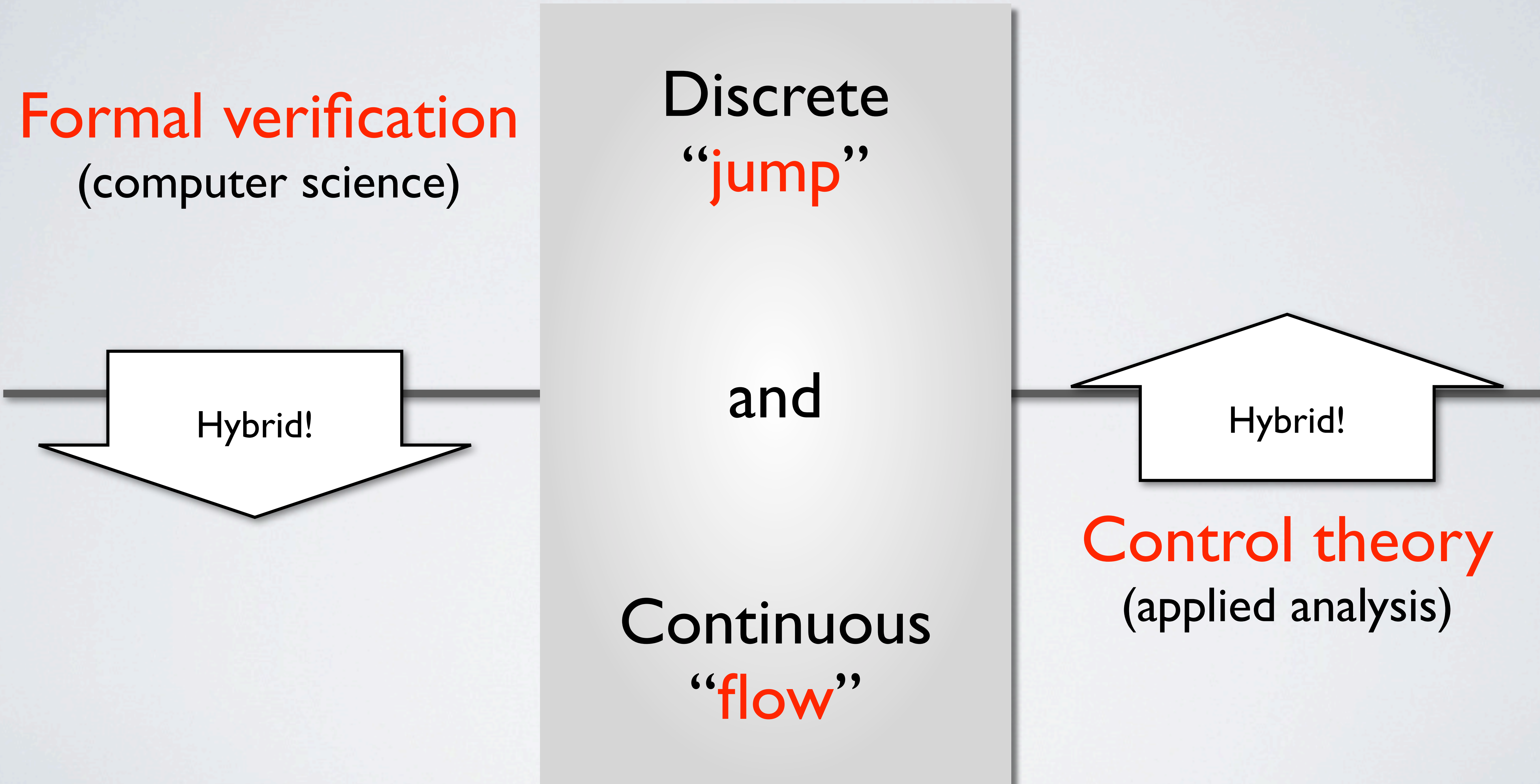
Discrete
“jump”

and

Continuous
“flow”

Control theory
(applied analysis)

Hybrid System



Hybrid System

Formal verification
(computer science)

Hybrid!

- Flow?
- With minimal cost?

Discrete
"jump"

and

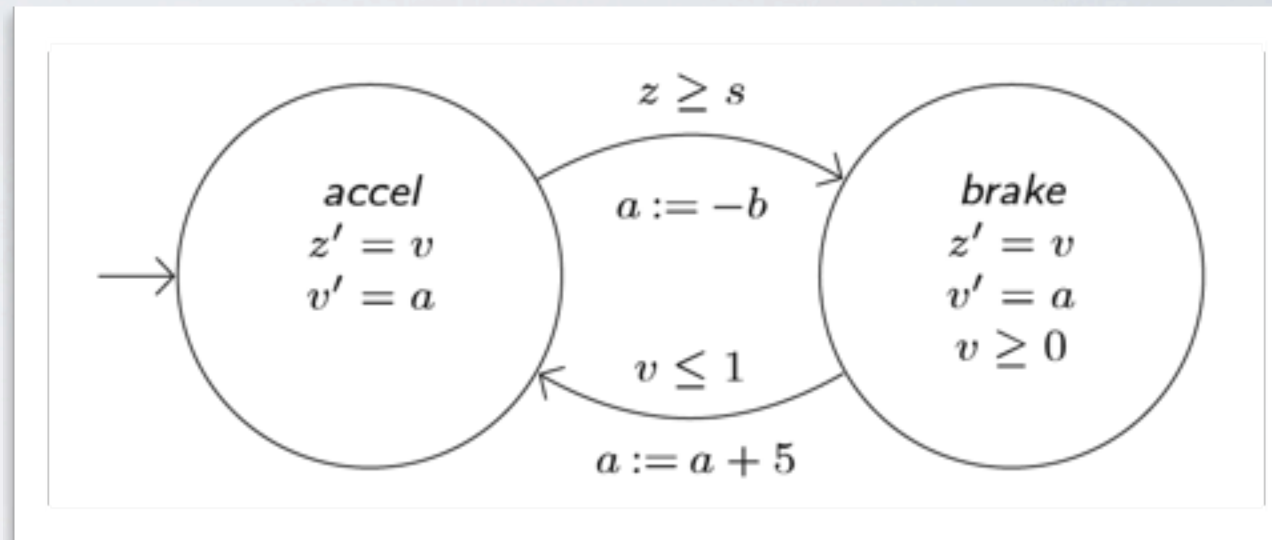
Continuous
"flow"

Hybrid!

Control theory
(applied analysis)

Formal Verification Approaches

- Hybrid automata [Alur & others, ’90s-]



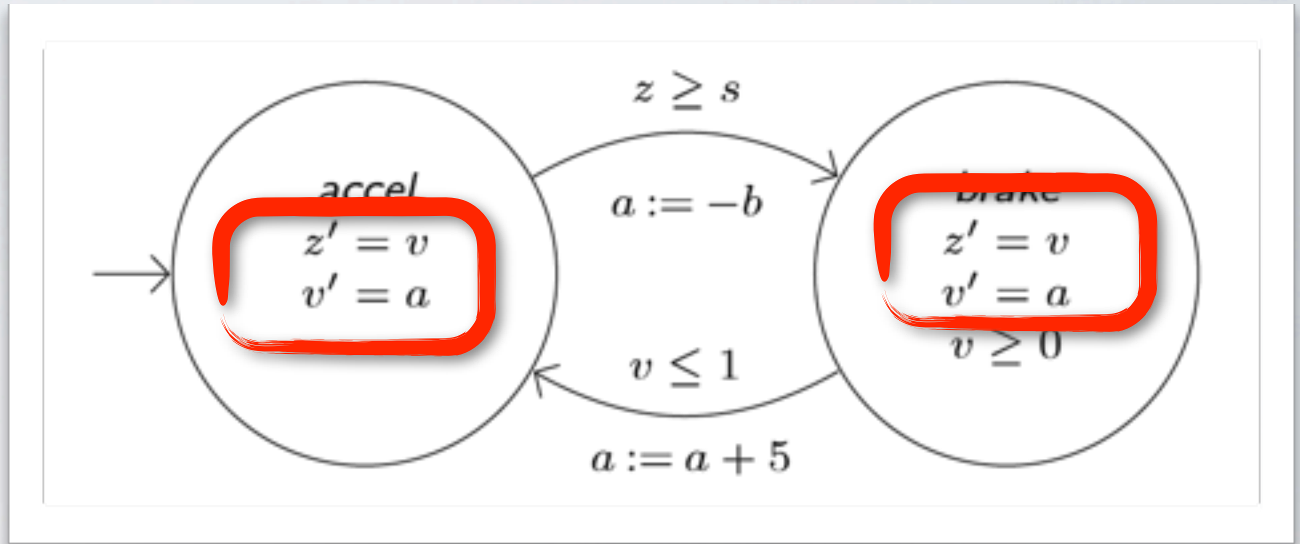
- Differential dynamic logic [Platzer & others, ’07-]

$$[\dot{x} = 1 \text{ while } x \leq 3]\varphi$$

- Differential equations, explicitly ➜ distinction jump vs. flow

Formal Verification Approaches

- Hybrid automata [Alur & others, ’90s-]



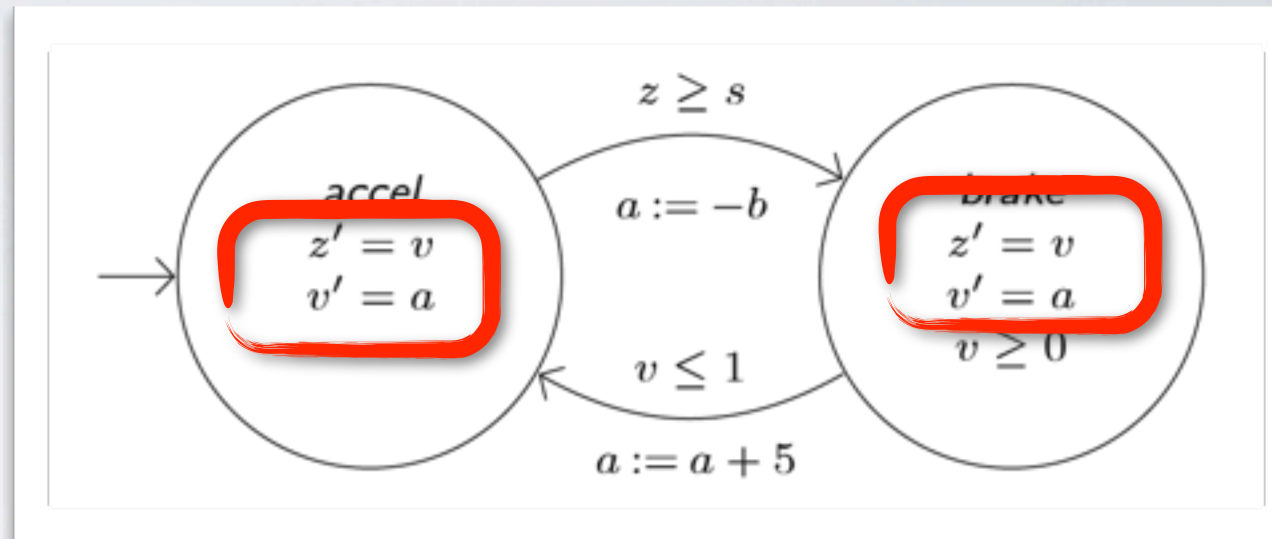
- Differential dynamic logic [Platzer & others, ’07-]

$$[\dot{x} = 1 \text{ while } x \leq 3]\varphi$$

- Differential equations, explicitly ➜ distinction jump vs. flow

Formal Verification Approaches

- Hybrid automata [Alur & others, '90s-]



- Differential dynamic logic [Platzer & others, '07-]

$$[\dot{\boldsymbol{x}} = \mathbf{1}\ \text{while}\ \boldsymbol{x} \leq \mathbf{3}]\varphi$$

- Differential equations, explicitly ➜ distinction jump vs. flow

"Turn Flow into Jump"

$$
\begin{array}{l}
\boldsymbol{t} := \mathbf{0}\ ; \\
\texttt{while}\ (\boldsymbol{t} \leq \mathbf{1})\ \texttt{do}\ \{ \\
\quad \boldsymbol{t} := \boldsymbol{t} + \texttt{dt} \\
\}
\end{array}
$$

“Turn Flow into Jump”

$$
\begin{array}{l}
t := 0\ ; \\
\texttt{while}\ (t \le 1)\ \texttt{do}\ \{ \\
\quad t := t + \texttt{dt} \\
\}
\end{array}
$$

- Infinitesimal number $\texttt{dt}$
  - “Infinitely small” : $0 < \texttt{dt} < r$ for any positive real r

"Turn Flow into Jump"

$$
\begin{array}{l}
\boldsymbol{t} := \mathbf{0}\ ; \\
\texttt{while}\ (t \le 1)\ \texttt{do}\ \{ \\
\quad t := t + \mathrm{dt} \\
\}
\end{array}
$$

- Infinitesimal number dt
  - "Infinitely small" :
    0 < dt < r for any positive real r
- t = 1 after the execution?

"Turn Flow into Jump"

$$
\begin{aligned}
&t := 0\ ;\\
&\texttt{while}\ (t \leq 1)\ \texttt{do}\ \{\\
&\quad t := t + \mathrm{dt}\\
&\}
\end{aligned}
$$

- Infinitesimal number dt
  - "Infinitely small" :
    $0 < \mathrm{dt} < r$ for any positive real r
- $t = 1$ after the execution?
- Non-standard analysis!
  [Robinson '60s]

Contribution

Contribution

The standard textbook [Winskel]

Contribution

The standard textbook [Winskel]

While

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Contribution

The standard textbook [Winskel]

While

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn

First-order assertion lang.

$\exists z(x=2*z \wedge y=3*z)$

Contribution

The standard textbook [Winskel]

While

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn

First-order assertion lang.

$\exists$z(x=2*z $\wedge$ y=3*z)

Hoare

Hoare-style program logic

$$\frac{\{A \wedge b\}\, c\, \{A\}}{\{A\}\ \texttt{while}\ b\ \texttt{do}\ c\ \{A \wedge \neg b\}}$$

Contribution

The standard textbook [Winskel]

While^{dt}

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn^{dt}

First-order assertion lang.

```
∃z(x=2*z ∧ y=3*z)
```

Hoare^{dt}

Hoare-style program logic

$$\frac{\{A \wedge b\}\; c\; \{A\}}{\{A\}\ \texttt{while}\ b\ \texttt{do}\ c\ \{A \wedge \neg b\}}$$

Contribution

The standard textbook [Winskel]

While$^{\mathrm{dt}}$

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn$^{\mathrm{dt}}$

First-order assertion lang.

$\exists z(x=2*z \land y=3*z)$

Hoare$^{\mathrm{dt}}$

Hoare-style program logic

$$\frac{\{A \land b\}\, c\, \{A\}}{\{A\}\ \texttt{while}\ b\ \texttt{do}\ c\ \{A \land \neg b\}}$$

Rigorous semantics by non-standard analysis

Contribution

The standard textbook [Winskel]

While$^{\mathrm{dt}}$

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn$^{\mathrm{dt}}$

First-order assertion lang.

```
∃z(x=2*z ∧ y=3*z)
```

Hoare$^{\mathrm{dt}}$

Hoare-style program logic

$$\frac{\{A \wedge b\}\, c\, \{A\}}{\{A\}\ \texttt{while}\ b\ \texttt{do}\ c\ \{A \wedge \neg b\}}$$

Rigorous semantics by non-standard analysis

- **Hoare**$^{\mathrm{dt}}$: sound and relatively complete

Contribution

The standard textbook [Winskel]

While$^{\mathrm{dt}}$

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn$^{\mathrm{dt}}$

First-order assertion lang.

$\exists$z(x=2*z $\wedge$ y=3*z)

Hoare$^{\mathrm{dt}}$

Hoare-style program logic

$$\frac{\{A \wedge b\}\, c\, \{A\}}{\{A\}\, \texttt{while}\ b\ \texttt{do}\ c\, \{A \wedge \neg b\}}$$

Rigorous semantics by non-standard analysis

- **Hoare**$^{\mathrm{dt}}$: sound and relatively complete
- Program verification/static analysis of hybrid systems

Contribution

The standard textbook [Winskel]

While$^{\text{dt}}$

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn$^{\text{dt}}$

First-order assertion lang.

$\exists z(x=2*z \wedge y=3*z)$

Hoare$^{\text{dt}}$

Hoare-style program logic

$$\frac{\{A \wedge b\}\, c\, \{A\}}{\{A\}\, \texttt{while}\ b\ \texttt{do}\ c\, \{A \wedge \neg b\}}$$

Rigorous semantics by non-standard analysis

- **Hoare**$^{\text{dt}}$: sound and relatively complete
- Program verification/static analysis of hybrid systems
- Actual verification with NSA

Contribution

Assn$^{\mathtt{dt}}$

First-order assertion lang.

`∃z(x=2*z ∧ y=3*z)`

Contribution

First-order language
for hyperreals

```
∃z(x=2*z ∧ y=3*z)
```

+

semantics via
ultraproducts

Contribution

First-order language
for reals

$\exists z(x=2*z \land y=3*z)$

+

usual semantics

First-order language
for hyperreals

$\exists z(x=2*z \land y=3*z)$

+

semantics via
ultraproducts

Contribution

First-order language for reals

∃z(x=2*z ∧ y=3*z)

+

usual semantics

Transfer Principle

[Robinson] (Łos' Theorem)

First-order language for hyperreals

∃z(x=2*z ∧ y=3*z)

+

semantics via ultraproducts

Contribution

First-order language
for reals

∃z(x=2*z ∧ y=3*z)

+

usual semantics

Transfer Principle

[Robinson] (Łos' Theorem)

First-order language
for hyperreals

∃z(x=2*z ∧ y=3*z)

+

semantics via
ultraproducts

Computational
aspects

- While-language
- Hoare logic

Contribution

First-order language for reals

∃z(x=2*z ∧ y=3*z)

\+

usual semantics

Transfer Principle

[Robinson] (Łos' Theorem)

First-order language for hyperreals

∃z(x=2*z ∧ y=3*z)

\+

semantics via ultraproducts

Computational aspects

- While-language
- Hoare logic

Computational aspects

- While-language
- Hoare logic

Semantics: Challenge

$$
\begin{array}{l}
\boldsymbol{t} := \mathbf{0}\ ; \\
\texttt{while}\ (\boldsymbol{t} \leq \mathbf{1})\ \texttt{do}\ \{ \\
\quad \boldsymbol{t} := \boldsymbol{t} + \texttt{dt} \\
\}
\end{array}
$$

Semantics: Challenge

$$
\begin{array}{l}
t := \mathbf{0}\ ; \\
\mathtt{while}\ (t \leq 1)\ \mathtt{do}\ \{ \\
\quad t := t + \mathtt{dt} \\
\}
\end{array}
$$

$$
\begin{array}{l}
t := \mathbf{0}\ ; \\
\mathtt{while}\ (\mathtt{true})\ \mathtt{do}\ \{ \\
\quad t := t + \mathtt{dt} \\
\}
\end{array}
$$

Semantics: Challenge

$$
\begin{array}{l}
\boldsymbol{t} := \mathbf{0}\ ; \\
\texttt{while}\ (\boldsymbol{t} \leq \mathbf{1})\ \texttt{do}\ \{ \\
\quad \boldsymbol{t} := \boldsymbol{t} + \texttt{dt} \\
\}
\end{array}
$$

$$\boldsymbol{t} = \mathbf{1} + \texttt{dt}$$

$$
\begin{array}{l}
\boldsymbol{t} := \mathbf{0}\ ; \\
\texttt{while}\ (\texttt{true})\ \texttt{do}\ \{ \\
\quad \boldsymbol{t} := \boldsymbol{t} + \texttt{dt} \\
\}
\end{array}
$$

$$\perp \text{ (divergence)}$$

Semantics: Challenge

$$\begin{array}{l} t := 0\ ; \\ \texttt{while}\ (t \leq 1)\ \texttt{do}\ \{ \\ \quad t := t + \mathtt{dt} \\ \} \end{array}$$

$$\begin{array}{l} t := 0\ ; \\ \texttt{while}\ (\texttt{true})\ \texttt{do}\ \{ \\ \quad t := t + \mathtt{dt} \\ \} \end{array}$$

$$t = 1 + \mathtt{dt}$$

$$\perp \text{ (divergence)}$$

- Semantics by “sectionwise execution”

Semantics: Challenge

$$
\begin{array}{l}
t := 0\ ; \\
\texttt{while}\ (t \leq 1)\ \texttt{do}\ \{ \\
\quad t := t + \texttt{dt} \\
\}
\end{array}
\qquad\qquad
\begin{array}{l}
t := 0\ ; \\
\texttt{while}\ (\texttt{true})\ \texttt{do}\ \{ \\
\quad t := t + \texttt{dt} \\
\}
\end{array}
$$

$$t = 1 + \texttt{dt} \qquad\qquad \bot\ \text{(divergence)}$$

- Semantics by "sectionwise execution"
- Sectionwise execution/satisfaction ➜ much like Łos' thm.

I

$\textbf{While}^{\mathrm{dt}}$

SYNTAX

$\mathbf{While}^{\mathsf{dt}}$: Syntax

$$
\begin{array}{llll}
\mathbf{AExp} \ni & a & ::= & x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt} \mid \infty \\
& & & \text{where } x \in \mathbf{Var},\ \mathtt{c}_r \text{ is a constant for } r \in \mathbb{R}, \text{ and } \mathtt{aop} \in \{+, -, \cdot, \hat{}\,\} \\
\mathbf{BExp} \ni & b & ::= & \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2 \\
\mathbf{Cmd} \ni & c & ::= & \mathtt{skip} \mid x := a \mid c_1; c_2 \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c
\end{array}
$$

$\mathbf{While}^{\mathrm{dt}}$: Syntax

const. for reals

const. for an infinitesimal & an infinite

$$\begin{array}{llll}
\mathbf{AExp} \ni & a & ::= & x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt} \mid \infty \\
& & & \text{where } x \in \mathbf{Var},\ \mathtt{c}_r \text{ is a constant for } r \in \mathbb{R}, \text{ and } \mathtt{aop} \in \{+, -, \cdot, ^\wedge\} \\
\mathbf{BExp} \ni & b & ::= & \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2 \\
\mathbf{Cmd} \ni & c & ::= & \mathtt{skip} \mid x := a \mid c_1; c_2 \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c
\end{array}$$

$\mathbf{While}^{\mathtt{dt}}$: Syntax

const. for reals

const. for an infinitesimal & an infinite

$$\begin{array}{llll}
\mathbf{AExp} \ni & a & ::= & x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt} \mid \infty \\
& & & \text{where } x \in \mathbf{Var},\ \mathtt{c}_r \text{ is a constant for } r \in \mathbb{R}, \text{ and } \mathtt{aop} \in \{+, -, \cdot, \hat{}\,\} \\
\mathbf{BExp} \ni & b & ::= & \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2 \\
\mathbf{Cmd} \ni & c & ::= & \mathtt{skip} \mid x := a \mid c_1; c_2 \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c
\end{array}$$

- **While** + reals + `dt`

While$^{\text{dt}}$: Syntax

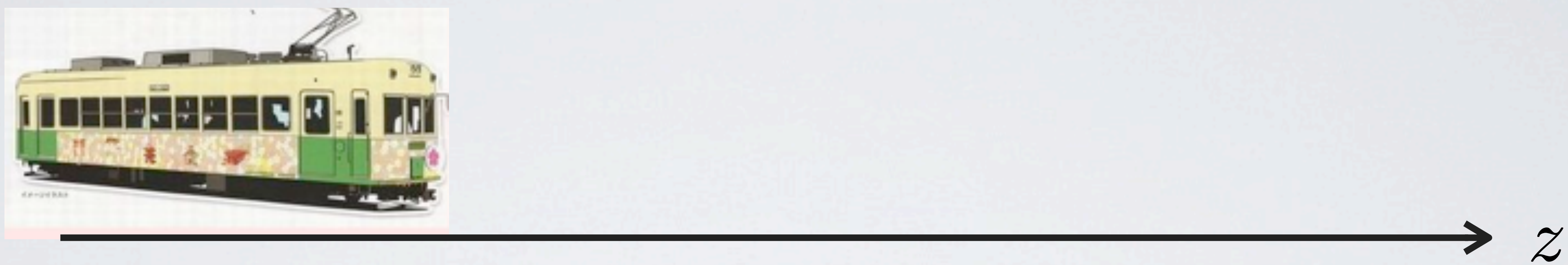
const. for reals

const. for an infinitesimal & an infinite

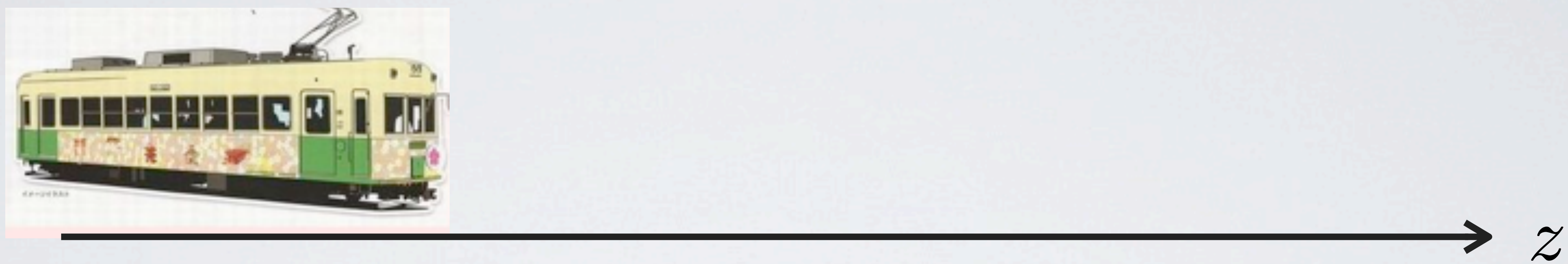
$$\begin{array}{llll}
\textbf{AExp} \ni & a & ::= & x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt} \mid \infty \\
& & & \text{where } x \in \mathbf{Var},\ \mathtt{c}_r \text{ is a constant for } r \in \mathbb{R}, \text{ and } \mathtt{aop} \in \{+, -, \cdot, \hat{}\} \\
\textbf{BExp} \ni & b & ::= & \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2 \\
\textbf{Cmd} \ni & c & ::= & \mathtt{skip} \mid x := a \mid c_1; c_2 \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c
\end{array}$$

- **While** + reals + dt
- Not meant to be executed; for modeling

While$^{\mathrm{dt}}$: Example [Platzer '07]

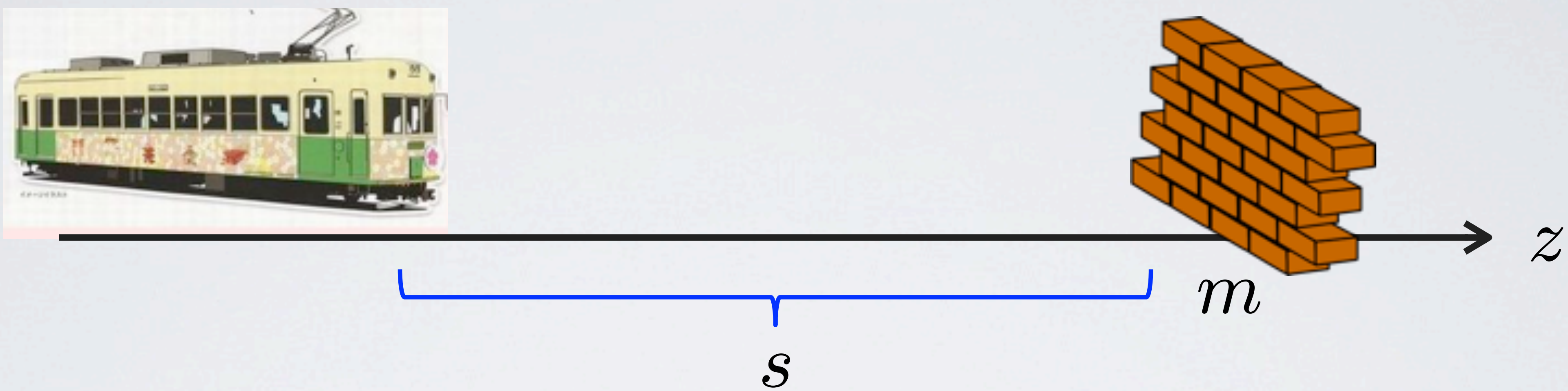


$\textbf{While}^{\mathrm{dt}}$: Example [Platzer '07]



$$
\begin{array}{l}
\texttt{while}\ t < \varepsilon\ \texttt{do}\ \{\\
\quad\quad t := t + \mathtt{dt}\,;\\
\quad\quad v := v + a \cdot \mathtt{dt}\,;\\
\quad\quad z := z + v \cdot \mathtt{dt}\\
\}
\end{array}
$$

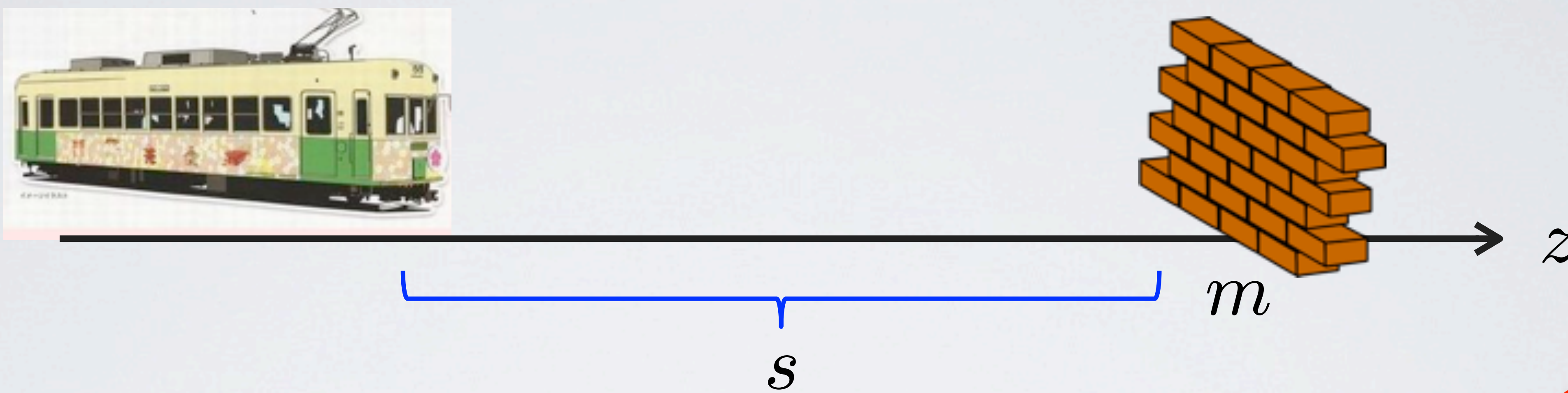
$\mathbf{While}^{\mathrm{dt}}$: Example [Platzer '07]



$$
\begin{array}{l}
\texttt{while}\ t < \varepsilon\ \texttt{do}\ \{ \\
\qquad t := t + \mathrm{dt}\,; \\
\qquad v := v + a \cdot \mathrm{dt}\,; \\
\qquad z := z + v \cdot \mathrm{dt} \\
\}
\end{array}
$$

$$
\begin{array}{l}
\texttt{while}\ v > 0\ \texttt{do}\ \{ \\
\qquad t := 0\,; \\
\qquad \texttt{if}\ m - z < s\ \texttt{then}\ a := -b\ \texttt{else}\ a := 0\,; \\
\qquad \texttt{while}\ t < \varepsilon\ \texttt{do}\ \{ \\
\qquad\qquad t := t + \mathrm{dt}\,; \\
\qquad\qquad v := v + a \cdot \mathrm{dt}\,; \\
\qquad\qquad z := z + v \cdot \mathrm{dt} \\
\qquad \}\}
\end{array}
$$

While$^{\mathrm{dt}}$: Example [Platzer '07]

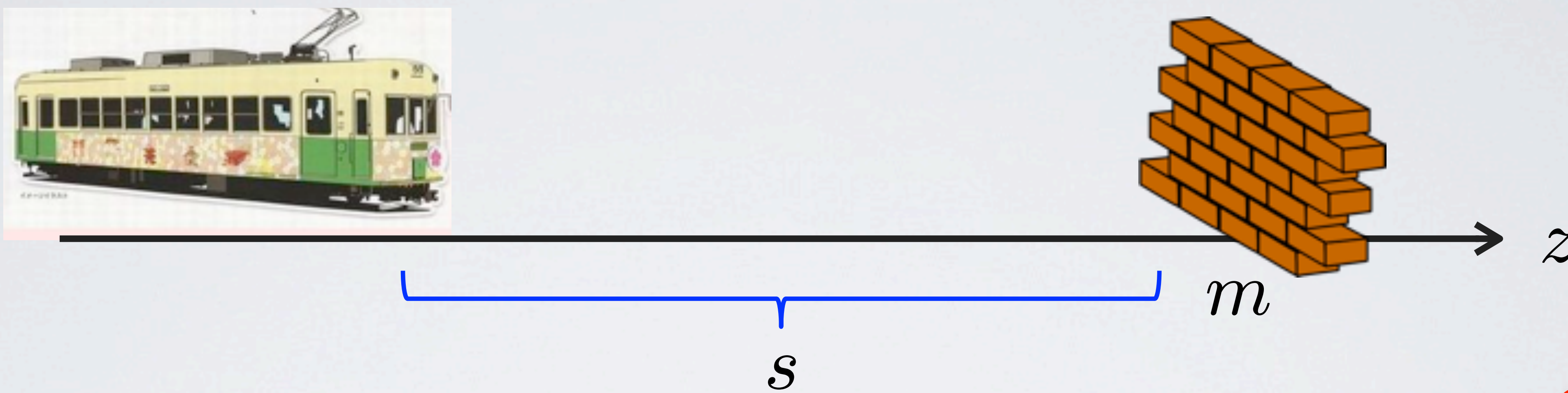



```
while t < ε do {
    t := t + dt ;
    v := v + a · dt ;
    z := z + v · dt
}
```

```
while v > 0 do {
    t := 0 ;
    if m − z < s then a := −b else a := 0 ;
    while t < ε do {
        t := t + dt ;
        v := v + a · dt ;
        z := z + v · dt
    }}
```

start braking

$\textbf{While}^{\mathrm{dt}}$: Example [Platzer '07]



$$
\begin{array}{l}
\texttt{while}\ t < \varepsilon\ \texttt{do}\ \{\\
\qquad t := t + \mathrm{dt}\,;\\
\qquad v := v + a \cdot \mathrm{dt}\,;\\
\qquad z := z + v \cdot \mathrm{dt}\\
\}
\end{array}
$$

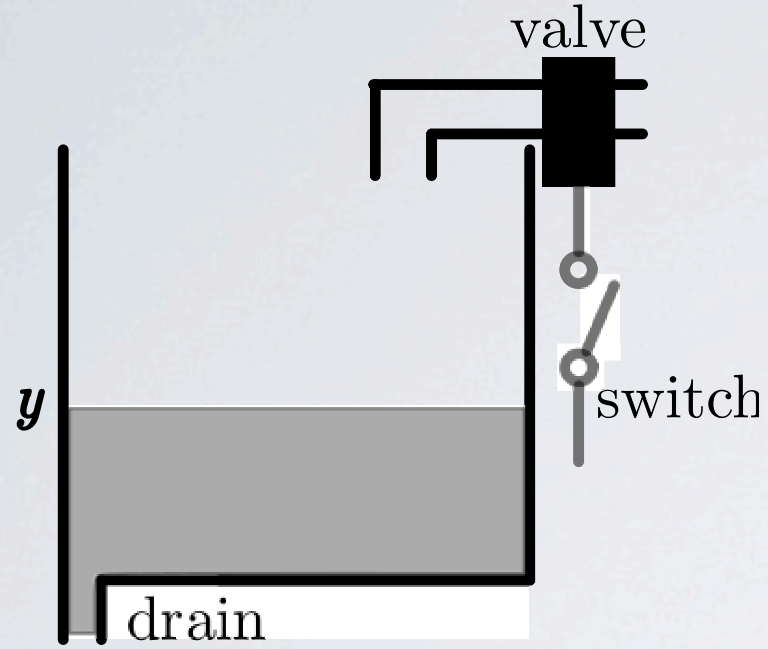
2nd derivative

(Unusual with hybrid automata)

$$
\begin{array}{l}
\texttt{while}\ v > 0\ \texttt{do}\ \{\\
\qquad t := 0\,;\\
\qquad \texttt{if}\ m - z < s\ \texttt{then}\ a := -b\ \texttt{else}\ a := 0\,;\\
\qquad \texttt{while}\ t < \varepsilon\ \texttt{do}\ \{\\
\qquad\qquad t := t + \mathrm{dt}\,;\\
\qquad\qquad v := v + a \cdot \mathrm{dt}\,;\\
\qquad\qquad z := z + v \cdot \mathrm{dt}\\
\qquad \}\}
\end{array}
$$

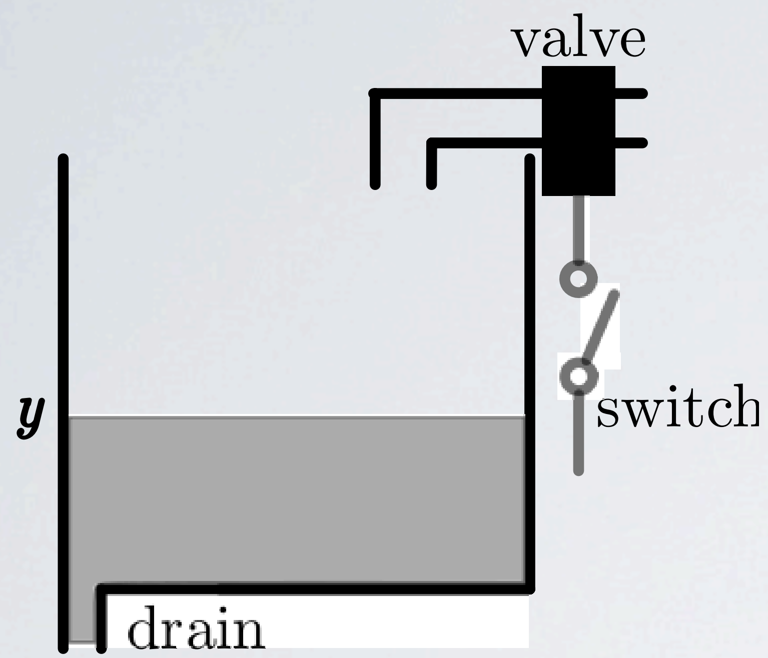
start braking

$\mathbf{While}^{\mathrm{dt}}$: Example [Alur et al. '95]



$$
\begin{array}{l}
x := 0;\ y := 1;\ s := 1;\ v := 1; \\
\texttt{while}\ t < t_{\mathsf{max}}\ \texttt{do}\quad \{ \\
\quad\quad x := x + \mathrm{dt};\quad t := t + \mathrm{dt}; \\
\quad\quad \texttt{if}\ v = 0\ \texttt{then}\ y := y - 2 \cdot \mathrm{dt}\ \texttt{else}\ y := y + \mathrm{dt}; \\
\quad\quad \texttt{case}\quad \{\ \begin{array}{ll}
s = 0\ \wedge\ v = 0\ \wedge\ y \leq 5 : & s := 1;\ x := 0; \\
s = 1\ \wedge\ v = 0\ \wedge\ x \geq 2 : & v := 1; \\
s = 1\ \wedge\ v = 1\ \wedge\ 10 \leq y : & s := 0;\ x := 0; \\
s = 0\ \wedge\ v = 1\ \wedge\ x \geq 2 : & v := 0; \\
\texttt{else} & \texttt{skip}\quad \}\}
\end{array}
\end{array}
$$

$\mathbf{While}^{\mathrm{dt}}$: Example [Alur et al. '95]



More involved jump structure

$$
\begin{array}{l}
x := 0;\; y := 1;\; s := 1;\; v := 1; \\
\texttt{while}\ t < t_{\mathsf{max}}\ \mathsf{do}\quad \{ \\
\qquad x := x + \mathrm{dt};\quad t := t + \mathrm{dt}; \\
\qquad \texttt{if}\ v = 0\ \texttt{then}\ y := y - 2 \cdot \mathrm{dt}\ \texttt{else}\ y := y + \mathrm{dt}; \\
\qquad \texttt{case}\quad \{\ \begin{array}{ll}
s = 0 \wedge v = 0 \wedge y \leq 5 : & s := 1;\; x := 0; \\
s = 1 \wedge v = 0 \wedge x \geq 2 : & v := 1; \\
s = 1 \wedge v = 1 \wedge 10 \leq y : & s := 0;\; x := 0; \\
s = 0 \wedge v = 1 \wedge x \geq 2 : & v := 0; \\
\texttt{else} & \texttt{skip}\quad \}\}
\end{array}
\end{array}
$$

$\mathbf{While}^{\mathrm{dt}}$: Semantics

$$
\begin{array}{l}
\boldsymbol{t} := \mathbf{0}\ ; \\
\texttt{while}\ (t \leq 1)\ \texttt{do}\ \{ \\
\quad t := t + \texttt{dt} \\
\}
\end{array}
$$

While$^{\mathrm{dt}}$: Semantics

$$t := 0\ ;$$
$$\texttt{while}\ (t \leq 1)\ \texttt{do}\ \{$$
$$\quad t := t + \mathtt{dt}$$
$$\}$$

$$t := 0\ ;$$
$$\texttt{while}\ (\texttt{true})\ \texttt{do}\ \{$$
$$\quad t := t + \mathtt{dt}$$
$$\}$$

While$^{\mathrm{dt}}$: Semantics

$$
\begin{array}{l}
\boldsymbol{t} := \boldsymbol{0}\ ; \\
\texttt{while}\ (t \leq 1)\ \texttt{do}\ \{ \\
\quad \boldsymbol{t} := \boldsymbol{t} + \texttt{dt} \\
\}
\end{array}
$$

$$\boldsymbol{t} = \boldsymbol{1} + \texttt{dt}$$

$$
\begin{array}{l}
\boldsymbol{t} := \boldsymbol{0}\ ; \\
\texttt{while}\ (\texttt{true})\ \texttt{do}\ \{ \\
\quad \boldsymbol{t} := \boldsymbol{t} + \texttt{dt} \\
\}
\end{array}
$$

$$\bot\ (\text{divergence})$$

$\mathbf{While}^{\mathrm{dt}}$: Semantics

$t := 0$;
while $(t \leq 1)$ do {
  $t := t + \mathrm{dt}$
}

??

$$t = 1 + \mathrm{dt}$$

$t := 0$;
while (true) do {
  $t := t + \mathrm{dt}$
}

??

$\perp$ (divergence)

While$^{\mathrm{dt}}$: Semantics

$$\begin{array}{l} t := 0\,; \\ \texttt{while}\ (t \leq 1)\ \texttt{do}\ \{ \\ \quad t := t + \mathtt{dt} \\ \} \end{array}$$

??

??

$$\begin{array}{l} t := 0\,; \\ \texttt{while}\ (\texttt{true})\ \texttt{do}\ \{ \\ \quad t := t + \mathtt{dt} \\ \} \end{array}$$

??

$$t = 1 + \mathtt{dt}$$

$\perp$ (divergence)

$\mathbf{While}^{\mathrm{dt}}$: Semantics

$$\begin{array}{l} t := 0\ ; \\ \texttt{while}\ (t \leq 1)\ \texttt{do}\ \{ \\ \quad t := t + \mathtt{dt} \\ \} \end{array}$$

??

??

$$\begin{array}{l} t := 0\ ; \\ \texttt{while}\ (\texttt{true})\ \texttt{do}\ \{ \\ \quad t := t + \mathtt{dt} \\ \} \end{array}$$

??

$$t = 1 + \mathtt{dt}$$

$$\perp \text{ (divergence)}$$

- Non-standard analysis (NSA) for "infinitesimal" dt
- While-loops ➜ sectionwise execution

II

“FIXING NOTATIONS” FOR NON-STANDARD ANALYSIS

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}}/\sim_{\mathcal{F}}\,.$$

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}}/\sim_{\mathcal{F}}\,.$$

- Reals are hyperreals : $\mathbb{R} \hookrightarrow {}^*\mathbb{R}$

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} \,.$$

- Reals are hyperreals : $\mathbb{R} \hookrightarrow {}^*\mathbb{R}$
- There are more than that:
  - An infinite ω : $\forall r \in \mathbb{R}.\ r < \omega$
  - An infinitesimal $\frac{1}{\omega}$: $\forall r \in \mathbb{R}.\ \left(0 < r \implies \frac{1}{\omega} < r\right)$

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} .$$

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$ {}^*\mathbb{R} := \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} . \quad \ni \left[(a_0, a_1, a_2, \ldots) \right] $$

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$ {}^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} . \quad \ni \; \big[\,(a_0, a_1, a_2, \ldots)\,\big] $$

- An infinite

$$ \omega = \big[\,(1, 2, 3, \ldots)\,\big] $$

- An infinitesimal

$$ \omega^{-1} = \Big[\,(1, \frac{1}{2}, \frac{1}{3}, \ldots)\,\Big] $$

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} := \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} . \quad \ni \Big[\,(a_0, a_1, a_2, \ldots)\,\Big]$$

- An infinite $\quad \omega = \big[\,(1, 2, 3, \ldots)\,\big]$
- An infinitesimal $\quad \omega^{-1} = \Big[\,(1, \frac{1}{2}, \frac{1}{3}, \ldots)\,\Big]$
- A real (via $\mathbb{R} \hookrightarrow {}^*\mathbb{R}$) $\quad r = \big[\,(r, r, r, \ldots)\,\big]$

(Prototype of) Hyperreals

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} . \quad \ni \quad \big[\,(a_0, a_1, a_2, \ldots)\,\big]$$

(Prototype of) Hyperreals

Defn.
The set of *hyperreal numbers* is

$$ {}^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} . \quad \ni \; \big[\,(\boldsymbol{a_0}, \boldsymbol{a_1}, \boldsymbol{a_2}, \ldots)\,\big] $$

- "Infinite stream of reals"
  - <u>Operations</u>: pointwise

$$\begin{array}{ll} & \big[\,(a_0, a_1, \ldots)\,\big] \\ + & \big[\,(b_0, b_1, \ldots)\,\big] \\ = & \big[\,(a_0 + b_0, a_1 + b_1, \ldots)\,\big] \end{array}$$

(Prototype of) Hyperreals

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} \; := \; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} \, . \quad \ni \quad \big[\, (\boldsymbol{a_0}, \boldsymbol{a_1}, \boldsymbol{a_2}, \ldots) \,\big]$$

- "Infinite stream of reals"
  - Operations: pointwise

$$\begin{array}{ll} & [\,(\boldsymbol{a_0}, \boldsymbol{a_1}, \ldots)\,] \\ + & [\,(\boldsymbol{b_0}, \boldsymbol{b_1}, \ldots)\,] \\ = & [\,(\boldsymbol{a_0} + \boldsymbol{b_0}, \boldsymbol{a_1} + \boldsymbol{b_1}, \ldots)\,] \end{array}$$

  - Predicates: pointwise, "for almost every i"

$$\begin{array}{lll} [(\boldsymbol{a_i})_{i \in \mathbb{N}}] < [(\boldsymbol{b_i})_{i \in \mathbb{N}}] & & \\ \iff \quad \boldsymbol{a_i} < \boldsymbol{b_i} & & \text{for almost every } \boldsymbol{i} \\ \iff \quad \{\boldsymbol{i} \in \mathbb{N} \mid \boldsymbol{a_i} \not< \boldsymbol{b_i}\} & & \text{is finite} \end{array}$$

(Prototype of) Hyperreals

Defn.
The set of *hyperreal numbers* is

$$ {}^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} \,. \quad \ni \; \Big[\, (\boldsymbol{a_0}, \boldsymbol{a_1}, \boldsymbol{a_2}, \ldots) \,\Big] $$

- "Infinite stream of reals"
  - Operations: pointwise
  - Predicates: pointwise, "for almost every i"

$$
\begin{array}{ll}
 & \big[\,(a_0, a_1, \ldots)\,\big] \\
+ & \big[\,(b_0, b_1, \ldots)\,\big] \\
= & \big[\,(a_0 + b_0, a_1 + b_1, \ldots)\,\big]
\end{array}
$$

$$
\begin{array}{lll}
\big[(a_i)_{i\in\mathbb{N}}\big] < \big[(b_i)_{i\in\mathbb{N}}\big] & & \\
\iff \quad a_i < b_i & & \text{for almost every } i \\
\iff \quad \{ i \in \mathbb{N} \mid a_i \not< b_i \} & & \text{is finite}
\end{array}
$$

"For sufficiently large i"
"Except for finitely many i"

An Infinitesimal

$$\forall r \in \mathbb{R}.\ (\ 0 < r \implies \omega^{-1} < r\ )$$

$$\left(1,\ \frac{1}{2},\ \frac{1}{3},\ \frac{1}{4},\ \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right)$$

$$\wedge$$

$$?$$

$$\left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots\ \frac{1}{N},\ \frac{1}{N},\ \frac{1}{N}, \ldots\right)$$

An Infinitesimal

$$\boxed{\forall r \in \mathbb{R}.\ \left(\ 0 < r \implies \omega^{-1} < r\ \right)}$$

ω^{-1}

$$\left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right)$$

$$\wedge$$

$$? \quad \left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots \quad \frac{1}{N}, \quad \frac{1}{N}, \quad\quad \frac{1}{N}, \ldots \right)$$

An Infinitesimal

$$\forall r \in \mathbb{R}.\ \left(0 < r \implies \omega^{-1} < r\right)$$

ω⁻¹

$$\wedge \quad \left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right)$$

$$? \quad \left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots \quad \frac{1}{N}, \quad \frac{1}{N}, \quad \frac{1}{N}, \ldots\right)$$

$1/N$

An Infinitesimal

$$\boxed{\forall r \in \mathbb{R}.\ (\ 0 < r \implies \omega^{-1} < r\ )}$$

ω^{-1}

$$\left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right)$$

$$\wedge \quad \cancel{\wedge}$$

$$? \quad \left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots\right)$$

$1/N$

An Infinitesimal

$$\boxed{\forall r \in \mathbb{R}.\ \left(0 < r \implies \omega^{-1} < r\right)}$$

ω^{-1}

$$\left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \dots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \dots\right)$$

$\wedge$ $\quad \cancel{\wedge} \quad \cancel{\wedge}$

?

$$\left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots\right)$$

$1/N$

An Infinitesimal

$$\boxed{\forall r \in \mathbb{R}.\ \left(0 < r \implies \omega^{-1} < r \right)}$$

ω^{-1}

$$\wedge \quad \left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right)$$

$$\cancel{\wedge} \quad \cancel{\wedge} \quad \cancel{\wedge}$$

$$? \quad \left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots\right)$$

$1/N$

An Infinitesimal

$$\forall r \in \mathbb{R}.\ (\ 0 < r \implies \omega^{-1} < r\ )$$

ω^-1

$$\wedge \quad \begin{array}{l} \left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right) \\ \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \cancel{\wedge} \\ ?\ \left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots \frac{1}{N}, \frac{1}{N}, \quad \frac{1}{N}, \ldots\right) \end{array}$$

1/N

An Infinitesimal

$$\forall r \in \mathbb{R}.\ \left(\, 0 < r \implies \omega^{-1} < r \,\right)$$

ω^{-1}

$$\left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right)$$

$\wedge$

$\cancel{\wedge}$ $\cancel{\wedge}$ $\cancel{\wedge}$ $\cancel{\wedge}$ ...

?

$$\left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots\right)$$

$1/N$

An Infinitesimal

$$\boxed{\forall r \in \mathbb{R}.\ \left(0 < r \implies \omega^{-1} < r \right)}$$

ω^{-1}

$$\left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \dots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \dots\right)$$

$$\wedge \quad \cancel{\wedge}\ \ \cancel{\wedge}\ \ \cancel{\wedge}\ \ \cancel{\wedge}\ \ \dots\ \ \cancel{\wedge}$$

$$? \quad \left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots\right)$$

$1/N$

An Infinitesimal

$$\boxed{\forall r \in \mathbb{R}.\ \left(0 < r \implies \omega^{-1} < r\right)}$$

$$\left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right)$$

ω^{-1}

$$\wedge \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \ldots \quad \cancel{\wedge} \quad \wedge$$

?

$$\left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots\right)$$

$1/N$

An Infinitesimal

$$\forall r \in \mathbb{R}.\ \left(0 < r \implies \omega^{-1} < r\right)$$

ω^{-1}

$$\left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right)$$

$$\wedge \qquad \cancel{\wedge} \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \ldots \quad \cancel{\wedge} \quad \wedge \quad \wedge$$

?

$$\left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots\right)$$

$1/N$

An Infinitesimal

$$\forall r \in \mathbb{R}.\ (\ 0 < r \implies \omega^{-1} < r\ )$$

$$\left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right)$$

ω^{-1}

$$\overset{?}{\wedge} \quad \cancel{\wedge}\ \cancel{\wedge}\ \cancel{\wedge}\ \cancel{\wedge}\ \ldots\ \cancel{\wedge}\ \wedge\ \wedge\ \ldots$$

$$\left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots\right)$$

$1/N$

An Infinitesimal

$$\boxed{\forall r \in \mathbb{R}.\ \left(0 < r \implies \omega^{-1} < r \right)}$$

ω⁻¹

$$\left(1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots \frac{1}{N}, \frac{1}{N+1}, \frac{1}{N+2}, \ldots\right)$$

$$\wedge \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \ldots \quad \cancel{\wedge} \quad \wedge \quad \wedge \quad \ldots$$

?

✓

$$\left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots \frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \ldots\right)$$

1/N

Infinites

$$(1, 1, 1, 1, \ldots) \overset{?}{>} (1, 2, 3, 4, \ldots)$$

$$(1, 2, 3, 4, \ldots) \overset{?}{>} (0, 1, 2, 3, \ldots)$$

Infinites

I

(1, 1, 1, 1, ...)

^

?

(1, 2, 3, 4, ...)

(1, 2, 3, 4, ...)

^

?

(0, 1, 2, 3, ...)

Infinites

I

(1, 1, 1, 1, ...)

^

?

(1, 2, 3, 4, ...)

ω

(1, 2, 3, 4, ...)

^

?

(0, 1, 2, 3, ...)

Infinites

1

(1, 1, 1, 1, …)

$\wedge$

?

(1, 2, 3, 4, …)

ω

ω

(1, 2, 3, 4, …)

$\wedge$

?

(0, 1, 2, 3, …)

Infinites

1

(1, 1, 1, 1, ...)

$\wedge$
?

(1, 2, 3, 4, ...)

ω

ω

(1, 2, 3, 4, ...)

$\wedge$
?

(0, 1, 2, 3, ...)

ω − 1

Infinites

1

(1, 1, 1, 1, …)

∧ $\cancel{\wedge}$

(1, 2, 3, 4, …)

?

ω

ω

(1, 2, 3, 4, …)

∧

(0, 1, 2, 3, …)

?

ω − 1

Infinites

1

(1, 1, 1, 1, …)

$\wedge$?

$\cancel{\wedge}$ $\wedge$

(1, 2, 3, 4, …)

ω

ω

(1, 2, 3, 4, …)

$\wedge$?

(0, 1, 2, 3, …)

ω − 1

Infinites

1

(1, 1, 1, 1, ...)

^ ? $\cancel{\wedge}$ ^ ^

(1, 2, 3, 4, ...)

ω

ω

(1, 2, 3, 4, ...)

^ ?

(0, 1, 2, 3, ...)

ω − 1

Infinites

1

$(1, 1, 1, 1, \ldots)$

$\wedge$?

$\cancel{\wedge}$ $\wedge$ $\wedge$ $\wedge$

$(1, 2, 3, 4, \ldots)$

ω

ω

$(1, 2, 3, 4, \ldots)$

$\wedge$?

$(0, 1, 2, 3, \ldots)$

$\omega - 1$

Infinites

$\wedge$?

$(1, 1, 1, 1, \ldots)$ 1

$\cancel{\wedge}\ \wedge\ \wedge\ \wedge\ \ldots$

$(1, 2, 3, 4, \ldots)$ ω

$\wedge$?

$(1, 2, 3, 4, \ldots)$ ω

$(0, 1, 2, 3, \ldots)$ ω − 1

Infinites

1

$(1, 1, 1, 1, \ldots)$

$\wedge$ $\cancel{\wedge}$ $\wedge$ $\wedge$ $\wedge$ $\ldots$

$(1, 2, 3, 4, \ldots)$

ω

?

✓

ω

$(1, 2, 3, 4, \ldots)$

$\wedge$

$(0, 1, 2, 3, \ldots)$

ω − 1

?

Infinites

1

$(1, 1, 1, 1, \ldots)$

$\wedge$ $\cancel{\wedge}$ $\wedge$ $\wedge$ $\wedge$ $\ldots$

? ✓

$(1, 2, 3, 4, \ldots)$

ω

ω

$(1, 2, 3, 4, \ldots)$

$\wedge$ $\cancel{\wedge}$

?

$(0, 1, 2, 3, \ldots)$

$\omega - 1$

Infinites

1

$(1, 1, 1, 1, \ldots)$

$\wedge \quad \cancel{\wedge} \;\; \wedge \;\; \wedge \;\; \wedge \;\; \ldots$

?

✓

$(1, 2, 3, 4, \ldots)$

ω

ω

$(1, 2, 3, 4, \ldots)$

$\wedge \quad \cancel{\wedge} \;\; \cancel{\wedge}$

?

$(0, 1, 2, 3, \ldots)$

ω − 1

Infinites

1

$(1, 1, 1, 1, \ldots)$

$\wedge \quad \cancel{\wedge} \quad \wedge \quad \wedge \quad \wedge \quad \ldots$

?

✓

$(1, 2, 3, 4, \ldots)$

ω

ω

$(1, 2, 3, 4, \ldots)$

$\wedge \quad \cancel{\wedge} \quad \cancel{\wedge} \quad \cancel{\wedge}$

?

$(0, 1, 2, 3, \ldots)$

$\omega - 1$

Infinites

1

(1, 1, 1, 1, …)

$\cancel{\wedge}$ $\wedge$ $\wedge$ $\wedge$ …

(1, 2, 3, 4, …)

ω

$\bigwedge$

?

✓

ω

(1, 2, 3, 4, …)

$\cancel{\wedge}$ $\cancel{\wedge}$ $\cancel{\wedge}$ $\cancel{\wedge}$

(0, 1, 2, 3, …)

ω − 1

$\bigwedge$

?

Infinites

1

$$\wedge \qquad \begin{array}{c} (1,\ 1,\ 1,\ 1,\ \ldots) \\ \cancel{\wedge}\ \ \wedge\ \ \wedge\ \ \wedge\ \ \ldots \\ (1,\ 2,\ 3,\ 4,\ \ldots) \end{array}$$

?

✓

ω

ω

$$\wedge \qquad \begin{array}{c} (1,\ 2,\ 3,\ 4,\ \ldots) \\ \cancel{\wedge}\ \ \cancel{\wedge}\ \ \cancel{\wedge}\ \ \cancel{\wedge}\ \ \ldots \\ (0,\ 1,\ 2,\ 3,\ \ldots) \end{array}$$

?

ω − 1

Infinites

$\wedge$

?

✓

1

$$(1, 1, 1, 1, \ldots)$$

$$\cancel{\wedge}\quad \wedge\quad \wedge\quad \wedge\quad \ldots$$

$$(1, 2, 3, 4, \ldots)$$

ω

$\cancel{\wedge}$

?

ω

$$(1, 2, 3, 4, \ldots)$$

$$\cancel{\wedge}\quad \cancel{\wedge}\quad \cancel{\wedge}\quad \cancel{\wedge}\quad \ldots$$

$$(0, 1, 2, 3, \ldots)$$

$\omega - 1$

Trouble... Resolved

$$\mathbf{0} \qquad\qquad \left[\left(\,\mathbf{1}, -\mathbf{1},\ \mathbf{1}, -\mathbf{1}, \ldots\right)\right]$$

Trouble... Resolved

$$\mathbf{0} \quad \begin{matrix} > \\ = \\ < \\ ?? \end{matrix} \quad \left[(\, \mathbf{1}, -\mathbf{1},\ \mathbf{1}, -\mathbf{1}, \ldots \,) \right]$$

Trouble... Resolved

$$\mathbf{0} \quad \begin{matrix} > \\ = \\ < \\ {\color{red}??} \end{matrix} \quad \left[\left(\mathbf{1}, -\mathbf{1}, \mathbf{1}, -\mathbf{1}, \ldots \right)\right]$$

- Meaning of "almost every i" extended
  - ... so that

For each $\boldsymbol{S} \subset \mathbb{N}$, exactly one of

$$\boldsymbol{S} \quad \text{and} \quad \mathbb{N} \setminus \boldsymbol{S}$$

is "almost all $\boldsymbol{i}$."

Trouble... Resolved

$$\mathbf{0} \quad \begin{matrix} > \\ = \\ < \end{matrix} \quad \big[(\,\mathbf{1}, -\mathbf{1},\ \mathbf{1}, -\mathbf{1}, \ldots\,)\big]$$

??

- Meaning of "almost every i" extended
  - ... so that

For each $\boldsymbol{S} \subset \mathbb{N}$, exactly one of

$$\boldsymbol{S} \quad \text{and} \quad \mathbb{N} \setminus \boldsymbol{S}$$

is "almost all $\boldsymbol{i}$."

- ➜ Ultrafilter!

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} \ := \ \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} .$$

Trouble... Resolved

$$\mathbf{0} \quad \begin{matrix} > \\ = \\ < \end{matrix} \quad \big[(\,\mathbf{1}, -\mathbf{1},\ \mathbf{1}, -\mathbf{1}, \dots)\big]$$

??

- Meaning of "almost every i" extended
  - ... so that

For each $\boldsymbol{S} \subset \mathbb{N}$, exactly one of

$$\boldsymbol{S} \quad \text{and} \quad \mathbb{N} \setminus \boldsymbol{S}$$

is "almost all $\boldsymbol{i}$."

- ➜ Ultrafilter!

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} \ := \ \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}}.$$

Filters & Ultrafilters

Defn.
An *ultrafilter* $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is such that:

1. For each $X \subseteq \mathbb{N}$, exactly one of

$$X \quad \text{and} \quad \mathbb{N} \setminus X$$

   is in $\mathcal{F}$.

2. $X, Y \in \mathcal{F} \implies X \cap Y \in \mathcal{F}$
3. $X \in \mathcal{F}, X \subseteq Y \implies Y \in \mathcal{F}$
4. $\emptyset \notin \mathcal{F}$

Filters & Ultrafilters

Given $X \subseteq \mathbb{N}$ is
"yes, almost all!" or "no!"

Defn.
An *ultrafilter* $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is such that:

1. For each $X \subseteq \mathbb{N}$, exactly one of
$$X \quad \text{and} \quad \mathbb{N} \setminus X$$
is in $\mathcal{F}$.
2. $X, Y \in \mathcal{F} \implies X \cap Y \in \mathcal{F}$
3. $X \in \mathcal{F}, X \subseteq Y \implies Y \in \mathcal{F}$
4. $\emptyset \notin \mathcal{F}$

Filters & Ultrafilters

Given $\boldsymbol{X} \subseteq \mathbb{N}$ is
"yes, almost all!" or "no!"

Defn.
An *ultrafilter* $\boldsymbol{\mathcal{F}} \subseteq \boldsymbol{\mathcal{P}}(\mathbb{N})$ is such that:

1. For each $\boldsymbol{X} \subseteq \mathbb{N}$, exactly one of
$$\boldsymbol{X} \quad \text{and} \quad \mathbb{N} \setminus \boldsymbol{X}$$
is in $\boldsymbol{\mathcal{F}}$.
2. $\boldsymbol{X}, \boldsymbol{Y} \in \boldsymbol{\mathcal{F}} \implies \boldsymbol{X} \cap \boldsymbol{Y} \in \boldsymbol{\mathcal{F}}$
3. $\boldsymbol{X} \in \boldsymbol{\mathcal{F}}, \boldsymbol{X} \subseteq \boldsymbol{Y} \implies \boldsymbol{Y} \in \boldsymbol{\mathcal{F}}$
4. $\emptyset \notin \boldsymbol{\mathcal{F}}$

Defn.
A *filter* $\boldsymbol{\mathcal{F}} \subseteq \boldsymbol{\mathcal{P}}(\mathbb{N})$ is that which satisfies Cond. 2.–4.

Filters & Ultrafilters

Given $X \subseteq \mathbb{N}$ is
"yes, almost all!" or "no!"

Defn.
An *ultrafilter* $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is such that:

1. For each $X \subseteq \mathbb{N}$, exactly one of

$$X \quad \text{and} \quad \mathbb{N} \setminus X$$

   is in $\mathcal{F}$.

2. $X, Y \in \mathcal{F} \implies X \cap Y \in \mathcal{F}$
3. $X \in \mathcal{F}, X \subseteq Y \implies Y \in \mathcal{F}$
4. $\emptyset \notin \mathcal{F}$

Defn.
A *filter* $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is that which satisfies Cond. 2.–4.

Prop.

$$\mathcal{F}_c := \{S \subseteq \mathbb{N} \mid \mathbb{N} \setminus S \text{ is finite}\}$$

is a filter (the *cofinite/Frechet* filter).

Filters & Ultrafilters

Given $X \subseteq \mathbb{N}$ is
"yes, almost all!" or "no!"

Defn.
An *ultrafilter* $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is such that:

1. For each $X \subseteq \mathbb{N}$, exactly one of

$$X \quad \text{and} \quad \mathbb{N} \setminus X$$

   is in $\mathcal{F}$.

2. $X, Y \in \mathcal{F} \Longrightarrow X \cap Y \in \mathcal{F}$
3. $X \in \mathcal{F}, X \subseteq Y \Longrightarrow Y \in \mathcal{F}$
4. $\emptyset \notin \mathcal{F}$

Defn.
A *filter* $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is that which satisfies Cond. 2.–4.

Prop.

$$\mathcal{F}_c := \{S \subseteq \mathbb{N} \mid \mathbb{N} \setminus S \text{ is finite}\}$$

is a filter (the *cofinite/Frechet* filter).

Prop.
Any filter $\mathcal{F}'$ can be extended to an ultrafilter $\mathcal{F} \supseteq \mathcal{F}'$. (By Zorn's lemma)

Filters & Ultrafilters

Given $X \subseteq \mathbb{N}$ is
"yes, almost all!" or "no!"

Defn.
An *ultrafilter* $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is such that:

1. For each $X \subseteq \mathbb{N}$, exactly one of
$$X \quad \text{and} \quad \mathbb{N} \setminus X$$
is in $\mathcal{F}$.
2. $X, Y \in \mathcal{F} \implies X \cap Y \in \mathcal{F}$
3. $X \in \mathcal{F}, X \subseteq Y \implies Y \in \mathcal{F}$
4. $\emptyset \notin \mathcal{F}$

Defn.
A *filter* $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is that which satisfies Cond. 2.–4.

Prop.
$$\mathcal{F}_c := \{S \subseteq \mathbb{N} \mid \mathbb{N} \setminus S \text{ is finite}\}$$
is a filter (the *cofinite/Frechet* filter).

Prop.
Any filter $\mathcal{F}'$ can be extended to an ultrafilter $\mathcal{F} \supseteq \mathcal{F}'$. (By Zorn's lemma)

Cor.
There is an ultrafilter $\mathcal{F}$ such that $\mathcal{F}_c \subseteq \mathcal{F}$.

Filters & Ultrafilters

Given $X \subseteq \mathbb{N}$ is "yes, almost all!" or "no!"

Defn.
An *ultrafilter* $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is such that:

1. For each $X \subseteq \mathbb{N}$, exactly one of
$$X \quad \text{and} \quad \mathbb{N} \setminus X$$
is in $\mathcal{F}$.
2. $X, Y \in \mathcal{F} \implies X \cap Y \in \mathcal{F}$
3. $X \in \mathcal{F}, X \subseteq Y \implies Y \in \mathcal{F}$
4. $\emptyset \notin \mathcal{F}$

Defn.
A *filter* $\mathcal{F} \subseteq \mathcal{P}(\mathbb{N})$ is that which satisfies Cond. 2.–4.

Prop.
$$\mathcal{F}_c := \{S \subseteq \mathbb{N} \mid \mathbb{N} \setminus S \text{ is finite}\}$$
is a filter (the *cofinite/Frechet* filter).

Prop.
Any filter $\mathcal{F}'$ can be extended to an ultrafilter $\mathcal{F} \supseteq \mathcal{F}'$. (By Zorn's lemma)

Cor.
There is an ultrafilter $\mathcal{F}$ such that $\mathcal{F}_c \subseteq \mathcal{F}$.

Fix one such

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} \, .$$

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$ {}^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} . $$

$$ (a_0, a_1, \ldots) \sim_{\mathcal{F}} (b_0, b_1, \ldots) $$

$$ \overset{\text{def}}{\iff} \{ i \in \mathbb{N} \mid a_i = b_i \} \in \mathcal{F} $$

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} .$$

$$(a_0, a_1, \ldots) \sim_{\mathcal{F}} (b_0, b_1, \ldots)$$
$$\stackrel{\text{def}}{\Longleftrightarrow} \quad \{i \in \mathbb{N} \mid a_i = b_i\} \in \mathcal{F}$$

- Predicates: pointwise, "for almost every i"

$$\begin{aligned} & [(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \\ \Longleftrightarrow \quad & a_i < b_i \qquad \text{for "almost every } i\text{"} \\ \Longleftrightarrow \quad & \{i \in \mathbb{N} \mid a_i < b_i\} \in \mathcal{F} \end{aligned}$$

(For the other predicates, too)

Hyperreals

Defn.
The set of *hyperreal numbers* is

$$ {}^*\mathbb{R} \;:=\; \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}} . $$

$$ (a_0, a_1, \ldots) \sim_{\mathcal{F}} (b_0, b_1, \ldots) $$
$$ \overset{\text{def}}{\iff} \; \{ i \in \mathbb{N} \mid a_i = b_i \} \in \mathcal{F} $$

- Predicates: pointwise, "for almost every i"

$$ \big[(a_i)_{i \in \mathbb{N}} \big] < \big[(b_i)_{i \in \mathbb{N}} \big] $$
$$ \iff \quad a_i < b_i \qquad \text{for "almost every } i\text{"} $$
$$ \iff \quad \{ i \in \mathbb{N} \mid a_i < b_i \} \in \mathcal{F} $$

(For the other predicates, too)

- Consequences: ω is infinite; ω^{-1} is infinitesimal;
$^*\mathbb{R}$ is an ordered field; [(0,1,0,1,...)] is either 0 or 1; ...

The Sectionwise Paradigm

Thm. (Łos)
For any first-order formula $\varphi(x)$ and a hyperreal $\mathbf{a} = \left[(a_i)_{i\in\mathbb{N}}\right]$,

$$^*\mathbb{R} \models \varphi(\mathbf{a}) \iff \{i \in \mathbb{N} \mid \mathbb{R} \models \varphi(a_i)\} \in \mathcal{F}.$$

The Sectionwise Paradigm

Thm. (Łos) (Constants for reals, not hyperreals)
For any first-order formula $\varphi(x)$ and a hyperreal $\mathbf{a} = \left[(a_i)_{i\in\mathbb{N}}\right]$,

$$^*\mathbb{R} \models \varphi(\mathbf{a}) \quad \Longleftrightarrow \quad \left\{i \in \mathbb{N} \;\middle|\; \mathbb{R} \models \varphi(a_i)\right\} \in \mathcal{F}.$$

The Sectionwise Paradigm

Thm. (Łos) (Constants for reals, not hyperreals)
For any first-order formula $\varphi(x)$ and a hyperreal $\mathbf{a} = \left[(a_i)_{i\in\mathbb{N}}\right]$,

$$^*\mathbb{R} \models \varphi(\mathbf{a}) \iff \{i \in \mathbb{N} \mid \mathbb{R} \models \varphi(a_i)\} \in \mathcal{F}.$$

$$\mathbf{a} = \left[\,(a_0, a_1, a_2, \dots)\,\right]$$

The Sectionwise Paradigm

Thm. (Łos) (Constants for reals, not hyperreals)
For any first-order formula $\varphi(x)$ and a hyperreal $\mathbf{a} = \left[(a_i)_{i\in\mathbb{N}}\right]$,

$$^*\mathbb{R} \models \varphi(\mathbf{a}) \quad \Longleftrightarrow \quad \{i \in \mathbb{N} \mid \mathbb{R} \models \varphi(a_i)\} \in \mathcal{F}.$$

$$\mathbf{a} = \left[\,(a_0, a_1, a_2, \ldots)\,\right]$$

0th section

1st section

2nd section

The Sectionwise Paradigm

Thm. (Łos) (Constants for reals, not hyperreals)
For any first-order formula $\varphi(x)$ and a hyperreal $\mathbf{a} = [(a_i)_{i \in \mathbb{N}}]$,

$$^*\mathbb{R} \models \varphi(\mathbf{a}) \iff \{i \in \mathbb{N} \mid \mathbb{R} \models \varphi(a_i)\} \in \mathcal{F}.$$

$$\mathbf{a} = [\,(a_0, a_1, a_2, \ldots)\,]$$

0th section

1st section

2nd section

a satisfies φ

iff

almost every section does

The Transfer Principle

Thm. (Łos)
For any first-order formula $\varphi(x)$ and a hyperreal $\mathbf{a} = \big[(a_i)_{i\in\mathbb{N}}\big]$,

$$^*\mathbb{R} \models \varphi(\mathbf{a}) \iff \big\{i \in \mathbb{N} \;\big|\; \mathbb{R} \models \varphi(a_i)\big\} \in \mathcal{F}.$$

Thm. (Constants for reals, not hyperreals)
For any first-order sentence φ,

$$\mathbb{R} \models \varphi \iff {}^*\mathbb{R} \models \varphi \;.$$

The Transfer Principle

Thm. (Łos)
For any first-order formula $\varphi(x)$ and a hyperreal $\mathbf{a} = [(a_i)_{i\in\mathbb{N}}]$,

$$^*\mathbb{R} \models \varphi(\mathbf{a}) \iff \{i \in \mathbb{N} \mid \mathbb{R} \models \varphi(a_i)\} \in \mathcal{F}.$$

Thm. (Constants for reals, not hyperreals)
For any first-order sentence φ,

$$\mathbb{R} \models \varphi \iff {}^*\mathbb{R} \models \varphi .$$

$\forall x.\, \psi$

$\forall x.\, \psi$

The Transfer Principle

Thm. (Łos)
For any first-order formula $\varphi(x)$ and a hyperreal $\mathbf{a} = [(a_i)_{i\in\mathbb{N}}]$,

$$^*\mathbb{R} \models \varphi(\mathbf{a}) \iff \{i \in \mathbb{N} \mid \mathbb{R} \models \varphi(a_i)\} \in \mathcal{F}.$$

Thm. (Constants for reals, not hyperreals)
For any first-order sentence φ,

$$\mathbb{R} \models \varphi \iff {}^*\mathbb{R} \models \varphi \; .$$

$\forall x \in \mathbb{R}.\, \psi$

$\forall x \in {}^*\mathbb{R}.\, \psi$

The Transfer Principle

Thm. (Łos)
For any first-order formula $\varphi(x)$ and a hyperreal $\mathbf{a} = [(a_i)_{i \in \mathbb{N}}]$,

$$ {}^*\mathbb{R} \models \varphi(\mathbf{a}) \iff \{ i \in \mathbb{N} \mid \mathbb{R} \models \varphi(a_i) \} \in \mathcal{F}. $$

Thm. (Constants for reals, not hyperreals)
For any first-order sentence φ,

$$ \mathbb{R} \models \varphi \iff {}^*\mathbb{R} \models \varphi \,. $$

$\forall x \in \mathbb{R}.\, \psi$

*-transform

$\forall x \in {}^*\mathbb{R}.\, \psi$

The Transfer Principle

Thm. (Łos)
For any first-order formula $\varphi(x)$ and a hyperreal $\mathbf{a} = [(a_i)_{i \in \mathbb{N}}]$,

$$^*\mathbb{R} \models \varphi(\mathbf{a}) \quad \iff \quad \{i \in \mathbb{N} \mid \mathbb{R} \models \varphi(a_i)\} \in \mathcal{F}.$$

Thm. (Constants for reals, not hyperr

For any first-order sentence φ,

$$\mathbb{R} \models \varphi \quad \Longleftrightarrow \quad ^*\mathbb{R} \models \varphi \; .$$

$$\forall x, y.\, (x < y \lor x = y \lor x > y)$$

$$\forall x.\, (x \neq 0 \implies \exists y.\, (xy = 1))$$

$$\forall x \in \mathbb{R}.\, \psi$$

*-transform

$$\forall x \in {}^*\mathbb{R}.\, \psi$$

THE COAUTHOR

Kohei Suenaga

- PhD (U.Tokyo, 2008)
  with Naoki Kobayashi
- Program verification, static analysis, type systems
- Industiral experience
  (IBM Research)
- Assist. Prof. at Kyoto U. (2012-)

THE COAUTHOR

Kohei Suenaga

- PhD (U.Tokyo, 2008)
  with Naoki Kobayashi
- Program verification, static analysis, type systems
- Industiral experience
  (IBM Research)
- Assist. Prof. at Kyoto U. (2012-)

THE COAUTHOR

Kohei Suenaga

- PhD (U. Tokyo, 2008)
  with Naoki Kobayashi
- Program verification, static analysis, type systems
- Industiral experience
  (IBM Research)
- Assist. Prof. at Kyoto U. (2012-)

THE COAUTHOR

Kohei Suenaga

- PhD (U.Tokyo, 2008) with Naoki Kobayashi
- Program verification, static analysis, type systems
- Industiral experience (IBM Research)
- Assist. Prof. at Kyoto U. (2012-)

THE COAUTHOR

Kohei Suenaga

- PhD (U.Tokyo, 2008)
  with Naoki Kobayashi
- Program verification, static analysis, type systems
- Industiral experience
  (IBM Research)
- Assist. Prof. at Kyoto U. (2012-)

THE COAUTHOR

Kohei Suenaga

- PhD (U.Tokyo, 2008)
  with Naoki Kobayashi
- Program verification, static analysis, type systems
- Industiral experience
  (IBM Research)
- Assist. Prof. at Kyoto U. (2012.4-)

[Fixed pt. obs., Braga, PT, 2007]

III

$\mathbf{While}^{\mathrm{dt}}$

"SECTIONWISE" SEMANTICS

$\mathbf{While}^{\mathrm{dt}}$: Semantics

$$
\begin{array}{l}
t := 0\ ; \\
\texttt{while}\ (t \leq 1)\ \texttt{do}\ \{ \\
\quad t := t + \mathtt{dt} \\
\}
\end{array}
$$

$$
\begin{array}{l}
t := 0\ ; \\
\texttt{while}\ (\texttt{true})\ \texttt{do}\ \{ \\
\quad t := t + \mathtt{dt} \\
\}
\end{array}
$$

$\mathbf{While}^{\mathrm{dt}}$: Semantics

$$
\begin{array}{l}
t := 0\ ; \\
\texttt{while}\ (t \leq 1)\ \texttt{do}\ \{ \\
\quad t := t + \mathtt{dt} \\
\}
\end{array}
$$

$$t = 1 + \mathtt{dt}$$

$$
\begin{array}{l}
t := 0\ ; \\
\texttt{while}\ (\texttt{true})\ \texttt{do}\ \{ \\
\quad t := t + \mathtt{dt} \\
\}
\end{array}
$$

$$\perp\ \text{(divergence)}$$

$\mathbf{While}^{\mathtt{dt}}$: Semantics

$$
\begin{array}{l}
t := 0\ ; \\
\mathtt{while}\ (t \leq 1)\ \mathtt{do}\ \{ \\
\quad t := t + \mathtt{dt} \\
\}
\end{array}
$$

$$t = 1 + \mathtt{dt}$$

$$
\begin{array}{l}
t := 0\ ; \\
\mathtt{while}\ (\mathtt{true})\ \mathtt{do}\ \{ \\
\quad t := t + \mathtt{dt} \\
\}
\end{array}
$$

$[\![\mathtt{dt}]\!]$

$:= \omega^{-1}$

$= [\,(1, \frac{1}{2}, \frac{1}{3}, \ldots)\,]$

$\perp$ (divergence)

$\mathbf{While}^{\mathtt{dt}}$: Semantics

$$\begin{array}{l} t := 0\ ; \\ \mathtt{while}\ (t \leq 1)\ \mathtt{do}\ \{ \\ \quad t := t + \mathtt{dt} \\ \} \end{array}$$

??

$$\begin{array}{l} t := 0\ ; \\ \mathtt{while}\ (\mathtt{true})\ \mathtt{do}\ \{ \\ \quad t := t + \underline{\mathtt{dt}} \\ \} \end{array}$$

$[\![\mathtt{dt}]\!]$

$:= \omega^{-1}$

$= \left[\, (1, \frac{1}{2}, \frac{1}{3}, \ldots) \,\right]$

$t = 1 + \mathtt{dt}$

$\perp$ (divergence)

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := 0;
while (t < 1)
  t := t + dt;
```

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := 0;
while (t < 1)
  t := t + dt;
```

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := (0,0,0,...);
while (t < (1,1,1,...))
```

$$\texttt{t := t +} \left(1, \frac{1}{2}, \frac{1}{3}, \ldots\right) \texttt{;}$$

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

0th section

```
t := 0;
while (t < 1)
   t := t + 1 ;
```

1st section

```
t := 0;
while (t < 1)
   t := t + 1/2 ;
```

2nd section

```
t := 0;
while (t < 1)
   t := t + 1/3 ;
```

...

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

0th section	1st section	2nd section	
`t := 0;` `while (t < 1)` `t := t + 1 ;`	`t := 0;` `while (t < 1)` $\mathtt{t := t +} \frac{1}{2}$ `;`	`t := 0;` `while (t < 1)` $\mathtt{t := t +} \frac{1}{3}$ `;`	...
`t = 1`	`t = 1`	`t = 1`	...

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := (0,0,0,...);
while (t < (1,1,1,...))
```

$$\texttt{t := t +}\left(1, \frac{1}{2}, \frac{1}{3}, \ldots\right) \texttt{;}$$

```
t = (1,1,1,...)
```

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := 0;
while (t < 1)
  t := t + dt;
```

```
t = 1
```

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := 0;
while (t <= 1)
  t := t + dt;
```

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := 0;
while (t <= 1)
  t := t + dt;
```

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := (0,0,0,...);
while (t <= (1,1,1,...))
```

$$\texttt{t := t + } \left(1, \frac{1}{2}, \frac{1}{3}, \ldots\right) \texttt{ ;}$$

Sectionwise Semantics

- **Execute sectionwise** and **bundle up** the outcomes!

0th section

```
t := 0;
while (t <= 1)
  t := t + 1 ;
```

1st section

$$\texttt{t := 0; while (t <= 1)} \quad \texttt{t := t + } \frac{1}{2}\texttt{;}$$

2nd section

$$\texttt{t := 0; while (t <= 1)} \quad \texttt{t := t + } \frac{1}{3}\texttt{;}$$

...

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

0th section	1st section	2nd section	
`t := 0;` `while (t <= 1)` $t := t + 1\ ;$	`t := 0;` `while (t <= 1)` $t := t + \frac{1}{2}\ ;$	`t := 0;` `while (t <= 1)` $t := t + \frac{1}{3}\ ;$	...
$t = 1 + 1$	$t = 1 + \frac{1}{2}$	$t = 1 + \frac{1}{3}$	...

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := (0,0,0,...);
while (t <= (1,1,1,...))

  t := t + (1, 1/2, 1/3, ...) ;
```

$$t = (1,1,1,\ldots) + \left(1, \frac{1}{2}, \frac{1}{3}, \ldots\right)$$

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := 0;
while (t <= 1)
  t := t + dt;
```

```
t = 1 + dt
```

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := 0;
while (true)
  t := t + dt;
```

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := 0;
while (true)
  t := t + dt;
```

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := (0,0,0,...);
while (true)
```

$$\texttt{t := t +} \left(1, \frac{1}{2}, \frac{1}{3}, \ldots\right) \texttt{;}$$

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

0th section

```
t := 0;
while (true)
  t := t + 1 ;
```

1st section

```
t := 0;
while (true)
  t := t + 1/2 ;
```

2nd section

```
t := 0;
while (true)
  t := t + 1/3 ;
```

...

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

0th section	1st section	2nd section	

`t := 0;`
`while (true)`
`t := t +` 1 `;`
	`t := 0;`
	`while (true)`
	`t := t +` $\frac{1}{2}$ `;`
		`t := 0;`
		`while (true)`
		`t := t +` $\frac{1}{3}$ `;`
			...

$\perp$	$\perp$	$\perp$	...

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := (0,0,0,...);
while (true)
```

$$t := t + \left(1, \frac{1}{2}, \frac{1}{3}, \ldots\right);$$

$$t = (\perp, \perp, \perp, \ldots)$$

Sectionwise Semantics

- Execute sectionwise and bundle up the outcomes!

```
t := 0;
while (true)
  t := t + dt;
```

$\perp$

Sectionwise Semantics

Defn. (Section)
Let $\boldsymbol{e}$: a $\textsc{While}^{\mathtt{dt}}$-expr., and $\boldsymbol{i} \in \mathbb{N}$.
$\boldsymbol{e}|_{\boldsymbol{i}}$, the $\boldsymbol{i}$-*th section* of $\boldsymbol{e}$, is obtained by replacing

$$\mathtt{dt} \mapsto \frac{\mathbf{1}}{\boldsymbol{i}+\mathbf{1}} \quad \text{and} \quad \infty \mapsto \boldsymbol{i}+\mathbf{1}\ .$$

Sectionwise Semantics

Defn. (Section)
Let $\boldsymbol{e}$: a $\textsc{While}^{\mathtt{dt}}$-expr., and $\boldsymbol{i} \in \mathbb{N}$.
$\boldsymbol{e}|_{\boldsymbol{i}}$, the $\boldsymbol{i}$*-th section* of $\boldsymbol{e}$, is obtained by replacing

$$\mathtt{dt} \mapsto \frac{\mathbf{1}}{\boldsymbol{i}+\mathbf{1}} \quad \text{and} \quad \infty \mapsto \boldsymbol{i}+\mathbf{1} \ .$$

A **While**-expression
➜ standard semantics

Sectionwise Semantics

Defn. (Section)
Let e: a WHILE$^{\mathsf{dt}}$-expr., and $i \in \mathbb{N}$.
$e|_i$, the i-*th section* of e, is obtained by replacing

$$\mathsf{dt} \mapsto \frac{1}{i+1} \quad \text{and} \quad \infty \mapsto i+1 \ .$$

A **While**-expression
➜ standard semantics

$[\![\texttt{while } b' \texttt{ do } c']\!]\sigma = \sigma' \quad \overset{\text{def}}{\Longleftrightarrow}$

- $\sigma = \sigma' = \bot$;
- there exists a finite sequence $\sigma = \sigma_0, \sigma_1, \ldots, \sigma_n = \sigma'$ such that: $[\![b']\!]\sigma_n = \mathrm{ff}$; and for each $j \in [0, n)$. $\big([\![b']\!]\sigma_j = \mathrm{tt} \ \& \ [\![c']\!]\sigma_j = \sigma_{j+1} \big)$; or
- such a finite sequence does not exist and $\sigma' = \bot$.

$\mathbf{While}^{\mathrm{dt}}$: Denotational Semantics

$$[\![x]\!]\sigma := \sigma(x) \qquad [\![\mathtt{c}_r]\!]\sigma := r \quad \text{for each } r \in \mathbb{R}$$

$$[\![a_1 \ \mathtt{aop}\ a_2]\!]\sigma := [\![a_1]\!]\sigma \ \mathtt{aop}\ [\![a_2]\!]\sigma$$

$$[\![\mathtt{dt}]\!]\sigma := \omega^{-1} = \big[\,(1, \tfrac{1}{2}, \tfrac{1}{3}, \dots)\,\big] \qquad [\![\infty]\!]\sigma := \omega = \big[\,(1, 2, 3, \dots)\,\big]$$

$$[\![\mathtt{true}]\!]\sigma := \mathtt{tt} \qquad [\![\mathtt{false}]\!]\sigma := \mathtt{ff}$$

$$[\![b_1 \wedge b_2]\!]\sigma := [\![b_1]\!]\sigma \wedge [\![b_2]\!]\sigma \qquad [\![\neg b]\!]\sigma := \neg([\![b]\!]\sigma)$$

$$[\![a_1 < a_2]\!]\sigma := [\![a_1]\!]\sigma < [\![a_2]\!]\sigma$$

$$[\![\mathtt{skip}]\!]\sigma := \sigma \qquad [\![x := a]\!]\sigma := \sigma\big[\,x \mapsto [\![a]\!]\sigma\,\big] \qquad [\![c_1; c_2]\!]\sigma := [\![c_2]\!]\big(\,[\![c_1]\!]\sigma\,\big)$$

$$[\![\mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2]\!]\sigma := \begin{cases} [\![c_1]\!]\sigma & \text{if } [\![b]\!]\sigma = \mathtt{tt} \\ [\![c_2]\!]\sigma & \text{if } [\![b]\!]\sigma = \mathtt{ff} \end{cases}$$

$$[\![\mathtt{while}\ b\ \mathtt{do}\ c]\!]\sigma := \Big(\,[\![\,(\mathtt{while}\ b\ \mathtt{do}\ c)|_i\,]\!](\sigma|_i)\,\Big)_{i\in\mathbb{N}}$$

$\mathbf{While}^{\mathrm{dt}}$: Denotational Semantics

Hyperstate (stores hyperreals)

$$
\begin{aligned}
&[\![x]\!]\sigma := \sigma(x) \qquad && [\![\mathtt{c}_r]\!]\sigma := r \quad \text{for each } r \in \mathbb{R}\\
&[\![a_1 \ \mathtt{aop}\ a_2]\!]\sigma := [\![a_1]\!]\sigma \ \mathtt{aop}\ [\![a_2]\!]\sigma &&\\
&[\![\mathtt{dt}]\!]\sigma := \omega^{-1} = \left[\,(1, \tfrac{1}{2}, \tfrac{1}{3}, \dots)\,\right] && [\![\infty]\!]\sigma := \omega = \left[\,(1, 2, 3, \dots)\,\right]
\end{aligned}
$$

$$
\begin{aligned}
&[\![\mathtt{true}]\!]\sigma := \mathrm{tt} \qquad && [\![\mathtt{false}]\!]\sigma := \mathrm{ff}\\
&[\![b_1 \wedge b_2]\!]\sigma := [\![b_1]\!]\sigma \wedge [\![b_2]\!]\sigma && [\![\neg b]\!]\sigma := \neg([\![b]\!]\sigma)\\
&[\![a_1 < a_2]\!]\sigma := [\![a_1]\!]\sigma < [\![a_2]\!]\sigma &&
\end{aligned}
$$

$$
[\![\mathtt{skip}]\!]\sigma := \sigma \qquad [\![x := a]\!]\sigma := \sigma\left[\,x \mapsto [\![a]\!]\sigma\,\right] \qquad [\![c_1; c_2]\!]\sigma := [\![c_2]\!]\left(\,[\![c_1]\!]\sigma\,\right)
$$

$$
[\![\mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2]\!]\sigma := \begin{cases} [\![c_1]\!]\sigma & \text{if } [\![b]\!]\sigma = \mathrm{tt}\\ [\![c_2]\!]\sigma & \text{if } [\![b]\!]\sigma = \mathrm{ff}\end{cases}
$$

$$
[\![\mathtt{while}\ b\ \mathtt{do}\ c]\!]\sigma := \Big(\,[\![\,(\mathtt{while}\ b\ \mathtt{do}\ c)|_i\,]\!](\sigma|_i)\,\Big)_{i \in \mathbb{N}}
$$

While$^{\mathrm{dt}}$: Denotational Semantics

Hyperstate (stores hyperreals)

$$
\begin{array}{lll}
[\![x]\!]\sigma & := & \sigma(x) \\
[\![a_1 \ \mathtt{aop}\ a_2]\!]\sigma & := & [\![a_1]\!]\sigma \ \mathtt{aop}\ [\![a_2]\!]\sigma \\
[\![\mathtt{dt}]\!]\sigma & := & \omega^{-1} = \big[\,(1, \tfrac{1}{2}, \tfrac{1}{3}, \dots)\,\big]
\end{array}
\qquad
\begin{array}{lll}
[\![\mathtt{c}_r]\!]\sigma & := & r \quad \text{for each } r \in \mathbb{R} \\
& & \\
[\![\infty]\!]\sigma & := & \omega = \big[\,(1, 2, 3, \dots)\,\big]
\end{array}
$$

$$
\begin{array}{lll}
[\![\mathtt{true}]\!]\sigma & := & \mathrm{tt} \\
[\![b_1 \wedge b_2]\!]\sigma & := & [\![b_1]\!]\sigma \wedge [\![b_2]\!]\sigma \\
[\![a_1 < a_2]\!]\sigma & := & [\![a_1]\!]\sigma < [\![a_2]\!]\sigma
\end{array}
\qquad
\begin{array}{lll}
[\![\mathtt{false}]\!]\sigma & := & \mathrm{ff} \\
[\![\neg b]\!]\sigma & := & \neg([\![b]\!]\sigma)
\end{array}
$$

$$
[\![\mathtt{skip}]\!]\sigma := \sigma \qquad [\![x := a]\!]\sigma := \sigma\big[\,x \mapsto [\![a]\!]\sigma\,\big] \qquad [\![c_1; c_2]\!]\sigma := [\![c_2]\!]\big([\![c_1]\!]\sigma\big)
$$

$$
[\![\mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2]\!]\sigma := \begin{cases} [\![c_1]\!]\sigma & \text{if } [\![b]\!]\sigma = \mathrm{tt} \\ [\![c_2]\!]\sigma & \text{if } [\![b]\!]\sigma = \mathrm{ff} \end{cases}
$$

$$
[\![\mathtt{while}\ b\ \mathtt{do}\ c]\!]\sigma := \Big(\,\big[\!\big[\,(\mathtt{while}\ b\ \mathtt{do}\ c)|_i\,\big]\!\big](\sigma|_i)\,\Big)_{i \in \mathbb{N}}
$$

Section of a program

$\mathbf{While}^{\mathrm{dt}}$: Denotational Semantics

Hyperstate (stores hyperreals)

$$\llbracket x \rrbracket \sigma := \sigma(x) \qquad \llbracket \mathtt{c}_r \rrbracket \sigma := r \quad \text{for each } r \in \mathbb{R}$$

$$\llbracket a_1 \ \mathtt{aop}\ a_2 \rrbracket \sigma := \llbracket a_1 \rrbracket \sigma \ \mathtt{aop}\ \llbracket a_2 \rrbracket \sigma$$

$$\llbracket \mathtt{dt} \rrbracket \sigma := \omega^{-1} = \left[\, (1, \tfrac{1}{2}, \tfrac{1}{3}, \dots) \,\right] \qquad \llbracket \infty \rrbracket \sigma := \omega = \left[\, (1, 2, 3, \dots) \,\right]$$

$$\llbracket \mathtt{true} \rrbracket \sigma := \mathrm{tt} \qquad \llbracket \mathtt{false} \rrbracket \sigma := \mathrm{ff}$$

$$\llbracket b_1 \wedge b_2 \rrbracket \sigma := \llbracket b_1 \rrbracket \sigma \wedge \llbracket b_2 \rrbracket \sigma \qquad \llbracket \neg b \rrbracket \sigma := \neg(\llbracket b \rrbracket \sigma)$$

$$\llbracket a_1 < a_2 \rrbracket \sigma := \llbracket a_1 \rrbracket \sigma < \llbracket a_2 \rrbracket \sigma$$

$$\llbracket \mathtt{skip} \rrbracket \sigma := \sigma \qquad \llbracket x := a \rrbracket \sigma := \sigma\left[\, x \mapsto \llbracket a \rrbracket \sigma \,\right] \qquad \llbracket c_1 ; c_2 \rrbracket \sigma := \llbracket c_2 \rrbracket \left(\llbracket c_1 \rrbracket \sigma \right)$$

$$\llbracket \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \rrbracket \sigma := \begin{cases} \llbracket c_1 \rrbracket \sigma & \text{if } \llbracket b \rrbracket \sigma = \mathrm{tt} \\ \llbracket c_2 \rrbracket \sigma & \text{if } \llbracket b \rrbracket \sigma = \mathrm{ff} \end{cases}$$

$$\llbracket \mathtt{while}\ b\ \mathtt{do}\ c \rrbracket \sigma := \Big(\llbracket (\mathtt{while}\ b\ \mathtt{do}\ c)|_i \rrbracket (\sigma|_i) \Big)_{i \in \mathbb{N}}$$

Section of a program

Applied to a section of a memory state

$\mathbf{While}^{\mathrm{dt}}$: Denotational Semantics

Hyperstate (stores hyperreals)

$$[\![x]\!]\sigma := \sigma(x) \qquad [\![\mathtt{c}_r]\!]\sigma := r \quad \text{for each } r \in \mathbb{R}$$

$$[\![a_1 \ \mathtt{aop}\ a_2]\!]\sigma := [\![a_1]\!]\sigma \ \mathtt{aop}\ [\![a_2]\!]\sigma$$

$$[\![\mathtt{dt}]\!]\sigma := \omega^{-1} = \left[\,(1, \tfrac{1}{2}, \tfrac{1}{3}, \dots)\,\right] \qquad [\![\infty]\!]\sigma := \omega = \left[\,(1, 2, 3, \dots)\,\right]$$

$$[\![\mathtt{true}]\!]\sigma := \mathrm{tt} \qquad [\![\mathtt{false}]\!]\sigma := \mathrm{ff}$$

$$[\![b_1 \wedge b_2]\!]\sigma := [\![b_1]\!]\sigma \wedge [\![b_2]\!]\sigma \qquad [\![\neg b]\!]\sigma := \neg([\![b]\!]\sigma)$$

$$[\![a_1 < a_2]\!]\sigma := [\![a_1]\!]\sigma < [\![a_2]\!]\sigma$$

$$[\![\mathtt{skip}]\!]\sigma := \sigma \qquad [\![x := a]\!]\sigma := \sigma\left[\,x \mapsto [\![a]\!]\sigma\,\right] \qquad [\![c_1; c_2]\!]\sigma := [\![c_2]\!]\left([\![c_1]\!]\sigma\right)$$

$$[\![\mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2]\!]\sigma := \begin{cases} [\![c_1]\!]\sigma & \text{if } [\![b]\!]\sigma = \mathrm{tt} \\ [\![c_2]\!]\sigma & \text{if } [\![b]\!]\sigma = \mathrm{ff} \end{cases}$$

$$[\![\mathtt{while}\ b\ \mathtt{do}\ c]\!]\sigma := \Big(\,[\![\,\underline{(\mathtt{while}\ b\ \mathtt{do}\ c)|_i}\,]\!](\underline{\sigma|_i})\,\Big)_{\underline{i\in\mathbb{N}}}$$

Bundled up

Section of a program

Applied to a section
of a memory state

"Sectionwise" Lemmas

Sectionwise Execution Lemma. For any expr. $\boldsymbol{e}$ and $\boldsymbol{i} \in \mathbb{N}$,

$$[\![e]\!]\boldsymbol{\sigma} = \big[\, \big([\![e|_{\boldsymbol{i}}]\!](\boldsymbol{\sigma}|_{\boldsymbol{i}}) \big)_{\boldsymbol{i}\in\mathbb{N}} \,\big] \ .$$

Sectionwise Satisfaction Lemma. For any hyperstate $\boldsymbol{\sigma}$ and an $\textsc{Assn}^{\mathrm{dt}}$ formula φ:

$$\boldsymbol{\sigma} \models \varphi \quad \Longleftrightarrow \quad \boldsymbol{\sigma}|_i \models \varphi|_i \quad \text{for almost every } \boldsymbol{i}.$$

Q. Is a **While**^{dt} program executable?

Q. Is a **While**^{dt} program executable?

- A. Not exactly.
  - A modeling language
    - Advantage: close to a common programming style

Q. Does the choice of `dt` matter?

Q. Does the choice of dt matter?

- A. Yes, for "pathological" programs

```
t := 0;
while (t ≠ 1)
  t := t + dt;
```

Terminates with dt = (1, 1/2, 1/3, …)
Doesn't with dt = (1/π, 1/2π, 1/3π, …)

IV

$\mathbf{Assn}^{dt}$ AND $\mathbf{Hoare}^{dt}$

Assertion Language $\mathbf{Assn}^{\mathrm{dt}}$

$$\boldsymbol{A} \quad ::= \quad \mathtt{true} \mid \mathtt{false} \mid \boldsymbol{A_1} \wedge \boldsymbol{A_2} \mid \neg \boldsymbol{A} \mid \boldsymbol{a_1} < \boldsymbol{a_2} \mid \forall \boldsymbol{x} \in {}^*\mathbb{N}.\, \boldsymbol{A} \mid \forall \boldsymbol{x} \in {}^*\mathbb{R}.\, \boldsymbol{A}$$

Assertion Language $\mathbf{Assn}^{\mathrm{dt}}$

$$\boldsymbol{A} \quad ::= \quad \texttt{true} \mid \texttt{false} \mid \boldsymbol{A_1} \wedge \boldsymbol{A_2} \mid \neg \boldsymbol{A} \mid \boldsymbol{a_1} < \boldsymbol{a_2} \mid \underline{\forall \boldsymbol{x} \in {}^*\mathbb{N}.}\ \boldsymbol{A} \mid \underline{\forall \boldsymbol{x} \in {}^*\mathbb{R}.}\ \boldsymbol{A}$$

- Only hyperquantifiers are allowed

Assertion Language $\mathbf{Assn}^{\mathrm{dt}}$

$$\begin{array}{lll} \boldsymbol{A} & ::= & \texttt{true} \mid \texttt{false} \mid \boldsymbol{A_1} \wedge \boldsymbol{A_2} \mid \neg \boldsymbol{A} \mid \boldsymbol{a_1} < \boldsymbol{a_2} \mid \\ & & \forall \boldsymbol{x} \in {}^*\mathbb{N}.\ \boldsymbol{A} \mid \forall \boldsymbol{x} \in {}^*\mathbb{R}.\ \boldsymbol{A} \end{array}$$

- Only hyperquantifiers are allowed
- Standard quantifiers ➜ failure of
  - sectionwise semantics
  - soundness of Hoare logic ...

Assertion Language $\mathbf{Assn}^{\mathrm{dt}}$

$$\begin{aligned} \boldsymbol{A} \quad ::= \quad & \texttt{true} \mid \texttt{false} \mid \boldsymbol{A_1} \wedge \boldsymbol{A_2} \mid \neg \boldsymbol{A} \mid \boldsymbol{a_1} < \boldsymbol{a_2} \mid \\ & \underline{\forall \boldsymbol{x} \in {}^{*}\mathbb{N}.}\ \boldsymbol{A} \mid \underline{\forall \boldsymbol{x} \in {}^{*}\mathbb{R}.}\ \boldsymbol{A} \end{aligned}$$

- Only hyperquantifiers are allowed
- Standard quantifiers ➜ failure of
  - sectionwise semantics
  - soundness of Hoare logic ...
- Drawback: can't say "infinitely close": $\forall r \in \mathbb{R}.\ |x - y| < r$

"Sectionwise" Lemmas

Sectionwise Execution Lemma. For any expr. $\boldsymbol{e}$ and $\boldsymbol{i} \in \mathbb{N}$,

$$[\![e]\!]\boldsymbol{\sigma} = \left[\,\left(\,[\![e|_{i}]\!](\boldsymbol{\sigma}|_{i})\,\right)_{\boldsymbol{i}\in\mathbb{N}}\,\right] .$$

Sectionwise Satisfaction Lemma. For any hyperstate $\boldsymbol{\sigma}$ and an $\textsc{Assn}^{\mathsf{dt}}$ formula $\boldsymbol{\varphi}$:

$$\boldsymbol{\sigma} \models \boldsymbol{\varphi} \quad \Longleftrightarrow$$
$$\boldsymbol{\sigma}|_{i} \models \boldsymbol{\varphi}|_{i} \quad \text{for almost every } \boldsymbol{i}.$$

"Sectionwise" Lemmas

Sectionwise Execution
For any expr. e and $i \in \mathbb{N}$,

$$[\![e]\!]\boldsymbol{\sigma} = \big[\,\big(\,[\![e|_i]\!](\boldsymbol{\sigma}|_i)\,\big)$$

- $\mathtt{dt} \mapsto \frac{1}{i+1}$
  (same for ∞)
- $\forall x \in {}^*\mathbb{R} \quad \mapsto \quad \forall x \in \mathbb{R}$
  (same for $\mathbb{N}$, $\exists$)

Sectionwise Satisfaction Lemma.
For any hyperstate $\boldsymbol{\sigma}$ and an $\textsc{Assn}^{\mathtt{dt}}$ formula φ:

$$\boldsymbol{\sigma} \models \varphi \quad \Longleftrightarrow$$
$$\boldsymbol{\sigma}|_i \models \varphi|_i \quad \text{for almost every } i.$$

“Sectionwise” Lemmas

Sectionwise Execution Lemma. For any expr. $\boldsymbol{e}$ and $\boldsymbol{i} \in \mathbb{N}$,

$$[\![e]\!]\boldsymbol{\sigma} = \left[\, \big(\, [\![e|_i]\!](\boldsymbol{\sigma}|_i)\, \big)_{\boldsymbol{i} \in \mathbb{N}} \,\right] .$$

Sectionwise Satisfaction Lemma. For any hyperstate $\boldsymbol{\sigma}$ and an $\text{ASSN}^{\mathsf{dt}}$ formula $\boldsymbol{\varphi}$:

$$\boldsymbol{\sigma} \models \boldsymbol{\varphi} \quad \Longleftrightarrow \quad \boldsymbol{\sigma}|_i \models \boldsymbol{\varphi}|_i \quad \text{for almost every } \boldsymbol{i}.$$

"Sectionwise" Lemmas

Sectionwise Execution Lemma. For any expr. $\boldsymbol{e}$ and $\boldsymbol{i} \in \mathbb{N}$,

$$[\![e]\!] \boldsymbol{\sigma} = \Big[\Big([\![e|_i]\!] (\boldsymbol{\sigma}|_i) \Big)_{\boldsymbol{i} \in \mathbb{N}} \Big] \ .$$

Sectionwise Satisfaction Lemma. For any hyperstate $\boldsymbol{\sigma}$ and an $\textsc{Assn}^{\mathrm{dt}}$ formula $\boldsymbol{\varphi}$:

$$\boldsymbol{\sigma} \models \boldsymbol{\varphi} \quad \Longleftrightarrow \quad \boldsymbol{\sigma}|_{\boldsymbol{i}} \models \boldsymbol{\varphi}|_{\boldsymbol{i}} \quad \text{for almost every } \boldsymbol{i}.$$

Łos' Theorem

Wednesday, November 9, 2011

Program Logic **Hoare**^{dt}

- Hoare triple

$$\{A\}\, c\, \{B\}$$

- Hoare logic: rule-based derivation of valid Hoare triples
  - ➜ Formal/automated verif. of programs (hybrid systems!)

Program Logic $\mathbf{Hoare}^{\mathrm{dt}}$

- Hoare triple

$$\{A\}\, c\, \{B\}$$

If A holds prior to execution, (precondition)

- Hoare logic: rule-based derivation of valid Hoare triples
  - ➜ Formal/automated verif. of programs (hybrid systems!)

Program Logic **Hoare**^{dt}

- Hoare triple

$$\{A\}\, c\, \{B\}$$

If A holds prior to execution, (precondition)

after successful execution of c,

- Hoare logic: rule-based derivation of valid Hoare triples
  - ➜ Formal/automated verif. of programs (hybrid systems!)

Program Logic $\mathbf{Hoare}^{\mathrm{dt}}$

- Hoare triple

$$\{A\}\, c\, \{B\}$$

If A holds prior to execution, (precondition)

after successful execution of c,

B holds (postcondition)

- Hoare logic: rule-based derivation of valid Hoare triples
  - ➜ Formal/automated verif. of programs (hybrid systems!)

Program Logic $\mathbf{Hoare}^{dt}$

- Hoare triple

$$\{A\}\, c\, \{B\}$$

If A holds prior to execution, (precondition)

after successful execution of c,

B holds (postcondition)

- Hoare logic: rule-based derivation of valid Hoare triples

Program Logic **Hoare**^{dt}

- Hoare triple

$$\{A\}\, c\, \{B\}$$

If A holds prior to execution, (precondition)

after successful execution of c,

B holds (postcondition)

- Hoare logic: rule-based derivation of valid Hoare triples
  - ➜ Formal/automated verif. of programs (hybrid systems!)

Program Logic $\mathbf{Hoare}^{\mathrm{dt}}$

$$\frac{}{\{A\}\ \texttt{skip}\ \{A\}}\ (\textsc{Skip}) \qquad \frac{}{\{\,A[a/x]\,\}\ x := a\ \{A\}}\ (\textsc{Assign})$$

$$\frac{\{A\}\ c_1\ \{C\} \quad \{C\}\ c_2\ \{B\}}{\{A\}\ c_1; c_2\ \{B\}}\ (\textsc{Seq}) \qquad \frac{\{A \wedge b\}\ c_1\ \{B\} \quad \{A \wedge \neg b\}\ c_2\ \{B\}}{\{A\}\ \texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2\ \{B\}}\ (\textsc{If})$$

$$\frac{\{A \wedge b\}\ c\ \{A\}}{\{A\}\ \texttt{while}\ b\ \texttt{do}\ c\ \{A \wedge \neg b\}}\ (\textsc{While}) \qquad \frac{\models A \Rightarrow A' \quad \{A'\}\ c\ \{B'\} \quad \models B' \Rightarrow B}{\{A\}\ c\ \{B\}}\ (\textsc{Conseq})$$

- Precisely the same rules as with **Hoare**

Program Logic **Hoare**^{dt}

$$\frac{}{\{A\}\ \mathtt{skip}\ \{A\}}\ (\textsc{Skip}) \qquad \frac{}{\{\ A[a/x]\ \}\ x := a\ \{A\}}\ (\textsc{Assign})$$

$$\frac{\{A\}\ c_1\ \{C\} \quad \{C\}\ c_2\ \{B\}}{\{A\}\ c_1; c_2\ \{B\}}\ (\textsc{Seq}) \qquad \frac{\{A \wedge b\}\ c_1\ \{B\} \quad \{A \wedge \neg b\}\ c_2\ \{B\}}{\{A\}\ \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2\ \{B\}}\ (\textsc{If})$$

$$\frac{\{A \wedge b\}\ c\ \{A\}}{\{A\}\ \mathtt{while}\ b\ \mathtt{do}\ c\ \{A \wedge \neg b\}}\ (\textsc{While}) \qquad \frac{\models A \Rightarrow A' \quad \{A'\}\ c\ \{B'\} \quad \models B' \Rightarrow B}{\{A\}\ c\ \{B\}}\ (\textsc{Conseq})$$

- Precisely the same rules as with **Hoare**

Thm.
HOARE^{dt} rules are *sound* and *relatively complete*.

"Sectionwise" Lemmas

Sectionwise Execution Lemma. For any expr. $\boldsymbol{e}$ and $\boldsymbol{i} \in \mathbb{N}$,

$$[\![\boldsymbol{e}]\!]\boldsymbol{\sigma} = \left[\, \left(\, [\![\boldsymbol{e}|_{\boldsymbol{i}}]\!](\boldsymbol{\sigma}|_{\boldsymbol{i}})\, \right)_{\boldsymbol{i}\in\mathbb{N}} \,\right] .$$

Sectionwise Satisfaction Lemma. For any hyperstate $\boldsymbol{\sigma}$ and an $\textsc{Assn}^{\mathsf{dt}}$ formula $\boldsymbol{\varphi}$:

$$\boldsymbol{\sigma} \models \boldsymbol{\varphi} \quad \Longleftrightarrow \quad \boldsymbol{\sigma}|_{\boldsymbol{i}} \models \boldsymbol{\varphi}|_{\boldsymbol{i}} \quad \text{for almost every } \boldsymbol{i}.$$

Verification example

```
t := 0; x := 0;
v := 0; a := 1;
while (t < 4) {
  v' := v + a * dt;
  x' := x + v * dt;
  v := v'; x := x';
  if (t < 2) then a := 1
  else a:= -1;
  t := t + dt;
}
```

Verification example

```
t := 0; x := 0;
v := 0; a := 1;
while (t < 4) {
  v' := v + a * dt;
  x' := x + v * dt;
  v := v'; x := x';
  if (t < 2) then a := 1
  else a:= -1;
  t := t + dt;
}
```

Loop invariant:

$\exists n \in {}^*\mathbb{N}.\ t = n * dt\ \&$

$t < 2 + dt \rightarrow$

$v = n * dt\ \&\ a = 1\ \&$

$x = (n-1)n * dt^2 / 2$

$t >= 2 + dt \rightarrow$

$v = (2n_0 + 4 - n) * dt\ \&$

$a = -1\ \&$

$x = x_0 + (3n_0 + 7 - n)(n - n_0 - 2) * dt^2 / 2$

Verification example

```
t := 0; x := 0;
v := 0; a := 1;
while (t < 4) {
  v' := v + a * dt;
  x' := x + v * dt;
  v := v'; x := x';
  if (t < 2) then a := 1
  else a:= -1;
  t := t + dt;
}
```

Loop invariant:

$\exists n \in {}^*\mathbb{N}.\ t = n * dt\ \&$

$t < 2 + dt \rightarrow$

$v = n * dt\ \&\ a = 1\ \&$

$x = (n-1)n * dt^2 / 2$

$t >= 2 + dt \rightarrow$

$v = (2n_0 + 4 - n) * dt\ \&$

$a = -1\ \&$

$x = x_0 + (3n_0 + 7 - n)(n - n_0 - 2) * dt^2 / 2$

Verification example

```
t := 0; x := 0;
v := 0; a := 1;
while (t < 4) {
  v' := v + a * dt;
  x' := x + v * dt;
  v := v'; x := x';
  if (t < 2) then a := 1
  else a:= -1;
  t := t + dt;
}
```

Loop invariant:

$\exists n \in {}^*\mathbb{N}.\ t = n * dt\ \&$

$t < 2 + dt \rightarrow$

$v = n * dt\ \&\ a = 1\ \&$

$x = (n-1)n * dt^2 / 2$

$t >= 2 + dt \rightarrow$

$v = (2n_0 + 4 - n) * dt\ \&$

$a = -1\ \&$

$x = x_0 + (3n_0 + 7 - n)(n - n_0 - 2) * dt^2 / 2$

Verification example

$\vdots$ Using the loop invariant

$\{\mathbf{true}\}$

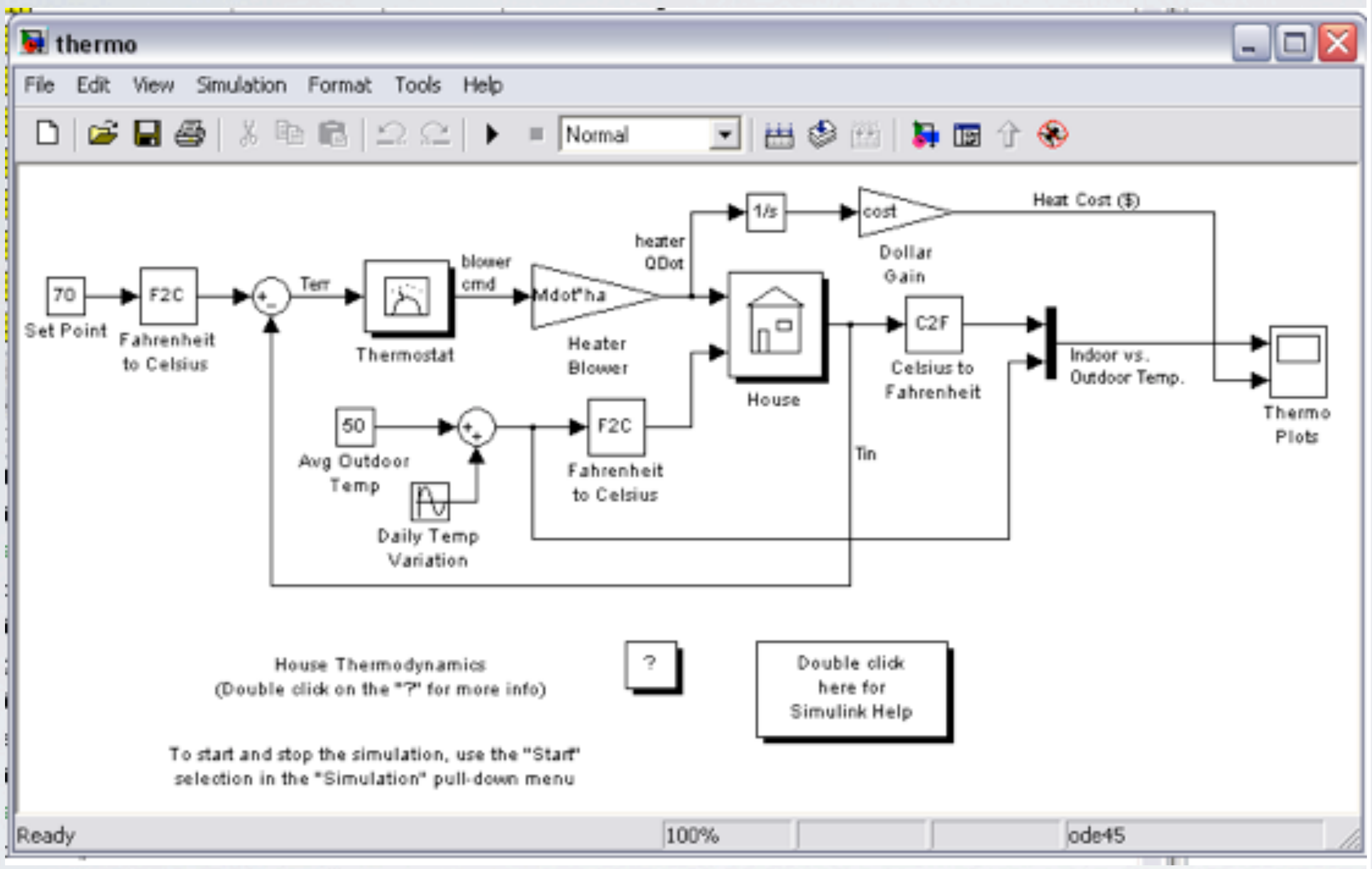
```
t := 0; x := 0; v := 0; a := 1;
while (t < 4) {
  v' := v + a * dt;
  x' := x + v * dt;
  v := v'; x := x';
  t := t + dt;
  a := (t < 2) ? 1 : -1;
}
```

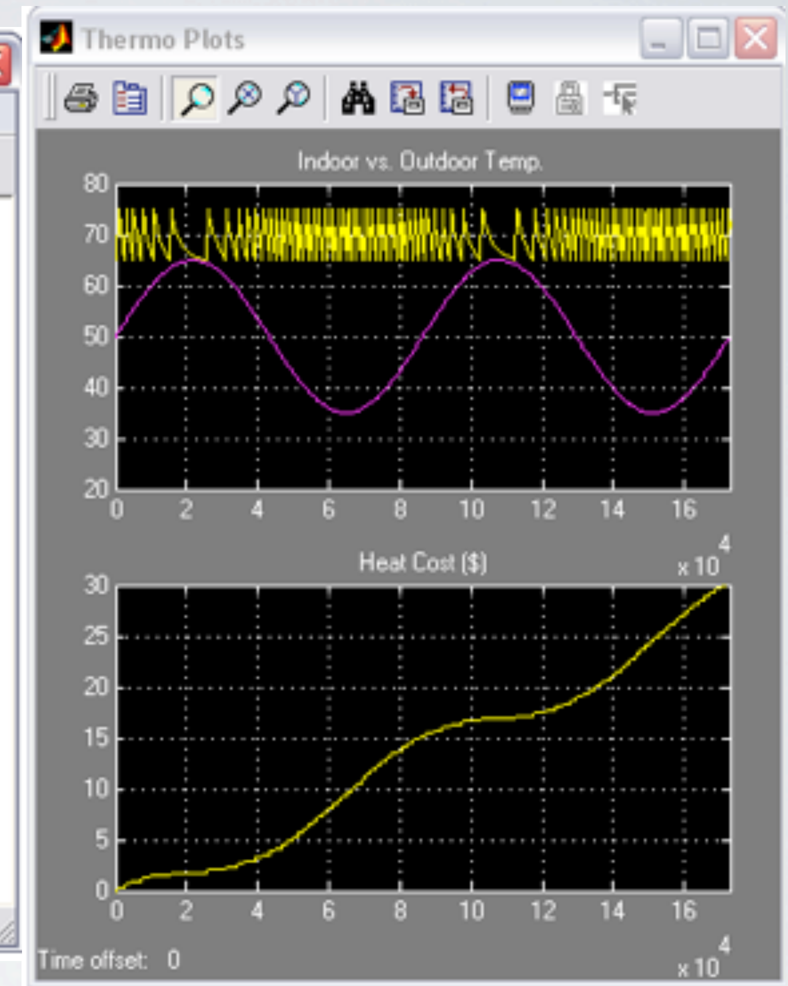
$\{x < 4.01\}$

V

RELATED & FUTURE WORK

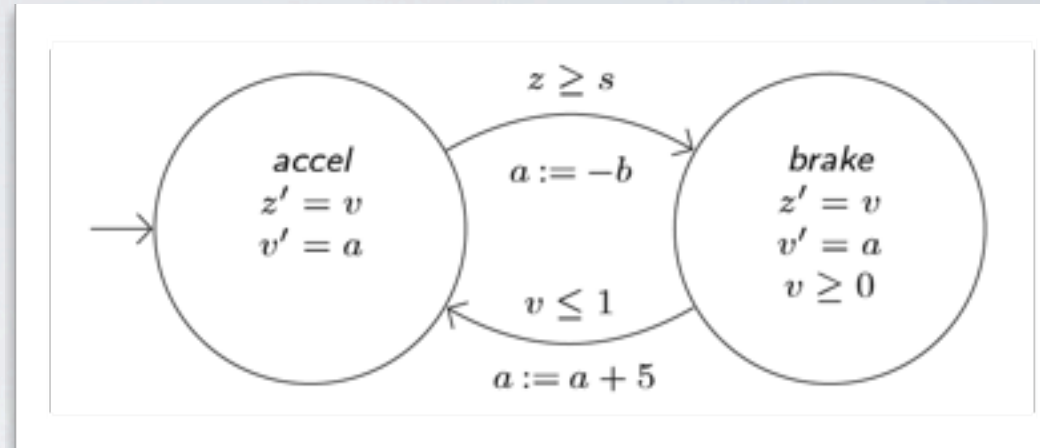
Related Work



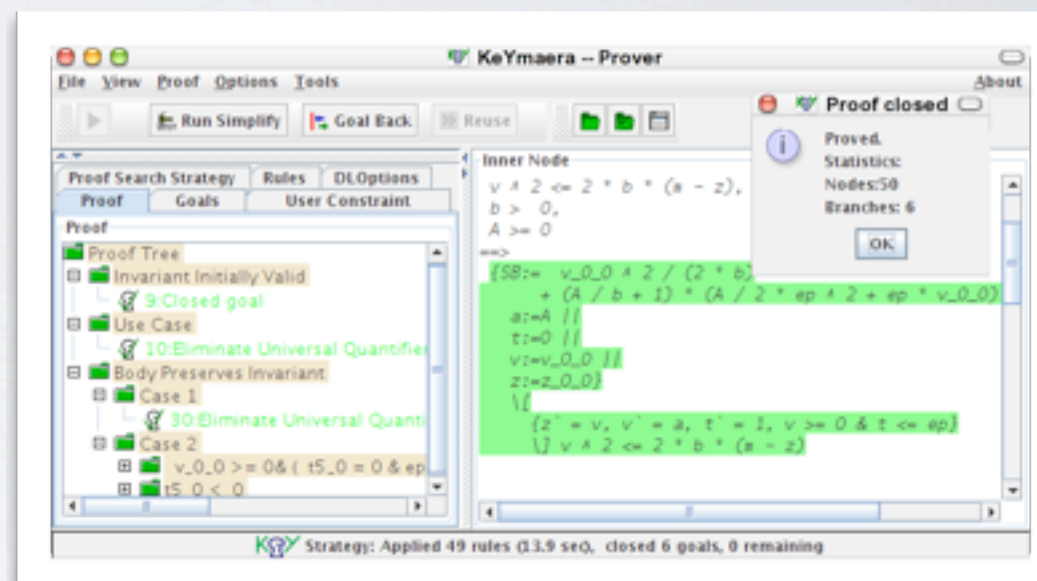


- Simulink [Mathwork Inc.]
  - Industrial standard for hybrid system design
  - Control-theory oriented
  - Test/simulation, rather than verification

Related Work



- Hybrid automaton [Alur & others, '90s-]
  - Model-checking: "push-button" verif. algorithms
  - Flow-dynamics restricted



- Differential dynamic logic [Platzer & others, '07-]
  - Dynamic logic (≈ Hoare logic) + differential equations
  - Automatic prover KeYmaera

Related Work

- Use of NSA for hybrid systems
  [Benveniste et al., Bliudze & Krob, Nakamura & Fusaoka, Rust]

  - Ours: clean integration with existing verification framework; actually proves something!

Future Work: Practical

- Automatic prover (prototype under development)
- (Any program verification techniques)^{dt}
- Simulink as (a stream processing language)^{dt}

Future Work: Theoretical

- Internalize the whole framework in a topos
  - Ultrapower construction of toposes?
- Characterize computing power of $\mathbf{While}^{\mathrm{dt}}$
  - Preliminary results: [Miyabe-Suenaga]

Summary

While

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn

First-order assertion lang.

∃z(x=2*z ∧ y=3*z)

Hoare

Hoare-style program logic

$$\frac{\{A \wedge b\}\; c\; \{A\}}{\{A\}\; \texttt{while}\; b\; \texttt{do}\; c\; \{A \wedge \neg b\}}$$

Summary

While^{dt}

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn^{dt}

First-order assertion lang.

$\exists$z(x=2*z $\wedge$ y=3*z)

Hoare^{dt}

Hoare-style program logic

$$\frac{\{A \wedge b\}\, c\, \{A\}}{\{A\}\ \texttt{while}\ b\ \texttt{do}\ c\, \{A \wedge \neg b\}}$$

Summary

While$^{\mathrm{dt}}$

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn$^{\mathrm{dt}}$

First-order assertion lang.

$\exists z(x=2*z \wedge y=3*z)$

Hoare$^{\mathrm{dt}}$

Hoare-style program logic

$$\frac{\{A \wedge b\}\, c\, \{A\}}{\{A\}\ \texttt{while}\ b\ \texttt{do}\ c\, \{A \wedge \neg b\}}$$

Rigorous semantics by non-standard analysis

Summary

While$^{\text{dt}}$

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn$^{\text{dt}}$

First-order assertion lang.

∃z(x=2*z ∧ y=3*z)

Hoare$^{\text{dt}}$

Hoare-style program logic

$$\frac{\{A \wedge b\}\, c\, \{A\}}{\{A\}\ \texttt{while}\ b\ \texttt{do}\ c\, \{A \wedge \neg b\}}$$

Rigorous semantics by non-standard analysis

- **Hoare**$^{\text{dt}}$: sound and relatively complete
- Program verification/static analysis of hybrid systems
- Actual verification with NSA

Thank you for your attention!
Ichiro Hasuo (Dept. CS, U Tokyo)
http://www-mmm.is.s.u-tokyo.ac.jp/~ichiro/

Summary

While$^{\mathrm{dt}}$

Programming lang.

```
while (t<a) do {
  t:=t+1;
  if ...
}
```

Assn$^{\mathrm{dt}}$

First-order assertion lang.

$\exists$z(x=2*z $\wedge$ y=3*z)

Hoare$^{\mathrm{dt}}$

Hoare-style program logic

$$\frac{\{A \wedge b\}\, c\, \{A\}}{\{A\}\, \texttt{while}\, b\, \texttt{do}\, c\, \{A \wedge \neg b\}}$$

Rigorous semantics by non-standard analysis

- **Hoare**$^{\mathrm{dt}}$: sound and relatively complete
- Program verification/static analysis of hybrid systems
- Actual verification with NSA