[1] KS and IH. **Programming with Infinitesimals: A While-Language for Hybrid System Modeling**. Proc. ICALP 2011, Track B.

[2] IH and KS. **Exercises in Nonstandard Static Analysis of Hybrid Systems**. Proc. CAV 2012.

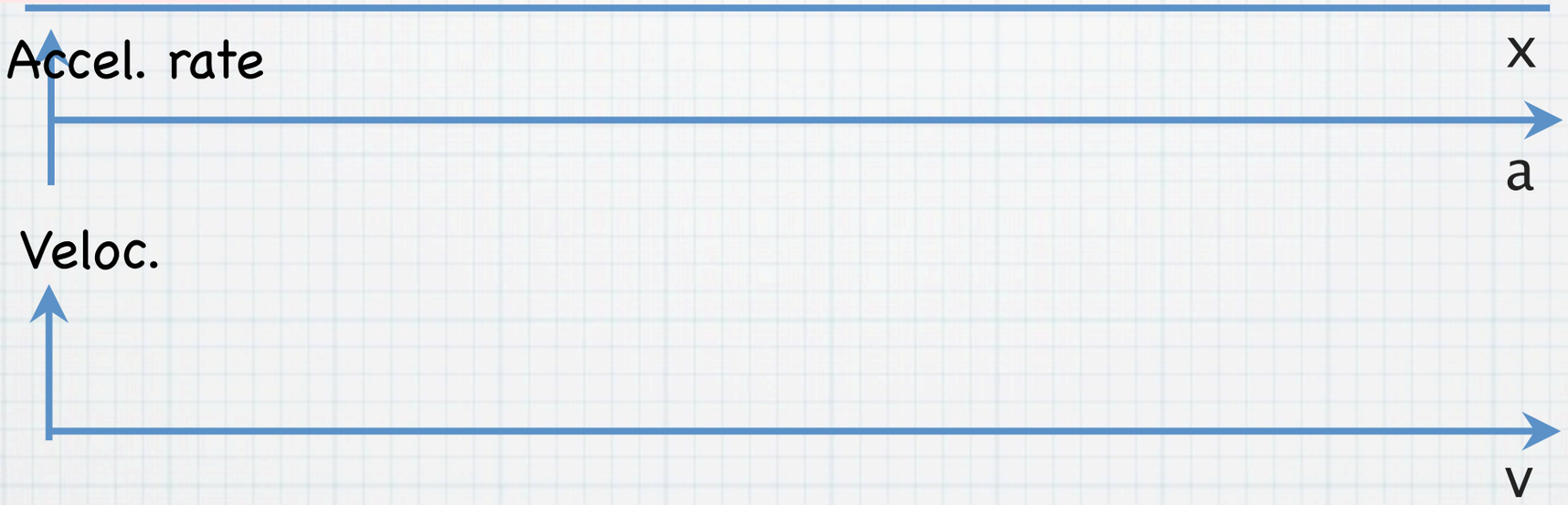
Nonstandard Static Analysis

Transfer Verification to Hybrid Systems

Ichiro Hasuo
University of Tokyo (JP)

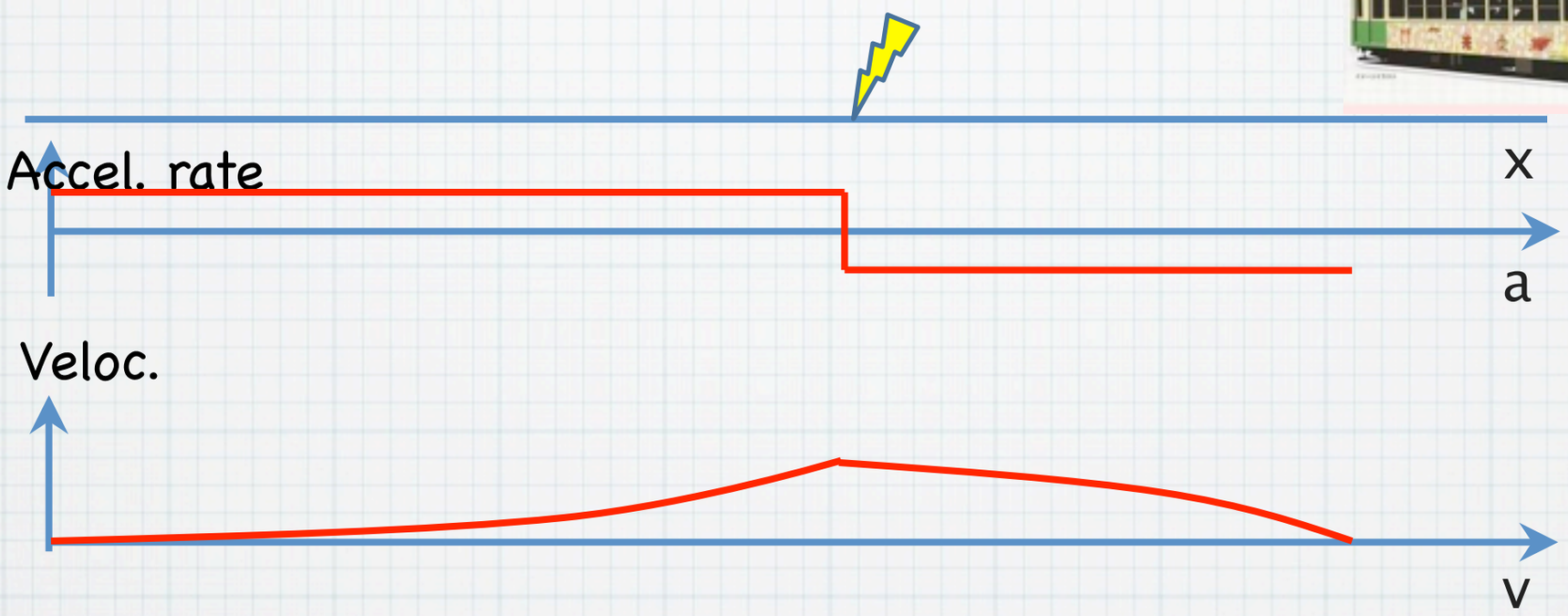
Kohei Suenaga
Kyoto University (JP)

Hybrid System



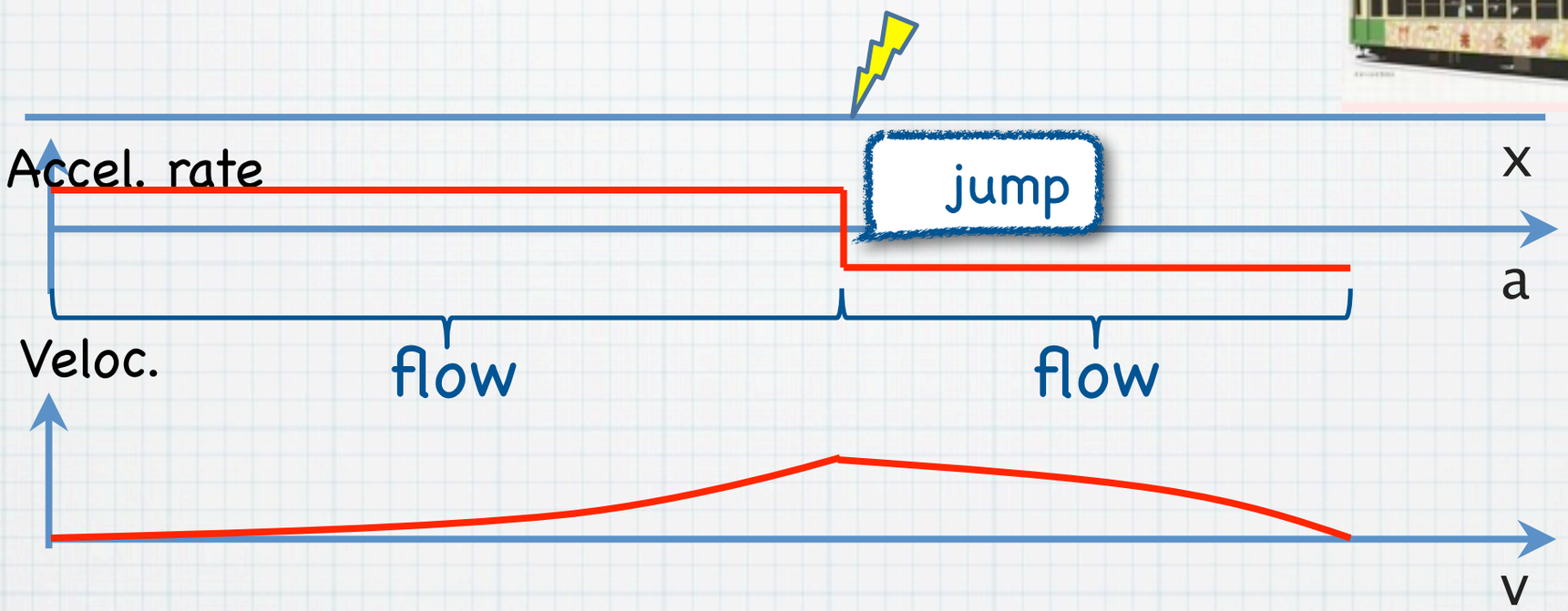
Hasuo (Tokyo)

Hybrid System



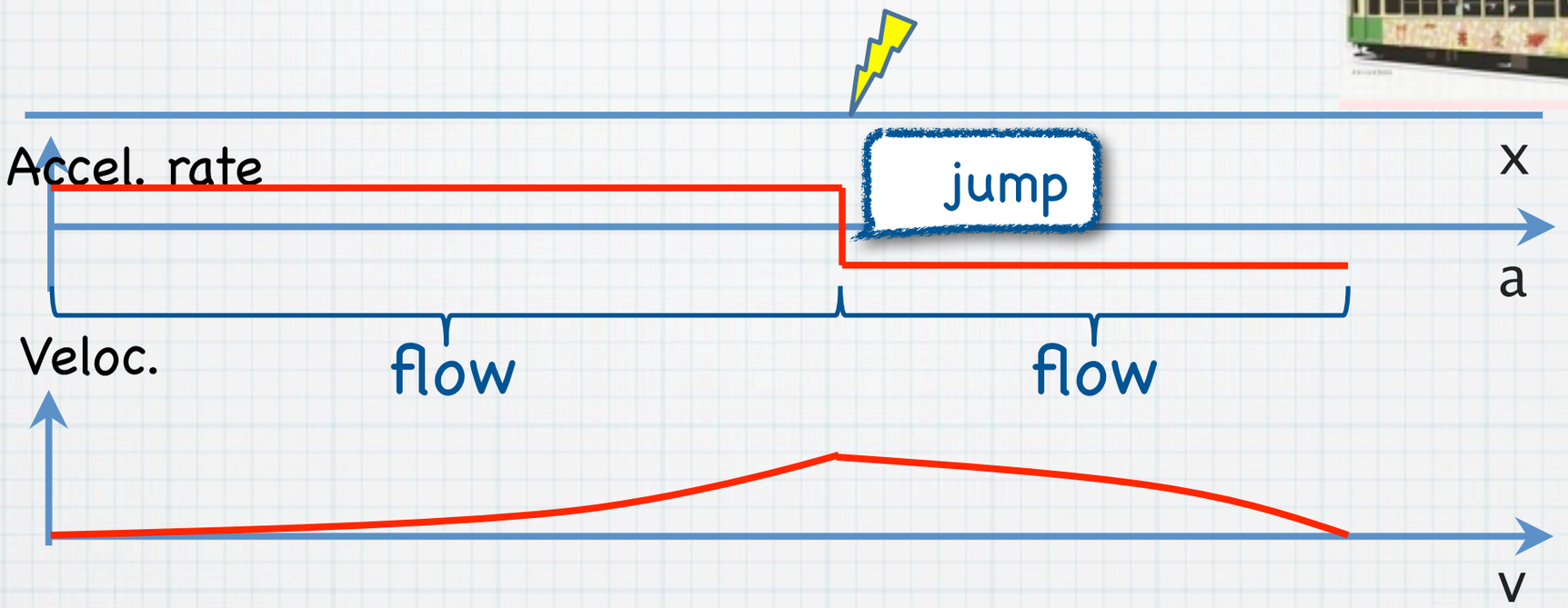
Hasuo (Tokyo)

Hybrid System



Hasuo (Tokyo)

Hybrid System



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

Hasuo (Tokyo)

Hybrid System

Discrete

"jump"

and

Continuous

"flow"

Hasuo (Tokyo)

Hybrid System

Discrete

"jump"

and

Continuous

"flow"

Hasuo (Tokyo)

Hybrid System

Formal verification
(computer science)

Discrete
"**jump**"

and

Continuous
"**flow**"

Control theory
(applied analysis)

Hasuo (Tokyo)

Hybrid System

Formal verification
(computer science)

Hybrid!

Discrete
"**jump**"

and

Continuous
"**flow**"

Hybrid!

Control theory
(applied analysis)

Hasuo (Tokyo)

Hybrid System

Formal verification
(computer science)

Hybrid!

- Flow?
- With minimal cost?

Discrete
"**jump**"

and

Continuous
"**flow**"

Hybrid!

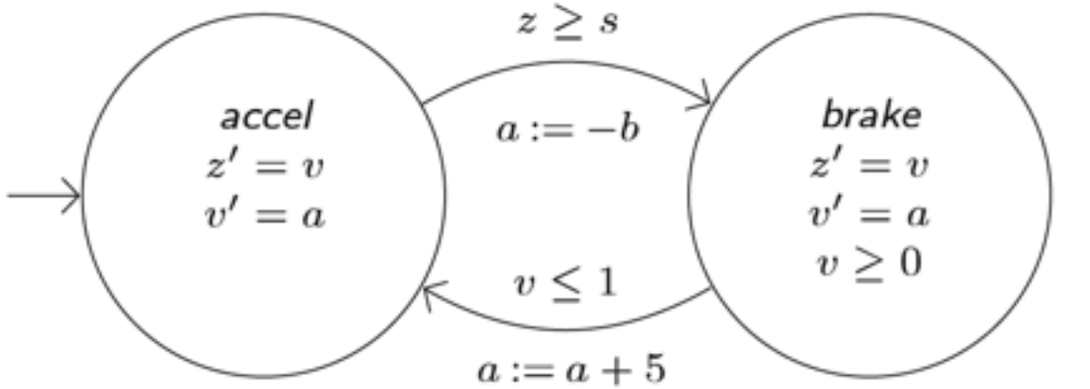
Control theory
(applied analysis)

Hasuo (Tokyo)

Formal Verification Approaches

* Hybrid automata

  [Alur, Henzinger, ...; '90s-]



* Differential dynamic logic

  [Platzer & others, '07-]

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

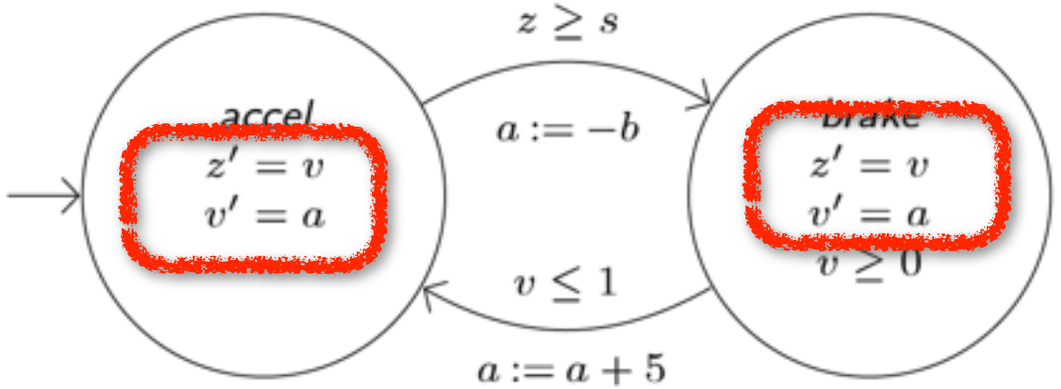
* Differential equations, explicitly

  ➜ distinction jump vs. flow

Formal Verification Approaches

* Hybrid automata

  [Alur, Henzinger, …; '90s-]



* Differential dynamic logic

  [Platzer & others, '07-]

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

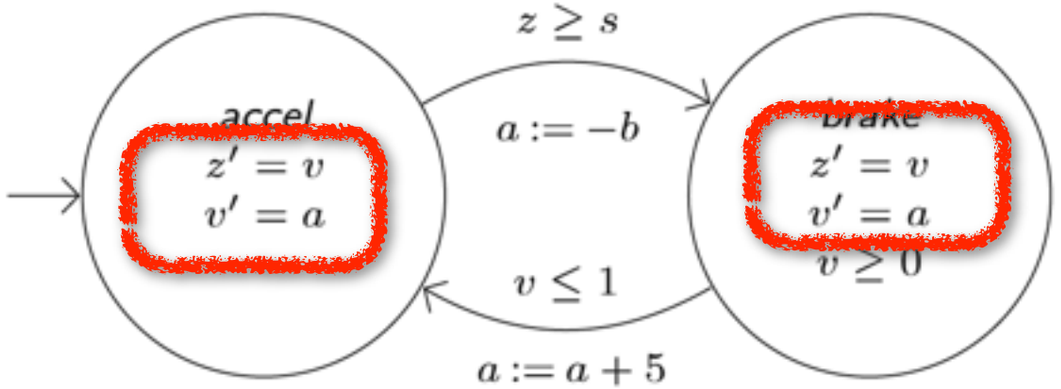
* Differential equations, explicitly

  ➜ distinction jump vs. flow

Formal Verification Approaches

* Hybrid automata

  [Alur, Henzinger, …; '90s-]



* Differential dynamic logic

  [Platzer & others, '07-]

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

* Differential equations, explicitly

  ➜ distinction jump vs. flow

"Turn Flow into Jump"

```
t := 0 ;
while (t ≤ 1) do {
   t := t + dt
}
```

Hasuo (Tokyo)

"Turn Flow into Jump"

```
t := 0 ;
while (t ≤ 1) do {
   t := t + dt
}
```

* Infinitesimal number dt

* "Infinitely small" :

  0 < dt < r

  for **any** positive real r

Hasuo (Tokyo)

"Turn Flow into Jump"

```
t := 0 ;
while (t ≤ 1) do {
   t := t + dt
}
```

* Infinitesimal number dt
* "Infinitely small" :

  0 < dt < r

  for **any** positive real r
* t = 1 after the execution?

Hasuo (Tokyo)

"Turn Flow into Jump"

```
t := 0 ;
while (t ≤ 1) do {
   t := t + dt
}
```

* Infinitesimal number dt
* "Infinitely small" :

  0 < dt < r

  for **any** positive real r
* t = 1 after the execution?
* Non-standard analysis!

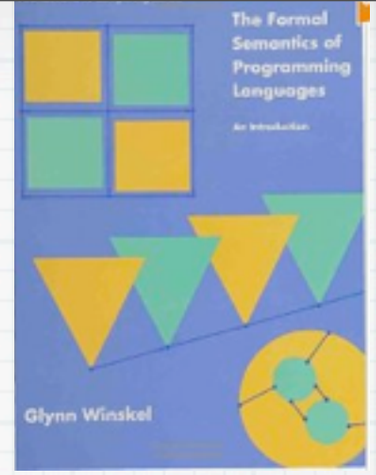
  [Robinson '60s]

Hasuo (Tokyo)

Theoretical Framework [Suenaga&H., ICALP'11]

Theoretical Framework

[Suenaga&H., ICALP'11]

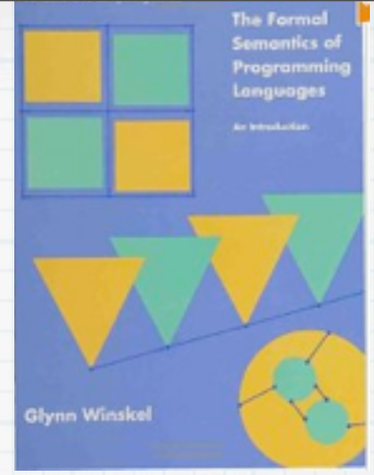


The standard textbook [Winskel]

Hasuo (Tokyo)

Theoretical Framework

[Suenaga&H., ICALP'11]



The standard textbook [Winskel]

While

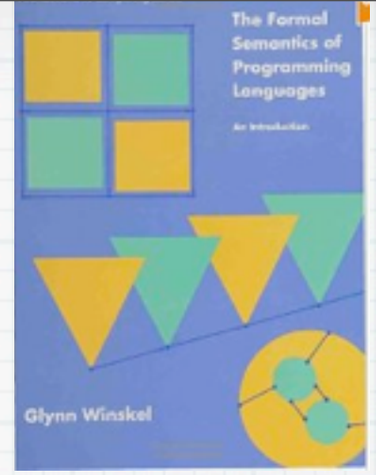
Programming lang.

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

Hasuo (Tokyo)

Theoretical Framework

[Suenaga&H., ICALP'11]



The standard textbook [Winskel]

While

Programming lang.

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

Assn

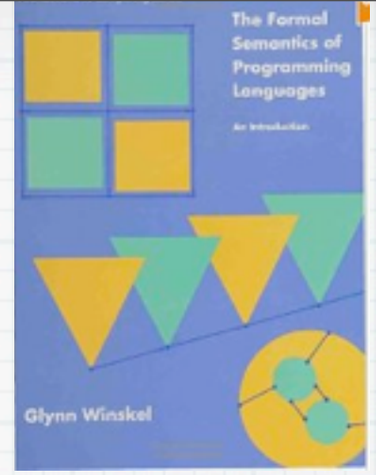
First-order assertion lang.

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

Hasuo (Tokyo)

Theoretical Framework

[Suenaga&H., ICALP'11]



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

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

Hasuo (Tokyo)

Theoretical Framework

[Suenaga&H., ICALP'11]

The standard textbook [Winskel]

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

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

Theoretical Framework

[Suenaga&H., ICALP'11]

The standard textbook [Winskel]

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

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

Rigorous semantics by non-standard analysis

Hasuo (Tokyo)

Theoretical Framework

[Suenaga&H., ICALP'11]

The standard textbook [Winskel]

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

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

Rigorous semantics by non-standard analysis

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

Theoretical Framework

[Suenaga&H., ICALP'11]

The standard textbook [Winskel]

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

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

Rigorous semantics by non-standard analysis

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

Theoretical Framework

[Suenaga&H., ICALP'11]

The standard textbook [Winskel]

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

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

Rigorous semantics by non-standard analysis

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

Hasuo (Tokyo)

Static Analysis

Nonstandard Analysis

Hasuo (Tokyo)

Static Analysis

Program Verif. Techniques

* Esp. invariant discovery

Nonstandard Analysis

Infinitesimal dt

Hasuo (Tokyo)

Nonstandard Static Analysis

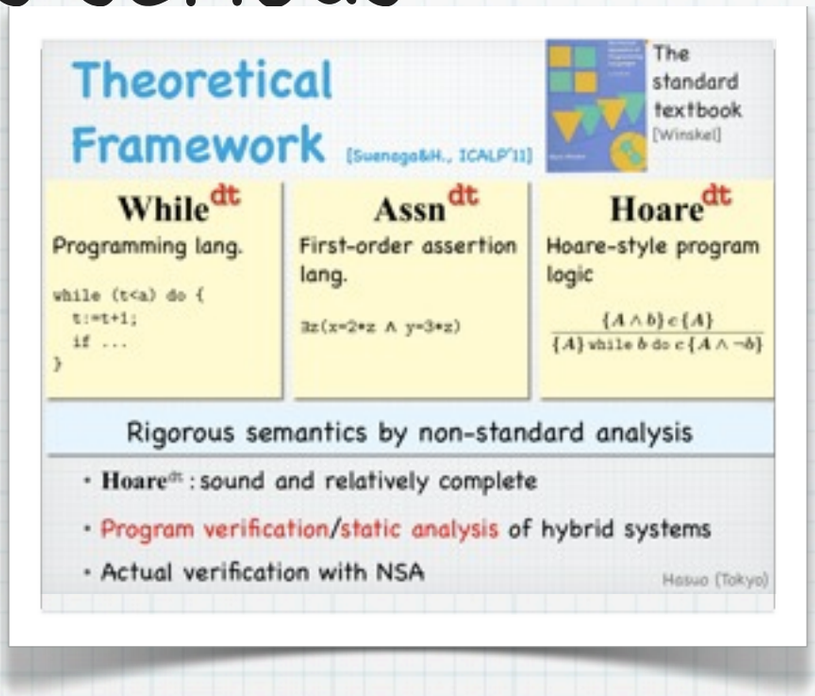
Hasuo (Tokyo)

Nonstandard Static Analysis

Hasuo (Tokyo)

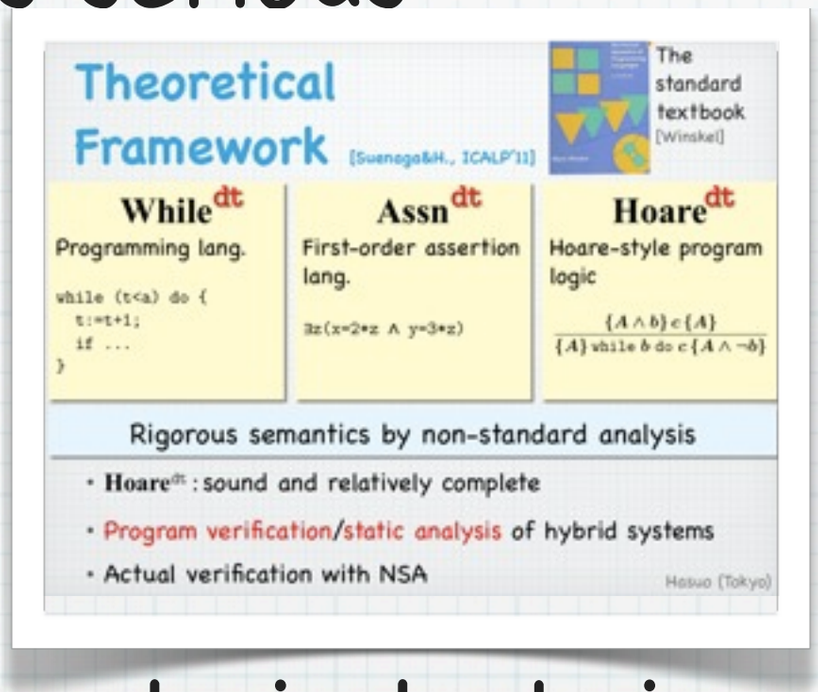
Nonstandard Static Analysis

* Towards serious use of



Nonstandard Static Analysis

* Towards serious use of

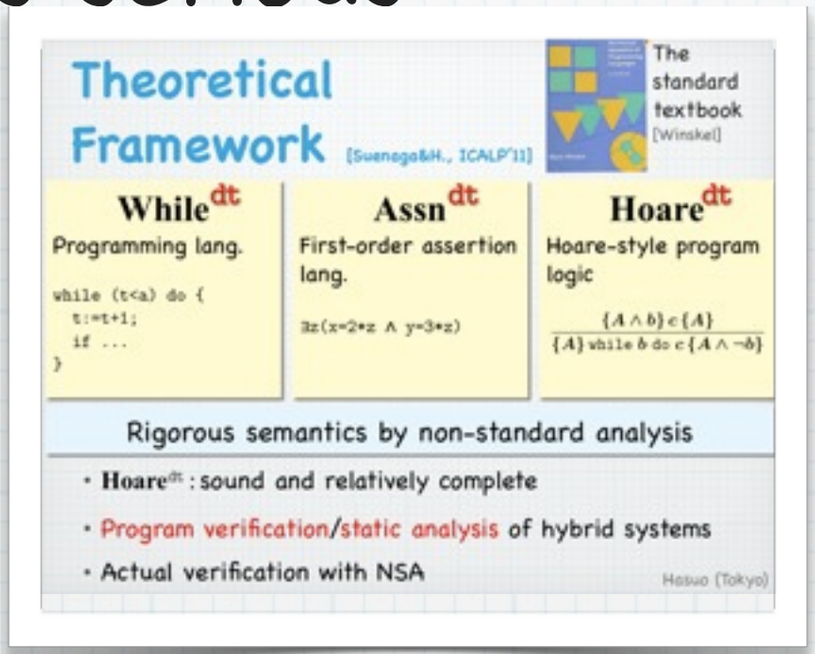


* Static analysis techniques **transferred** to hybrid appl.

Hasuo (Tokyo)

Nonstandard Static Analysis

* Towards serious use of



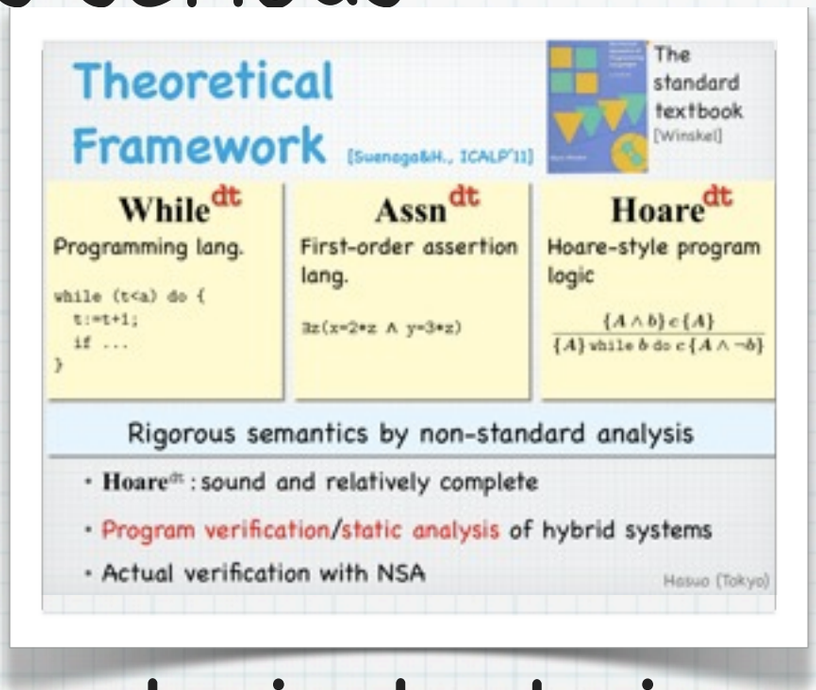
Exactly as they are!

* Static analysis techniques **transferred** to hybrid appl.

Hasuo (Tokyo)

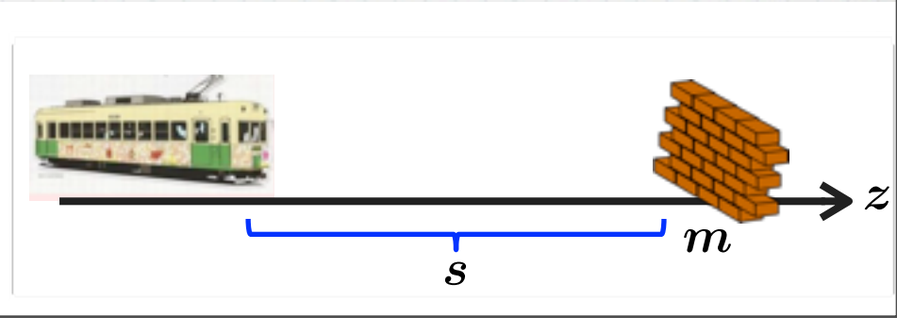
Nonstandard Static Analysis

* Towards serious use of



Exactly as they are!

* Static analysis techniques **transferred** to hybrid appl.
* Leading example: ETCS



Prototype Automatic Prover

Hasuo (Tokyo)

Prototype Automatic Prover

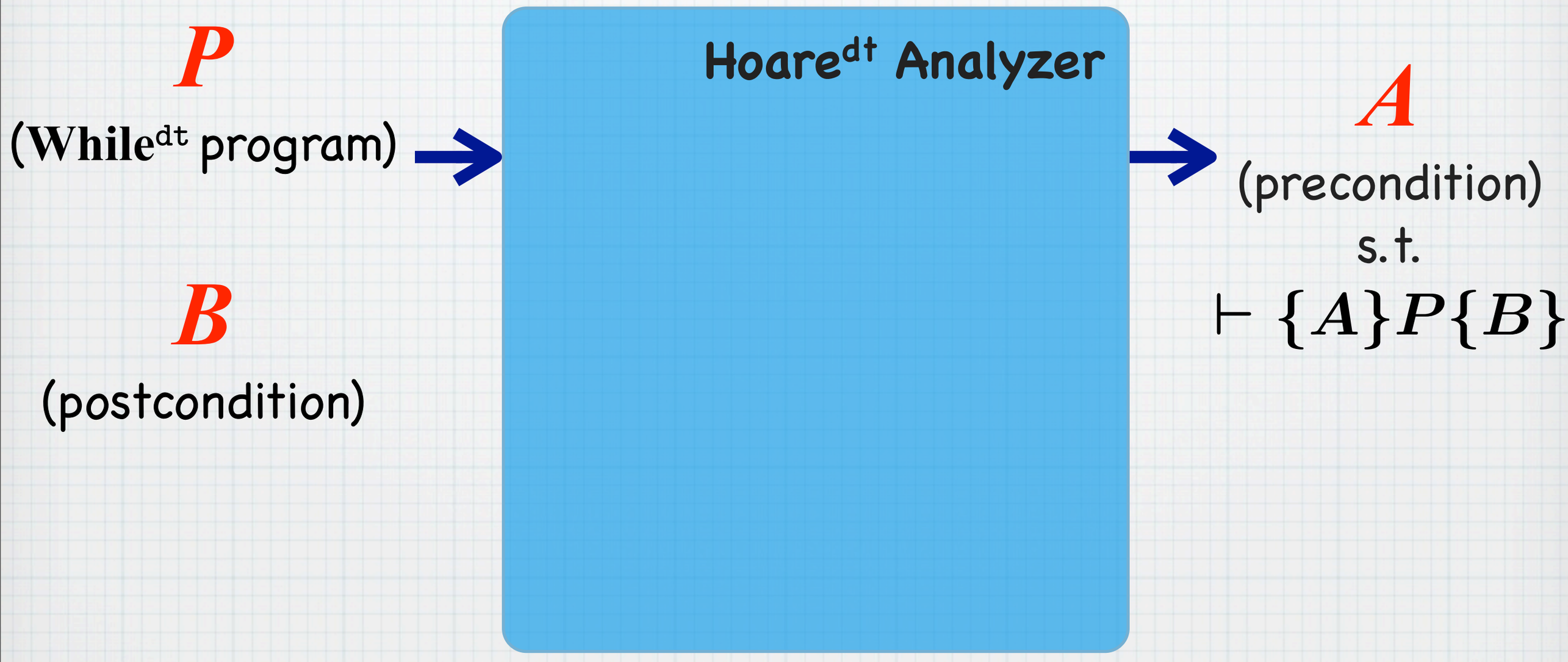
P

(**While**^{dt} program)

B

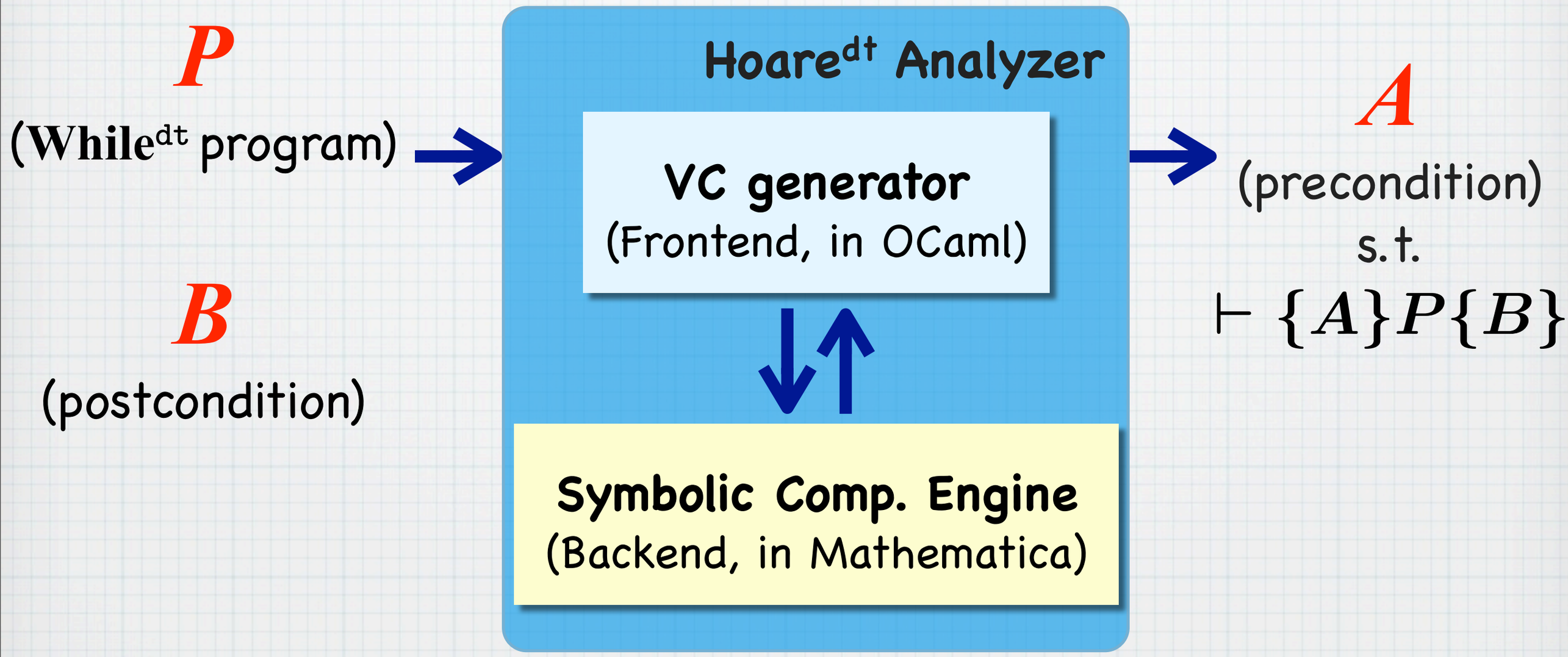
(postcondition)

Prototype Automatic Prover



Hasuo (Tokyo)

Prototype Automatic Prover



Hasuo (Tokyo)

Prototype Automatic Prover

P

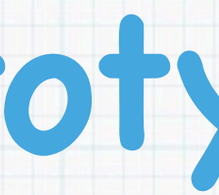
(While^{dt} program)

B

(postcondition)

Hoare^{dt} Analyzer

VC generator
(Frontend, in OCaml)



Symbolic Comp. Engine
(Backend, in Mathematica)

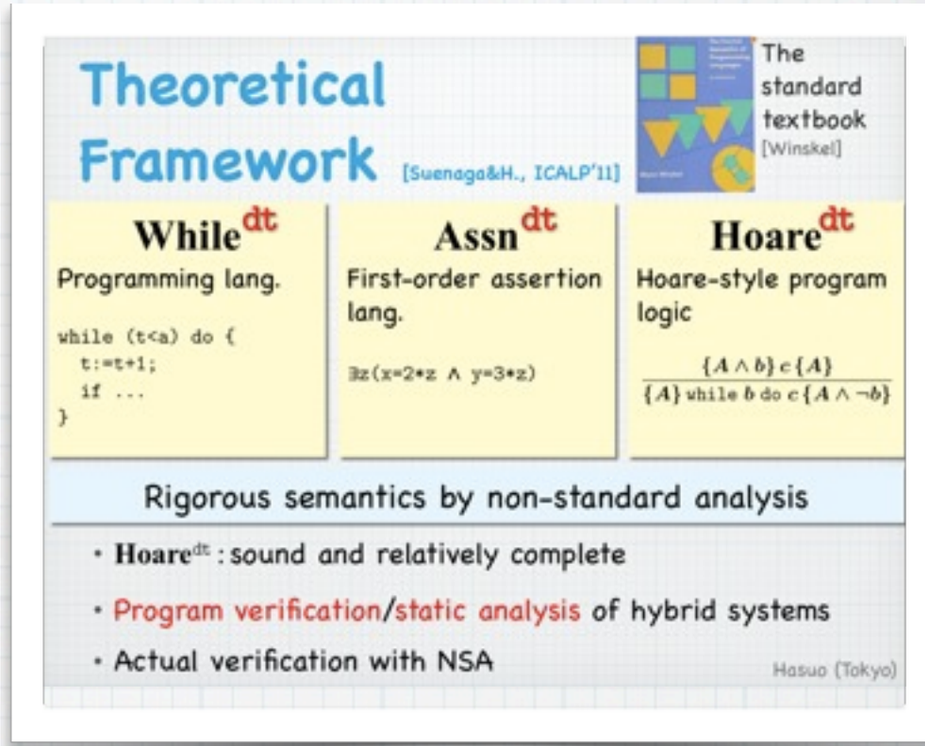
A

(precondition)
s.t.
$\vdash \{A\}P\{B\}$

For reals, not hyperreals
➜ justified by the
transfer principle

Outline

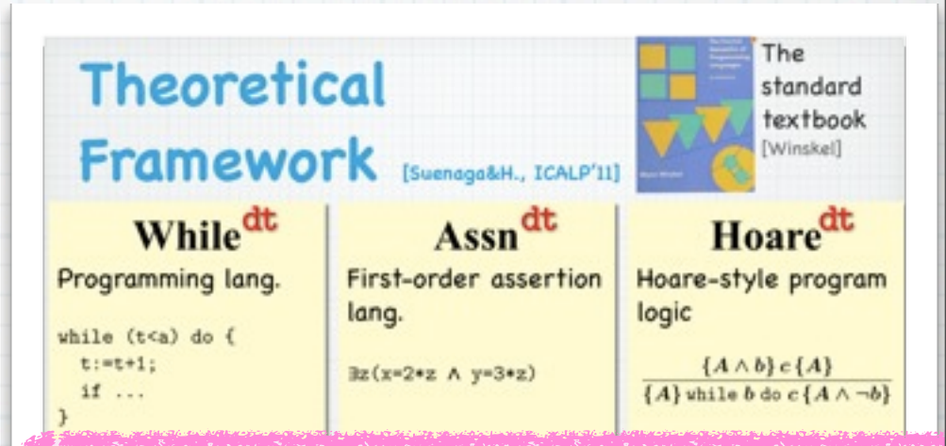
- **Theoretical foundations**
  - **While^{dt}, Assn^{dt}, Hoare^{dt}**
  - Rigorous semantics via NSA
    - Transfer principle, "sectionwise lemmas"
- **Static analysis techniques, transferred as they are**
  - Phase split [Sharma,Dillig,Dillig,Aiken; CAV'11] [Balakrishnan,Sankaranarayanan,Ivancic,Gupta; EMSOFT'09] [Gopan,Reps; SAS'07]
  - Differential invariant [Platzer,Clarke; CAV'08]
  - … and more!



Hasuo (Tokyo)

Outline

- **Theoretical foundations**
  - **While^{dt}, Assn^{dt}, Hoare^{dt}**
  - Rigorous semantics via NSA
  - Transfer principle, "sectionwise lemmas"
- **Static analysis techniques, transferred as they are**
  - Phase split [Sharma,Dillig,Dillig,Aiken; CAV'11] [Balakrishnan,Sankaranarayanan,Ivancic,Gupta; EMSOFT'09] [Gopan,Reps; SAS'07]
  - Differential invariant [Platzer,Clarke; CAV'08]
  - ... and more!



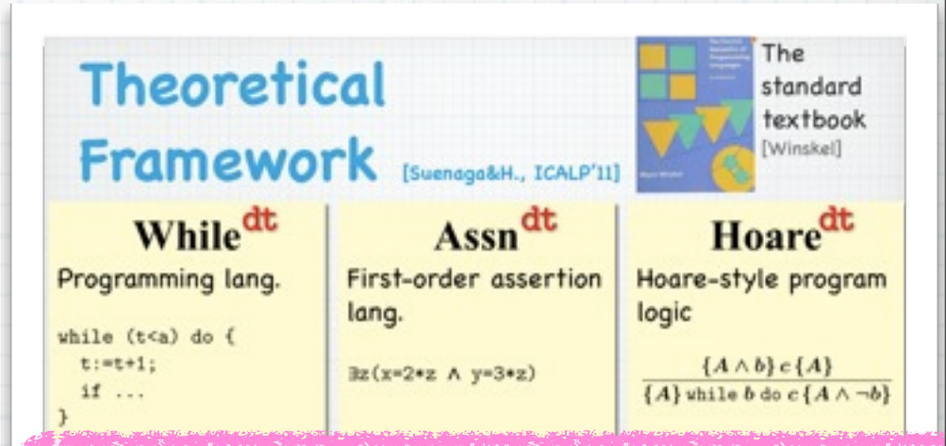
w/ or w/o dt ...

➜ logically "**the same**"

Hasuo (Tokyo)

Outline

- **Theoretical foundations**
  - **While^{dt}, Assn^{dt}, Hoare^{dt}**
  - Rigorous semantics via NSA
  - Transfer principle, "sectionwise lemmas"
- **Static analysis techniques, transferred as they are**
  - Phase split [Sharma,Dillig,Dillig,Aiken; CAV'11] [Balakrishnan,Sankaranarayanan,Ivancic,Gupta; EMSOFT'09] [Gopan,Reps; SAS'07]
  - Differential invariant [Platzer,Clarke; CAV'08]
  - ... and more!



w/ or w/o dt ...

➜ logically "**the same**"

Hasuo (Tokyo)

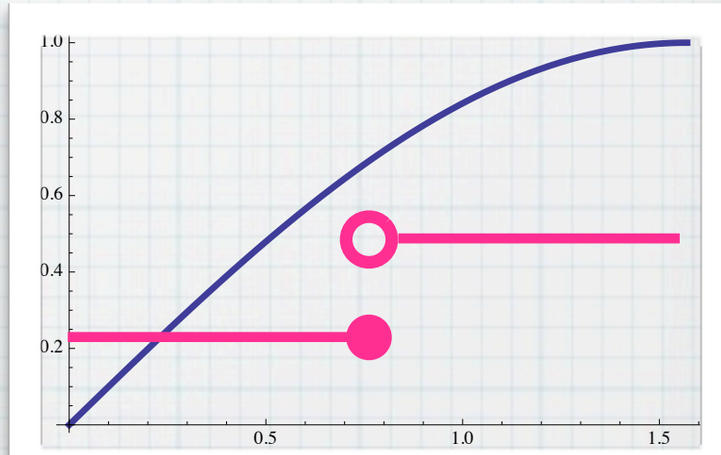
Part I:

Theoretical Foundations

Nonstandard Analysis

* Analysis with an **infinitesimal** δ, e.g.

$$\boldsymbol{f} \text{ is continuous} \iff \left(\begin{array}{l} |\boldsymbol{x} - \boldsymbol{x'}| \text{ is infinitesimal} \\ \Longrightarrow\ |\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x'})| \text{ is infinitesimal} \end{array} \right)$$



* Cf. Leibniz's **monad**
* Done naively ➜ contradiction!

Hasuo (Tokyo)

Nonstandard Analysis

* Analysis with an **infinitesimal** δ, e.g.

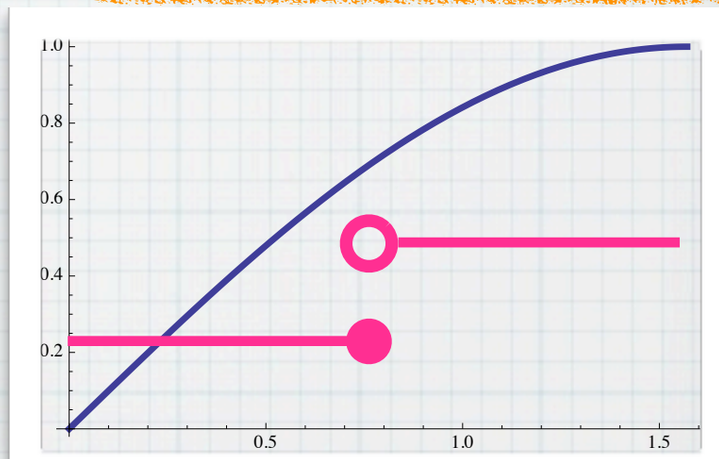
$\boldsymbol{f}$ is continuous $\iff$

$$\left(\begin{array}{l} |\boldsymbol{x} - \boldsymbol{x'}| \text{ is infinitesimal} \\ \Longrightarrow\ |\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x'})| \text{ is infinitesimal} \end{array} \right)$$

"Infinitely small"

$$0 < \delta < r$$

$$(\forall r \in \mathbb{R}_+)$$



* Cf. Leibniz's **monad**

* Done naively ➜ contradiction!

Nonstandard Analysis

* Analysis with an **infinitesimal** δ, e.g.

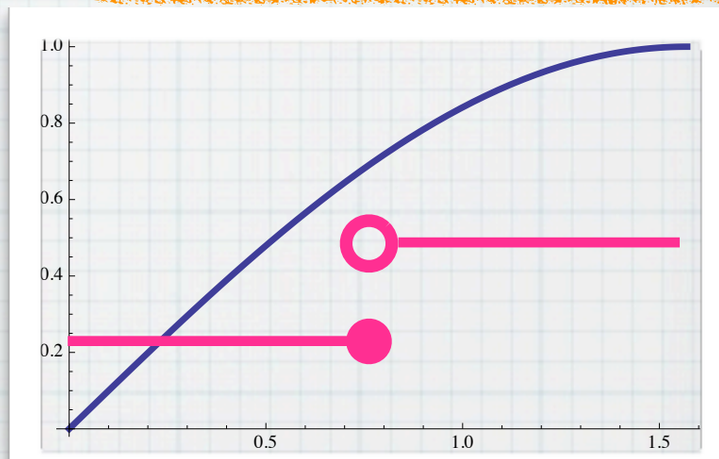
f is continuous $\iff$

$$\left(\begin{array}{l} |x - x'| \text{ is infinitesimal} \\ \Longrightarrow |f(x) - f(x')| \text{ is infinitesimal} \end{array} \right)$$

"Infinitely small"

$$0 < \delta < r$$

$$(\forall r \in \mathbb{R}_+)$$



* Cf. Leibniz's **monad**

Nonstandard Analysis

* Analysis with an **infinitesimal** δ, e.g.

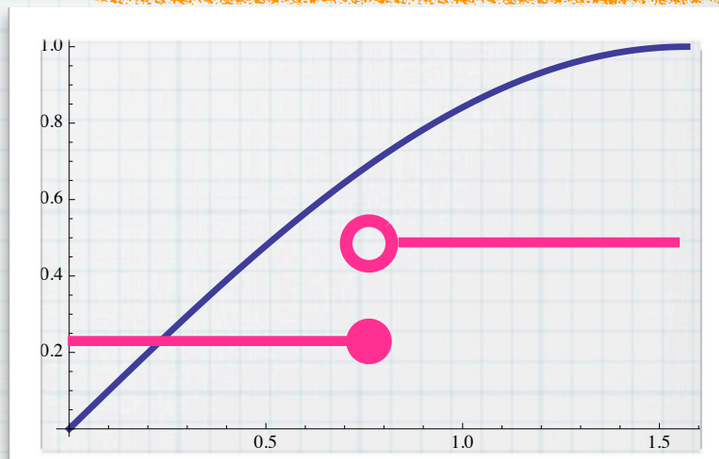
$\boldsymbol{f}$ is continuous $\iff$

$$\left(\begin{array}{l} |\boldsymbol{x} - \boldsymbol{x'}| \text{ is infinitesimal} \\ \Longrightarrow\ |\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x'})| \text{ is infinitesimal} \end{array} \right)$$

"Infinitely small"

$$0 < \delta < r$$

$$(\forall r \in \mathbb{R}_+)$$



* Cf. Leibniz's **monad**

* Done naively ➜ contradiction!

Nonstandard Analysis

* Analysis with an **infinitesimal** δ, e.g.

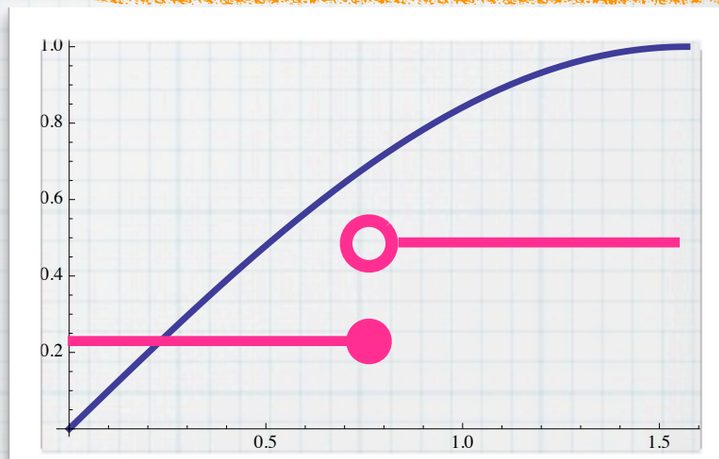
$\boldsymbol{f}$ is continuous $\iff$

$$\left(\begin{array}{l} |\boldsymbol{x} - \boldsymbol{x'}| \text{ is infinitesimal} \\ \Longrightarrow |\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x'})| \text{ is infinitesimal} \end{array} \right)$$

"Infinitely small"

$$0 < \delta < r$$

$$(\forall r \in \mathbb{R}_+)$$



* Cf. Leibniz's **monad**

* Done naively ➜ contradiction!

Logical foundation via an ultrafilter

[Robinson,1960]

Hyperreals

= Reals + Infinitesimals + ...

Defn.

The set of *hyperreal numbers* is

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

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.

The set of *hyperreal numbers* is

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

Ignore

Hyperreals

= Reals + Infinitesimals + ...

Defn.

The set of *hyperreal numbers* is

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

Ignore

Hyperreals

= Reals + Infinitesimals + ...

Defn.

The set of *hyperreal numbers* is

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

Ignore

0th section

1st section

2nd section

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.

The set of *hyperreal numbers* is

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

0th section

1st section

2nd section

Ignore

* Operations: sectionwise

$$\begin{array}{rl} & [(a_0, a_1, \dots)] \\ + & [(b_0, b_1, \dots)] \\ = & [(a_0 + b_0, a_1 + b_1, \dots)] \end{array}$$

Hyperreals

= Reals + Infinitesimals + ...

Defn.

The set of *hyperreal numbers* is

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

0th section 1st section 2nd section

Ignore

- Operations: sectionwise

$$\begin{aligned} & \left[(a_0, a_1, \dots)\right] \\ + \quad & \left[(b_0, b_1, \dots)\right] \\ = \quad & \left[(a_0 + b_0, a_1 + b_1, \dots)\right] \end{aligned}$$

- Reals are hyperreals

$$\mathbb{R} \hookrightarrow {}^*\mathbb{R},$$

$$r \mapsto \left[(r, r, \dots)\right]$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

* Predicates:
sectionwise,
"for almost all i"

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + …

Defn.
The set of *hyperreal numbers* is

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

* Predicates:
sectionwise,
"for almost all i"

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

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + …

Defn.
The set of *hyperreal numbers* is

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

* Predicates:
sectionwise,
"for almost all i"

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

$$\big[(a_i)_{i\in\mathbb{N}}\big] < \big[(b_i)_{i\in\mathbb{N}}\big]$$
$$\iff \quad a_i < b_i \qquad \text{"for almost every } i\text{"}$$
$$\impliedby \quad \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + …

Defn.
The set of *hyperreal numbers* is

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

* Predicates:
sectionwise,
"for almost all i"

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

$$\left[(a_i)_{i\in\mathbb{N}}\right] < \left[(b_i)_{i\in\mathbb{N}}\right]$$
$$\iff \quad a_i < b_i \qquad \text{"for almost every } i\text{"}$$
$$\impliedby \quad \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}$$

Precise defn. is via an ultrafilter $\mathcal{F}$:

$$\left[(a_i)_{i\in\mathbb{N}}\right] < \left[(b_i)_{i\in\mathbb{N}}\right]$$
$$\iff \quad \{i \in \mathbb{N} \mid a_i < b_i\} \in \mathcal{F}$$

Hyperreals

= Reals + Infinitesimals + …

Defn.
The set of *hyperreal numbers* is

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

* Predicates:
sectionwise,
"for almost all i"

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

$$\big[(a_i)_{i\in\mathbb{N}}\big] < \big[(b_i)_{i\in\mathbb{N}}\big]$$
$$\iff \quad a_i < b_i \qquad \text{"for almost every } i\text{"}$$
$$\impliedby \quad \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}$$

Precise defn. is via an ultrafilter $\mathcal{F}$:

$$\big[(a_i)_{i\in\mathbb{N}}\big] < \big[(b_i)_{i\in\mathbb{N}}\big]$$
$$\iff \quad \{i \in \mathbb{N} \mid a_i < b_i\} \in \mathcal{F}$$

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$\begin{aligned}
&[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \\
&\iff \quad a_i < b_i \qquad \text{“for almost every } i\text{”} \\
&\impliedby \quad \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}
\end{aligned}$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

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

Prop. $\omega^{-1} = \big[\,(1, \frac{1}{2}, \frac{1}{3}, \dots)\,\big]$ is infinitesimal.

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$\begin{aligned}
&[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \\
\iff\quad & a_i < b_i \qquad \text{“for almost every } i\text{”} \\
\impliedby\quad & \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}
\end{aligned}$$

Prop. $\omega^{-1} = [(1, \frac{1}{2}, \frac{1}{3}, \dots)]$ is infinitesimal.

$$\omega^{-1} = (1, \frac{1}{2}, \frac{1}{3}, \dots, \frac{1}{N}, \frac{1}{N+1}, \dots)$$

$$\frac{1}{N} = (\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}, \frac{1}{N}, \dots)$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}]$$
$$\iff \quad a_i < b_i \qquad \text{“for almost every } i\text{”}$$
$$\impliedby \quad \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}$$

Prop. $\omega^{-1} = [(1, \frac{1}{2}, \frac{1}{3}, \dots)]$ is infinitesimal.

$$\omega^{-1} = (1, \frac{1}{2}, \frac{1}{3}, \dots, \frac{1}{N}, \frac{1}{N+1}, \dots)$$

$$\frac{1}{N} = (\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}, \frac{1}{N}, \dots)$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}]$$
$$\iff \quad a_i < b_i \qquad \text{“for almost every } i\text{”}$$
$$\impliedby \quad \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}$$

Prop. $\omega^{-1} = [(1, \frac{1}{2}, \frac{1}{3}, \dots)]$ is infinitesimal.

$$\omega^{-1} = (1, \frac{1}{2}, \frac{1}{3}, \dots, \frac{1}{N}, \frac{1}{N+1}, \dots)$$

$$\frac{1}{N} = (\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}, \frac{1}{N}, \dots)$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}]$$
$$\iff \quad a_i < b_i \qquad \text{“for almost every } i\text{”}$$
$$\impliedby \quad \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}$$

Prop. $\omega^{-1} = [(1, \frac{1}{2}, \frac{1}{3}, \dots)]$ is infinitesimal.

$$\omega^{-1} = (1, \frac{1}{2}, \frac{1}{3}, \dots, \frac{1}{N}, \frac{1}{N+1}, \dots)$$

$$\frac{1}{N} = (\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}, \frac{1}{N}, \dots)$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$\begin{aligned}
&[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \\
\iff\quad & a_i < b_i \qquad \text{“for almost every } i\text{”} \\
\impliedby\quad & \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}
\end{aligned}$$

Prop. $\omega^{-1} = [(1, \frac{1}{2}, \frac{1}{3}, \dots)]$ is infinitesimal.

$$\omega^{-1} = (1, \frac{1}{2}, \frac{1}{3}, \dots, \frac{1}{N}, \frac{1}{N+1}, \dots)$$

$$\frac{1}{N} = (\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}, \frac{1}{N}, \dots)$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$\begin{aligned}
&[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \\
&\iff \quad a_i < b_i \qquad \text{“for almost every } i\text{”} \\
&\impliedby \quad \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}
\end{aligned}$$

Prop. $\omega^{-1} = [(1, \frac{1}{2}, \frac{1}{3}, \dots)]$ is infinitesimal.

$$\omega^{-1} = (1, \frac{1}{2}, \frac{1}{3}, \dots, \frac{1}{N}, \frac{1}{N+1}, \dots)$$

$$\frac{1}{N} = (\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}, \frac{1}{N}, \dots)$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$\begin{aligned}
&[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \\
\iff\quad & a_i < b_i \qquad \text{“for almost every } i\text{”} \\
\impliedby\quad & \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}
\end{aligned}$$

Prop. $\omega^{-1} = [(1, \frac{1}{2}, \frac{1}{3}, \dots)]$ is infinitesimal.

$$\omega^{-1} = (1, \frac{1}{2}, \frac{1}{3}, \dots, \frac{1}{N}, \frac{1}{N+1}, \dots)$$

$$\frac{1}{N} = (\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}, \frac{1}{N}, \dots)$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$\begin{aligned}
&[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \\
&\iff \quad a_i < b_i \qquad \text{“for almost every } i\text{”} \\
&\impliedby \quad \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}
\end{aligned}$$

Prop. $\omega^{-1} = [(1, \frac{1}{2}, \frac{1}{3}, \dots)]$ is infinitesimal.

$$\omega^{-1} = (1, \frac{1}{2}, \frac{1}{3}, \dots, \frac{1}{N}, \frac{1}{N+1}, \dots)$$

$$\frac{1}{N} = (\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}, \frac{1}{N}, \dots)$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$\begin{aligned}
&[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \\
\iff\quad & a_i < b_i \qquad \text{“for almost every } i\text{”} \\
\impliedby\quad & \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}
\end{aligned}$$

Prop. $\omega^{-1} = [(1, \frac{1}{2}, \frac{1}{3}, \dots)]$ is infinitesimal.

$$\omega^{-1} = (1, \frac{1}{2}, \frac{1}{3}, \dots, \frac{1}{N}, \frac{1}{N+1}, \dots)$$

OK! ^ ✗ ✗ ✗ ✗ ^ ^ ...

$$\frac{1}{N} = (\frac{1}{N}, \frac{1}{N}, \frac{1}{N}, \dots, \frac{1}{N}, \frac{1}{N}, \dots)$$

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$\begin{aligned}
&\left[(a_i)_{i\in\mathbb{N}}\right] < \left[(b_i)_{i\in\mathbb{N}}\right] \\
\iff\quad & a_i < b_i \qquad \text{“for almost every } i\text{”} \\
\impliedby\quad & \{i \in \mathbb{N} \mid a_i \not< b_i\} \qquad \text{is finite}
\end{aligned}$$

Prop. $\omega^{-1} = \left[\left(1, \frac{1}{2}, \frac{1}{3}, \dots\right)\right]$ is infinitesimal.

Prop. $\omega = \left[(1, 2, 3, \dots)\right]$ is infinite.

Hasuo (Tokyo)

Hyperreals

= Reals + Infinitesimals + ...

Defn.
The set of *hyperreal numbers* is

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

$$[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}]$$
$$\iff \quad \{i \in \mathbb{N} \mid a_i < b_i\} \in \mathcal{F}$$

Hasuo (Tokyo)

Hype

= Reals + Inf

Ultrafilter

(existence by AC)

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.
The set of *hyperreal numbers* is
$$^*\mathbb{R} := \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}}$$

$$[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}]$$
$$\iff \quad \{i \in \mathbb{N} \mid a_i < b_i\} \in \mathcal{F}$$

Hasuo (Tokyo)

Hype

= Reals + Inf

Ultrafilter

(existence by AC)

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.
The set of *hyperreal numbers* is
$$^*\mathbb{R} := \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}}$$

$$[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \iff \{i \in \mathbb{N} \mid a_i < b_i\} \in \mathcal{F}$$

Thm. (Transfer Principle)
A: a first-order formula.
$^*A$: its $*$-*transform*. Then
$$\mathbb{R} \models A \iff {}^*\mathbb{R} \models {}^*A \ .$$

Hasuo (Tokyo)

Hype

= Reals + Inf

Ultrafilter

(existence by AC)

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.
The set of *hyperreal numbers* is
$$^*\mathbb{R} := \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}}$$

$$[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \iff \{i \in \mathbb{N} \mid a_i < b_i\} \in \mathcal{F}$$

Thm. (Transfer Principle)
A: a first-order formula.
$^*A$: its $*$-*transform*. Then
$$\mathbb{R} \models A \iff {}^*\mathbb{R} \models {}^*A \ .$$

Same as A, except:
$\forall x \in \mathbb{R}$ in A is
$\forall x \in {}^*\mathbb{R}$ in $^*A$

Hasuo (Tokyo)

Hype

= Reals + Inf

Ultrafilter

(existence by AC)

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.
The set of *hyperreal numbers* is
$$^*\mathbb{R} := \mathbb{R}^{\mathbb{N}} / \sim_{\mathcal{F}}$$

$$[(a_i)_{i\in\mathbb{N}}] < [(b_i)_{i\in\mathbb{N}}] \iff \{i \in \mathbb{N} \mid a_i < b_i\} \in \mathcal{F}$$

Thm. (Transfer Principle)
A: a first-order formula.
$^*A$: its $*$-*transform*. Then
$$\mathbb{R} \models A \iff {}^*\mathbb{R} \models {}^*A \ .$$

Same as A, except:
$\forall x \in \mathbb{R}$ in A is
$\forall x \in {}^*\mathbb{R}$ in $^*A$

$\mathbb{R}$ and $^*\mathbb{R}$ are "logically the same"

Theoretical Framework

[Suenaga&H., ICALP'11]

The standard textbook [Winskel]

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

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

Rigorous semantics by non-standard analysis

Hasuo (Tokyo)

Syntax

While^{dt}

$$
\begin{array}{lll}
\textbf{AExp} \ni & a & ::= \quad x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt} \\
& \multicolumn{2}{l}{\text{where } \mathtt{c}_r \text{ is a const. for } r \in \mathbb{R},\ \mathtt{aop} \in \{+, -, \cdot, \hat{}, /\}} \\
\textbf{BExp} \ni & b & ::= \quad \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2 \\
\textbf{Cmd} \ni & c & ::= \quad \mathtt{skip} \mid x := a \mid c_1 ; c_2 \\
& & \quad \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c
\end{array}
$$

Assn^{dt}

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

Hoare^{dt}

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

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

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

Tokyo)

Syntax

While$^{\mathtt{dt}}$

$$\textbf{AExp} \ni a ::= x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt}$$

where $\mathtt{c}_r$ is a const. for $r \in \mathbb{R}$, $\mathtt{aop} \in \{+, -, \cdot, \hat{}, /\}$

$$\textbf{BExp} \ni b ::= \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2$$

$$\textbf{Cmd} \ni c ::= \mathtt{skip} \mid x := a \mid c_1; c_2 \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c$$

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

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

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

Tokyo)

Syntax

While$^{\mathtt{dt}}$

$$\textbf{AExp} \ni \quad a \quad ::= \quad x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt}$$

where $\mathtt{c}_r$ is a const. for $r \in \mathbb{R}$, $\mathtt{aop} \in \{+, -, \cdot, \hat{}, /\}$

$$\textbf{BExp} \ni \quad b \quad ::= \quad \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2$$

$$\textbf{Cmd} \ni \quad c \quad ::= \quad \mathtt{skip} \mid x := a \mid c_1 ; c_2 \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c$$

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

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

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

Tokyo)

Syntax

While$^{\mathtt{dt}}$

While + dt

$$\textbf{AExp} \ni \quad a \quad ::= \quad x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt}$$

where $\mathtt{c}_r$ is a const. for $r \in \mathbb{R}$, $\mathtt{aop} \in \{+, -, \cdot, \hat{}, /\}$

$$\textbf{BExp} \ni \quad b \quad ::= \quad \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2$$

$$\textbf{Cmd} \ni \quad c \quad ::= \quad \mathtt{skip} \mid x := a \mid c_1 ; c_2 \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c$$

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

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

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

Tokyo)

Syntax

While^{dt}

While + dt

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

Assn^{dt}

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

Hoare^{dt}

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

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

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

Tokyo)

Syntax

While$^{\mathtt{dt}}$

While + dt

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

Assn$^{\mathtt{dt}}$

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

Hoare$^{\mathtt{dt}}$

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

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

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

Tokyo)

Syntax

While^dt

While + dt

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

Assn^dt

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

Hoare^dt

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

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

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

Tokyo)

Syntax

While + dt

While^{dt}

$$
\begin{array}{lll}
\textbf{AExp} \ni & a & ::= \quad x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt} \\
& \multicolumn{2}{l}{\text{where } \mathtt{c}_r \text{ is a const. for } r \in \mathbb{R},\ \mathtt{aop} \in \{+, -, \cdot, \hat{}, /\}} \\
\textbf{BExp} \ni & b & ::= \quad \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2 \\
\textbf{Cmd} \ni & c & ::= \quad \mathtt{skip} \mid x := a \mid c_1 ; c_2 \\
& & \quad \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c
\end{array}
$$

Assn^{dt}

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

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

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

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

Tokyo)

Syntax

While + dt

While^{dt}

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

Assn^{dt}

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

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

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

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

Tokyo)

Syntax

While^dt

While + dt

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

Assn^dt

Assn, *-transformed

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

$$\frac{}{\{A\}\ \mathtt{skip}\ \{A\}}\ (\textsc{Skip})$$

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

$$\frac{\{A\}\ c_1\ \{C\} \quad \{C\}\ c_2\ \{B\}}{\{A\}\ c_1; c_2\ \{B\}}\ (\textsc{Seq})$$

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

$$\frac{\{A \wedge b\}\ c\ \{A\}}{\{A\}\ \mathtt{while}\ b\ \mathtt{do}\ c\ \{A \wedge \neg b\}}\ (\textsc{While})$$

$$\frac{\models A \Rightarrow A' \quad \{A'\}\ c\ \{B'\} \quad \models B' \Rightarrow B}{\{A\}\ c\ \{B\}}\ (\textsc{Conseq})$$

Tokyo)

Syntax

While^dt

While + dt

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

Assn^dt

Assn, *-transformed

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

Hoare^dt

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

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

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

Tokyo)

Syntax

While^dt

While + dt

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

Assn^dt

Assn, *-transformed

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

Hoare^dt

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

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

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

Tokyo)

Syntax

While^dt

While + dt

$$
\begin{array}{lll}
\textbf{AExp} \ni & a & ::= \quad x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop}\ a_2 \mid \mathtt{dt} \\
& \multicolumn{2}{l}{\text{where } \mathtt{c}_r \text{ is a const. for } r \in \mathbb{R},\ \mathtt{aop} \in \{+, -, \cdot, \hat{}, /\}} \\
\textbf{BExp} \ni & b & ::= \quad \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2 \\
\textbf{Cmd} \ni & c & ::= \quad \mathtt{skip} \mid x := a \mid c_1; c_2 \\
& & \quad \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c
\end{array}
$$

Assn^dt

Assn, *-transformed

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

Hoare^dt

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

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

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

Tokyo)

While^{dt}

While + dt

Syntax

$$\textbf{AExp} \ni a ::= x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt}$$

where $\mathtt{c}_r$ is a const. for $r \in \mathbb{R}$, $\mathtt{aop} \in \{+, -, \cdot, \hat{}, /\}$

$$\textbf{BExp} \ni b ::= \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2$$

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

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

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

Hoare^{dt}

Hasuo (Tokyo)

Syntax

While^{dt}

While + dt

$$\begin{array}{lll} \textbf{AExp} \ni & a & ::= \quad x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt} \\ & \text{where } \mathtt{c}_r \text{ is a const. for } r \in \mathbb{R},\ \mathtt{aop} \in \{+, -, \cdot, \hat{}, /\} \\ \textbf{BExp} \ni & b & ::= \quad \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2 \end{array}$$

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

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

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

Hoare^{dt}

Precisely the same rules

Hasuo (Tokyo)

Syntax

While^{dt}

While + dt

$$
\begin{array}{lll}
\textbf{AExp} \ni & a & ::= \quad x \mid \mathtt{c}_r \mid a_1 \ \mathtt{aop} \ a_2 \mid \mathtt{dt} \\
& \multicolumn{2}{l}{\text{where } \mathtt{c}_r \text{ is a const. for } r \in \mathbb{R},\ \mathtt{aop} \in \{+, -, \cdot, \hat{}, /\}} \\
\textbf{BExp} \ni & b & ::= \quad \mathtt{true} \mid \mathtt{false} \mid b_1 \wedge b_2 \mid \neg b \mid a_1 < a_2 \\
\textbf{Cmd} \ni & c & ::= \quad \mathtt{skip} \mid x := a \mid c_1 ; c_2 \\
& & \quad \mid \mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2 \mid \mathtt{while}\ b\ \mathtt{do}\ c
\end{array}
$$

Assn^{dt}

Assn, *-transformed

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

Hoare^{dt}

Precisely the same rules

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

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

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

Tokyo)

Syntax

While^{dt}

While + dt

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

Thm.

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

Hoare^{dt}

Precisely the same rules

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

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

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

Tokyo)

Denotational Semantics: Challenge

```
t := 0 ;
while (t ≤ 1) do {
   t := t + dt
}
```

Denotational Semantics: Challenge

```
t := 0 ;
while (t ≤ 1) do {
   t := t + dt
}
```

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

Hasuo (Tokyo)

Denotational Semantics: Challenge

```
t := 0 ;
while (t ≤ 1) do {
   t := t + dt
}
```

$t = 1 + \mathrm{dt}$

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

$\perp$ (divergence)

Hasuo (Tokyo)

Denotational Semantics: Challenge

```
t := 0 ;
while (t ≤ 1) do {
   t := t + dt
}
```

$t = 1 + \mathrm{dt}$

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

$\perp$ (divergence)

* Semantics by "sectionwise execution"

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

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

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

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

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

```
t := (0,0,0,...);
while (t < (1,1,1,...))
    t := t + (1, 1/2, 1/3, ...) ;
```

Hasuo (Tokyo)

Denotational 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 ;
```

...

Hasuo (Tokyo)

Denotational 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 + 1/2 ;`
		`t := 0;`
		`while (t < 1)`
		`  t := t + 1/3 ;`
			…

`t = 1`	`t = 1`	`t = 1`	…

Hasuo (Tokyo)

Denotational 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,...)
```

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

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

```
t = 1
```

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

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

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

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

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

```
t := (0,0,0,...);
while (t <= (1,1,1,...))
  t := t + (1, 1/2, 1/3, ...) ;
```

Hasuo (Tokyo)

Denotational 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 ;
```

...

Hasuo (Tokyo)

Denotational 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 + 1/2 ;`
		`t := 0;`
		`while (t <= 1)`
		`  t := t + 1/3 ;`
			…

`t = 1 + 1`	`t = 1 + 1/2`	`t = 1 + 1/3`	…

Yasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

```
t := (0,0,0,...);
while (t <= (1,1,1,...))
   t := t + (1, 1/2, 1/3, ...) ;
```

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

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

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

```
t = 1 + dt
```

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

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

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

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

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

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

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

Hasuo (Tokyo)

Denotational 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 ;
```

...

Hasuo (Tokyo)

Denotational 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 + 1/2;`	`t := 0; while (true) t := t + 1/3;`	…
$\bot$	$\bot$	$\bot$	…

Hasuo (Tokyo)

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

```
t := (0,0,0,...);
while (true)
   t := t + (1, 1/2, 1/3, ...) ;
```

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

Denotational Semantics

* Execute sectionwise and bundle up the outcomes!

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

⊥

Denotational Semantics

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

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

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

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

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

Denotational Semantics

Hyperstate (stores hyperreals)

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

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

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

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

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

Denotational Semantics

Hyperstate (stores hyperreals)

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

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

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

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

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

Section of a program

Hasuo (Tokyo)

Denotational Semantics

Hyperstate (stores hyperreals)

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

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

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

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

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

Section of a program

Applied to a section of a memory state

Hasuo (Tokyo)

Denotational Semantics

Hyperstate (stores hyperreals)

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

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

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

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

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

Bundled up

Section of a program

Applied to a section of a memory state

Hasuo (Tokyo)

Denotational Semantics

Hyperstate (stores hyperreals)

$$
\begin{array}{lcl}
[\![x]\!]\sigma & := & \sigma(x) \\
[\![a_1 \text{ aop } a_2]\!]\sigma & := & [\![a_1]\!]\sigma \text{ aop } [\![a_2]\!]\sigma \\
[\![\mathtt{dt}]\!]\sigma & := & \omega^{-1} = \left[(1, \tfrac{1}{2}, \tfrac{1}{3}, \ldots) \right]
\end{array}
$$

$$
\begin{array}{lcl}
[\![\mathtt{true}]\!]\sigma & := & \mathrm{tt} \\
[\![b_1 \wedge b_2]\!]\sigma & := & [\![b_1]\!]\sigma \wedge [\![b_2]\!]\sigma \\
[\![a_1 < a_2]\!]\sigma & := & [\![a_1]\!]\sigma < [\![a_2]\!]\sigma
\end{array}
$$

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

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

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

Def.
The *i-th section* of a While$^{\mathtt{dt}}$ expression e is

$$
e|_i \quad :\equiv \quad e\left[\frac{1}{i+1} \,/\, \mathtt{dt} \right] .
$$

Section of a program

Applied to a section of a memory state

Bundled up

Hasuo (Tokyo)

Denot

$$\left[\begin{array}{l} t := 0\,; \\ \texttt{while}\ (t \le 1)\ \texttt{do} \\ \quad t := t + \mathrm{dt} \end{array}\right] \overset{i\text{-th section}}{\longmapsto} \left[\begin{array}{l} t := 0\,; \\ \texttt{while}\ (t \le 1)\ \texttt{do} \\ \quad t := t + \frac{1}{i+1} \end{array}\right]$$

Hyperstate (stores hyperreals)

$$\begin{array}{lcl}
[\![x]\!]\sigma & := & \sigma(x) \\
[\![a_1 \ \texttt{aop}\ a_2]\!]\sigma & := & [\![a_1]\!]\sigma \ \texttt{aop}\ [\![a_2]\!]\sigma \\
[\![\mathrm{dt}]\!]\sigma & := & \omega^{-1} = [\,(1, \tfrac{1}{2}, \tfrac{1}{3}, \ldots)\,]
\end{array}$$

$$\begin{array}{lcl}
[\![\texttt{true}]\!]\sigma & := & \mathrm{tt} \\
[\![b_1 \wedge b_2]\!]\sigma & := & [\![b_1]\!]\sigma \wedge [\![b_2]\!]\sigma \\
[\![a_1 < a_2]\!]\sigma & := & [\![a_1]\!]\sigma < [\![a_2]\!]\sigma
\end{array}$$

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

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

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

Def.
The *i-th section* of a WHILE$^{\mathrm{dt}}$ expression e is

$$e|_i \quad :\equiv \quad e\left[\,\frac{1}{i+1} \,/\, \mathrm{dt}\,\right] .$$

Section of a program

Applied to a section of a memory state

Bundled up

Hasuo (Tokyo)

"Sectionwise Lemmas"

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

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

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

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

Hasuo (Tokyo)

"Sectionwise Lemmas"

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

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

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

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

Łoś'

"Sectionwise Lemmas"

Lem. (Sectionwise validity of Hoare triples)

$$\models \{A\}c\{B\} \iff \models \{A|_i\}\, c|_i\, \{B|_i\} \text{ for almost every } i.$$

Hasuo (Tokyo)

"Sectionwise Lemmas"

Lem. (Sectionwise validity of Hoare triples)

$$\models \{A\}c\{B\} \iff \models \{A|_i\}\, c|_i\, \{B|_i\} \text{ for almost every } i.$$

Interface for **transferring** static analysis techniques

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

* A. Not exactly.
  * A **modeling** language
    * Not numerical approx., but **exact** modeling
    * Advantage:
      close to a common programming style

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

* A. Not exactly.
  * A **modeling** language
    * Not numerical approx., but **exact** modeling
    * Advantage:
      close to a common programming style

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

* A. Not exactly.
  * A **modeling** language
    * Not numerical approx., but **exact** modeling
    * Advantage:
      close to a common programming style

* Static analysis ➜ **no need to execute**!
  * Mathematical semantics suffices

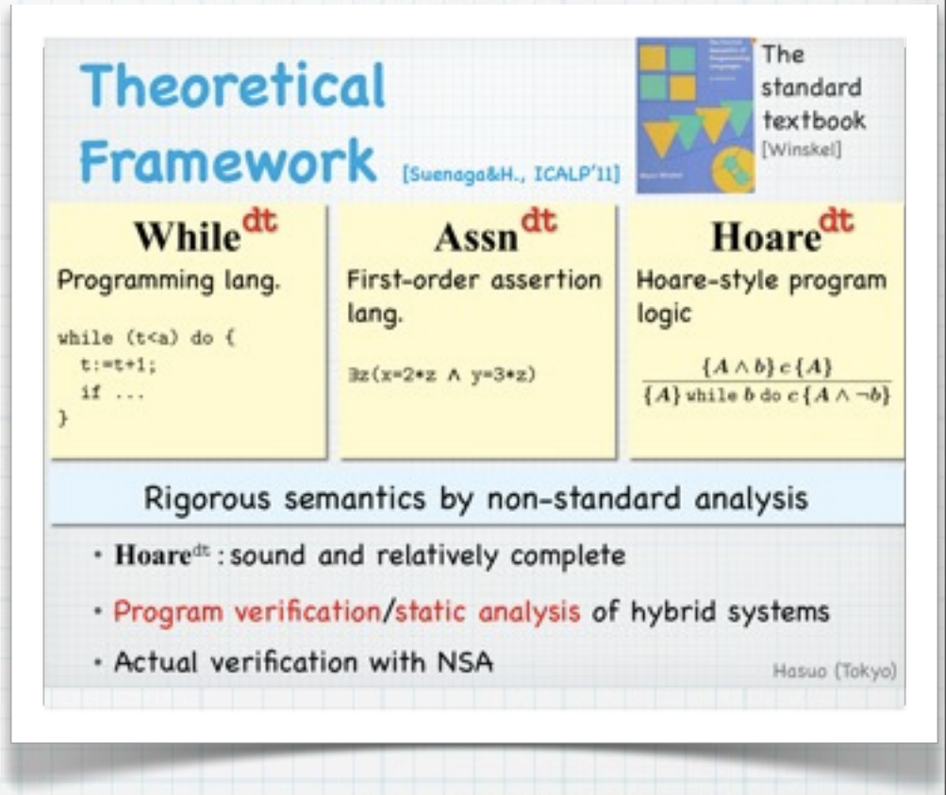
Hasuo (Tokyo)

Outline

Suenaga & H., ICALP'11

* **Theoretical foundations**
  * **While^{dt}, Assn^{dt}, Hoare^{dt}**
  * Rigorous semantics via NSA
  * Transfer principle, "sectionwise lemmas"
* **Static analysis techniques, transferred as they are**
  * Phase split [Sharma,Dillig,Dillig,Aiken; CAV'11] [Balakrishnan,Sankaranarayanan,Ivancic,Gupta; EMSOFT'09] [Gopan,Reps; SAS'07]
  * Differential invariant [Platzer,Clarke; CAV'08]
  * ... and more!

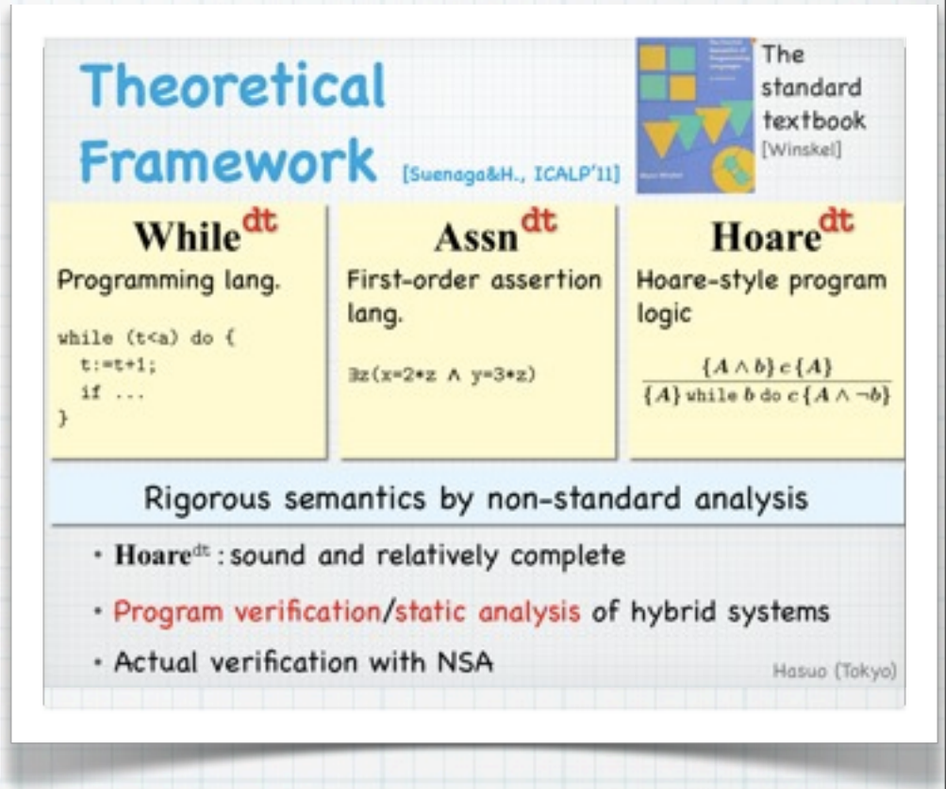
H. & Suenaga, CAV'12



Outline

Suenaga & H., ICALP'11

* **Theoretical foundations**
  * **While^{dt}, Assn^{dt}, Hoare^{dt}**
  * Rigorous semantics via NSA
  * Transfer principle, "sectionwise lemmas"



Done

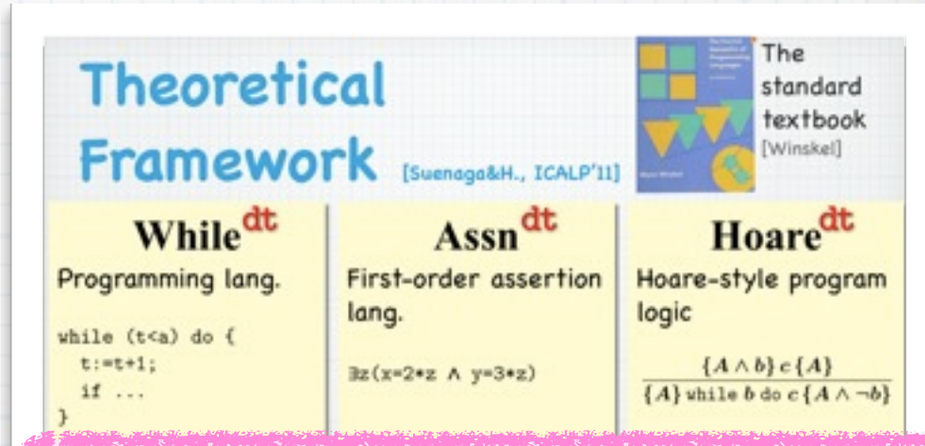
H. & Suenaga, CAV'12

* **Static analysis techniques, transferred as they are**
  * Phase split [Sharma,Dillig,Dillig,Aiken; CAV'11] [Balakrishnan,Sankaranarayanan,Ivancic,Gupta; EMSOFT'09] [Gopan,Reps; SAS'07]
  * Differential invariant [Platzer,Clarke; CAV'08]
  * ... and more!

Outline

Suenaga & H., ICALP'11

* **Theoretical foundations**
  * **While^{dt}, Assn^{dt}, Hoare^{dt}**
  * Rigorous semantics via NSA
  * Transfer principle, "sectionwise lemmas"



w/ or w/o dt ...

➜ logically "**the same**"

Done

H. & Suenaga, CAV'12

* **Static analysis techniques, transferred as they are**
  * Phase split [Sharma,Dillig,Dillig,Aiken; CAV'11] [Balakrishnan,Sankaranarayanan,Ivancic,Gupta; EMSOFT'09] [Gopan,Reps; SAS'07]
  * Differential invariant [Platzer,Clarke; CAV'08]
  * ... and more!

Part II:

Exercises in Nonstandard Static Analysis

Exercise 1.1


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

Hasuo (Tokyo)

Exercise 1.1


```
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 := a0;
    while t < ε do {
        t := t + dt;
        v := v + a · dt;
        z := z + v · dt
    }}
```

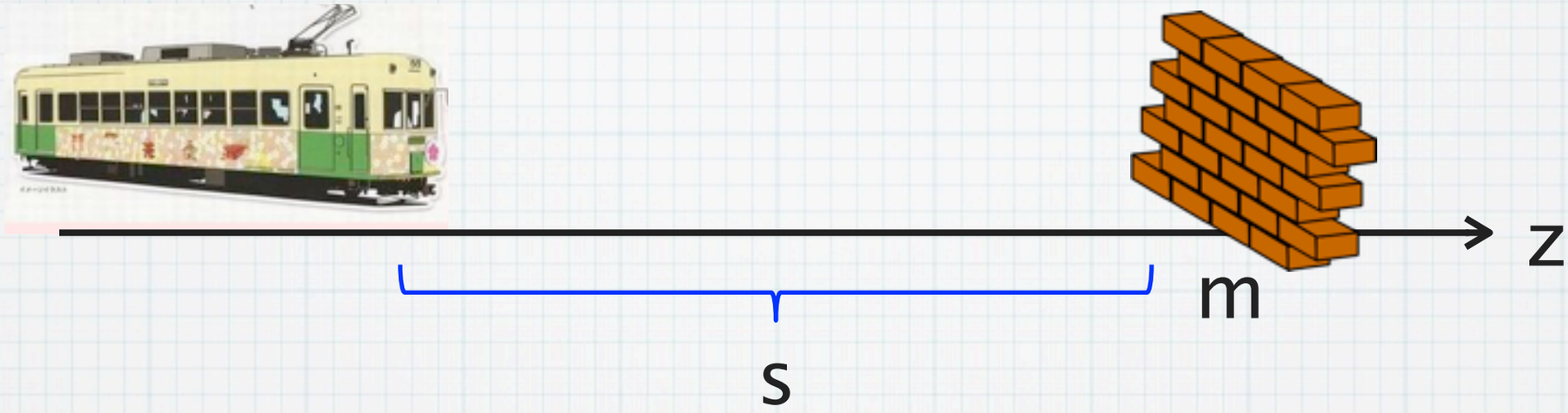
Hasuo (Tokyo)

Exercise 1.1


```
while v > 0 do {
    t := 0;
    if m − z < s then a := −b else a := a0;
    while t < ε do {
        t := t + dt;
        v := v + a · dt;
        z := z + v · dt
    }}
```

Hasuo (Tokyo)

Exercise 1.1




```
while v > 0 do {
    t := 0;
    if m − z < s then a := −b else a := a0;
    while t < ε do {
        t := t + dt;
        v := v + a · dt;
        z := z + v · dt
    }}
```

Hasuo (Tokyo)

Exercise 1.1

(Tiny) fragment of
Euro. Train Ctrl. Sys. (ETCS)


```
while v > 0 do {
    t := 0;
    if m − z < s then a := −b else a := a0;
    while t < ε do {
        t := t + dt;
        v := v + a · dt;
        z := z + v · dt
    }}
```

ETCS_0

Hasuo (Tokyo)

Exercise 1.1

(Tiny) fragment of
Euro. Train Ctrl. Sys. (ETCS)


```
while v > 0 do {
    t := 0;
    if m − z < s then a := −b else a := a0;
    while t < ε do {
        t := t + dt;
        v := v + a · dt;
        z := z + v · dt
    }}
```

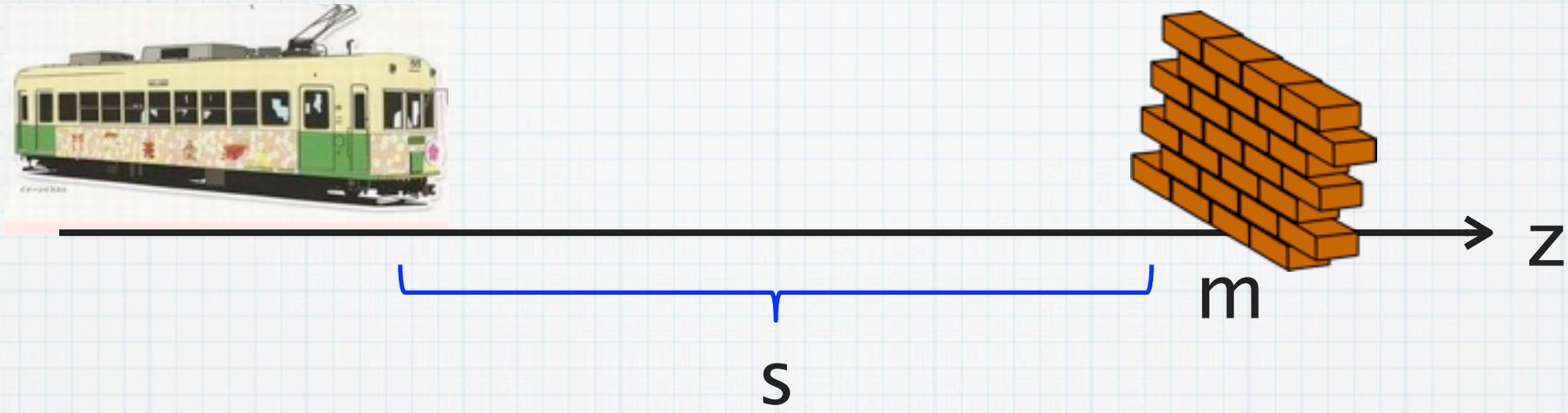
ETCS_0

Q. Find A s.t. $\models \{A\}\,\mathrm{ETCS}_0\,\{z < m\}$

Hasuo (Tokyo)

Exercise 1.1

(Tiny) fragment of
Euro. Train Ctrl. Sys. (ETCS)



s: big enough
b: big enough
a_0: small enough
…

$$
\begin{array}{l}
\texttt{while } v > 0 \texttt{ do } \{ \\
\quad t := 0\,; \\
\quad \texttt{if } m - z < s \texttt{ then } a := -b \texttt{ else } a := a_0\,; \\
\quad \texttt{while } t < \varepsilon \texttt{ do } \{ \\
\qquad t := t + \mathrm{dt}\,; \\
\qquad v := v + a \cdot \mathrm{dt}\,; \\
\qquad z := z + v \cdot \mathrm{dt} \\
\quad \}\}
\end{array}
$$

ETCS_0

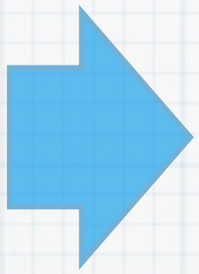
Q. Find A s.t. $\models \{A\}\,\mathrm{ETCS}_0\,\{z < m\}$

Hasuo (Tokyo)

```
while (v > 0) {
  if m - z < s
    then a := -b
    else a := a0;
  t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a * dt;
    t := t + dt }}


{z < m}
```

```
while (v > 0) {
  if m - z < s
    then a := -b
    else a := a0;
  t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a * dt;
    t := t + dt }}


{z < m}
```

```
while (v > 0) {
  if m - z < s
    then a := -b
    else a := a0;
  t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a * dt;
    t := t + dt }}

{z < m}
```



```
while (v > 0 && m - z >= s) {
  a := a0;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }};
while (v > 0 && m - z < s) {
  a := -b;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v - b * dt;
    t := t + dt }}

{z < m}
```

```
while (v > 0) {
  if m - z < s
    then a := -b
    else a := a0;
  t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a * dt;
    t := t + dt }}

{z < m}
```

```
while (v > 0 && m - z >= s) {
  a := a0;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }};
while (v > 0 && m - z < s) {
  a := -b;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v - b * dt;
    t := t + dt }}

{z < m}
```

accel.

brake

```
while (v > 0) {
  if m - z < s
    then a := -b
    else a := a0;
  t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a * dt;
    t := t + dt }}

{z < m}
```

```
while (v > 0 && m - z >= s) {
  a := a0;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }};
while (v > 0 && m - z < s) {
  a := -b;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v - b * dt;
    t := t + dt }}

{z < m}
```

accel.

brake

Strategy1 "Phase split"

[Sharma,Dillig,Dillig,Aiken; CAV'11]
[Balakrishnan,Sankaranarayanan,Ivancic,Gupta; EMSOFT'09] [Gopan,Reps; SAS'07]

Phase Split
(Standard Ver., for While & Hoare)

[Sharma,Dillig,Dillig,Aiken; CAV'11]

Defn.
The set of *holed commands* $\mathbf{Cmd}_{[\_]}$ is:

$$\mathbf{Cmd}_{[\_]} \ni h \ ::= \ \texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2 \mid h; c \mid c; h \mid \texttt{if } b \texttt{ then } h \texttt{ else } c \mid \texttt{if } b \texttt{ then } c \texttt{ else } h$$

For each holed command h, its *pre-hole fragment* $\overline{h}$ is:

$$\begin{array}{lll} \overline{\texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2} & :\equiv & \texttt{skip} \\ \overline{h; c} \ :\equiv \ \overline{h} \qquad \overline{c; h} & :\equiv & c; \overline{h} \\ \overline{\texttt{if } b \texttt{ then } h \texttt{ else } c} & :\equiv & \texttt{assert } b \,; \overline{h} \\ \overline{\texttt{if } b \texttt{ then } c \texttt{ else } h} & :\equiv & \texttt{assert } \neg b \,; \overline{h} \end{array}$$

Lem.
If a Boolean expression $b_s \in \mathbf{BExp}$ satisfies

$$\models \{b_s\}\, \overline{h}\, \{b_c\} \ , \quad \models \{\neg b_s\}\, \overline{h}\, \{\neg b_c\} \ , \quad \text{and} \quad \models \{b_g \wedge b_s\}\, h[b_c]\, \{\neg b_g \vee b_s\} \ ,$$

then we have

$$[\![\texttt{while } b_g \texttt{ do } h[b_c]]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (b_g \wedge \neg b_s) \texttt{ do } h[\texttt{false}] \,; \\ \texttt{while } (b_g \wedge b_s) \texttt{ do } h[\texttt{true}] \end{array} \right]\!\!\right] .$$

Phase Split
(Standard Ver., for While & Hoare)

[Sharma,Dillig,Dillig,Aiken; CAV'11]

Defn.
The set of *holed commands* $\mathbf{Cmd}_{[\_]}$ is:

$$\mathbf{Cmd}_{[\_]} \ni h \ ::= \ \texttt{if}\ [\_]\ \texttt{then}\ c_1\ \texttt{else}\ c_2 \mid h; c \mid c; h \mid \texttt{if}\ b\ \texttt{then}\ h\ \texttt{else}\ c \mid \texttt{if}\ b\ \texttt{then}\ c\ \texttt{else}\ h$$

For each holed command h, its *pre-hole fragment* $\overline{h}$ is:

$$\begin{array}{lll} \overline{\texttt{if}\ [\_]\ \texttt{then}\ c_1\ \texttt{else}\ c_2} & :\equiv & \texttt{skip} \\ \overline{h; c} :\equiv \overline{h} \qquad \overline{c; h} & :\equiv & c; \overline{h} \\ \overline{\texttt{if}\ b\ \texttt{then}\ h\ \texttt{else}\ c} & :\equiv & \texttt{assert}\ b\ ; \overline{h} \\ \overline{\texttt{if}\ b\ \texttt{then}\ c\ \texttt{else}\ h} & :\equiv & \texttt{assert}\ \neg b\ ; \overline{h} \end{array}$$

Lem.
If a Boolean expression $b_s \in \mathbf{BExp}$ satisfies

$$\models \{b_s\}\,\overline{h}\,\{b_c\}\ , \quad \models \{\neg b_s\}\,\overline{h}\,\{\neg b_c\}\ , \quad \text{and} \quad \models \{b_g \wedge b_s\}\,h[b_c]\,\{\neg b_g \vee b_s\}\ ,$$

then we have

$$[\![\ \texttt{while}\ b_g\ \texttt{do}\ h[b_c]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while}\ (b_g \wedge \neg b_s)\ \texttt{do}\ h[\texttt{false}]\ ; \\ \texttt{while}\ (b_g \wedge b_s)\ \texttt{do}\ h[\texttt{true}] \end{array}\right]\!\!\right] .$$

h[_] is a command containing

```
if [ _ ] then ... else ...
```

Hasuo (Tokyo)

Phase Split
(Standard Ver., for While & Hoare)

[Sharma,Dillig,Dillig,Aiken; CAV'11]

Defn.
The set of *holed commands* $\mathbf{Cmd}_{[\_]}$ is:

$$\mathbf{Cmd}_{[\_]} \ni h \ ::= \ \texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2 \mid h;c \mid c;h \mid \texttt{if } b \texttt{ then } h \texttt{ else } c \mid \texttt{if } b \texttt{ then } c \texttt{ else } h$$

For each holed command h, its *pre-hole fragment* $\overline{h}$ is:

$$\begin{array}{lll} \overline{\texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2} & :\equiv & \texttt{skip} \\ \overline{h;c} \ :\equiv \ \overline{h} \qquad \overline{c;h} & :\equiv & c;\overline{h} \\ \overline{\texttt{if } b \texttt{ then } h \texttt{ else } c} & :\equiv & \texttt{assert } b \,;\overline{h} \\ \overline{\texttt{if } b \texttt{ then } c \texttt{ else } h} & :\equiv & \texttt{assert } \neg b \,;\overline{h} \end{array}$$

$\texttt{while } b_g \texttt{ do } \boxed{\ldots(\texttt{if}\ldots)\ldots}$

into $\left[\begin{array}{l} \texttt{while } b_g \wedge \neg b_s \texttt{ do } \boxed{\ldots} \ ; \\ \texttt{while } b_g \wedge b_s \texttt{ do } \boxed{\ldots} \end{array}\right]$

Lem.
If a Boolean expression $b_s \in \mathbf{BExp}$ satisfies

$$\models \{b_s\}\,\overline{h}\,\{b_c\} \ , \quad \models \{\neg b_s\}\,\overline{h}\,\{\neg b_c\} \ , \quad \text{and} \quad \models \{b_g \wedge b_s\}\,h[b_c]\,\{\neg b_g \vee b_s\} \ ,$$

then we have

$$[\![\ \texttt{while } b_g \texttt{ do } h[b_c]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (b_g \wedge \neg b_s) \texttt{ do } h[\texttt{false}] \,; \\ \texttt{while } (b_g \wedge b_s) \texttt{ do } h[\texttt{true}] \end{array}\right]\!\!\right] .$$

$h[\ \_\ ]$ is a command containing

`if [ _ ] then ... else ...`

Phase Split
(Nonstandard Ver., for While^{dt} & Hoare^{dt})

Defn.
The set of *holed commands* $\mathbf{Cmd}_{[\_]}$ is:

$$\mathbf{Cmd}_{[\_]} \ni \boldsymbol{h} \ ::= \ \texttt{if } [\_] \texttt{ then } \boldsymbol{c_1} \texttt{ else } \boldsymbol{c_2} \mid \boldsymbol{h};\boldsymbol{c} \mid \boldsymbol{c};\boldsymbol{h} \mid \texttt{if } \boldsymbol{b} \texttt{ then } \boldsymbol{h} \texttt{ else } \boldsymbol{c} \mid \texttt{if } \boldsymbol{b} \texttt{ then } \boldsymbol{c} \texttt{ else } \boldsymbol{h}$$

For each holed command $\boldsymbol{h}$, its *pre-hole fragment* $\overline{\boldsymbol{h}}$ is:

$$\begin{array}{lll}
\overline{\texttt{if } [_] \texttt{ then } \boldsymbol{c_1} \texttt{ else } \boldsymbol{c_2}} & :\equiv & \texttt{skip} \\
\overline{\boldsymbol{h};\boldsymbol{c}} :\equiv \overline{\boldsymbol{h}} \qquad \overline{\boldsymbol{c};\boldsymbol{h}} & :\equiv & \boldsymbol{c};\overline{\boldsymbol{h}} \\
\overline{\texttt{if } \boldsymbol{b} \texttt{ then } \boldsymbol{h} \texttt{ else } \boldsymbol{c}} & :\equiv & \texttt{assert } \boldsymbol{b} \,;\overline{\boldsymbol{h}} \\
\overline{\texttt{if } \boldsymbol{b} \texttt{ then } \boldsymbol{c} \texttt{ else } \boldsymbol{h}} & :\equiv & \texttt{assert } \neg\boldsymbol{b} \,;\overline{\boldsymbol{h}}
\end{array}$$

Lem.
If a Boolean expression $\boldsymbol{b_s} \in \mathbf{BExp}$ satisfies

$$\models \{\boldsymbol{b_s}\}\,\overline{\boldsymbol{h}}\,\{\boldsymbol{b_c}\}\ , \quad \models \{\neg\boldsymbol{b_s}\}\,\overline{\boldsymbol{h}}\,\{\neg\boldsymbol{b_c}\}\ , \quad \text{and} \quad \models \{\boldsymbol{b_g} \wedge \boldsymbol{b_s}\}\,\boldsymbol{h}[\boldsymbol{b_c}]\,\{\neg\boldsymbol{b_g} \vee \boldsymbol{b_s}\}\ ,$$

then we have

$$[\![\ \texttt{while } \boldsymbol{b_g} \texttt{ do } \boldsymbol{h}[\boldsymbol{b_c}]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (\boldsymbol{b_g} \wedge \neg\boldsymbol{b_s}) \texttt{ do } \boldsymbol{h}[\texttt{false}]\,; \\ \texttt{while } (\boldsymbol{b_g} \wedge \boldsymbol{b_s}) \texttt{ do } \boldsymbol{h}[\texttt{true}] \end{array}\right]\!\!\right] .$$

Hasuo (Tokyo)

Phase Split
(Nonstandard Ver., for While^{dt} & Hoare^{dt})

Defn.
The set of *holed commands* $\mathbf{Cmd}_{[\_]}$ is:

$$\mathbf{Cmd}_{[\_]} \ni \boldsymbol{h} \ ::= \ \texttt{if } [\_] \texttt{ then } \boldsymbol{c_1} \texttt{ else } \boldsymbol{c_2} \mid \boldsymbol{h};\boldsymbol{c} \mid \boldsymbol{c};\boldsymbol{h} \mid \texttt{if } \boldsymbol{b} \texttt{ then } \boldsymbol{h} \texttt{ else } \boldsymbol{c} \mid \texttt{if } \boldsymbol{b} \texttt{ then } \boldsymbol{c} \texttt{ else } \boldsymbol{h}$$

For each holed command $\boldsymbol{h}$, its *pre-hole fragment* $\overline{\boldsymbol{h}}$ is:

$$\begin{array}{lll}
\overline{\texttt{if } [_] \texttt{ then } \boldsymbol{c_1} \texttt{ else } \boldsymbol{c_2}} & :\equiv & \texttt{skip} \\
\overline{\boldsymbol{h};\boldsymbol{c}} :\equiv \overline{\boldsymbol{h}} \qquad \overline{\boldsymbol{c};\boldsymbol{h}} & :\equiv & \boldsymbol{c};\overline{\boldsymbol{h}} \\
\overline{\texttt{if } \boldsymbol{b} \texttt{ then } \boldsymbol{h} \texttt{ else } \boldsymbol{c}} & :\equiv & \texttt{assert } \boldsymbol{b} \,;\overline{\boldsymbol{h}} \\
\overline{\texttt{if } \boldsymbol{b} \texttt{ then } \boldsymbol{c} \texttt{ else } \boldsymbol{h}} & :\equiv & \texttt{assert } \neg\boldsymbol{b} \,;\overline{\boldsymbol{h}}
\end{array}$$

Lem.
If a Boolean expression $\boldsymbol{b_s} \in \mathbf{BExp}$ satisfies

$$\models \{\boldsymbol{b_s}\}\,\overline{\boldsymbol{h}}\,\{\boldsymbol{b_c}\}\ , \quad \models \{\neg\boldsymbol{b_s}\}\,\overline{\boldsymbol{h}}\,\{\neg\boldsymbol{b_c}\}\ , \quad \text{and} \quad \models \{\boldsymbol{b_g} \wedge \boldsymbol{b_s}\}\,\boldsymbol{h}[\boldsymbol{b_c}]\,\{\neg\boldsymbol{b_g} \vee \boldsymbol{b_s}\}\ ,$$

then we have

$$[\![\ \texttt{while } \boldsymbol{b_g} \texttt{ do } \boldsymbol{h}[\boldsymbol{b_c}]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (\boldsymbol{b_g} \wedge \neg\boldsymbol{b_s}) \texttt{ do } \boldsymbol{h}[\texttt{false}]\,; \\ \texttt{while } (\boldsymbol{b_g} \wedge \boldsymbol{b_s}) \texttt{ do } \boldsymbol{h}[\texttt{true}] \end{array}\right]\!\!\right] .$$

Hasuo (Tokyo)

Phase Split
(Nonstandard Ver., for While^{dt} & Hoare^{dt})

Defn.
The set of *holed commands* $\mathbf{Cmd}_{[\_]}$ is:

$$\mathbf{Cmd}_{[\_]} \ni h \ ::= \ \texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2 \mid h;c \mid c;h \mid \texttt{if } b \texttt{ then } h \texttt{ else } c \mid \texttt{if } b \texttt{ then } c \texttt{ else } h$$

For each holed command h, its *pre-hole fragment* $\overline{h}$ is:

$$\begin{array}{lll} \overline{\texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2} & :\equiv & \texttt{skip} \\ \overline{h;c} \ :\equiv \ \overline{h} \qquad \overline{c;h} & :\equiv & c;\overline{h} \\ \overline{\texttt{if } b \texttt{ then } h \texttt{ else } c} & :\equiv & \texttt{assert } b \,;\overline{h} \\ \overline{\texttt{if } b \texttt{ then } c \texttt{ else } h} & :\equiv & \texttt{assert } \neg b \,;\overline{h} \end{array}$$

Lem.
If a Boolean expression $b_{\mathbf{s}} \in \mathbf{BExp}$ satisfies

$$\models \{b_{\mathbf{s}}\}\,\overline{h}\,\{b_{\mathbf{c}}\}\ , \quad \models \{\neg b_{\mathbf{s}}\}\,\overline{h}\,\{\neg b_{\mathbf{c}}\}\ , \quad \text{and} \quad \models \{b_{\mathbf{g}} \wedge b_{\mathbf{s}}\}\,h[b_{\mathbf{c}}]\,\{\neg b_{\mathbf{g}} \vee b_{\mathbf{s}}\}\ ,$$

then we have

$$[\![\ \texttt{while } b_{\mathbf{g}} \texttt{ do } h[b_{\mathbf{c}}]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (b_{\mathbf{g}} \wedge \neg b_{\mathbf{s}}) \texttt{ do } h[\texttt{false}]\,; \\ \texttt{while } (b_{\mathbf{g}} \wedge b_{\mathbf{s}}) \texttt{ do } h[\texttt{true}] \end{array}\right]\!\!\right] .$$

Proof.

$\models \{b_{\mathbf{s}}\}\,\overline{h}\,\{b_{\mathbf{c}}\}$

$\models \{\neg b_{\mathbf{s}}\}\,\overline{h}\,\{\neg b_{\mathbf{c}}\}$

$\models \{b_{\mathbf{g}} \wedge b_{\mathbf{s}}\}\,h[b_{\mathbf{c}}]\,\{\neg b_{\mathbf{g}} \vee b_{\mathbf{s}}\}$

Hasuo (Tokyo)

Phase Split
(Nonstandard Ver., for While^{dt} & Hoare^{dt})

Defn.
The set of *holed commands* $\mathbf{Cmd}_{[\_]}$ is:

$$\mathbf{Cmd}_{[\_]} \ni h \ ::= \ \texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2 \mid h;c \mid c;h \mid \texttt{if } b \texttt{ then } h \texttt{ else } c \mid \texttt{if } b \texttt{ then } c \texttt{ else } h$$

For each holed command h, its *pre-hole fragment* $\overline{h}$ is:

$$\begin{array}{lll} \overline{\texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2} & :\equiv & \texttt{skip} \\ \overline{h;c} \ :\equiv \ \overline{h} \qquad \overline{c;h} & :\equiv & c;\overline{h} \\ \overline{\texttt{if } b \texttt{ then } h \texttt{ else } c} & :\equiv & \texttt{assert } b \,;\overline{h} \\ \overline{\texttt{if } b \texttt{ then } c \texttt{ else } h} & :\equiv & \texttt{assert } \neg b \,;\overline{h} \end{array}$$

Lem.
If a Boolean expression $b_{\mathbf{s}} \in \mathbf{BExp}$ satisfies

$$\models \{b_{\mathbf{s}}\}\,\overline{h}\,\{b_{\mathbf{c}}\}\ , \quad \models \{\neg b_{\mathbf{s}}\}\,\overline{h}\,\{\neg b_{\mathbf{c}}\}\ , \quad \text{and} \quad \models \{b_{\mathbf{g}} \wedge b_{\mathbf{s}}\}\,h[b_{\mathbf{c}}]\,\{\neg b_{\mathbf{g}} \vee b_{\mathbf{s}}\}\ ,$$

then we have

$$[\![\ \texttt{while } b_{\mathbf{g}} \texttt{ do } h[b_{\mathbf{c}}]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (b_{\mathbf{g}} \wedge \neg b_{\mathbf{s}}) \texttt{ do } h[\texttt{false}]\,; \\ \texttt{while } (b_{\mathbf{g}} \wedge b_{\mathbf{s}}) \texttt{ do } h[\texttt{true}] \end{array}\right]\!\!\right] .$$

Proof.

$$\begin{array}{l} \models \{b_{\mathbf{s}}\}\,\overline{h}\,\{b_{\mathbf{c}}\} \\ \models \{\neg b_{\mathbf{s}}\}\,\overline{h}\,\{\neg b_{\mathbf{c}}\} \\ \models \{b_{\mathbf{g}} \wedge b_{\mathbf{s}}\}\,h[b_{\mathbf{c}}]\,\{\neg b_{\mathbf{g}} \vee b_{\mathbf{s}}\} \end{array}$$

$\Leftrightarrow$ sectionwise

$$\begin{array}{l} \models \{b_{\mathbf{s}}|_i\}\,\overline{h|_i}\,\{b_{\mathbf{c}}|_i\} \\ \models \{\neg b_{\mathbf{s}}|_i\}\,\overline{h|_i}\,\{\neg b_{\mathbf{c}}|_i\} \\ \models \{b_{\mathbf{g}}|_i \wedge b_{\mathbf{s}}|_i\}\,h|_i[b_{\mathbf{c}}|_i]\,\{\neg b_{\mathbf{g}}|_i \vee b_{\mathbf{s}}|_i\} \end{array}$$

$\vdots$

(for almost all i)

Hasuo (Tokyo)

Phase Split
(Nonstandard Ver., for While^{dt} & Hoare^{dt})

Defn.
The set of *holed commands* $\mathbf{Cmd}_{[\_]}$ is:

$$\mathbf{Cmd}_{[\_]} \ni h \ ::= \ \texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2 \mid h;c \mid c;h \mid \texttt{if } b \texttt{ then } h \texttt{ else } c \mid \texttt{if } b \texttt{ then } c \texttt{ else } h$$

For each holed command h, its *pre-hole fragment* $\overline{h}$ is:

$$\begin{array}{lll} \overline{\texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2} & :\equiv & \texttt{skip} \\ \overline{h;c} \ :\equiv \ \overline{h} \qquad \overline{c;h} & :\equiv & c;\overline{h} \\ \overline{\texttt{if } b \texttt{ then } h \texttt{ else } c} & :\equiv & \texttt{assert } b \,;\overline{h} \\ \overline{\texttt{if } b \texttt{ then } c \texttt{ else } h} & :\equiv & \texttt{assert } \neg b \,;\overline{h} \end{array}$$

Lem.
If a Boolean expression $b_{\mathbf{s}} \in \mathbf{BExp}$ satisfies

$$\models \{b_{\mathbf{s}}\}\,\overline{h}\,\{b_{\mathbf{c}}\}\ , \quad \models \{\neg b_{\mathbf{s}}\}\,\overline{h}\,\{\neg b_{\mathbf{c}}\}\ , \quad \text{and} \quad \models \{b_{\mathbf{g}} \wedge b_{\mathbf{s}}\}\,h[b_{\mathbf{c}}]\,\{\neg b_{\mathbf{g}} \vee b_{\mathbf{s}}\}\ ,$$

then we have

$$[\![\ \texttt{while } b_{\mathbf{g}} \texttt{ do } h[b_{\mathbf{c}}]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (b_{\mathbf{g}} \wedge \neg b_{\mathbf{s}}) \texttt{ do } h[\texttt{false}]\,; \\ \texttt{while } (b_{\mathbf{g}} \wedge b_{\mathbf{s}}) \texttt{ do } h[\texttt{true}] \end{array}\right.\right] .$$

Proof.

$$\models \{b_{\mathbf{s}}\}\,\overline{h}\,\{b_{\mathbf{c}}\}$$
$$\models \{\neg b_{\mathbf{s}}\}\,\overline{h}\,\{\neg b_{\mathbf{c}}\}$$
$$\models \{b_{\mathbf{g}} \wedge b_{\mathbf{s}}\}\,h[b_{\mathbf{c}}]\,\{\neg b_{\mathbf{g}} \vee b_{\mathbf{s}}\}$$

$\Leftrightarrow$ sectionwise

$$\models \{b_{\mathbf{s}}|_i\}\,\overline{h|_i}\,\{b_{\mathbf{c}}|_i\}$$
$$\models \{\neg b_{\mathbf{s}}|_i\}\,\overline{h|_i}\,\{\neg b_{\mathbf{c}}|_i\}$$
$$\models \{b_{\mathbf{g}}|_i \wedge b_{\mathbf{s}}|_i\}\,h|_i[b_{\mathbf{c}}|_i]\,\{\neg b_{\mathbf{g}}|_i \vee b_{\mathbf{s}}|_i\}$$

(for almost all i)

$\Rightarrow\Rightarrow\Rightarrow$ std. ver.

$$[\![\ \texttt{while } b_{\mathbf{g}}|_i \texttt{ do } h|_i[b_{\mathbf{c}}|_i]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (b_{\mathbf{g}}|_i \wedge \neg b_{\mathbf{s}}|_i) \texttt{ do } h|_i[\texttt{false}]\,; \\ \texttt{while } (b_{\mathbf{g}}|_i \wedge b_{\mathbf{s}}|_i) \texttt{ do } h|_i[\texttt{true}] \end{array}\right.\right]$$

Hasuo (Tokyo)

Phase Split
(Nonstandard Ver., for While^{dt} & Hoare^{dt})

Defn.
The set of *holed commands* $\mathbf{Cmd}_{[\_]}$ is:

$$\mathbf{Cmd}_{[\_]} \ni h \ ::= \ \texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2 \mid h;c \mid c;h \mid \texttt{if } b \texttt{ then } h \texttt{ else } c \mid \texttt{if } b \texttt{ then } c \texttt{ else } h$$

For each holed command h, its *pre-hole fragment* $\overline{h}$ is:

$$\begin{array}{lll} \overline{\texttt{if } [\_] \texttt{ then } c_1 \texttt{ else } c_2} & :\equiv & \texttt{skip} \\ \overline{h;c} \ :\equiv \ \overline{h} \qquad \overline{c;h} & :\equiv & c;\overline{h} \\ \overline{\texttt{if } b \texttt{ then } h \texttt{ else } c} & :\equiv & \texttt{assert } b \,;\overline{h} \\ \overline{\texttt{if } b \texttt{ then } c \texttt{ else } h} & :\equiv & \texttt{assert } \neg b \,;\overline{h} \end{array}$$

Lem.
If a Boolean expression $b_s \in \mathbf{BExp}$ satisfies

$$\models \{b_s\}\,\overline{h}\,\{b_c\}\ , \quad \models \{\neg b_s\}\,\overline{h}\,\{\neg b_c\}\ , \quad \text{and} \quad \models \{b_g \wedge b_s\}\,h[b_c]\,\{\neg b_g \vee b_s\}\ ,$$

then we have

$$[\![\ \texttt{while } b_g \texttt{ do } h[b_c]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (b_g \wedge \neg b_s) \texttt{ do } h[\texttt{false}]\,; \\ \texttt{while } (b_g \wedge b_s) \texttt{ do } h[\texttt{true}] \end{array}\right]\!\!\right] .$$

Proof.

$$\models \{b_s\}\,\overline{h}\,\{b_c\}$$
$$\models \{\neg b_s\}\,\overline{h}\,\{\neg b_c\}$$
$$\models \{b_g \wedge b_s\}\,h[b_c]\,\{\neg b_g \vee b_s\}$$

$\Leftrightarrow$ sectionwise

$$\models \{b_s|_i\}\,\overline{h|_i}\,\{b_c|_i\}$$
$$\models \{\neg b_s|_i\}\,\overline{h|_i}\,\{\neg b_c|_i\}$$
$$\models \{b_g|_i \wedge b_s|_i\}\,h|_i[b_c|_i]\,\{\neg b_g|_i \vee b_s|_i\}$$

$\vdots$ (for almost all i)

$\Rightarrow\Rightarrow\Rightarrow$ std. ver.

$$[\![\ \texttt{while } b_g|_i \texttt{ do } h|_i[b_c|_i]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (b_g|_i \wedge \neg b_s|_i) \texttt{ do } h|_i[\texttt{false}]\,; \\ \texttt{while } (b_g|_i \wedge b_s|_i) \texttt{ do } h|_i[\texttt{true}] \end{array}\right]\!\!\right]$$

$\vdots$

sectionwise $\Leftrightarrow$

$$[\![\ \texttt{while } b_g \texttt{ do } h[b_c]\ ]\!] = \left[\!\!\left[\begin{array}{l} \texttt{while } (b_g \wedge \neg b_s) \texttt{ do } h[\texttt{false}]\,; \\ \texttt{while } (b_g \wedge b_s) \texttt{ do } h[\texttt{true}] \end{array}\right]\!\!\right]$$

Transferring Static Analysis Strategies

$$\models \{b_s\}\, \overline{h}\, \{b_c\}$$
$$\models \{\neg b_s\}\, \overline{h}\, \{\neg b_c\}$$
$$\models \{b_g \wedge b_s\}\, h[b_c]\, \{\neg b_g \vee b_s\}$$

⇕ sectionwise

$$\models \{b_s|_i\}\, \overline{h|_i}\, \{b_c|_i\}$$
$$\models \{\neg b_s|_i\}\, \overline{h|_i}\, \{\neg b_c|_i\}$$
$$\models \{b_g|_i \wedge b_s|_i\}\, h|_i[b_c|_i]\, \{\neg b_g|_i \vee b_s|_i\}$$

⋮

(for almost all i)

⇒⇒⇒ ⋮ std. ver.

$$[\![\texttt{while}\ b_g|_i\ \texttt{do}\ h|_i[b_c|_i]]\!]$$
$$= [\![\texttt{while}\ (b_g|_i \wedge \neg b_s|_i)\ \texttt{do}\ h|_i[\texttt{false}]\ ;\ \texttt{while}\ (b_g|_i \wedge b_s|_i)\ \texttt{do}\ h|_i[\texttt{true}]]\!]$$

⋮

sectionwise ⇕

$$[\![\texttt{while}\ b_g\ \texttt{do}\ h[b_c]]\!]$$
$$= [\![\texttt{while}\ (b_g \wedge \neg b_s)\ \texttt{do}\ h[\texttt{false}]\ ;\ \texttt{while}\ (b_g \wedge b_s)\ \texttt{do}\ h[\texttt{true}]]\!]$$

* Doesn't matter what "std. ver." is
* ➜ **modular method** for transfer

Hasuo (Tokyo)

```
while (v > 0) {
  if m - z < s
    then a := -b
    else a := a0;
  t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a * dt;
    t := t + dt }}

{z < m}
```

```
while (v > 0 && m - z >= s) {
  a := a0;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }};
while (v > 0 && m - z < s) {
  a := -b;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v - b * dt;
    t := t + dt }}

{z < m}
```

accel.

brake

Strategy 1 "Phase split"

[Sharma,Dillig,Dillig,Aiken; CAV'11]
[Balakrishnan,Sankaranarayanan,Ivancic,Gupta; EMSOFT'09]
[Gopan,Reps; SAS'07]

```
while (v > 0 && m - z >= s) {
  a := a0;   t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }};
while (v > 0 && m - z < s) {
  a := -b;   t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v - b * dt;
    t := t + dt }}
```

{z < m}

```
while (v > 0 && m - z >= s) {
  a := a0;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }};
while (v > 0 && m - z < s) {
  a := -b;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v - b * dt;
    t := t + dt }}

{z < m}
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;    t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }

{z < m}
```

```
while (v > 0 && m - z >= s) {
  a := a0;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }};
while (v > 0 && m - z < s) {
  a := -b;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v - b * dt;
    t := t + dt }}

{z < m}
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;    t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }

{z < m}
```

Startegies 2,3

"Superfluous guard elim." "Time elapse"

```
while (v > 0 && m - z >= s) {
  a := a0;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }};
while (v > 0 && m - z < s) {
  a := -b;    t := 0;
  while (t < eps && v > 0) {
    z := z + v * dt;
    v := v - b * dt;
    t := t + dt }}

{z < m}
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;    t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
while (v > 0) {
  a := -b;
```

Strategy 4

"Differential invariant"

[Platzer,Clarke; CAV'08]

Startegies 2,3

"Superfluous guard elim." "Time elapse"

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;   t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }

{z < m}
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;   t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }

{z < m}
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;   t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
```

$(v > 0 \vee m > z) \wedge$

$\{ \; (b^2 \mathrm{dt}^2 + 4b\mathrm{dt}v + 8bz + 4v^2 < 8bm \quad \}$

$\quad \vee b\mathrm{dt}v + 2bz + v^2 \leq 2bm)$

```
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;   t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }

{z < m}
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;   t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
  (v > 0 ∨ m > z)∧
{ (b²dt² + 4bdtv + 8bz + 4v² < 8bm        }
    ∨ bdtv + 2bz + v² ≤ 2bm)
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

Strategy 5

"QE Invariant"

QE Invariant

Lem. In $\textsc{Hoare}^{\mathtt{dt}}$,

$$\vdash \left\{ \begin{array}{l} (\neg \boldsymbol{b} \Rightarrow \boldsymbol{A}) \wedge \\ \forall \boldsymbol{y} \in {}^*\mathbb{N}.\big(\big(\boldsymbol{b}[\boldsymbol{a}/\boldsymbol{x}]^{\boldsymbol{y}} \wedge \neg \boldsymbol{b}[\boldsymbol{a}/\boldsymbol{x}]^{\boldsymbol{y}+1} \big) \Rightarrow \boldsymbol{A}[\boldsymbol{a}/\boldsymbol{x}]^{\boldsymbol{y}+1} \big) \end{array} \right\}$$

$$\mathtt{while}\ \boldsymbol{b}\ \mathtt{do}\ \boldsymbol{x} := \boldsymbol{a}\ \{\boldsymbol{A}\}\ .$$

Hasuo (Tokyo)

QE Invariant

Lem. In $\textsc{Hoare}^{\mathrm{dt}}$,

$$\vdash \left\{ \begin{array}{l} (\neg b \Rightarrow A) \wedge \\ \forall y \in {}^*\mathbb{N}.\big(\big(b[a/x]^y \wedge \neg b[a/x]^{y+1} \big) \Rightarrow A[a/x]^{y+1} \big) \end{array} \right\}$$

$$\texttt{while}\ b\ \texttt{do}\ x := a\ \{A\}\ .$$

quantifier must go!
(to manage complexity)

Hasuo (Tokyo)

QE Invariant

Lem. In $\textsc{Hoare}^{\mathrm{dt}}$,

$$\vdash \left\{ \begin{array}{l} (\neg b \Rightarrow A) \wedge \\ \forall y \in {}^*\mathbb{N}.\big(\big(b[a/x]^y \wedge \neg b[a/x]^{y+1} \big) \Rightarrow A[a/x]^{y+1} \big) \end{array} \right\} \texttt{while } b \texttt{ do } x := a \ \{A\} \ .$$

quantifier must go!
(to manage complexity)

* **Quantifier elimination**
  * Tarski, CAD algorithm, `Resolve` in Mathematica

QE Invariant

Lem. In $\textsc{Hoare}^{\mathrm{dt}}$,

$$\vdash \left\{ \begin{array}{l} (\neg b \Rightarrow A) \wedge \\ \forall y \in {}^*\mathbb{N}.\big(\big(b[a/x]^y \wedge \neg b[a/x]^{y+1} \big) \Rightarrow A[a/x]^{y+1} \big) \end{array} \right\} \texttt{while}\ b\ \texttt{do}\ x := a\ \{A\}\ .$$

quantifier must go!
(to manage complexity)

* **Quantifier elimination**
  * Tarski, CAD algorithm, `Resolve` in Mathematica
  * e.g. $\models\ \forall x \in \mathbb{R}.\,(x^2 + ax + b > 0) \iff a^2 - 4b < 0$

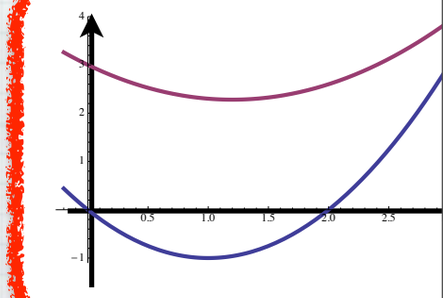
Hasuo (Tokyo)

QE Invariant

Lem. In $\textsc{Hoare}^{\mathrm{dt}}$,

$$\vdash \left\{ \begin{array}{l} (\neg b \Rightarrow A) \wedge \\ \forall y \in {}^*\mathbb{N}.\big(\big(b[a/x]^y \wedge \neg b[a/x]^{y+1} \big) \Rightarrow A[a/x]^{y+1} \big) \end{array} \right\} \texttt{while } b \texttt{ do } x := a \ \{A\} \ .$$

quantifier must go!
(to manage complexity)



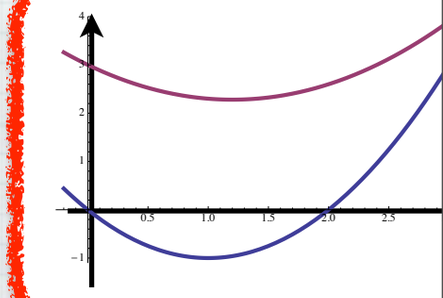
* **Quantifier elimination**
  * Tarski, CAD algorithm, `Resolve` in Mathematica
  * e.g. $\models \forall x \in \mathbb{R}.\, (x^2 + ax + b > 0) \iff a^2 - 4b < 0$

QE Invariant

Lem. In $\textsc{Hoare}^{\mathrm{dt}}$,

$$\vdash \left\{ \begin{array}{l} (\neg b \Rightarrow A) \wedge \\ \forall y \in {}^*\mathbb{N}.\big(\big(b[a/x]^y \wedge \neg b[a/x]^{y+1} \big) \Rightarrow A[a/x]^{y+1} \big) \end{array} \right\} \texttt{while}\ b\ \texttt{do}\ x := a\ \{A\}\ .$$

quantifier must go!
(to manage complexity)



* **Quantifier elimination**
  * Tarski, CAD algorithm, `Resolve` in Mathematica
  * e.g. $\models\ \forall x \in \mathbb{R}.\,(x^2 + ax + b > 0) \iff a^2 - 4b < 0$
  * then $\models\ \forall x \in {}^*\mathbb{R}.\,(x^2 + ax + b > 0) \iff a^2 - 4b < 0$

by **transfer**!

Hasuo (Tokyo)

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;   t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }

{z < m}
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;   t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
  (v > 0 ∨ m > z)∧
{ (b²dt² + 4bdtv + 8bz + 4v² < 8bm        }
    ∨ bdtv + 2bz + v² ≤ 2bm)
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

Strategy 5

"QE Invariant"

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;    t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
```

$(v > 0 \vee m > z) \wedge$

$\{\ (b^2 \mathrm{dt}^2 + 4b\mathrm{dt}v + 8bz + 4v^2 < 8bm \quad \}$

$\vee\ b\mathrm{dt}v + 2bz + v^2 \leq 2bm)$

```
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;    t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
```

$(v > 0 \lor m > z) \land$

$\{\ (b^2 \mathrm{dt}^2 + 4b\mathrm{dt}v + 8bz + 4v^2 < 8bm \quad \}$

$\quad \lor\ b\mathrm{dt}v + 2bz + v^2 \leq 2bm)$

```
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

+ some fwd. propagation

```
{ ... (long fml. with dt) }
while (m - z >= s) {
  a := a0;    t := 0;
  while (t < eps) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }}
{ ... }
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;    t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
```

$(v > 0 \vee m > z) \wedge$

$\{ \; (b^2 \mathrm{dt}^2 + 4b\mathrm{dt}v + 8bz + 4v^2 < 8bm \quad \}$

$\quad \vee b\mathrm{dt}v + 2bz + v^2 \leq 2bm)$

```
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

+ some fwd. propagation

```
{ ... (long fml. with dt) }
while (m - z >= s) {
  a := a0;    t := 0;
  while (t < eps) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }}
{ ... }
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

Strategy 6

"Iteration count"

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;    t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
```

$(v > 0 \vee m > z) \wedge$

$\{\ (b^2 dt^2 + 4b dt v + 8bz + 4v^2 < 8bm \quad \}$

$\quad \vee\ b dt v + 2bz + v^2 \leq 2bm)$

```
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

+ some fwd. propagation

```
{ ... (long fml. with dt) }
while (m - z >= s) {
  a := a0;    t := 0;
  while (t < eps) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }}
{ ... }
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

Strategy 6

"Iteration count"

```
x := 0;
while (x < x0) {
  x := x + a
}
```

```
if (v > 0)
  then
    while (m - z >= s) {
      a := a0;    t := 0;
      while (t < eps) {
        z := z + v * dt;
        v := v + a0 * dt;
        t := t + dt }}
  else skip;
```

$(v > 0 \vee m > z) \wedge$

$\{\ (b^2 dt^2 + 4bdtv + 8bz + 4v^2 < 8bm \quad \}$

$\quad \vee\ bdtv + 2bz + v^2 \leq 2bm)$

```
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

+ some fwd. propagation

```
{ ... (long fml. with dt) }
while (m - z >= s) {
  a := a0;    t := 0;
  while (t < eps) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }}
{ ... }
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

Strategy 6

"Iteration count"

```
x := 0;
while (x < x0) {
  x := x + a
}
```

iteration: x0/a times?

* approximated by ⌊x0/a⌋ or ⌈x0/a⌉
* ➜ monotonicity reqm. must be discharged

```
{ ... (long fml. with dt) }
while (m - z >= s) {
  a := a0;   t := 0;
  while (t < eps) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }}
{ ... }
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

```
{ ... (long fml. with dt) }
while (m - z >= s) {
  a := a0;   t := 0;
  while (t < eps) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }}
{ ... }
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

long fml. w/o `dt`, whose core is

$$a_0(2\varepsilon\sqrt{2a_0(m-s-z_0)+v_0^2}+b\epsilon^2+2m-2s-2z_0)$$
$$+2b\varepsilon\sqrt{2a_0(m-s-z_0)+v_0^2}+a_0^2\epsilon^2+v_0^2<2bs$$

```
{ ... (long fml. with dt) }
while (m - z >= s) {
  a := a0;   t := 0;
  while (t < eps) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }}
{ ... }
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

long fml. w/o `dt`, whose core is

$$a_0(2\varepsilon\sqrt{2a_0(m-s-z_0)+v_0^2}+b\epsilon^2+2m-2s-2z_0)$$
$$+2b\varepsilon\sqrt{2a_0(m-s-z_0)+v_0^2}+a_0^2\epsilon^2+v_0^2<2bs$$

Strategy 7

"Cast to shadow"

(Eliminates `dt`, strengthens the precond.)

```
{ ... (long fml. with dt) }
while (m - z >= s) {
  a := a0;   t := 0;
  while (t < eps) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }}
{ ... }
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

long fml. w/o `dt`, whose core is

$$a_0(2\varepsilon\sqrt{2a_0(m-s-z_0)+v_0^2}+b\epsilon^2+2m-2s-2z_0)$$
$$+2b\varepsilon\sqrt{2a_0(m-s-z_0)+v_0^2}+a_0^2\epsilon^2+v_0^2<2bs$$

Lem. If:

1. $\boldsymbol{a}$ is *closed*
2. $\boldsymbol{r} \mapsto [\![\boldsymbol{a}[\boldsymbol{r}/\mathtt{dt}]]\!]$ is continuous at $\boldsymbol{r} = \mathbf{0}$,

then $\models \boldsymbol{a}[\mathbf{0}/\mathtt{dt}] < \mathbf{0} \Longrightarrow \boldsymbol{a} < \mathbf{0}$.

Strategy 7

"Cast to shadow"

(Eliminates `dt`, strengthens the precond.)

```
{ ... (long fml. with dt) }
while (m - z >= s) {
  a := a0;   t := 0;
  while (t < eps) {
    z := z + v * dt;
    v := v + a0 * dt;
    t := t + dt }}
{ ... }
while (v > 0) {
  a := -b;
  z := z + v * dt;
  v := v - b * dt }
```

long fml. w/o dt, whose core is

$$a_0(2\varepsilon\sqrt{2a_0(m-s-z_0)+v_0^2}+b\epsilon^2+2m-2s-2z_0)$$
$$+2b\varepsilon\sqrt{2a_0(m-s-z_0)+v_0^2}+a_0^2\epsilon^2+v_0^2<2bs$$

the final outcome

Lem. If:

1. $\boldsymbol{a}$ is *closed*
2. $\boldsymbol{r} \mapsto [\![\boldsymbol{a}[\boldsymbol{r}/\mathtt{dt}]]\!]$ is continuous at $\boldsymbol{r} = \mathbf{0}$,

then $\models \boldsymbol{a}[\mathbf{0}/\mathtt{dt}] < \mathbf{0} \Longrightarrow \boldsymbol{a} < \mathbf{0}$.

Strategy 7

"Cast to shadow"

(Eliminates dt, strengthens the precond.)

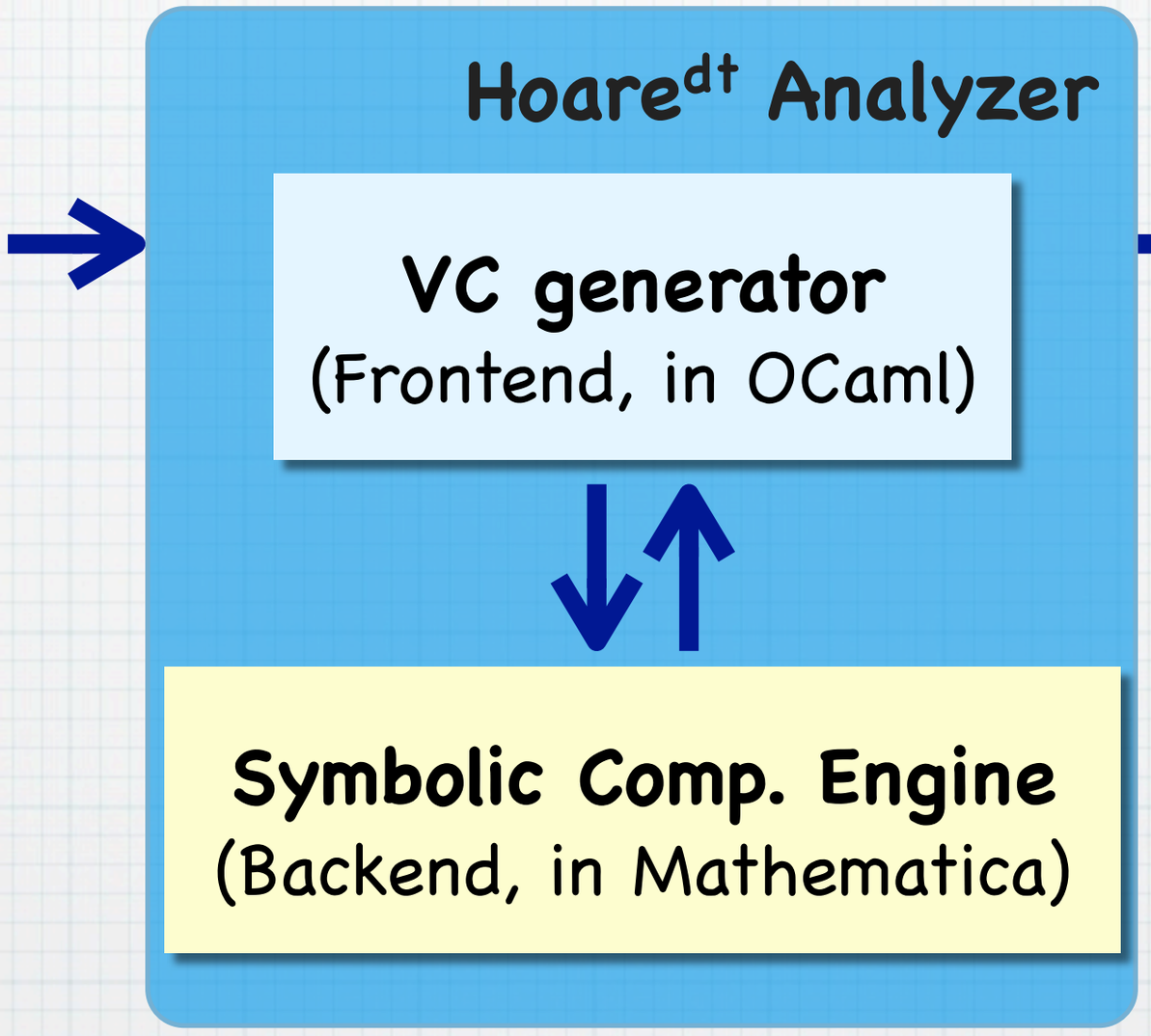
Prototype Automatic Prover

P

(While^{dt} program)

B

(postcondition)



A

(precondition)

s.t.

$\vdash \{A\}P\{B\}$

* Fujitsu HX600 with Quad Core AMD Opteron 2.3GHz CPU, 32GB memory. Mathematica 7.0 for Linux x86 (64-bit)
  * ETCS: 40.96 sec.
  * Bouncing ball: runs with one manual insertion of invariants

Hasuo (Tokyo)

Prototype Automatic Prover

P

(While^{dt} program)

B

(postcondition)

Hoare^{dt} Analyzer

VC generator
(Frontend, in OCaml)

Symbolic Comp. Engine
(Backend, in Mathematica)

A

(precondition)
s.t.
$\vdash \{A\}P\{B\}$

Totally symbolic
(crucial for transfer)

* Fujitsu HX600 with Quad Core AMD Opteron 2.3GHz CPU, 32GB memory. Mathematica 7.0 for Linux x86 (64-bit)
  * ETCS: 40.96 sec.
  * Bouncing ball: runs with one manual insertion of invariants

Related Work

- **Deductive verification** of hybrid sys. [Platzer, '10] [Platzer, LICS'12]
  - Automatic prover KeYmaera
- **Static analysis techniques**
  - A **LOT** in CAV, SAS, VMCAI, ...
  - Applied to hybrid systems (w/ diff. eq.)
    [Rodriguez-Carbonell, Tiwari; HSCC'05] [Sankaranarayanan; HSCC'10]
    [Sankaranarayanan, Sipma, Manna; Formal Methods Sys. Design '08]
- Use of **NSA for hybrid systems**
  [Benveniste, Bourke, Caillaud, Pouzet; J. Comput. Syst. Sci. '12]
  [Bliudze, Krob; Fundam. Inform. '09] [Gamboa, Kaufmann; J. Autom. Reason. '01]
- **Continuous techniques applied to discrete appl.**
  [Chaudhuri, Gulwani, Lublinerman, NavidPour; FSE '11]
  - Not contending! Combination?

Conclusions

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

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

Rigorous semantics by non-standard analysis

Hasuo (Tokyo)

Conclusions

Nonstandard Static Analysis

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

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

Rigorous semantics by non-standard analysis

Hasuo (Tokyo)

Conclusions

Nonstandard Static Analysis

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

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

Rigorous semantics by non-standard analysis

- Tool's effectivity. More heuristics?
- (Any discrete frmwk.)^{dt} ?
- Simulink as stream processing?
- With (explicit) differential equations?

Hasuo (Tokyo)

Conclusions

Nonstandard Static Analysis

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

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

Rigorous semantics by non-standard analysis

- Tool's effectivity. More heuristics?
- (Any discrete frmwk.)[dt] ?
- Simulink as stream processing?
- With (explicit) differential equations?

Thank you for your attention!

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

Hasuo (Tokyo)