Verifying iRRAM-like implementation of exact real computation

Sewon Park j.w.w. Holger Thies
Kyoto University

CCC 2023
September 25-29, 2023, Kyoto, Japan

Sewon Park is a JSPS International Research Fellow supported by JSPS KAKENHI (Grant-in-Aid for JSPS Fellows) JP22F22071.

Holger Thies  is supported by JSPS KAKENHI Grant Numbers JP20K19744 and JP23H03346.

Motivation and Overview

Motivation: Verified exact real computation

$\{x, y \in \mathbb{R}, y \neq 0\}$

$\texttt{Real}\ z := 333.75y^6 + x^2(11x^2y^2 - y^6 - 121y^4 - 2) + 5.5y^8 + x/(2y)$

$*$*example* $R(x, y)$ *by Siegfried Rump*

$\{z = R(x, y)\}$

- *Imperative programming* with *abstract type of real numbers* and TTE-based *primitive operations*.
- Verification calculus trusting the primitive operations

$$\{P\}\ c\ \{Q\}$$

based on Hoare-style logic. E.g.,

$$\{x, y \in \mathbb{R}\}\ \texttt{Real}\ z := x + y\ \{z = x + y\}$$

- Verify the implementation(s) of the primitive operations.
- **IRRAM** by Norbert Müller

iRRAM-style implementation of ERC

x	$\sqrt{2}$
y	π

$\{\mathsf{T}\}$

$\texttt{Real}\ z := x + y$

$\{z = x + y\}$

x	$[1, 2]$	$[1.4, 1.5]$	$[1.41, 1.42]$
y	$[3, 4]$	$[3.1, 3.2]$	$[3.14, 3.15]$

$\{x \Vdash \hat{x}, y \Vdash \hat{y}\}$

$\texttt{Intvl}\ z := \texttt{IntvlAdd}(x, y)$

$\{z \Vdash \hat{z}, \hat{z} = \hat{x} + \hat{y}\}$

x	$\sqrt{2}$
y	π
z	$\sqrt{2}+\pi$

x	$[1, 2]$	$[1.4, 1.5]$	$[1.41, 1.42]$
y	$[3, 4]$	$[3.1, 3.2]$	$[3.14, 3.15]$
z	$[4, 6]$	$[4.5, 4.7]$	$[4.55, 4.57]$

Reiteration exception

x	$\pi + 0.2$
y	π

$\{x \neq y\}$

$\texttt{Bool}\ z := x > y$

$\{z = (x > y)\}$

x	$[3, 4]$	$[3.3, 3.4]$	$[3.34, 3.44]$
y	$[3, 4]$	$[3.1, 3.2]$	$[3.14, 3.15]$

$\{\hat{x} \neq \hat{y}\}$

if $\texttt{IntvlGt}(x, y)$ **then** $z := \mathsf{T}$

else if $\texttt{IntvlGt}(y, x)$ **then** $z := \mathsf{F}$

else reiterate

$\{\hat{z} = (\hat{x} > \hat{y})\}$

x	$\pi + 0.2$		
y	π		
z	T		

x	$\cdot$	$[3.3, 3.4]$	$[3.34, 3.44]$
y	$\cdot$	$[3.1, 3.2]$	$[3.14, 3.15]$
z	$\cdot$	T	T

Heap memory for inter-iteration communication

a	F	F	T
b	$\cdot$	$\cdot$	T
x	$[3, 4]$	$[3.3, 3.4]$	$[3.34, 3.44]$
y	$[3, 4]$	$[3.1, 3.2]$	$[3.14, 3.15]$

$\{(a \mapsto \mathsf{F}) \wedge (\hat{x} \neq \hat{y})\}$

if $[a]$ **then** $z := [b]$ **else**

if $\texttt{IntvlGt}(x, y)$ **then** $[a] := \mathsf{T}; [b], z := \mathsf{T}$

elif $\texttt{IntvlGt}(y, x)$ **then** $[a] := \mathsf{T}; [b], z := \mathsf{F}$

else reiterate

$\{\hat{z} = (\hat{x} > \hat{y})\}$

x	$\cdot$	$[3.3, 3.4]$	$[3.34, 3.44]$
y	$\cdot$	$[3.1, 3.2]$	$[3.14, 3.15]$
z	$\cdot$	T	T
a	$\cdot$	T	T
b	$\cdot$	T	T

x	$\pi + 0.2$
y	π

$\{x \neq y\}$

$\texttt{Bool}\ z := x > y$

$\{z = (x > y)\}$

x	$\pi + 0.2$
y	π
z	T

Overview of the project

- Imperative while language over discrete (approximating) data with
  Reiteration exception **reiterate**
  Heap memory that carry on over iterations
  $x := [e] \quad | \quad [e_1] := e_2 \quad | \quad x := \mathbf{cons}(e) \quad | \quad \mathbf{dispose}(e)$

- **This work**: Natural way to formally specify reiterations (for compositional reasoning justifying verifying only the base operations):
  $$\{P\}\ c\ \{Q\}$$

- **Future work**: Prove correctness of base operations: assignments, compositions, loops, arithmetic

$$\{\hat{x} \in \text{domain of } f\} \text{ implementation of } y := f(x)\ \{\hat{y} = f(\hat{x})\}$$

Specifying reiterations

Type-2 semantics

- Step semantics: $[\![c]\!] : (\Sigma \times H) \to \left(\Sigma \times H + \{\bot, \mathbf{reiterate}(h)\}\right)$

  Σ: the set of stores

  H: the set of heap memories

$$[\![c]\!] : (s, h) \mapsto \begin{cases} (s', h') & \text{computation terminates} \\ \mathbf{reiterate}(h') & \text{reiteration exception at heap } h' \\ \bot & \text{infinite loop or memory failures} \end{cases}$$

- Reit. semantics:

  $[\![c]\!]^\omega : (\Sigma^\omega \times H) \to \left(\Sigma \times H + \{\bot, \mathbf{reiterate}(h)\}\right)^\omega$

$$[\![c]\!]^\omega(\mathbf{s}, h)(n+1) = [\![c]\!](\mathsf{s}_{n+1}, h')$$

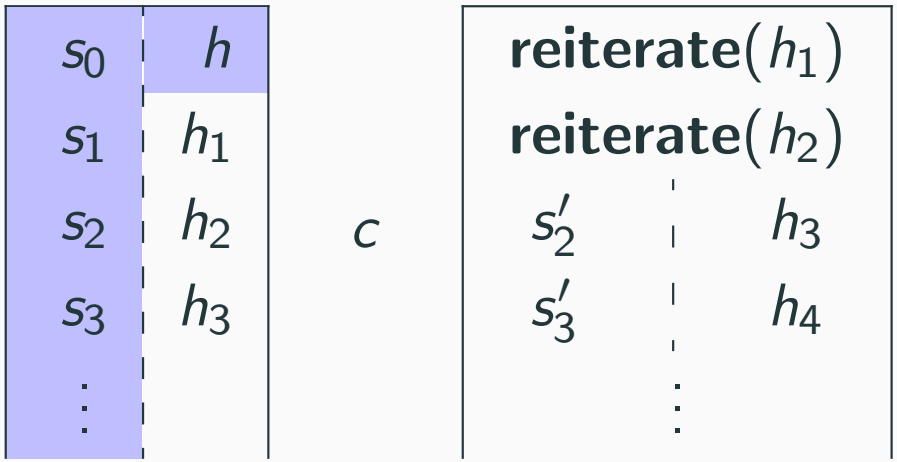
  when $[\![c]\!]^\omega(\mathsf{s}, h) = (s', h')$ or $\mathbf{reiterate}(h')$.

Type-2 semantics (2)

- Reit. semantics:
  $[\![c]\!]^\omega : (\Sigma^\omega \times H) \to \left(\Sigma \times H + \{\bot, \mathbf{reiterate}(h)\}\right)^\omega$

$$[\![c]\!]^\omega(\mathbf{s}, h)(n+1) = [\![c]\!](\mathrm{s}_{n+1}, h')$$

when $[\![c]\!]^\omega(\mathrm{s}, h)(n) = (s', h')$ or $\mathbf{reiterate}(h')$.



Reiteration invariants

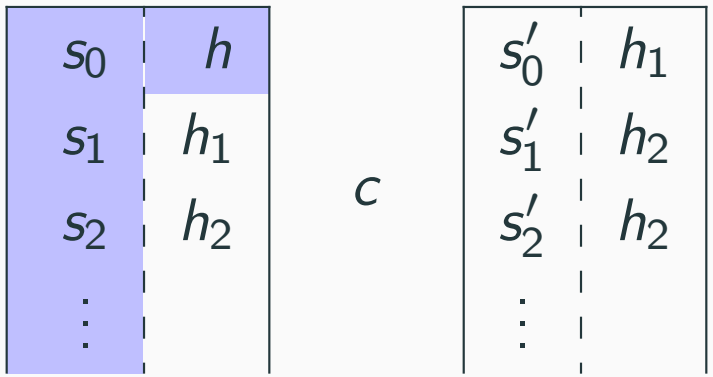
Mission! Specifying reiterations for compositional reasoning

$$\{P\}\ c\ \{Q\}$$

- Reit.-invariant precondition $P \subseteq \Sigma^\omega \times H$
- Reit.-invariant postcondition $Q \subseteq \Sigma^\omega \times H$

  For all $(\mathbf{s}, h) \in P$, the computation succeeds with $(\mathbf{s}', h_i) \in Q$.

  Moreover, all heaps h_i appear in reiteration satisfy $(\mathbf{s}, h_i) \in P$.



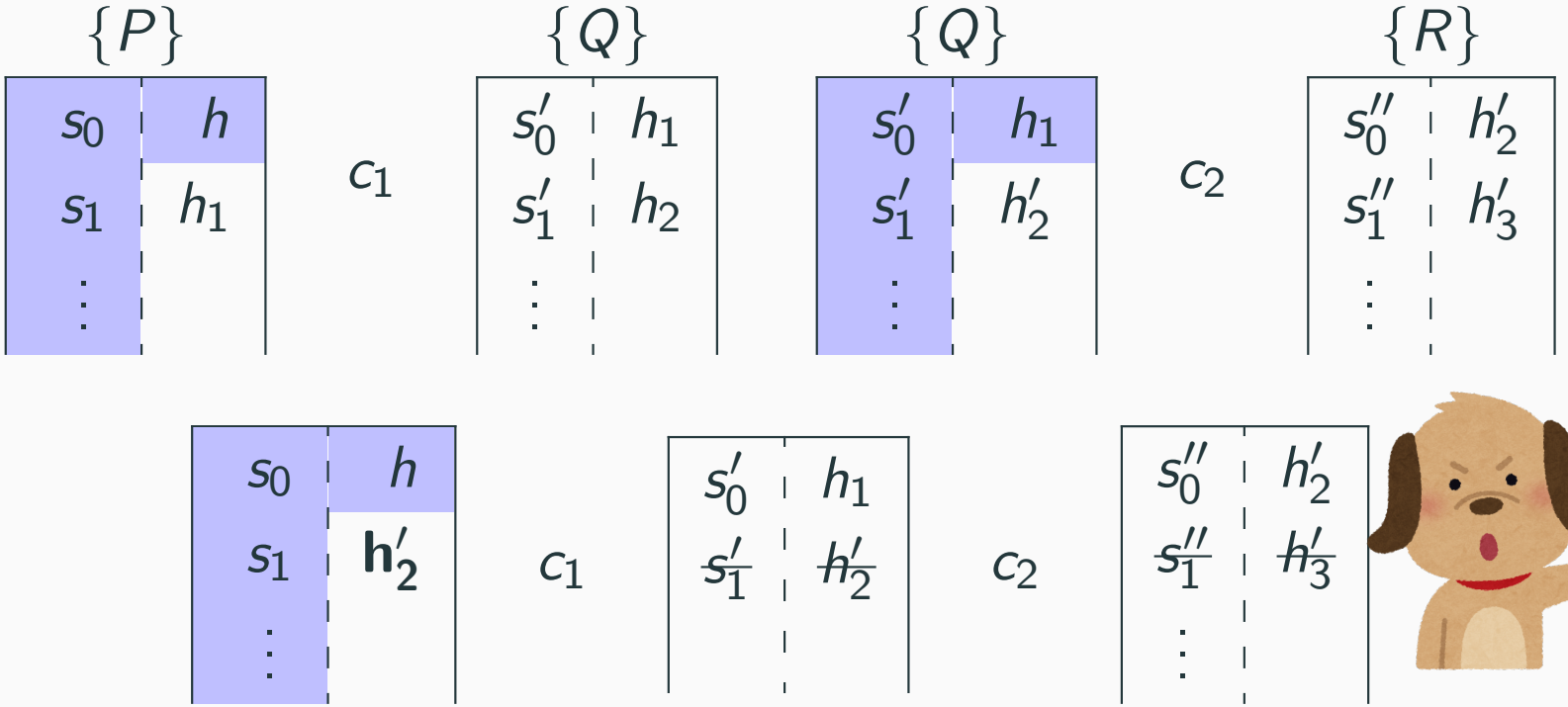
$\{a \mapsto \mathsf{T} \Rightarrow (b \mapsto \mathsf{T} \wedge \hat{x} > \hat{y}) \vee (b \mapsto \mathsf{F} \wedge \hat{x} < \hat{y}),\ \ \hat{x} \neq \hat{y}\}$

$z := \mathtt{Gt}(x, y;\ a, b)$

$\{a \mapsto \mathsf{T} \wedge (b \mapsto \mathsf{T} \wedge \hat{x} > \hat{y}) \vee (b \mapsto \mathsf{F} \wedge \hat{x} < \hat{y}),\ \ \hat{z} = (\hat{x} > \hat{y})\}$

Reiteration invariants - not compositional

- Rule for compositions fails:



- E.g.,

$$\frac{\{P\}\ z := \mathtt{Gt}(x, y;\ a, b)\ \{Q\}\quad \{Q\}\ z' := \mathtt{Gt}(x', y';\ a', b')\ \{R\}\qquad a, b, a', b' \text{ all distinct}}{\{P\}\ z := \mathtt{Gt}(x, y;\ a, b); z' := \mathtt{Gt}(x', y';\ a', b')\ \{R\}}$$

Reiteration invariants and consistency conditions

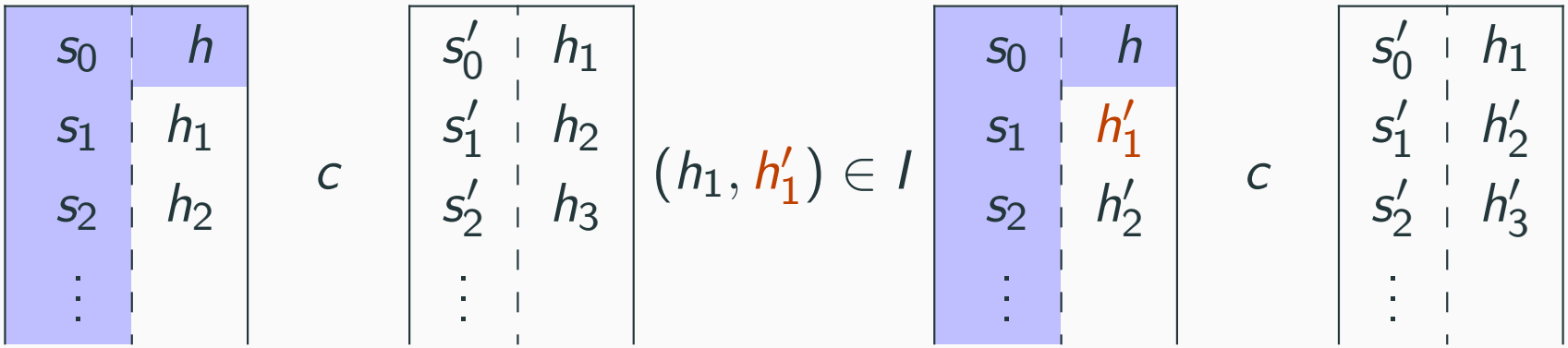
Mission! Specifying computation for compositional reasoning

$$I \triangleright \{P\}\ c\ \{Q\} \triangleright J$$

- Reit.-invariant precondition $P \subseteq \Sigma^\omega \times H$
- Reit.-invariant postcondition $Q \subseteq \Sigma^\omega \times H$
- Consistency condition $I \subseteq H \times H$

  For all $(h, h') \in I$, computation results the same store.
- Heap specification $J \subseteq H \times H$

  For all h' which is a result of h, $(h, h') \in J$



E.g., $(h_1, h_2) \in J$

Specification example

$$I \triangleright \{P\}\ c\ \{Q\} \triangleright J$$

- Reit.-invariant precondition $P \subseteq \Sigma^\omega \times H$
- Reit.-invariant postcondition $Q \subseteq \Sigma^\omega \times H$
- Consistency condition $I \subseteq H \times H$
- Heap specification $J \subseteq H \times H$

values at a, b do not change $\triangleright$

$\{a \mapsto \mathsf{T} \Rightarrow (b \mapsto \mathsf{T} \wedge \hat{x} > \hat{y}) \vee (b \mapsto \mathsf{F} \wedge \hat{x} < \hat{y}),\ \ \hat{x} \neq \hat{y}\}$

$z := \mathtt{Gt}(x, y;\ a, b)$

$\{a \mapsto \mathsf{T} \wedge (b \mapsto \mathsf{T} \wedge \hat{x} > \hat{y}) \vee (b \mapsto \mathsf{F} \wedge \hat{x} < \hat{y}),\ \ \hat{z} = (\hat{x} > \hat{y})\}$

$\triangleright$ values at a, b do not change

Compositional reasoning about reiterations

Rule for composition:

$$\frac{I \triangleright \{P\}\ c_1\ \{Q\} \triangleright J \qquad J \triangleright \{Q\}\ c_2\ \{R\} \triangleright I}{I \triangleright \{P\}\ c_1; c_2\ \{R\} \triangleright I \circ J}$$

the values at a, b, a', b' do not change $\triangleright$

$\{\cdots\}\ z := \mathtt{Gt}(x, y;\ \ a, b);\ \ \{\cdots\}$

$\triangleright$ the values at a, b, a', b' do not change

the values at a', b', a, b do not change $\triangleright$

$\{\cdots\}\ z' := \mathtt{Gt}(x', y';\ \ a', b');\ \ \{\cdots\}$

$\triangleright$ the values at a', b', a, b do not change

Conclusion and future work

We have

- devised a framework for compositional reasoning about iRRAM-style implementations of ERC
- and done case studies on some basic operations

We aim to

- extend this to more operations

  including the MPFR library

  and multi-value cache in iRRAM
- add function objects and pointers for implementations of limit operations
- and specify a formal logic for reiteration conditions

% Thank you for your attention!