Proving Hypersafety Compositionally

Appeared on OOPSLA 2022

Emanuele D'Osualdo, Azadeh Farzan, Derek Dreyer

MPI-SWS, University of Toronto, MPI-SWS

Presented by Jialu Bao, March 29th, 2023

Hyperproperties

- Hyperproperties: Properties of multiple program traces.

Hyperproperties

- Hyperproperties: Properties of multiple program traces.
- Examples:
  - Program Equivalence, e.g.,

Hyperproperties

- Hyperproperties: Properties of multiple program traces.
- Examples:
  - Program Equivalence, e.g.,
    - Commutativity (2-property): $f(a, b) = f(b, a)$.
    - Associativity (4-property): $f(a, f(b, c)) = f(f(a, b), c)$.

Relational Hoare Logic (RHL)

Relational Hoare Logic (RHL)

Judgments are written as

$$\vdash \ \{\Psi\} \ [1 : t_1, 2 : t_2] \ \{\Phi\} \,,$$

where Ψ, Φ are assertions on pairs of stores and t_1, t_2 are programs.

Relational Hoare Logic (RHL)

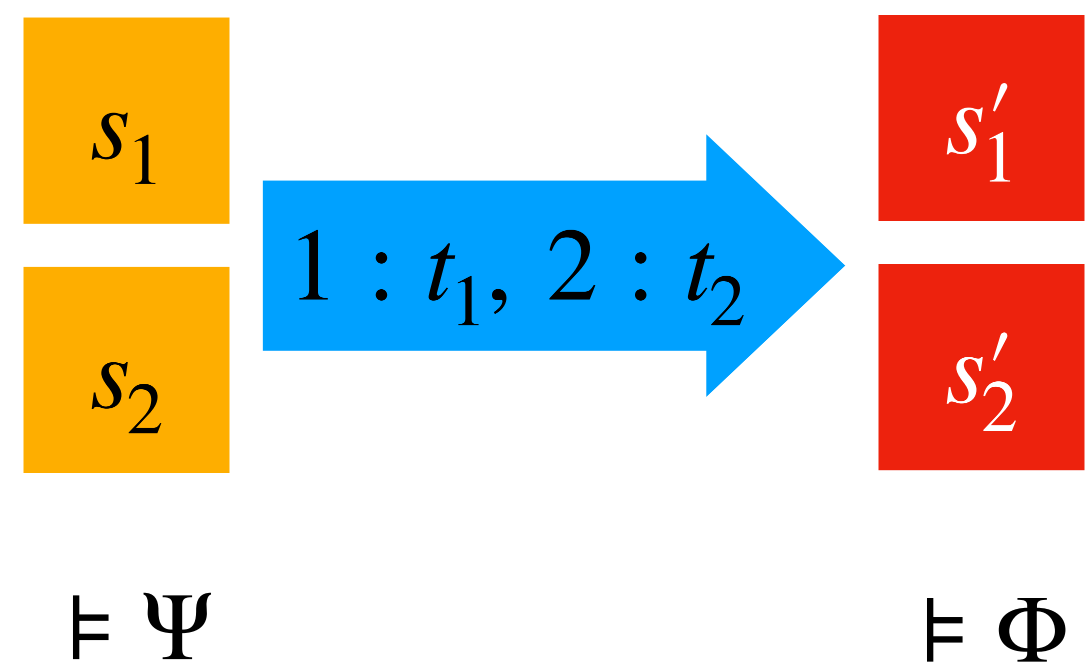
Judgments are written as

$$\vdash \ \{\Psi\} \ [1 : t_1, 2 : t_2] \ \{\Phi\} \, ,$$

where Ψ, Φ are assertions on pairs of stores and t_1, t_2 are programs. For example, let

- Store s_1 with $s_1(x) = 0$, $s_1(y) = 1$;
- Store s_2 with $s_2(x) = 0$, $s_2(y) = 2$.

Then, $(s_1, s_2) \models x\langle 1\rangle = x\langle 2\rangle \wedge y\langle 1\rangle + 1 = y\langle 2\rangle$.



Relational Hoare Logic (RHL)

- Sample rules in standard RHL:

$$\frac{}{\vdash \left\{\Phi[e\langle 1\rangle/x\langle 1\rangle, e'\langle 2\rangle/y\langle 2\rangle]\right\} \ \left[1 : x := e,\ 2 : y := e'\right] \ \{\Phi\}} \textsc{Assn}$$

Relational Hoare Logic (RHL)

- Sample rules in standard RHL:

$$\frac{}{\vdash \left\{\Phi[e\langle 1\rangle/x\langle 1\rangle, e'\langle 2\rangle/y\langle 2\rangle]\right\}\ \left[1 : x := e,\ 2 : y := e'\right]\ \{\Phi\}}\ \textsc{Assn}$$

$$\frac{\vdash \{\Phi\}\ [1 : t_1,\ 2 : t_2]\ \{\Phi'\} \qquad \vdash \{\Phi'\}\ [1 : t_1',\ 2 : t_2']\ \{\Phi''\}}{\vdash \{\Phi\}\ [1 : t_1; t_1',\ 2 : t_2; t_2']\ \{\Phi''\}}\ \textsc{Seq}$$

Relational Hoare Logic (RHL)

- Sample rules in standard RHL:

$$\frac{}{\vdash \{\Phi[e\langle 1\rangle / x\langle 1\rangle, e'\langle 2\rangle / y\langle 2\rangle]\} \; [1 : x := e, \; 2 : y := e'] \; \{\Phi\}} \; \textsc{Assn}$$

$$\frac{\vdash \{\Phi\} \; [1 : t_1, \; 2 : t_2] \; \{\Phi'\} \qquad \vdash \{\Phi'\} \; [1 : t_1', \; 2 : t_2'] \; \{\Phi''\}}{\vdash \{\Phi\} \; [1 : t_1; t_1', \; 2 : t_2; t_2'] \; \{\Phi''\}} \; \textsc{Seq}$$

- Pros: Exploit the similar structures of related programs
- Cons: Rigid in the number and the alignment of related programs.

Motivating Example

Consider a deterministic program *op* that is also commutative, i.e.,

$$\vdash \{\top\}\ [1 : r_1 := op(a, b),\ 2 : r_2 := op(b, a)]\ \{r_1\langle 1\rangle = r_2\langle 2\rangle\} \qquad (\text{Comm}_{op})$$

Motivating Example

Consider a deterministic program *op* that is also commutative, i.e.,

$$\vdash \{\top\}\ [1 : r_1 := op(a, b),\ 2 : r_2 := op(b, a)]\ \{r_1\langle 1\rangle = r_2\langle 2\rangle\} \qquad (\mathrm{Comm}_{op})$$

How do we prove the following?

$$\vdash \{\top\} \left[\begin{array}{l} 1 : x := op(a, b); z := op(x, x), \\ 2 : x := op(a, b); y := op(b, a); z := op(x, y) \end{array} \right] \{z\langle 1\rangle = z\langle 2\rangle\}$$

Derivation Sketch for Motivating Example

Derivation Sketch for Motivating Example

$$\text{Seq}\ \dfrac{\vdash \{\top\} \left[\begin{array}{l} 1 : x := op(a, b), \\ \\ 2 : x := op(a, b); y := op(b, a) \end{array}\right] \left\{\begin{array}{l} x\langle 1\rangle = x\langle 2\rangle \\ x\langle 1\rangle = y\langle 2\rangle \end{array}\right\} \qquad \bigstar}{\vdash \{\top\} \left[\begin{array}{l} 1 : x := op(a, b); z := op(x, x) \\ 2 : x := op(a, b); y := op(b, a); z := op(x, y) \end{array}\right] \{z\langle 1\rangle = z\langle 2\rangle\}}$$

where $\bigstar$ abbreviates $\left\{\begin{array}{l} x\langle 1\rangle = x\langle 2\rangle \\ x\langle 1\rangle = y\langle 2\rangle \end{array}\right\} \left[\begin{array}{l} 1 : z := op(x, x) \\ 2 : z := op(x, y) \end{array}\right] \{z\langle 1\rangle = z\langle 2\rangle\}$

Derivation Sketch for Motivating Example

$$
\text{Seq}\ \dfrac{\dfrac{?}{\vdash \{\top\} \left[\begin{array}{l} 1 : x := op(a, b), \\ \\ 2 : x := op(a, b); y := op(b, a) \end{array}\right] \left\{\begin{array}{l} x\langle 1\rangle = x\langle 2\rangle \\ x\langle 1\rangle = y\langle 2\rangle \end{array}\right\} \qquad \bigstar}{\vdash \{\top\} \left[\begin{array}{l} 1 : x := op(a, b); z := op(x, x) \\ 2 : x := op(a, b); y := op(b, a); z := op(x, y) \end{array}\right] \{z\langle 1\rangle = z\langle 2\rangle\}}
$$

where $\bigstar$ abbreviates $\left\{\begin{array}{l} x\langle 1\rangle = x\langle 2\rangle \\ x\langle 1\rangle = y\langle 2\rangle \end{array}\right\} \left[\begin{array}{l} 1 : z := op(x, x) \\ 2 : z := op(x, y) \end{array}\right] \{z\langle 1\rangle = z\langle 2\rangle\}$

Derivation Sketch for Motivating Example

$$
\textsf{Seq}\;\dfrac{\dfrac{?}{\vdash \{\top\} \left[\begin{array}{l} 1 : x := op(a, b), \\ \\ 2 : x := op(a, b); y := op(b, a) \end{array}\right] \left\{\begin{array}{c} x\langle 1\rangle = x\langle 2\rangle \\ x\langle 1\rangle = y\langle 2\rangle \end{array}\right\} \qquad \bigstar}{\vdash \{\top\} \left[\begin{array}{l} 1 : x := op(a, b); z := op(x, x) \\ 2 : x := op(a, b); y := op(b, a); z := op(x, y) \end{array}\right] \{z\langle 1\rangle = z\langle 2\rangle\}}
$$

where $\bigstar$ abbreviates $\left\{\begin{array}{c} x\langle 1\rangle = x\langle 2\rangle \\ x\langle 1\rangle = y\langle 2\rangle \end{array}\right\} \left[\begin{array}{l} 1 : z := op(x, x) \\ 2 : z := op(x, y) \end{array}\right] \{z\langle 1\rangle = z\langle 2\rangle\}$

Derivation Sketch for Motivating Example

$$
\text{Seq}\ \dfrac{\dfrac{?}{\vdash \{\top\} \begin{bmatrix} 1 : x := op(a, b), \\ \\ 2 : x := op(a, b); y := op(b, a) \end{bmatrix} \begin{Bmatrix} x\langle 1\rangle = x\langle 2\rangle \\ x\langle 1\rangle = y\langle 2\rangle \end{Bmatrix}} \qquad \bigstar}{\vdash \{\top\} \begin{bmatrix} 1 : x := op(a, b); z := op(x, x) \\ 2 : x := op(a, b); y := op(b, a); z := op(x, y) \end{bmatrix} \{z\langle 1\rangle = z\langle 2\rangle\}}
$$

where $\bigstar$ abbreviates $\begin{Bmatrix} x\langle 1\rangle = x\langle 2\rangle \\ x\langle 1\rangle = y\langle 2\rangle \end{Bmatrix} \begin{bmatrix} 1 : z := op(x, x) \\ 2 : z := op(x, y) \end{bmatrix} \{z\langle 1\rangle = z\langle 2\rangle\}$

This Paper: Logic for Hyper-triple Composition (LHC)

Extends RHL rules for *n* related programs.

- Lockstep rules: WP-SEQ, WP-ASSN, WP-IF...
- Structural rules: WP-FRAME, ...

Proposes proof rules for aligning programs in new ways.

- Hyper-structure rules: ...

Proposes proof rules for moving between judgments relating different number of programs.

- Reindexing rules: ...

Preliminaries

Programming language

- A minimal untyped imperative language:

$$\mathbb{E} \ni g, e ::= v \mid x \mid * \mid e + e \mid e - e \mid e \leq e \mid \ldots$$
$$\mathbb{T} \ni t ::= \textbf{skip} \mid x := e \mid t; t \mid \textbf{if } g \textbf{ then } t \textbf{ else } t \mid \textbf{while } g : t$$

- A minimal untyped imperative language:

$$\mathbb{E} \ni g, e ::= v \mid x \mid * \mid e + e \mid e - e \mid e \leq e \mid \ldots$$

$$\mathbb{T} \ni t ::= \textbf{skip} \mid x := e \mid t; t \mid \textbf{if } g \textbf{ then } t \textbf{ else } t \mid \textbf{while } g : t$$

- Big-step semantics: for stores $s, s' \in \mathbb{S}$,

$\langle t, s \rangle \Downarrow s'$ iff the execution from $\langle t, s \rangle$ ends with s'

$\langle t, s \rangle \Downarrow$ iff $\exists s', \langle t, s \rangle \Downarrow s'$

Preliminaries

Hyper-everything

- Hyper-program: a finite partial function $\mathbf{t} : \mathbb{I} \rightharpoonup \mathbb{T}$.
  - $[1 : t_1, 2 : t_2, \ldots, n : t_n]$

- Hyper-program: a finite partial function $\mathbf{t} : \mathbb{I} \rightharpoonup \mathbb{T}$.
  - $[1 : t_1, 2 : t_2, \ldots, n : t_n]$
- Hyper-store: a finite partial function $\mathbf{s} : \mathbb{I} \rightharpoonup \mathbb{S}$.

Preliminaries

Hyper-everything

- Hyper-program: a finite partial function $\mathbf{t} : \mathbb{I} \rightharpoonup \mathbb{T}$.
  - $[1 : t_1, 2 : t_2, \ldots, n : t_n]$
- Hyper-store: a finite partial function $\mathbf{s} : \mathbb{I} \rightharpoonup \mathbb{S}$.
- Big-step semantics for hyper-programs: for hyperstores $\mathbf{s}, \mathbf{s}'$,

$$\langle \mathbf{t}, \mathbf{s} \rangle \Downarrow \mathbf{s}' \text{ iff the execution from } \langle \mathbf{t}, \mathbf{s} \rangle \text{ ends with } \mathbf{s}'$$

$$\langle \mathbf{t}, \mathbf{s} \rangle \Downarrow \text{ iff } \exists \mathbf{s}', \langle \mathbf{t}, \mathbf{s} \rangle \Downarrow \mathbf{s}'$$

Preliminaries

Hyper-everything

- Hyper-program: a finite partial function $\mathbf{t} : \mathbb{I} \rightharpoonup \mathbb{T}$.
  - $[1 : t_1, 2 : t_2, \ldots, n : t_n]$
- Hyper-store: a finite partial function $\mathbf{s} : \mathbb{I} \rightharpoonup \mathbb{S}$.
- Big-step semantics for hyper-programs: for hyperstores $\mathbf{s}, \mathbf{s}'$,

$$\langle \mathbf{t}, \mathbf{s} \rangle \Downarrow \mathbf{s}' \text{ iff the execution from } \langle \mathbf{t}, \mathbf{s} \rangle \text{ ends with } \mathbf{s}'$$

$$\langle \mathbf{t}, \mathbf{s} \rangle \Downarrow \text{ iff } \exists \mathbf{s}', \langle \mathbf{t}, \mathbf{s} \rangle \Downarrow \mathbf{s}'$$

- Hyper-assertions map hyper-stores to Booleans.

Preliminaries

Weakest Precondition

- Weakest pre-condition $\mathbf{wp}\ [\mathbf{t}]\ \{Q\}$:
  - $\vdash \{P\}\ [\mathbf{t}]\ \{Q\}$ iff $P \models \mathbf{wp}\ [\mathbf{t}]\ \{Q\}$.

- Weakest pre-condition $\textbf{wp}\ [\textbf{t}]\ \{Q\}$:
    - $\vdash \{P\}\ [\textbf{t}]\ \{Q\}$ iff $P \models \textbf{wp}\ [\textbf{t}]\ \{Q\}$.
    - Semantics definition:

$$\textbf{wp}\ [\textbf{t}]\ \{Q\} := \lambda \textbf{s}.(\forall \textbf{s}'.\langle \textbf{t}, \textbf{s}\rangle \Downarrow \textbf{s}' \implies Q(\textbf{s}'))$$

    - Enables assertions to mention programs.

Logic for Hyper-triple Composition (LHC)

Extends RHL rules for *n* related programs.

- Lockstep rules: wp-Seq, wp-Assn, wp-If...
- Structural rules: wp-Frame, ...

Proposes proof rules for aligning programs in new ways.

- Hyper-structure rules: ...

Proposes proof rules for moving between judgments relating different number of programs.

- Reindexing rules: ...

Logic for Hyper-triple Composition (LHC)

Extends RHL rules for n related programs.

- Lockstep rules: WP-SEQ, WP-ASSN, WP-IF...
- Structural rules: WP-FRAME, ...

Proposes proof rules for aligning programs in new ways.

- Hyper-structure rules: ...

Proposes proof rules for moving between judgments relating different number of programs.

- Reindexing rules: ...

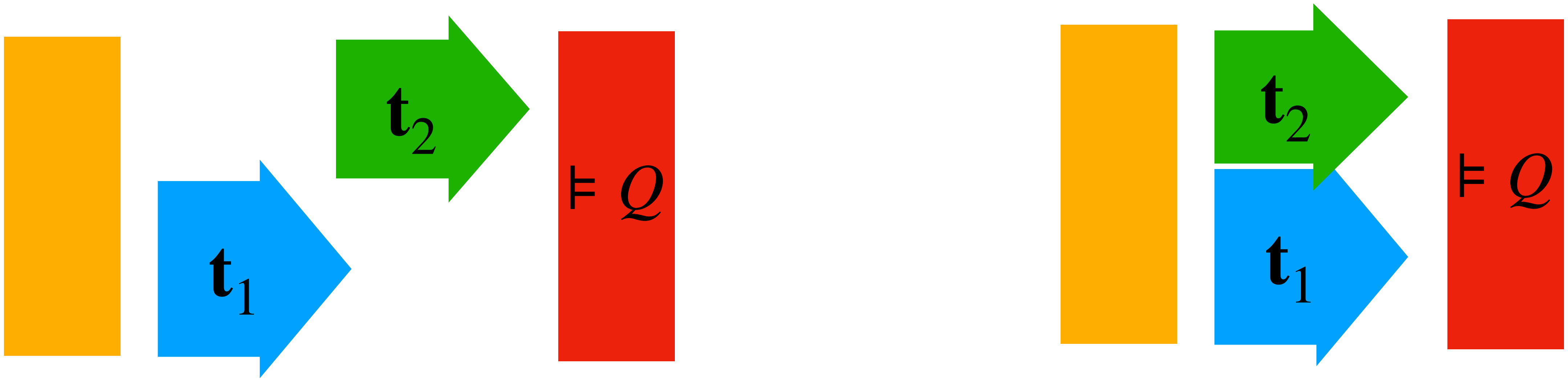
Definition (Union of hyper-programs)

Given hyper-programs $f, g : \mathbb{I} \rightharpoonup \mathbb{T}$ such that for any $i \in \mathrm{supp}(f) \cap \mathrm{supp}(g)$, $f(i) = g(i)$. Then the union of f and g, written $f + g : \mathbb{I} \rightharpoonup \mathbb{T}$ is defined as

$$(f+g)(i) = \begin{cases} f(i) & \text{if } i \in \mathrm{supp}(f) \setminus \mathrm{supp}(g) \\ g(i) & \text{if } i \in \mathrm{supp}(g) \\ \perp & \text{otherwise} \end{cases}$$

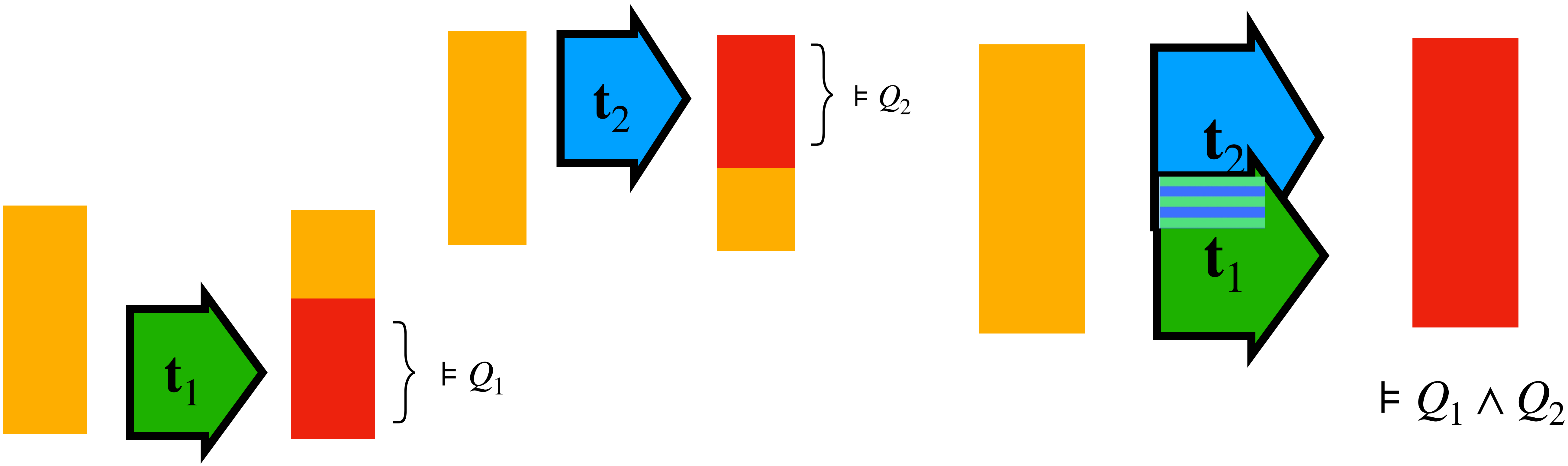
Hyper-structural Rules

Novel Structural Rules for Hyper-programs



Hyper-structural Rules

Novel Structural Rules for Hyper-programs



Application on Motivating Example

Apply WP-NEST:

$$\frac{\vdash \textbf{wp}\ [1 : x := op(a,b)]\ \left\{\textbf{wp}\ \begin{bmatrix} 2 : x := op(a,b); \\ y := op(b,a) \end{bmatrix} \begin{Bmatrix} x\langle 1\rangle = x\langle 2\rangle \\ y\langle 2\rangle = x\langle 1\rangle \end{Bmatrix}\right\}}{\vdash \textbf{wp}\ \begin{bmatrix} 1 : x := op(a,b) \\ 2 : x := op(a,b); y := op(b,a) \end{bmatrix} \begin{Bmatrix} x\langle 1\rangle = x\langle 2\rangle \\ y\langle 2\rangle = x\langle 1\rangle \end{Bmatrix}}$$

Application on Motivating Example

Apply WP-SEQ:

$$\frac{\vdash \textbf{wp}\ [1 : x := op(a,b)]\ \left\{\textbf{wp}\ [2 : x := op(a,b)]\ \left\{\textbf{wp}\ [2 : y := op(b,a)]\ \left\{\begin{array}{c} x\langle 1\rangle = x\langle 2\rangle \\ y\langle 2\rangle = x\langle 1\rangle \end{array}\right\}\right\}\right\}}{\vdash \textbf{wp}\ [1 : x := op(a,b)]\ \left\{\textbf{wp}\ \left[\begin{array}{r} 2 : x := op(a,b); \\ y := op(b,a) \end{array}\right]\ \left\{\begin{array}{c} x\langle 1\rangle = x\langle 2\rangle \\ y\langle 2\rangle = x\langle 1\rangle \end{array}\right\}\right\}}$$

Application on Motivating Example

Apply wp-Nest again:

$$\frac{\vdash \mathbf{wp}\ \begin{bmatrix} 1 : x := op(a,b) \\ 2 : x := op(a,b) \end{bmatrix} \left\{ \mathbf{wp}\ \left[\ 2 : y := op(b,a)\ \right] \left\{ \begin{array}{c} x\langle 1\rangle = x\langle 2\rangle \\ y\langle 2\rangle = x\langle 1\rangle \end{array} \right\} \right\}}{\vdash \mathbf{wp}\ [1 : x := op(a,b)] \left\{ \mathbf{wp}\ [2 : x := op(a,b)] \left\{ \mathbf{wp}\ [2 : y := op(b,a)] \left\{ \begin{array}{c} x\langle 1\rangle = x\langle 2\rangle \\ y\langle 2\rangle = x\langle 1\rangle \end{array} \right\} \right\} \right\}}$$

Application on Motivating Example

Apply $\textsc{wp-Frame}$:

$$\frac{\vdash \textbf{wp}\begin{bmatrix} 1 : x := op(a, b) \\ 2 : x := op(a, b) \end{bmatrix}\Big\{x\langle 1\rangle = x\langle 2\rangle \wedge \textbf{wp}\left[\ 2 : y := op(b, a)\ \right]\left\{\ y\langle 2\rangle = x\langle 1\rangle\ \right\}\Big\}}{\vdash \textbf{wp}\begin{bmatrix} 1 : x := op(a, b) \\ 2 : x := op(a, b) \end{bmatrix}\left\{\textbf{wp}\left[\ 2 : y := op(b, a)\ \right]\begin{Bmatrix} x\langle 1\rangle = x\langle 2\rangle \\ y\langle 2\rangle = x\langle 1\rangle \end{Bmatrix}\right\}}$$

Application on Motivating Example

Apply wp-Conj:

$$\frac{\vdash \mathbf{wp}\begin{bmatrix} 1 : x := op(a,b) \\ 2 : x := op(a,b) \end{bmatrix} \left\{ \mathbf{wp} \left[\ 2 : y := op(b,a)\ \right] \left\{\ y\langle 2\rangle = x\langle 1\rangle\ \right\} \right\} \qquad \bigstar}{\vdash \mathbf{wp}\begin{bmatrix} 1 : x := op(a,b) \\ 2 : x := op(a,b) \end{bmatrix} \left\{ x\langle 1\rangle = x\langle 2\rangle \wedge \mathbf{wp} \left[\ 2 : y := op(b,a)\ \right] \left\{\ y\langle 2\rangle = x\langle 1\rangle\ \right\} \right\}}$$

where $\bigstar$ is

$$\vdash \mathbf{wp}\begin{bmatrix} 1 : x := op(a,b) \\ 2 : x := op(a,b) \end{bmatrix} \{x\langle 1\rangle = x\langle 2\rangle\}$$

Application on Motivating Example

$$
\dfrac{\dfrac{\vdash \mathbf{wp} \begin{bmatrix} 1 : x := op(a,b) \\ 2 : y := op(b,a) \end{bmatrix} \left\{\ y\langle 2\rangle = x\langle 1\rangle\ \right\}}{\vdash \mathbf{wp} \left[\ 2 : x := op(a,b)\ \right] \left\{ \mathbf{wp} \begin{bmatrix} 1 : x := op(a,b) \\ 2 : y := op(b,a) \end{bmatrix} \left\{\ y\langle 2\rangle = x\langle 1\rangle\ \right\} \right\}}\ \textsc{WP-Cons}}{\vdash \mathbf{wp} \begin{bmatrix} 1 : x := op(a,b) \\ 2 : x := op(a,b) \end{bmatrix} \left\{ \mathbf{wp} \left[\ 2 : y := op(b,a)\ \right] \left\{\ y\langle 2\rangle = x\langle 1\rangle\ \right\} \right\}}\ \textsc{WP-Nest}
$$

Application on Motivating Example

Putting everything together,

$$
\begin{array}{c}
\vdash \mathbf{wp}\begin{bmatrix} 1\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \\ 2\colon \mathsf{y} \coloneqq \mathsf{op}(b,a) \end{bmatrix} \{\mathsf{y}(2) = \mathsf{x}(1)\} \\
\hline
\vdash \mathbf{wp}\,[2\colon \mathsf{x} \coloneqq \mathsf{op}(a,b)] \left\{\mathbf{wp}\begin{bmatrix} 1\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \\ 2\colon \mathsf{y} \coloneqq \mathsf{op}(b,a) \end{bmatrix} \{\mathsf{y}(2) = \mathsf{x}(1)\}\right\} \quad \textsc{wp-cons} \\
\hline
\vdash \mathbf{wp}\begin{bmatrix} 1\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \\ 2\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \end{bmatrix} \{\mathsf{x}(1) = \mathsf{x}(2)\} \qquad \vdash \mathbf{wp}\begin{bmatrix} 1\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \\ 2\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \end{bmatrix} \{\mathbf{wp}\,[2\colon \mathsf{y} \coloneqq \mathsf{op}(b,a)]\,\{\mathsf{y}(2) = \mathsf{x}(1)\}\} \quad \textsc{wp-nest}_0 \\
\hline
\vdash \mathbf{wp}\begin{bmatrix} 1\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \\ 2\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \end{bmatrix} \left\{\begin{matrix} \mathsf{x}(1) = \mathsf{x}(2) \land \\ \mathbf{wp}\,[2\colon \mathsf{y} \coloneqq \mathsf{op}(b,a)]\,\{\mathsf{y}(2) = \mathsf{x}(1)\} \end{matrix}\right\} \quad \textsc{wp-conj}_0 \\
\hline
\vdash \mathbf{wp}\begin{bmatrix} 1\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \\ 2\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \end{bmatrix} \left\{\mathbf{wp}\,[2\colon \mathsf{y} \coloneqq \mathsf{op}(b,a)] \begin{Bmatrix} \mathsf{x}(1) = \mathsf{x}(2) \\ \mathsf{y}(2) = \mathsf{x}(1) \end{Bmatrix}\right\} \quad \textsc{wp-frame} \\
\hline
\vdash \mathbf{wp}\,[1\colon \mathsf{x} \coloneqq \mathsf{op}(a,b)]\,\{\mathbf{wp}\,[2\colon \mathsf{x} \coloneqq \mathsf{op}(a,b)]\,\{\mathbf{wp}\,[2\colon \mathsf{y} \coloneqq \mathsf{op}(b,a)]\,\{\mathsf{x}(1) = \mathsf{x}(2) \land \mathsf{y}(2) = \mathsf{x}(1)\}\}\} \quad \textsc{wp-nest}_0 \\
\hline
\vdash \mathbf{wp}\,[1\colon \mathsf{x} \coloneqq \mathsf{op}(a,b)]\,\{\mathbf{wp}\,[2\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) ; \mathsf{y} \coloneqq \mathsf{op}(b,a)]\,\{\mathsf{x}(1) = \mathsf{x}(2) \land \mathsf{y}(2) = \mathsf{x}(1)\}\} \quad \textsc{wp-seq}_1 \\
\hline
\vdash \mathbf{wp}\begin{bmatrix} 1\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) \\ 2\colon \mathsf{x} \coloneqq \mathsf{op}(a,b) ; \mathsf{y} \coloneqq \mathsf{op}(b,a) \end{bmatrix} \begin{Bmatrix} \mathsf{x}(1) = \mathsf{x}(2) \\ \mathsf{y}(2) = \mathsf{x}(1) \end{Bmatrix} \quad \textsc{wp-nest}_0
\end{array}
$$

Logic for Hyper-triple Composition (LHC)

Extends RHL rules for n related programs.

- Lockstep rules: WP-SEQ, WP-ASSN, WP-IF...
- Structural rules: WP-FRAME, ...

Proposes proof rules for aligning programs in new ways.

- Hyper-structure rules: ...

Proposes proof rules for moving between judgments relating different number of programs.

- Reindexing rules: ...

Reindexing Rules

Reindexing rules tell us

- when it is possible to offload some reasoning to another index.
- when reindexing of pre-conditions can be propagated to post-conditions.
- when reindexing of hyper-programs can be propagated to post-conditions.

Questions

Extends RHL rules for *n* related programs.

Lockstep rules: WP-SEQ, WP-ASSN, WP-IF...

Structural rules: WP-FRAME, ...

Proposes proof rules for aligning programs in new ways.

Hyper-structure rules: ...

Proposes proof rules for moving between judgments relating different number of programs.

Reindexing rules: ...

Motivating Reindexing Rules

Two encodings of idempotence:

$$\vdash \{\vec{x}\langle 1\rangle = \vec{x}\langle 2\rangle\}\ [1 : t, 2 : (t; t)]\ \{\vec{x}\langle 1\rangle = \vec{x}\langle 2\rangle\} \qquad (\textsc{IdemSeq})$$

$$\vdash \{\vec{x}\langle 1\rangle = \vec{v}\}\ [1 : t, 2 : t]\ \{\vec{x}\langle 1\rangle = \vec{v} \implies \vec{x}\langle 2\rangle = \vec{v}\} \qquad (\textsc{Idem})$$

Q: Are they equally strong?

A: Idem together with Det_{op} implies IdemSeq.

Q: How do we prove that?

Motivating Reindexing Rules

A Proof Sketch

$$\dfrac{\text{IDEM}}{\vec{x}\langle 3\rangle = \vec{v} \vdash \textbf{wp}\ [2 : t, 3 : t]\ \{\vec{x}\langle 2\rangle = \vec{v} \implies \vec{x}\langle 3\rangle = \vec{v}\}}$$

$$\vdots$$

$$\vdash \textbf{wp}\ [2 : t]\ \{\exists \vec{v}. \vec{x}\langle 2\rangle = \vec{v} \wedge \vec{x}\langle 3\rangle = \vec{v} \wedge \textbf{wp}\ [3 : \mathbf{t}]\ \{\vec{x}(3) = \vec{v}\}\}$$

$\vdots$ We fork the store at 2 to 3 and offload the reasoning to 3.

$$\dfrac{\bigstar \qquad \vdash \textbf{wp}\ [2 : t]\ \{\exists \vec{v}. \vec{x}\langle 2\rangle = \vec{v} \wedge \textbf{wp}\ [2 : \mathbf{t}]\ \{\vec{x}(2) = \vec{v}\}\}}{\vec{x}\langle 1\rangle = \vec{x}\langle 2\rangle \vdash \textbf{wp}\ [1 : t, 2 : (t; t)]\ \{\vec{x}\langle 1\rangle = \vec{x}\langle 2\rangle\}}$$

where $\bigstar$ is $\vec{x}\langle 1\rangle = \vec{x}\langle 2\rangle \vdash \textbf{wp}\ [1 : t, 2 : t]\ \{\vec{x}\langle 1\rangle = \vec{x}\langle 2\rangle\}$.