Iris: A Modular Foundation for Higher-Order Concurrent Separation Logic[1]

Jacques-Henri Jourdan[2]   Robbert Krebbers[3]

[2]CNRS, LRI, Université Paris-Sud, France

[3]Delft University of Technology, The Netherlands

January 8, 2018 @ POPL Tutorials, Los Angeles

[1]Iris is joint work with: Ralf Jung, Aleš Bizjak, Hoang-Hai Dang, Jan-Oliver Kaiser, David Swasey, Filip Sieczkowski, Kasper Svendsen, Aaron Turon, Amin Timany, Derek Dreyer and Lars Birkedal

This tutorial

- First part (45 min): “theoretical” overview of Iris, by me
- Second part (45 min): Coq demo by Robbert Krebbers
- Third part (after break): hands-on session

Get ready:

- Download the tutorial lecture material
  `http://iris-project.org/tutorial`
- Follow README to install Iris **3.1**

What is Iris?

Language-independent higher-order separation logic with simple foundations for verifying fine-grained concurrent programs in Coq.

What is Iris?

Language-independent higher-order separation logic with simple foundations for verifying fine-grained concurrent programs in Coq.

- **Higher-order separation logic:** Supports modular reasoning about higher-order stateful programs

What is Iris?

Language-independent higher-order separation logic with simple foundations for verifying fine-grained concurrent programs in Coq.

- **Higher-order separation logic:** Supports modular reasoning about higher-order stateful programs
- **Fine-grained concurrent programs:** Programs that use low-level synchronization primitives for more parallelism

What is Iris?

Language-independent higher-order separation logic with simple foundations for verifying fine-grained concurrent programs in Coq.

- **Higher-order separation logic:** Supports modular reasoning about higher-order stateful programs
- **Fine-grained concurrent programs:** Programs that use low-level synchronization primitives for more parallelism
- **Language-independent:** Parameterized by the language

What is Iris?

Language-independent higher-order separation logic with simple foundations for verifying fine-grained concurrent programs in Coq.

- **Higher-order separation logic:** Supports modular reasoning about higher-order stateful programs
- **Fine-grained concurrent programs:** Programs that use low-level synchronization primitives for more parallelism
- **Language-independent:** Parameterized by the language
- **Simple foundations:** Small, "canonical" set of primitive rules

What is Iris?

Language-independent higher-order separation logic with simple foundations for verifying fine-grained concurrent programs in Coq.

- **Higher-order separation logic:** Supports modular reasoning about higher-order stateful programs
- **Fine-grained concurrent programs:** Programs that use low-level synchronization primitives for more parallelism
- **Language-independent:** Parameterized by the language
- **Simple foundations:** Small, "canonical" set of primitive rules
- **Coq:** Provides practical support for machine-checked proof

The versatility of Iris

The scope of Iris goes beyond proving traditional program correctness using Hoare triples:

- The Rust type system (Jung, Jourdan, Krebbers, Dreyer)
- Logical relations (Krogh-Jespersen, Svendsen, Timany, Birkedal, Krebbers)
- Termination-preserving refinement (Tassarotti, Jung, Harper)
- Weak memory concurrency (Kaiser, Dang, Dreyer, Lahav, Vafeiadis)
- Object capability patterns (Swasey, Garg, Dreyer)
- Logical atomicity (Jung, Swasey, Krogh-Jespersen, Zhang, Dreyer, Birkedal)
- Defining Iris (Krebbers, Jung, Jourdan, Bizjak, Dreyer, Birkedal)

Most of these projects are formalized in Iris in Coq

The Iris story, Part 1:

Working with ghost state and invariants

Hoare triples

Hoare triples for partial program correctness:

$$\{\underbrace{P}_{\text{Precondition}}\}\, e\, \{\underbrace{w}_{\text{Binder for return value}}.\, \underbrace{Q}_{\text{Postcondition}}\}$$

If the initial state satisfies P, then:

- e does not get stuck/crash
- if e terminates with value v, the final state satisfies $Q[v/w]$

Separation logic [O'Hearn, Reynolds, Yang]

The points-to connective $x \mapsto v$

- provides the knowledge that location x has value v, and
- provides exclusive ownership of x

Separating conjunction $P * Q$: the state consists of *disjoint parts* satisfying P and Q

Separation logic [O'Hearn, Reynolds, Yang]

The points-to connective $x \mapsto v$

- provides the knowledge that location x has value v, and
- provides exclusive ownership of x

Separating conjunction $P * Q$: the state consists of *disjoint parts* satisfying P and Q

Example:

$$\{x \mapsto v_1 * y \mapsto v_2\} \mathit{swap}(x, y) \{w.\, w = () \wedge x \mapsto v_2 * y \mapsto v_1\}$$

the $*$ ensures that x and y are different

Concurrent separation logic [O'Hearn, Brookes]

The *par* rule:

$$\frac{\{P_1\}\, e_1\, \{Q_1\} \qquad \{P_2\}\, e_2\, \{Q_2\}}{\{P_1 * P_2\}\, e_1 || e_2\, \{Q_1 * Q_2\}}$$

Concurrent separation logic [O'Hearn, Brookes]

The *par* rule:

$$\frac{\{P_1\}\, e_1 \,\{Q_1\} \qquad \{P_2\}\, e_2 \,\{Q_2\}}{\{P_1 * P_2\}\, e_1 || e_2 \,\{Q_1 * Q_2\}}$$

For example:

$$\{x \mapsto 4 * y \mapsto 6\}$$

$$x := {!}\,x + 2 \;\Big\|\; y := {!}\,y + 2$$

$$\{x \mapsto 6 * y \mapsto 8\}$$

Concurrent separation logic [O'Hearn, Brookes]

The *par* rule:

$$\frac{\{P_1\}\, e_1 \,\{Q_1\} \qquad \{P_2\}\, e_2 \,\{Q_2\}}{\{P_1 * P_2\}\, e_1 || e_2 \,\{Q_1 * Q_2\}}$$

For example:

$$\begin{array}{c}
\{x \mapsto 4 * y \mapsto 6\} \\
\begin{array}{l||l}
\{x \mapsto 4\} & \{y \mapsto 6\} \\
x := !\,x + 2 & y := !\,y + 2 \\
 &
\end{array} \\
\{x \mapsto 6 * y \mapsto 8\}
\end{array}$$

Concurrent separation logic [O'Hearn, Brookes]

The *par* rule:

$$\frac{\{P_1\}\, e_1\, \{Q_1\} \qquad \{P_2\}\, e_2\, \{Q_2\}}{\{P_1 * P_2\}\, e_1 || e_2\, \{Q_1 * Q_2\}}$$

For example:

$$\begin{array}{c}
\{x \mapsto 4 * y \mapsto 6\} \\
\begin{array}{l||l}
\{x \mapsto 4\} & \{y \mapsto 6\} \\
x := !\, x + 2 & y := !\, y + 2 \\
\{x \mapsto 6\} & \{y \mapsto 8\}
\end{array} \\
\{x \mapsto 6 * y \mapsto 8\}
\end{array}$$

Concurrent separation logic [O'Hearn, Brookes]

The *par* rule:

$$\frac{\{P_1\}\, e_1 \,\{Q_1\} \qquad \{P_2\}\, e_2 \,\{Q_2\}}{\{P_1 * P_2\}\, e_1 || e_2 \,\{Q_1 * Q_2\}}$$

For example:

$$\begin{array}{c}
\{x \mapsto 4 * y \mapsto 6\} \\
\begin{array}{l|l}
\{x \mapsto 4\} & \{y \mapsto 6\} \\
x := !\, x + 2 & y := !\, y + 2 \\
\{x \mapsto 6\} & \{y \mapsto 8\}
\end{array} \\
\{x \mapsto 6 * y \mapsto 8\}
\end{array}$$

Works great for concurrent programs without shared memory: concurrent quick sort, concurrent merge sort, ...

What about shared state/racy programs?

A classic problem:

$$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$$

$$\begin{array}{l||l} \texttt{fetchandadd}(x, 2) & \texttt{fetchandadd}(x, 2) \end{array}$$

$$!\, x$$

Where $\texttt{fetchandadd}(x, y)$ is the atomic version of $x := !\, x + y$.

What about shared state/racy programs?

A classic problem:

$$
\begin{array}{l}
\{\mathsf{True}\} \\
\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in} \\
\left(\texttt{fetchandadd}(x, 2) \;\middle\|\; \texttt{fetchandadd}(x, 2)\right) \\
!\, x \\
\{w.\, w = 4\}
\end{array}
$$

Where $\texttt{fetchandadd}(x, y)$ is the atomic version of $x := !\, x + y$.

What about shared state/racy programs?

A classic problem:

$$
\begin{array}{l}
\{\mathsf{True}\} \\
\mathtt{let}\, x = \mathtt{ref}(0)\,\mathtt{in} \\
\{x \mapsto 0\} \\
\mathtt{fetchandadd}(x, 2) \;\Big\|\; \mathtt{fetchandadd}(x, 2) \\
!\, x \\
\{w.\, w = 4\}
\end{array}
$$

Where $\mathtt{fetchandadd}(x, y)$ is the atomic version of $x := !\, x + y$.

What about shared state/racy programs?

A classic problem:

$$
\begin{array}{l}
\{\mathsf{True}\} \\
\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in} \\
\{x \mapsto 0\} \\
\left(
\begin{array}{l}
\{??\} \\
\texttt{fetchandadd}(x, 2) \\
\{??\}
\end{array}
\;\middle\|\;
\begin{array}{l}
\{??\} \\
\texttt{fetchandadd}(x, 2) \\
\{??\}
\end{array}
\right) \\
!\, x \\
\{w.\, w = 4\}
\end{array}
$$

Where $\texttt{fetchandadd}(x, y)$ is the atomic version of $x := !\, x + y$.

Problem: can only give ownership of x to one thread

Invariants

The invariant assertion $\boxed{R}$ expresses that R is maintained as an invariant on the state

Invariants

The invariant assertion $\boxed{R}$ expresses that R is maintained as an invariant on the state

Invariant opening:

$$\frac{\{R * P\}\, e\, \{R * Q\} \qquad e \text{ atomic}}{\boxed{R} \;\vdash \{P\}\, e\, \{Q\}}$$

Invariants

The invariant assertion $\boxed{R}$ expresses that R is maintained as an invariant on the state

Invariant opening:

$$\frac{\{R * P\}\, e\, \{R * Q\} \qquad e \text{ atomic}}{\boxed{R} \vdash \{P\}\, e\, \{Q\}}$$

Invariant allocation:

$$\frac{\boxed{R} \vdash \{P\}\, e\, \{Q\}}{\{R * P\}\, e\, \{Q\}}$$

Invariants

The invariant assertion $\boxed{R}^{\mathcal{N}}$ expresses that R is maintained as an invariant on the state

Invariant opening:

$$\frac{\{R * P\}\, e\, \{R * Q\}_{\mathcal{E}} \qquad e \text{ atomic}}{\boxed{R}^{\mathcal{N}} \vdash \{P\}\, e\, \{Q\}_{\mathcal{E} \uplus \mathcal{N}}}$$

Invariant allocation:

$$\frac{\boxed{R}^{\mathcal{N}} \vdash \{P\}\, e\, \{Q\}_{\mathcal{E}}}{\{R * P\}\, e\, \{Q\}_{\mathcal{E}}}$$

Technical detail: names are needed to avoid *reentrancy*, i.e., opening the same invariant twice

Invariants

The invariant assertion $\boxed{R}^{\mathcal{N}}$ expresses that R is maintained as an invariant on the state

Invariant opening:

$$\frac{\{\triangleright R * P\}\, e\, \{\triangleright R * Q\}_{\mathcal{E}} \qquad e \text{ atomic}}{\boxed{R}^{\mathcal{N}} \vdash \{P\}\, e\, \{Q\}_{\mathcal{E} \uplus \mathcal{N}}}$$

Invariant allocation:

$$\frac{\boxed{R}^{\mathcal{N}} \vdash \{P\}\, e\, \{Q\}_{\mathcal{E}}}{\{\triangleright R * P\}\, e\, \{Q\}}$$

Technical detail: names are needed to avoid *reentrancy*, i.e., opening the same invariant twice

Other technical detail: the later $\triangleright$ is needed to support impredicative invariants, i.e., $\boxed{\ldots \boxed{R}^{\mathcal{N}_2} \ldots}^{\mathcal{N}_1}$

Invariants in action

Let us consider a simpler problem first:

$\{\mathsf{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$

$$\texttt{fetchandadd}(x, 2) \quad \Big\| \quad \texttt{fetchandadd}(x, 2)$$

$!\, x$

$\{n.\, \mathit{even}(n)\}$

Invariants in action

Let us consider a simpler problem first:

$\{\text{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$

$$\texttt{fetchandadd}(x, 2) \quad \Big\| \quad \texttt{fetchandadd}(x, 2)$$

$!\, x$

$\{n.\, even(n)\}$

Invariants in action

Let us consider a simpler problem first:

$\{\mathsf{True}\}$
$\texttt{let}\; x = \texttt{ref}(0)\; \texttt{in}$
$\{x \mapsto 0\}$
allocate $\boxed{\exists n.\, x \mapsto n \land even(n)}$

$\texttt{fetchandadd}(x, 2)$	$\texttt{fetchandadd}(x, 2)$

$!\, x$

$\{n.\, even(n)\}$

Invariants in action

Let us consider a simpler problem first:

$\{\mathsf{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$
allocate $\boxed{\exists n.\, x \mapsto n \wedge \mathit{even}(n)}$

$$
\begin{array}{l|l}
\{\mathsf{True}\} & \{\mathsf{True}\} \\
\quad \texttt{fetchandadd}(x, 2) & \quad \texttt{fetchandadd}(x, 2) \\
\{\mathsf{True}\} & \{\mathsf{True}\}
\end{array}
$$

$\quad !\, x$

$\{n.\, \mathit{even}(n)\}$

Invariants in action

Let us consider a simpler problem first:

$$
\begin{array}{l}
\{\mathsf{True}\} \\
\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in} \\
\{x \mapsto 0\} \\
\text{allocate } \boxed{\exists n.\, x \mapsto n \wedge \mathit{even}(n)} \\
\begin{array}{l|l}
\{\mathsf{True}\} & \{\mathsf{True}\} \\
\quad \{x \mapsto n \wedge \mathit{even}(n)\} & \\
\quad \texttt{fetchandadd}(x, 2) & \quad \texttt{fetchandadd}(x, 2) \\
\quad \{x \mapsto n + 2 \wedge \mathit{even}(n + 2)\} & \\
\{\mathsf{True}\} & \{\mathsf{True}\}
\end{array} \\
\\
\quad !\, x \\
\\
\{n.\, \mathit{even}(n)\}
\end{array}
$$

Invariants in action

Let us consider a simpler problem first:

$\{\mathsf{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\ \texttt{in}$
$\{x \mapsto 0\}$
allocate $\boxed{\exists n.\, x \mapsto n \wedge \mathit{even}(n)}$

$$
\begin{array}{l|l}
\{\mathsf{True}\} & \{\mathsf{True}\} \\
\quad \{x \mapsto n \wedge \mathit{even}(n)\} & \quad \{x \mapsto n \wedge \mathit{even}(n)\} \\
\quad \texttt{fetchandadd}(x, 2) & \quad \texttt{fetchandadd}(x, 2) \\
\quad \{x \mapsto n + 2 \wedge \mathit{even}(n + 2)\} & \quad \{x \mapsto n + 2 \wedge \mathit{even}(n + 2)\} \\
\{\mathsf{True}\} & \{\mathsf{True}\}
\end{array}
$$

$!\,x$

$\{n.\, \mathit{even}(n)\}$

Invariants in action

Let us consider a simpler problem first:

$$
\begin{array}{l}
\{\mathsf{True}\} \\
\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in} \\
\{x \mapsto 0\} \\
\text{allocate } \boxed{\exists n.\, x \mapsto n \wedge \mathit{even}(n)}
\end{array}
$$

$$
\begin{array}{l|l}
\{\mathsf{True}\} & \{\mathsf{True}\} \\
\quad \{x \mapsto n \wedge \mathit{even}(n)\} & \quad \{x \mapsto n \wedge \mathit{even}(n)\} \\
\quad \texttt{fetchandadd}(x, 2) & \quad \texttt{fetchandadd}(x, 2) \\
\quad \{x \mapsto n + 2 \wedge \mathit{even}(n + 2)\} & \quad \{x \mapsto n + 2 \wedge \mathit{even}(n + 2)\} \\
\{\mathsf{True}\} & \{\mathsf{True}\} \\
\quad \{x \mapsto n \wedge \mathit{even}(n)\} & \\
\quad !\, x & \\
\quad \{n.\, x \mapsto n \wedge \mathit{even}(n)\} & \\
\{n.\, \mathit{even}(n)\} &
\end{array}
$$

Invariants in action

Let us consider a simpler problem first:

$$
\begin{array}{l}
\{\mathsf{True}\} \\
\texttt{let}\, x = \texttt{ref}(0)\ \texttt{in} \\
\{x \mapsto 0\} \\
\text{allocate } \boxed{\exists n.\, x \mapsto n \wedge \mathit{even}(n)} \\
\left(\begin{array}{l}
\{\mathsf{True}\} \\
\left|\begin{array}{l}
\{x \mapsto n \wedge \mathit{even}(n)\} \\
\texttt{fetchandadd}(x, 2) \\
\{x \mapsto n + 2 \wedge \mathit{even}(n + 2)\}
\end{array}\right. \\
\{\mathsf{True}\} \\
\left|\begin{array}{l}
\{x \mapsto n \wedge \mathit{even}(n)\} \\
!\, x \\
\{n.\, x \mapsto n \wedge \mathit{even}(n)\}
\end{array}\right. \\
\{n.\, \mathit{even}(n)\}
\end{array}\right.
\ \Bigg\|\
\begin{array}{l}
\{\mathsf{True}\} \\
\left|\begin{array}{l}
\{x \mapsto n \wedge \mathit{even}(n)\} \\
\texttt{fetchandadd}(x, 2) \\
\{x \mapsto n + 2 \wedge \mathit{even}(n + 2)\}
\end{array}\right. \\
\{\mathsf{True}\}
\end{array}
\end{array}
$$

Problem: still cannot prove it returns 4

Ghost variables

Consider the invariant:

$$\boxed{\exists n.\, x \mapsto n * \ldots}$$

How to relate the quantified value to the state of the threads?

Ghost variables

Consider the invariant:

$$\boxed{\exists n.\, x \mapsto n * \ldots}$$

How to relate the quantified value to the state of the threads?

Solution: ghost variables

Ghost variables

Consider the invariant:

$$\boxed{\exists n.\, x \mapsto n * \ldots}$$

How to relate the quantified value to the state of the threads?

Solution: ghost variables

Ghost variables are allocated in pairs:

$$\mathrm{True} \quad \Rrightarrow\!\!\ast \quad \exists\gamma.\ \underbrace{\gamma \hookrightarrow_{\bullet} n}_{\text{in the invariant}} \quad * \quad \underbrace{\gamma \hookrightarrow_{\circ} n}_{\text{in the Hoare triple}}$$

Ghost variables

Consider the invariant:

$$\boxed{\exists n_1, n_2.\, x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$$

How to relate the quantified value to the state of the threads?

Solution: ghost variables

Ghost variables are allocated in pairs:

$$\text{True} \quad \Rrightarrow\!\!\ast \quad \exists \gamma.\; \underbrace{\gamma \hookrightarrow_\bullet n}_{\text{in the invariant}} \quad * \quad \underbrace{\gamma \hookrightarrow_\circ n}_{\text{in the Hoare triple}}$$

Ghost variables

Consider the invariant:

$$\boxed{\exists n_1, n_2.\, x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$$

How to relate the quantified value to the state of the threads?

Solution: ghost variables

Ghost variables are allocated in pairs:

$$\text{True} \quad \Rrightarrow\!\!\!\ast \quad \exists \gamma.\ \underbrace{\gamma \hookrightarrow_\bullet n}_{\text{in the invariant}} \quad * \quad \underbrace{\gamma \hookrightarrow_\circ n}_{\text{in the Hoare triple}}$$

When you own both parts you obtain that the values are equal and can update both parts:

$$\gamma \hookrightarrow_\bullet n * \gamma \hookrightarrow_\circ m \quad \Rightarrow \quad n = m$$

$$\gamma \hookrightarrow_\bullet n * \gamma \hookrightarrow_\circ m \quad \Rrightarrow\!\!\!\ast \quad \gamma \hookrightarrow_\bullet n' * \gamma \hookrightarrow_\circ n'$$

Ghost variables in action

$\{\mathsf{True}\}$
$\mathtt{let}\, x = \mathtt{ref}(0)\, \mathtt{in}$

$\mathtt{fetchandadd}(x, 2) \quad \Big\| \quad \mathtt{fetchandadd}(x, 2)$

$!\,x$

$\{n.\ n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$

$\texttt{fetchandadd}(x, 2)$	$\texttt{fetchandadd}(x, 2)$

$!\, x$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\text{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$
$\{x \mapsto 0 * \gamma_1 \hookrightarrow_{\bullet} 0 * \gamma_1 \hookrightarrow_{\circ} 0 * \gamma_2 \hookrightarrow_{\bullet} 0 * \gamma_2 \hookrightarrow_{\circ} 0\}$

$\texttt{fetchandadd}(x, 2) \quad \Big\| \quad \texttt{fetchandadd}(x, 2)$

$!\, x$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$
$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$
allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$

$\texttt{fetchandadd}(x, 2)$	$\Vert$	$\texttt{fetchandadd}(x, 2)$

$!\, x$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$
$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$
allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$
$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\texttt{fetchandadd}(x, 2) \quad \| \quad \texttt{fetchandadd}(x, 2)$

$!\, x$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\ x = \texttt{ref}(0)\ \texttt{in}$
$\{x \mapsto 0\}$
$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$
allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$
$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$	$\{\gamma_2 \hookrightarrow_\circ 0\}$
$\texttt{fetchandadd}(x, 2)$	$\texttt{fetchandadd}(x, 2)$

$!\,x$

$\{n.\ n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$

$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$

$\{x \mapsto 0\}$

$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$

allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$

$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$	$\Vert$	$\{\gamma_2 \hookrightarrow_\circ 0\}$
$\texttt{fetchandadd}(x, 2)$	$\Vert$	$\texttt{fetchandadd}(x, 2)$
$\{\gamma_1 \hookrightarrow_\circ 2\}$	$\Vert$	$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$!\, x$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$
$\mathtt{let}\ x = \mathtt{ref}(0)\ \mathtt{in}$
$\{x \mapsto 0\}$
$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$
allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$
$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$	$\Vert$	$\{\gamma_2 \hookrightarrow_\circ 0\}$
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$		
$\mathtt{fetchandadd}(x, 2)$		$\mathtt{fetchandadd}(x, 2)$
$\{\gamma_1 \hookrightarrow_\circ 2\}$		$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$!\,x$

$\{n.\ n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$

$\texttt{let}\ x = \texttt{ref}(0)\ \texttt{in}$

$\{x \mapsto 0\}$

$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$

allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$

$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$	$\{\gamma_2 \hookrightarrow_\circ 0\}$
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1+n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\texttt{fetchandadd}(x, 2)$	$\texttt{fetchandadd}(x, 2)$
$\{\gamma_1 \hookrightarrow_\circ 2\}$	$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$!\,x$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\ x = \texttt{ref}(0)\ \texttt{in}$
$\{x \mapsto 0\}$
$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$
allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$
$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$	$\{\gamma_2 \hookrightarrow_\circ 0\}$
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\texttt{fetchandadd}(x, 2)$	$\texttt{fetchandadd}(x, 2)$
$\{\gamma_1 \hookrightarrow_\circ 2\}$	$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$!\,x$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$
$\mathtt{let}\ x = \mathtt{ref}(0)\ \mathtt{in}$
$\{x \mapsto 0\}$
$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$
allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$
$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$	$\{\gamma_2 \hookrightarrow_\circ 0\}$
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\mathtt{fetchandadd}(x, 2)$	$\mathtt{fetchandadd}(x, 2)$
$\{\gamma_1 \hookrightarrow_\circ 2\}$	$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$!\,x$

$\{n.\ n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\ x = \texttt{ref}(0)\ \texttt{in}$
$\{x \mapsto 0\}$
$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$
allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$
$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$	$\{\gamma_2 \hookrightarrow_\circ 0\}$
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1+n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\texttt{fetchandadd}(x, 2)$	$\texttt{fetchandadd}(x, 2)$
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (2+n_2) * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\{\gamma_1 \hookrightarrow_\circ 2\}$	$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$!\,x$

$\{n.\ n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\ x = \texttt{ref}(0)\ \texttt{in}$
$\{x \mapsto 0\}$
$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$
allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$
$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$	$\{\gamma_2 \hookrightarrow_\circ 0\}$
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\texttt{fetchandadd}(x, 2)$	$\texttt{fetchandadd}(x, 2)$
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\{\gamma_1 \hookrightarrow_\circ 2\}$	$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$!\,x$

$\{n.\ n = 4\}$

Ghost variables in action

$\{\mathrm{True}\}$
$\texttt{let}\ x = \texttt{ref}(0)\ \texttt{in}$
$\{x \mapsto 0\}$
$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$
allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$
$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$	$\{\gamma_2 \hookrightarrow_\circ 0\}$
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\texttt{fetchandadd}(x, 2)$	$\texttt{fetchandadd}(x, 2)$
$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\{\gamma_1 \hookrightarrow_\circ 2 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet n_2\}$	
$\{\gamma_1 \hookrightarrow_\circ 2\}$	$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$!\,x$

$\{n.\ n = 4\}$

Ghost variables in action

$\{\text{True}\}$

$\texttt{let}\; x = \texttt{ref}(0)\; \texttt{in}$

$\{x \mapsto 0\}$

$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$

allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$

$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$ || $\{\gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$ || $\{\ldots\}$

$\texttt{fetchandadd}(x, 2)$ || $\texttt{fetchandadd}(x, 2)$

$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$ || $\{\ldots\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\{\gamma_1 \hookrightarrow_\circ 2\}$ || $\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$!\,x$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$

$\texttt{let}\ x = \texttt{ref}(0)\ \texttt{in}$

$\{x \mapsto 0\}$

$\{x \mapsto 0 * \gamma_1 \hookrightarrow_{\bullet} 0 * \gamma_1 \hookrightarrow_{\circ} 0 * \gamma_2 \hookrightarrow_{\bullet} 0 * \gamma_2 \hookrightarrow_{\circ} 0\}$

allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_{\bullet} n_1 * \gamma_2 \hookrightarrow_{\bullet} n_2}$

$\{\gamma_1 \hookrightarrow_{\circ} 0 * \gamma_2 \hookrightarrow_{\circ} 0\}$

$\{\gamma_1 \hookrightarrow_{\circ} 0\}$

$\quad \{\gamma_1 \hookrightarrow_{\circ} 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_{\bullet} n_1 * \gamma_2 \hookrightarrow_{\bullet} n_2\}$

$\quad \{\gamma_1 \hookrightarrow_{\circ} 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_{\bullet} 0 * \gamma_2 \hookrightarrow_{\bullet} n_2\}$

$\quad \texttt{fetchandadd}(x, 2)$

$\quad \{\gamma_1 \hookrightarrow_{\circ} 0 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_{\bullet} 0 * \gamma_2 \hookrightarrow_{\bullet} n_2\}$

$\quad \{\gamma_1 \hookrightarrow_{\circ} 2 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_{\bullet} 2 * \gamma_2 \hookrightarrow_{\bullet} n_2\}$

$\{\gamma_1 \hookrightarrow_{\circ} 2\}$

$\|$

$\{\gamma_2 \hookrightarrow_{\circ} 0\}$

$\quad \{\ldots\}$

$\quad \texttt{fetchandadd}(x, 2)$

$\quad \{\ldots\}$

$\{\gamma_2 \hookrightarrow_{\circ} 2\}$

$\{\gamma_1 \hookrightarrow_{\circ} 2 * \gamma_2 \hookrightarrow_{\circ} 2\}$

$\quad \{\gamma_1 \hookrightarrow_{\circ} 2 * \gamma_2 \hookrightarrow_{\circ} 2 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_{\bullet} n_1 * \gamma_2 \hookrightarrow_{\bullet} n_2\}$

$\quad !\,x$

$\{n.\ n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$

$\texttt{let}\; x = \texttt{ref}(0)\; \texttt{in}$

$\{x \mapsto 0\}$

$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$

allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$

$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\texttt{fetchandadd}(x, 2)$

$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\{\gamma_1 \hookrightarrow_\circ 2\}$

$\parallel$

$\{\gamma_2 \hookrightarrow_\circ 0\}$

$\{\ldots\}$

$\texttt{fetchandadd}(x, 2)$

$\{\ldots\}$

$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$!\,x$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$

$\mathtt{let}\ x = \mathtt{ref}(0)\ \mathtt{in}$

$\{x \mapsto 0\}$

$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$

allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$

$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\mathtt{fetchandadd}(x, 2)$

$\{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\{\gamma_1 \hookrightarrow_\circ 2\}$

$\parallel$

$\{\gamma_2 \hookrightarrow_\circ 0\}$

$\{\ldots\}$

$\mathtt{fetchandadd}(x, 2)$

$\{\ldots\}$

$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2 * x \mapsto 4 * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet 2\}$

$!\,x$

$\{n.\ n = 4\}$

Ghost variables in action

$\{\text{True}\}$

$\texttt{let}\; x = \texttt{ref}(0)\; \texttt{in}$

$\{x \mapsto 0\}$

$\{x \mapsto 0 * \gamma_1 \hookrightarrow_{\bullet} 0 * \gamma_1 \hookrightarrow_{\circ} 0 * \gamma_2 \hookrightarrow_{\bullet} 0 * \gamma_2 \hookrightarrow_{\circ} 0\}$

allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_{\bullet} n_1 * \gamma_2 \hookrightarrow_{\bullet} n_2}$

$\{\gamma_1 \hookrightarrow_{\circ} 0 * \gamma_2 \hookrightarrow_{\circ} 0\}$

$$
\begin{array}{l||l}
\{\gamma_1 \hookrightarrow_{\circ} 0\} & \{\gamma_2 \hookrightarrow_{\circ} 0\} \\
\quad \{\gamma_1 \hookrightarrow_{\circ} 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_{\bullet} n_1 * \gamma_2 \hookrightarrow_{\bullet} n_2\} & \\
\quad \{\gamma_1 \hookrightarrow_{\circ} 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_{\bullet} 0 * \gamma_2 \hookrightarrow_{\bullet} n_2\} & \quad \{\ldots\} \\
\quad \texttt{fetchandadd}(x, 2) & \quad \texttt{fetchandadd}(x, 2) \\
\quad \{\gamma_1 \hookrightarrow_{\circ} 0 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_{\bullet} 0 * \gamma_2 \hookrightarrow_{\bullet} n_2\} & \quad \{\ldots\} \\
\quad \{\gamma_1 \hookrightarrow_{\circ} 2 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_{\bullet} 2 * \gamma_2 \hookrightarrow_{\bullet} n_2\} & \\
\{\gamma_1 \hookrightarrow_{\circ} 2\} & \{\gamma_2 \hookrightarrow_{\circ} 2\}
\end{array}
$$

$\{\gamma_1 \hookrightarrow_{\circ} 2 * \gamma_2 \hookrightarrow_{\circ} 2\}$

$\quad \{\gamma_1 \hookrightarrow_{\circ} 2 * \gamma_2 \hookrightarrow_{\circ} 2 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_{\bullet} n_1 * \gamma_2 \hookrightarrow_{\bullet} n_2\}$

$\quad \{\gamma_1 \hookrightarrow_{\circ} 2 * \gamma_2 \hookrightarrow_{\circ} 2 * x \mapsto 4 * \gamma_1 \hookrightarrow_{\bullet} 2 * \gamma_2 \hookrightarrow_{\bullet} 2\}$

$\quad !\, x$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$

$\texttt{let}\ x = \texttt{ref}(0)\ \texttt{in}$

$\{x \mapsto 0\}$

$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$

allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$

$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$\{\gamma_1 \hookrightarrow_\circ 0\}$

$\quad \{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\quad \{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\quad \texttt{fetchandadd}(x, 2)$

$\quad \{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\quad \{\gamma_1 \hookrightarrow_\circ 2 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\{\gamma_1 \hookrightarrow_\circ 2\}$

$\|$

$\{\gamma_2 \hookrightarrow_\circ 0\}$

$\quad \{\ldots\}$

$\quad \texttt{fetchandadd}(x, 2)$

$\quad \{\ldots\}$

$\{\gamma_2 \hookrightarrow_\circ 2\}$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$\quad \{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\quad \{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2 * x \mapsto 4 * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet 2\}$

$\quad !\, x$

$\quad \{n.\, n = 4 \land \gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2 * x \mapsto 4 * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet 2\}$

$\{n.\, n = 4\}$

Ghost variables in action

$\{\mathsf{True}\}$

$\texttt{let}\; x = \texttt{ref}(0)\; \texttt{in}$

$\{x \mapsto 0\}$

$\{x \mapsto 0 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\circ 0\}$

allocate $\boxed{\exists n_1, n_2.\, x \mapsto n_1 + n_2 * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2}$

$\{\gamma_1 \hookrightarrow_\circ 0 * \gamma_2 \hookrightarrow_\circ 0\}$

$$
\begin{array}{l||l}
\{\gamma_1 \hookrightarrow_\circ 0\} & \{\gamma_2 \hookrightarrow_\circ 0\} \\
\quad \{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\} & \\
\quad \{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto n_2 * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\} & \quad \{\ldots\} \\
\quad \texttt{fetchandadd}(x, 2) & \quad \texttt{fetchandadd}(x, 2) \\
\quad \{\gamma_1 \hookrightarrow_\circ 0 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 0 * \gamma_2 \hookrightarrow_\bullet n_2\} & \quad \{\ldots\} \\
\quad \{\gamma_1 \hookrightarrow_\circ 2 * x \mapsto (2 + n_2) * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet n_2\} & \\
\{\gamma_1 \hookrightarrow_\circ 2\} & \{\gamma_2 \hookrightarrow_\circ 2\}
\end{array}
$$

$\{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2\}$

$\quad \{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2 * x \mapsto (n_1 + n_2) * \gamma_1 \hookrightarrow_\bullet n_1 * \gamma_2 \hookrightarrow_\bullet n_2\}$

$\quad \{\gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2 * x \mapsto 4 * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet 2\}$

$\quad !\,x$

$\quad \{n.\, n = 4 \wedge \gamma_1 \hookrightarrow_\circ 2 * \gamma_2 \hookrightarrow_\circ 2 * x \mapsto 4 * \gamma_1 \hookrightarrow_\bullet 2 * \gamma_2 \hookrightarrow_\bullet 2\}$

$\{n.\, n = 4\}$

Ghost variables with fractional permissions [Boyland]

What if we have n threads? Using n different ghost variables, results in different proofs for each thread. *That is not modular.*

Better way: ghost variables with a *fractional permission* $(0,1]_{\mathbb{Q}}$:

$$\gamma \overset{\pi_1+\pi_2}{\hookrightarrow_{\circ}} (n_1 + n_2) \quad \Leftrightarrow \quad \gamma \overset{\pi_1}{\hookrightarrow_{\circ}} n_1 * \gamma \overset{\pi_2}{\hookrightarrow_{\circ}} n_2$$

Ghost variables with fractional permissions [Boyland]

What if we have n threads? Using n different ghost variables, results in different proofs for each thread. *That is not modular.*

Better way: ghost variables with a *fractional permission* $(0, 1]_{\mathbb{Q}}$:

$$\gamma \xhookrightarrow{\pi_1+\pi_2}_{\circ} (n_1 + n_2) \quad \Leftrightarrow \quad \gamma \xhookrightarrow{\pi_1}_{\circ} n_1 * \gamma \xhookrightarrow{\pi_2}_{\circ} n_2$$

You only get the equality when you have *full ownership* ($\pi = 1$):

$$\gamma \hookrightarrow_{\bullet} n * \gamma \xhookrightarrow{1}_{\circ} m \quad \Rightarrow \quad n = m$$

Ghost variables with fractional permissions [Boyland]

What if we have n threads? Using n different ghost variables, results in different proofs for each thread. *That is not modular.*

Better way: ghost variables with a *fractional permission* $(0, 1]_{\mathbb{Q}}$:

$$\gamma \overset{\pi_1+\pi_2}{\hookrightarrow}_{\circ} (n_1 + n_2) \quad \Leftrightarrow \quad \gamma \overset{\pi_1}{\hookrightarrow}_{\circ} n_1 * \gamma \overset{\pi_2}{\hookrightarrow}_{\circ} n_2$$

You only get the equality when you have *full ownership* ($\pi = 1$):

$$\gamma \hookrightarrow_{\bullet} n * \gamma \overset{1}{\hookrightarrow}_{\circ} m \quad \Rightarrow \quad n = m$$

Updating is possible with *partial ownership* ($0 < \pi \leq 1$):

$$\gamma \hookrightarrow_{\bullet} n * \gamma \overset{\pi}{\hookrightarrow}_{\circ} m \quad \Rrightarrow\!\!\!* \quad \gamma \hookrightarrow_{\bullet} (n + i) * \gamma \overset{\pi}{\hookrightarrow}_{\circ} (m + i)$$

Ghost variables with fractional permissions [Boyland]

What if we have n threads? Using n different ghost variables, results in different proofs for each thread. *That is not modular.*

Better way: ghost variables with a *fractional permission* $(0, 1]_{\mathbb{Q}}$:

$$\gamma \xhookrightarrow{\pi_1+\pi_2}_{\circ} (n_1 + n_2) \quad \Leftrightarrow \quad \gamma \xhookrightarrow{\pi_1}_{\circ} n_1 * \gamma \xhookrightarrow{\pi_2}_{\circ} n_2$$

You only get the equality when you have *full ownership* ($\pi = 1$):

$$\gamma \hookrightarrow_{\bullet} n * \gamma \xhookrightarrow{1}_{\circ} m \quad \Rightarrow \quad n = m$$

Updating is possible with *partial ownership* ($0 < \pi \leq 1$):

$$\gamma \hookrightarrow_{\bullet} n * \gamma \xhookrightarrow{\pi}_{\circ} m \quad \Rrightarrow \quad \gamma \hookrightarrow_{\bullet} (n + i) * \gamma \xhookrightarrow{\pi}_{\circ} (m + i)$$

Keeps the invariant that all $\gamma \xhookrightarrow{\pi_i}_{\circ} n_i$ sum up to $\gamma \hookrightarrow_{\bullet} \sum n_i$

Fractional ghost variables in action

$\{\text{True}\}$
$\texttt{let}\ x = \texttt{ref}(0)\ \texttt{in}$

$$\texttt{fetchandadd}(x, 2) \quad \Big\| \quad \texttt{fetchandadd}(x, 2) \quad \Big\| \quad \ldots$$

$!\,x$

$\{n.\ n = 2k\}$

Fractional ghost variables in action

$\{\text{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$

$$\texttt{fetchandadd}(x, 2) \;\Bigg\|\; \texttt{fetchandadd}(x, 2) \;\Bigg\|\; \ldots$$

$!\, x$

$\{n.\ n = 2k\}$

Fractional ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$
$\left\{x \mapsto 0 * \gamma \hookrightarrow_{\bullet} 0 * \gamma \overset{1}{\hookrightarrow}_{\circ} 0\right\}$

$\texttt{fetchandadd}(x, 2) \quad \Big\| \quad \texttt{fetchandadd}(x, 2) \quad \Big\| \quad \ldots$

$!\, x$

$\{n.\, n = 2k\}$

Fractional ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$
$\left\{x \mapsto 0 * \gamma \hookrightarrow_{\bullet} 0 * \gamma \overset{1}{\hookrightarrow}_{\circ} 0\right\}$
allocate $\boxed{\exists n.\, x \mapsto n * \gamma \hookrightarrow_{\bullet} n}$

$\texttt{fetchandadd}(x, 2) \;\Big\|\; \texttt{fetchandadd}(x, 2) \;\Big\|\; \ldots$

$!\, x$

$\{n.\, n = 2k\}$

Fractional ghost variables in action

$\{\text{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$
$\left\{x \mapsto 0 * \gamma \hookrightarrow_{\bullet} 0 * \gamma \overset{1}{\hookrightarrow}_{\circ} 0\right\}$
allocate $\boxed{\exists n.\, x \mapsto n * \gamma \hookrightarrow_{\bullet} n}$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$ $\Bigg\|$ $\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$ $\Bigg\|$ ...

$\texttt{fetchandadd}(x, 2)$ $\Bigg\|$ $\texttt{fetchandadd}(x, 2)$ $\Bigg\|$ ...

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$ $\Bigg\|$ $\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$ $\Bigg\|$

$!\,x$

$\{n.\, n = 2k\}$

Fractional ghost variables in action

$\{\mathsf{True}\}$
`let` $x =$ `ref`(0) `in`
$\{x \mapsto 0\}$
$\left\{x \mapsto 0 * \gamma \hookrightarrow_{\bullet} 0 * \gamma \overset{1}{\hookrightarrow}_{\circ} 0\right\}$
allocate $\boxed{\exists n.\, x \mapsto n * \gamma \hookrightarrow_{\bullet} n}$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0 * x \mapsto n * \gamma \hookrightarrow_{\bullet} n\right\}$
`fetchandadd`$(x, 2)$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$

$\parallel$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$
`fetchandadd`$(x, 2)$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$

$\parallel$ ...

$!\,x$

$\{n.\, n = 2k\}$

Fractional ghost variables in action

$\{\text{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$
$\left\{x \mapsto 0 * \gamma \hookrightarrow_{\bullet} 0 * \gamma \overset{1}{\hookrightarrow}_{\circ} 0\right\}$
allocate $\boxed{\exists n.\, x \mapsto n * \gamma \hookrightarrow_{\bullet} n}$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0 * x \mapsto n * \gamma \hookrightarrow_{\bullet} n\right\}$
$\texttt{fetchandadd}(x, 2)$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2 * x \mapsto (2+n) * \gamma_1 \hookrightarrow_{\bullet} (2+n)\right\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$

$\|$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$
$\texttt{fetchandadd}(x, 2)$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$

$\|$ ...

$!\, x$

$\{n.\, n = 2k\}$

Fractional ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$
$\left\{x \mapsto 0 * \gamma \hookrightarrow_{\bullet} 0 * \gamma \overset{1}{\hookrightarrow}_{\circ} 0\right\}$
allocate $\boxed{\exists n.\, x \mapsto n * \gamma \hookrightarrow_{\bullet} n}$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0 * x \mapsto n * \gamma \hookrightarrow_{\bullet} n\right\}$
$\texttt{fetchandadd}(x, 2)$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2 * x \mapsto (2{+}n) * \gamma_1 \hookrightarrow_{\bullet} (2{+}n)\right\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$

$\|$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$
$\{\ldots\}$
$\texttt{fetchandadd}(x, 2)$
$\{\ldots\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$

$\| \ \ldots$

$!\, x$

$\{n.\, n = 2k\}$

Fractional ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\ x = \texttt{ref}(0)\ \texttt{in}$
$\{x \mapsto 0\}$
$\left\{x \mapsto 0 * \gamma \hookrightarrow_{\bullet} 0 * \gamma \overset{1}{\hookrightarrow}_{\circ} 0\right\}$
allocate $\boxed{\exists n.\, x \mapsto n * \gamma \hookrightarrow_{\bullet} n}$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0 * x \mapsto n * \gamma \hookrightarrow_{\bullet} n\right\}$
$\texttt{fetchandadd}(x, 2)$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2 * x \mapsto (2+n) * \gamma_1 \hookrightarrow_{\bullet} (2+n)\right\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$
$\left\{\gamma \overset{1}{\hookrightarrow}_{\circ} 2k * x \mapsto n * \gamma \hookrightarrow_{\bullet} n\right\}$
$!\,x$
$\{n.\ n = 2k\}$

$\Bigg\|$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$
$\{\ldots\}$
$\texttt{fetchandadd}(x, 2)$
$\{\ldots\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$

$\Bigg\|$ $\ldots$

Fractional ghost variables in action

$\{\mathsf{True}\}$
$\texttt{let}\, x = \texttt{ref}(0)\, \texttt{in}$
$\{x \mapsto 0\}$
$\left\{x \mapsto 0 * \gamma \hookrightarrow_{\bullet} 0 * \gamma \overset{1}{\hookrightarrow}_{\circ} 0\right\}$
allocate $\boxed{\exists n.\, x \mapsto n * \gamma \hookrightarrow_{\bullet} n}$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0 * x \mapsto n * \gamma \hookrightarrow_{\bullet} n\right\}$
$\texttt{fetchandadd}(x, 2)$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2 * x \mapsto (2+n) * \gamma_1 \hookrightarrow_{\bullet} (2+n)\right\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$

$\|$

$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 0\right\}$
$\{\ldots\}$
$\texttt{fetchandadd}(x, 2)$
$\{\ldots\}$
$\left\{\gamma \overset{1/k}{\hookrightarrow}_{\circ} 2\right\}$

$\| \quad \ldots$

$\left\{\gamma \overset{1}{\hookrightarrow}_{\circ} 2k * x \mapsto n * \gamma \hookrightarrow_{\bullet} n\right\}$
$!\, x$
$\left\{n.\, n = 2k \wedge \gamma \overset{1}{\hookrightarrow}_{\circ} 2k * x \mapsto 2k * \gamma \hookrightarrow_{\bullet} 2k\right\}$
$\{n.\, n = 2k\}$

The Iris story, Part 2:
Modeling ghost state via “PCMs”

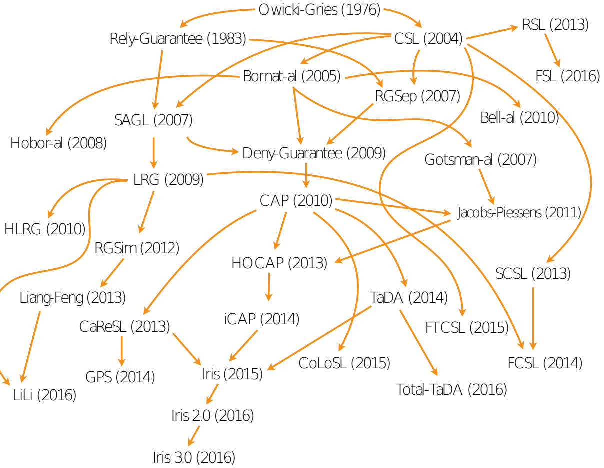
Mechanisms for concurrent reasoning

We have seen so far:

- Invariants $\boxed{R}^{\mathcal{N}}$
- Ghost variables $\gamma \hookrightarrow_{\bullet} n$ and $\gamma \hookrightarrow_{\circ} n$
- Fractional ghost variables $\gamma \hookrightarrow_{\bullet} n$ and $\gamma \overset{\pi}{\hookrightarrow}_{\circ} n$

Where do these mechanisms come from?

There are many CSLs with more powerful mechanisms. . .



Picture by Ilya Sergey

...and very complicated **primitive** rules

$$\frac{\begin{array}{c}\Gamma, \Delta \mid \Phi \vdash \mathsf{stable}(\mathsf{P}) \qquad \Gamma, \Delta \mid \Phi \vdash \forall y.\ \mathsf{stable}(\mathsf{Q}(y)) \\ \Gamma, \Delta \mid \Phi \vdash n \in C \qquad \Gamma, \Delta \mid \Phi \vdash \forall x \in X.\ (x, f(x)) \in \overline{T(A)} \vee f(x) = x \\ \Gamma \mid \Phi \vdash \forall x \in X.\ (\Delta).\langle \mathsf{P} * \circledast_{\alpha \in A}[\alpha]^{\mathsf{n}}_{g(\alpha)} * \triangleright I(x)\rangle\ c\ \langle \mathsf{Q}(x) * \triangleright I(f(x))\rangle^{C \setminus \{n\}}\end{array}}{\begin{array}{c}\Gamma \mid \Phi \vdash (\Delta).\ \langle \mathsf{P} * \circledast_{\alpha \in A}[\alpha]^{\mathsf{n}}_{g(\alpha)} * \mathsf{region}(X, T, I, n)\rangle \\ c \\ \langle \exists x.\ \mathsf{Q}(x) * \mathsf{region}(\{f(x)\}, T, I, n)\rangle^{C}\end{array}}\ \textsc{Atomic}$$

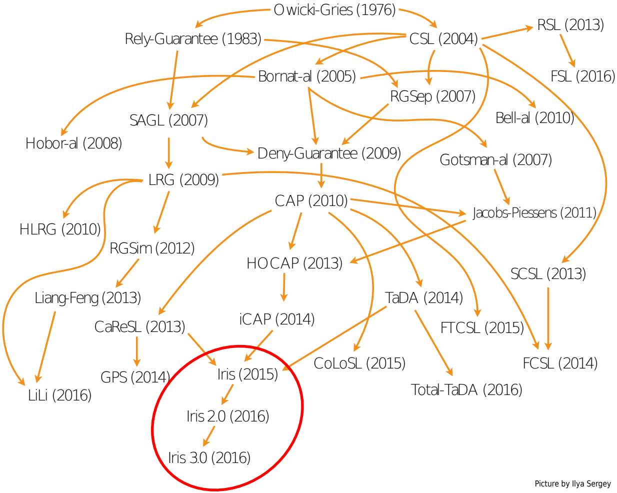
$$\frac{\mathcal{C} \vdash \forall b \overset{\text{rely}}{\sqsupseteq_{\pi}} b_0.\ (\!|\pi[\![b]\!] * P|\!)\ i \Rrightarrow_{\mathrm{I}} a\ (\!|x.\ \exists b' \overset{\text{guar}}{\sqsupseteq_{\pi}} b.\ \pi[\![b']\!] * Q|\!)}{\mathcal{C} \vdash \left\{\boxed{b_0}^{n}_{\pi} * \triangleright P\right\}\ i \Rrightarrow a\ \left\{x.\ \exists b'.\ \boxed{b'}^{n}_{\pi} * Q\right\}}\ \textsc{UpdIsl}$$

$$\frac{\begin{array}{c}\text{Use atomic rule} \\ a \notin \mathcal{A} \quad \forall x \in X.\ (x, f(x)) \in \mathcal{T}_{\mathbf{t}}(\mathrm{G})^{*} \\ \lambda; \mathcal{A} \vdash \mathbb{W} x \in X.\ \left\langle p_p \,\middle|\, I(\mathbf{t}^{\lambda}_{a}(x)) * p(x) * [\mathrm{G}]_a \right\rangle\ \mathbb{C}\ \ \exists\!\!\exists y \in Y.\ \left\langle q_p(x, y) \,\middle|\, I(\mathbf{t}^{\lambda}_{a}(f(x))) * q(x, y) \right\rangle\end{array}}{\lambda + 1; \mathcal{A} \vdash \mathbb{W} x \in X.\ \left\langle p_p \,\middle|\, \mathbf{t}^{\lambda}_{a}(x) * p(x) * [\mathrm{G}]_a \right\rangle\ \mathbb{C}\ \ \exists\!\!\exists y \in Y.\ \left\langle q_p(x, y) \,\middle|\, \mathbf{t}^{\lambda}_{a}(f(x)) * q(x, y) \right\rangle}$$

$$\frac{\begin{array}{c}\Gamma \mid \Phi \vdash x \in X \qquad \Gamma \mid \Phi \vdash \forall \alpha \in \mathsf{Action}.\ \forall x \in \mathsf{SId} \times \mathsf{SId}.\ up(T(\alpha)(x)) \\ \Gamma \mid \Phi \vdash A \text{ and } B \text{ are finite} \qquad \Gamma \mid \Phi \vdash C \text{ is infinite} \\ \Gamma \mid \Phi \vdash \forall n \in C.\ \mathsf{P} * \circledast_{\alpha \in A}[\alpha]^{n}_{1} \Rightarrow \triangleright I(n)(x) \\ \Gamma \mid \Phi \vdash \forall n \in C.\ \forall s.\ \mathsf{stable}(I(n)(s)) \qquad \Gamma \mid \Phi \vdash A \cap B = \emptyset\end{array}}{\Gamma \mid \Phi \vdash \mathsf{P} \sqsubseteq^{C} \exists n \in C.\ \mathsf{region}(X, T, I(n), n) * \circledast_{\alpha \in B}[\alpha]^{n}_{1}}\ \textsc{VAlloc}$$

$$\frac{\begin{array}{c}\text{Update region rule} \\ \lambda; \mathcal{A} \vdash \mathbb{W} x \in X.\ \left\langle p_p \,\middle|\, I(\mathbf{t}^{\lambda}_{a}(x)) * p(x) \right\rangle\ \mathbb{C}\ \ \exists\!\!\exists y \in Y.\ \left\langle q_p(x, y) \,\middle|\, \begin{array}{c} I(\mathbf{t}^{\lambda}_{a}(Q(x))) * q_1(x, y) \\ \vee\, I(\mathbf{t}^{\lambda}_{a}(x)) * q_2(x, y)\end{array} \right\rangle\end{array}}{\begin{array}{rl} & \mathbb{W} x \in X.\ \left\langle p_p \,\middle|\, \mathbf{t}^{\lambda}_{a}(x) * p(x) * a \Rrightarrow \blacklozenge \right\rangle \\ \lambda{+}1; a : x{\in}X \rightsquigarrow Q(x), \mathcal{A} \vdash & \mathbb{C} \\ & \exists\!\!\exists y{\in}Y.\ \left\langle q_p(x, y) \,\middle|\, \begin{array}{c}\exists z \in Q(x).\ \mathbf{t}^{\lambda}_{a}(z) * q_1(x, y) * a \Rrightarrow (x, z) \\ \vee\, \mathbf{t}^{\lambda}_{a}(x) * q_2(x, y) * a \Rrightarrow \blacklozenge\end{array} \right\rangle\end{array}}$$

The Iris story



Picture by Ilya Sergey

The Iris story: many of these mechanisms can be **encoded** using a simple mechanism of *resource ownership*

Generalizing ownership

All forms of ownership have common properties:

- Ownership of different threads can be composed
  For example:

$$\gamma \xhookrightarrow{\pi_1+\pi_2}_{\circ} (n_1 + n_2) \quad \Leftrightarrow \quad \gamma \xhookrightarrow{\pi_1}_{\circ} n_1 * \gamma \xhookrightarrow{\pi_2}_{\circ} n_2$$

Generalizing ownership

All forms of ownership have common properties:

- Ownership of different threads can be composed
  For example:

$$\gamma \xhookrightarrow{\pi_1+\pi_2}_{\circ} (n_1 + n_2) \quad \Leftrightarrow \quad \gamma \xhookrightarrow{\pi_1}_{\circ} n_1 * \gamma \xhookrightarrow{\pi_2}_{\circ} n_2$$

- Composition of ownership is associative and commutative
  Mirroring that parallel composition and separating conjunction is associative and commutative

Generalizing ownership

All forms of ownership have common properties:

- Ownership of different threads can be composed
  For example:

$$\gamma \overset{\pi_1+\pi_2}{\hookrightarrow_\circ} (n_1 + n_2) \quad \Leftrightarrow \quad \gamma \overset{\pi_1}{\hookrightarrow_\circ} n_1 * \gamma \overset{\pi_2}{\hookrightarrow_\circ} n_2$$

- Composition of ownership is associative and commutative
  Mirroring that parallel composition and separating conjunction is associative and commutative
- Combinations of ownership that do not make sense are ruled out
  For example:

$$\gamma \hookrightarrow_\bullet 5 * \gamma \overset{1/2}{\hookrightarrow_\circ} 3 * \gamma \overset{1/2}{\hookrightarrow_\circ} 4 \quad \Rightarrow \quad \text{False}$$

(because $5 \neq 3 + 4$)

Resource algebras (RAs): A generalization of PCMs

Resource algebra (RA) with carrier M:

- Composition $(\cdot) : M \to M \to M$
- Validity predicate $\mathcal{V} \subseteq M$

Satisfying:

$$a \cdot b = b \cdot a \qquad a \cdot (b \cdot c) = (a \cdot b) \cdot c \qquad (a \cdot b) \in \mathcal{V} \Rightarrow a \in \mathcal{V}$$

Resource algebras (RAs): A generalization of PCMs

Resource algebra (RA) with carrier M:

- Composition $(\cdot) : M \to M \to M$
- Validity predicate $\mathcal{V} \subseteq M$

Satisfying:

$$a \cdot b = b \cdot a \qquad a \cdot (b \cdot c) = (a \cdot b) \cdot c \qquad (a \cdot b) \in \mathcal{V} \Rightarrow a \in \mathcal{V}$$

Iris has ghost variables $\boxed{a : M}^{\gamma}$ for each resource algebra M

$$a \in \mathcal{V} \Rrightarrow \exists \gamma.\, \boxed{a}^{\gamma} \qquad \boxed{a}^{\gamma} * \boxed{b}^{\gamma} \Leftrightarrow \boxed{a \cdot b}^{\gamma} \qquad \boxed{a}^{\gamma} \Rightarrow \mathcal{V}(a)$$

$$\frac{\forall a_{\mathrm{f}}.\, a \cdot a_{\mathrm{f}} \in \mathcal{V} \Rightarrow b \cdot a_{\mathrm{f}} \in \mathcal{V}}{\boxed{a}^{\gamma} \Rrightarrow \boxed{b}^{\gamma}}$$

Ghost variables revisited

Resource algebra for ghost variables:

$$
\begin{aligned}
M &\triangleq \bullet\, n \mid \circ\, n \mid \bot \mid \bullet\!\!\circ\, n \\
\mathcal{V} &\triangleq \{a \neq \bot \mid a \in M\} \\
\bullet\, n \cdot \circ\, n' = \circ\, n' \cdot \bullet\, n &\triangleq \begin{cases} \bullet\!\!\circ\, n & \text{if } n = n' \\ \bot & \text{otherwise} \end{cases} \\
\text{other combinations} &\triangleq \bot
\end{aligned}
$$

And define:

$$
\gamma \hookrightarrow_{\bullet} n \triangleq \boxed{\bullet\, n}^{\gamma} \qquad\qquad \gamma \hookrightarrow_{\circ} n \triangleq \boxed{\circ\, n}^{\gamma}
$$

Ghost variables revisited

Resource algebra for ghost variables:

$$M \triangleq \bullet\, n \mid \circ\, n \mid \bot \mid \bullet\!\circ\, n$$

$$\mathcal{V} \triangleq \{a \neq \bot \mid a \in M\}$$

$$\bullet\, n \cdot \circ\, n' = \circ\, n' \cdot \bullet\, n \triangleq \begin{cases} \bullet\!\circ\, n & \text{if } n = n' \\ \bot & \text{otherwise} \end{cases}$$

$$\text{other combinations} \triangleq \bot$$

And define:

$$\gamma \hookrightarrow_{\bullet} n \triangleq \left[\bullet\, n\right]^{\gamma} \qquad\qquad \gamma \hookrightarrow_{\circ} n \triangleq \left[\circ\, n\right]^{\gamma}$$

The ghost variable rules follow directly from the general rules:

$$\text{True} \Rrightarrow \exists \gamma.\, \gamma \hookrightarrow_{\bullet} n * \gamma \hookrightarrow_{\circ} n$$

Ghost variables revisited

Resource algebra for ghost variables:

$$M \triangleq \bullet\, n \mid \circ\, n \mid \bot \mid \bullet\!\circ\, n$$

$$\mathcal{V} \triangleq \{a \neq \bot \mid a \in M\}$$

$$\bullet\, n \cdot \circ\, n' = \circ\, n' \cdot \bullet\, n \triangleq \begin{cases} \bullet\!\circ\, n & \text{if } n = n' \\ \bot & \text{otherwise} \end{cases}$$

$$\text{other combinations} \triangleq \bot$$

And define:

$$\gamma \hookrightarrow_{\bullet} n \triangleq \left[\bullet\, n\right]^{\gamma} \qquad\qquad \gamma \hookrightarrow_{\circ} n \triangleq \left[\circ\, n\right]^{\gamma}$$

The ghost variable rules follow directly from the general rules:

$$\text{True} \Rrightarrow\!\!\ast\; \exists \gamma.\, \left[\bullet\!\circ\, n\right]^{\gamma} \Rrightarrow\!\!\ast\; \exists \gamma.\, \gamma \hookrightarrow_{\bullet} n * \gamma \hookrightarrow_{\circ} n$$

Ghost variables revisited

Resource algebra for ghost variables:

$$
\begin{aligned}
M &\triangleq \bullet\, n \mid \circ\, n \mid \bot \mid \bullet\!\circ\, n \\
\mathcal{V} &\triangleq \{a \neq \bot \mid a \in M\} \\
\bullet\, n \cdot \circ\, n' = \circ\, n' \cdot \bullet\, n &\triangleq \begin{cases} \bullet\!\circ\, n & \text{if } n = n' \\ \bot & \text{otherwise} \end{cases} \\
\text{other combinations} &\triangleq \bot
\end{aligned}
$$

And define:

$$
\gamma \hookrightarrow_{\bullet} n \triangleq \boxed{\bullet\, n}^{\gamma} \qquad\qquad \gamma \hookrightarrow_{\circ} n \triangleq \boxed{\circ\, n}^{\gamma}
$$

The ghost variable rules follow directly from the general rules:

$$
\begin{gathered}
\mathrm{True} \Rrightarrow\!\!\ast\; \exists \gamma.\, \boxed{\bullet\!\circ\, n}^{\gamma} \Rrightarrow\!\!\ast\; \exists \gamma.\, \gamma \hookrightarrow_{\bullet} n * \gamma \hookrightarrow_{\circ} n \\
\gamma \hookrightarrow_{\bullet} n * \gamma \hookrightarrow_{\circ} m \Rightarrow n = m
\end{gathered}
$$

Ghost variables revisited

Resource algebra for ghost variables:

$$
\begin{aligned}
M &\triangleq \bullet\, n \mid \circ\, n \mid \bot \mid \bullet\!\!\circ\, n \\
\mathcal{V} &\triangleq \{a \neq \bot \mid a \in M\} \\
\bullet\, n \cdot \circ\, n' = \circ\, n' \cdot \bullet\, n &\triangleq \begin{cases} \bullet\!\!\circ\, n & \text{if } n = n' \\ \bot & \text{otherwise} \end{cases} \\
\text{other combinations} &\triangleq \bot
\end{aligned}
$$

And define:

$$
\gamma \hookrightarrow_{\bullet} n \triangleq \boxed{\bullet\, n}^{\gamma} \qquad\qquad \gamma \hookrightarrow_{\circ} n \triangleq \boxed{\circ\, n}^{\gamma}
$$

The ghost variable rules follow directly from the general rules:

$$
\begin{gathered}
\text{True} \Rrightarrow\!\!\!\ast\; \exists \gamma.\, \boxed{\bullet\!\!\circ\, n}^{\gamma} \Rrightarrow\!\!\!\ast\; \exists \gamma.\, \gamma \hookrightarrow_{\bullet} n * \gamma \hookrightarrow_{\circ} n \\
\gamma \hookrightarrow_{\bullet} n * \gamma \hookrightarrow_{\circ} m \Rightarrow (\bullet\, n \cdot \circ\, m) \in \mathcal{V} \Rightarrow n = m
\end{gathered}
$$

Updating resources

Resources can be *updated* using *frame-preserving updates:*

$$\frac{\forall a_{\mathrm{f}}.\, a \cdot a_{\mathrm{f}} \in \mathcal{V} \Rightarrow b \cdot a_{\mathrm{f}} \in \mathcal{V}}{\boxed{a}^{\gamma} \Rrightarrow\!\!\ast\ \boxed{b}^{\gamma}}$$

Key idea: a resource can be updated if the update does not invalidate the resources of concurrently-running threads

$$\begin{array}{ccccccccc}
\text{Thread 1} & & \text{Thread 2} & & \dots & & \text{Thread n} & & \\
\color{red}{a_1} & \cdot & a_2 & \cdot & \dots & \cdot & a_n & & \in \mathcal{V} \\
\color{red}{\rightsquigarrow} & & & & & & & & \\
\color{red}{b_1} & \cdot & a_2 & \cdot & \dots & \cdot & a_n & & \in \mathcal{V}
\end{array}$$

Updating resources

Resources can be *updated* using *frame-preserving updates:*

$$\frac{\forall a_{\mathrm{f}}.\, a \cdot a_{\mathrm{f}} \in \mathcal{V} \Rightarrow b \cdot a_{\mathrm{f}} \in \mathcal{V}}{\boxed{a}^{\gamma} \Rrightarrow\!\!\ast\; \boxed{b}^{\gamma}}$$

Key idea: a resource can be updated if the update does not invalidate the resources of concurrently-running threads

$$\begin{array}{ccccccccc} \text{Thread 1} & & \text{Thread 2} & & \ldots & & \text{Thread n} & & \\ a_1 & \cdot & a_2 & \cdot & \ldots & \cdot & a_n & & \in \mathcal{V} \\ \rightsquigarrow & & & & & & & & \\ b_1 & \cdot & a_2 & \cdot & \ldots & \cdot & a_n & & \in \mathcal{V} \end{array}$$

The rule $\gamma \hookrightarrow_{\bullet} n * \gamma \hookrightarrow_{\circ} m \Rrightarrow\!\!\ast\; \gamma \hookrightarrow_{\bullet} n' * \gamma \hookrightarrow_{\circ} n'$ follows directly

Generalizing to a library of RA combinators

Iris comes with a library of useful RA combinators

- $\textsc{Auth}(M)$: Generalizes the $\bullet$, $\circ$, $\bullet\!\circ$ construction over an arbitrary RA M – we call it the "authoritative" RA.
- $\textsc{Frac}$: The RA for fractions in $(0, 1]$ with addition.
- $\textsc{Excl}(X)$: The "exclusive" RA, whose valid elements are the elements of X, and where composition is always undefined.
- The expected RA liftings of products, sums, etc.

Using these combinators, we can easily construct the necessary models of many desired forms of ghost state:

- Ghost variables from this talk: $\textsc{Auth}\ (\textsc{Excl}\ \textsc{Nat})$
- Fractional ghost variables: $\textsc{Auth}\ (\textsc{Frac} \times \textsc{Nat}_{+})$

Many things I haven't covered

Modal basis of Iris: $\square$, $\triangleright$, $\Rrightarrow$

- Persistent modality $\square\, P$: Says P holds forever, i.e., only relying on duplicable resources, such as invariants
- Later modality $\triangleright P$: Says P holds one step-index later (lower); needed to model impredicative invariants
- Update modality $\Rrightarrow P$: Says P holds after some frame-preserving update to ghost state

Higher-order ghost state, e.g., named propositions $\gamma \mapsto P$

- $\gamma \mapsto P * \gamma \mapsto Q \Rightarrow \triangleright(P = Q)$
- Sounds arcane, but turns out to be surprisingly useful!
- Achieved by equipping RAs with a step-indexing structure

Encoding of Iris program logic (including invariants)
within the modal base logic (with higher-order ghost state)

Conclusion

The Iris methodology for concurrent reasoning:

- Divide up logical ownership of shared physical state using appropriately chosen **ghost state predicates and axioms**
- Tie ghost state assertions to physical state using **invariants**
- Build model of ghost state predicates by choosing an appropriate (step-indexed) **"PCM"**
- Verify ghost state axioms as instances of a few basic laws like **frame-preserving update**