Iris: Monoids and Invariants as an Orthogonal basis for Concurrent Reasoning

Kasper Svendsen

joint work with
Ralf Young, David Swasey, Filip Sieczkowski,
Aaron Turon, Lars Birkedal and Derek Dreyer

LRG

RGSep

CAP

HOCAP

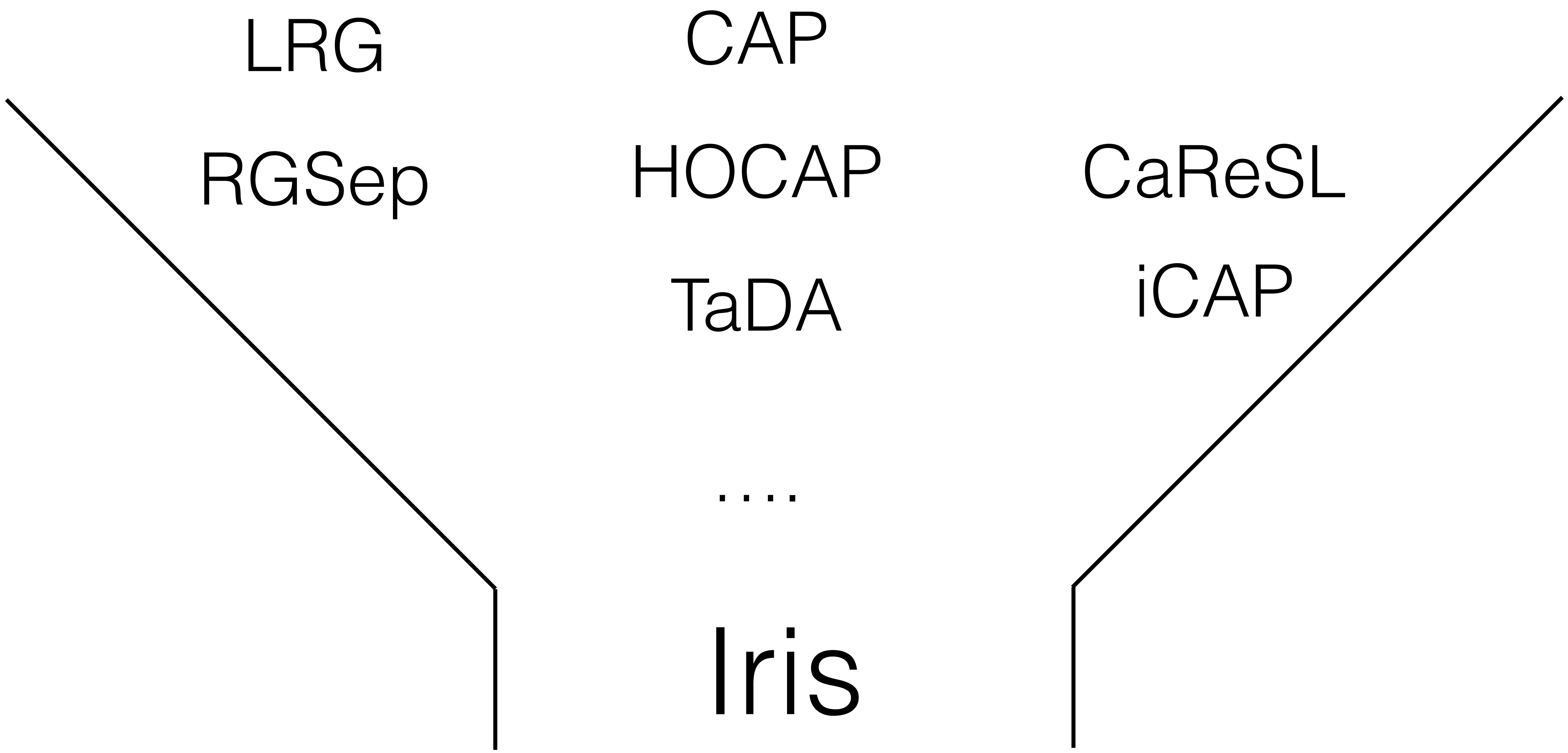
TaDA

CaReSL

iCAP

….

A uniform framework for describing interference



Iris

Higher-order separation logic + **Impredicative Invariants** + **Monoids**

- Supports encoding of existing reasoning principles
  - Monoids for **expressing** protocols on shared state
  - Invariants for **enforcing** protocols on shared state

Iris

Higher-order separation logic	+	Impredicative Invariants	+	Monoids

- Invariants and monoids are **orthogonal**
    - Treating them as such, leads to a simpler logic, and a **model** simple enough to **formalize in Coq**

Iris

Higher-order separation logic + **Impredicative Invariants** + **Monoids**

- Supports a notion of **logical atomicity**
  - extends reasoning principles usually reserved for atomic code to code that **appears** to be atomic
  - we can **define** logical atomicity in Iris

Iris

Higher-order separation logic + **Impredicative** + **Monoids**

Normally we have to reason about possible interference between every statement

R* | c1 | R* | c2 | R*

- Supports a notion of **logical atomicity**
  - extends reasoning principles usually reserved for atomic code to code that **appears** to be atomic
  - we can **define** logical atomicity in Iris

Iris

Higher-order separation logic + **Impredicative** + **Monoids**

No need to reason about interference **inside** logically atomic operations

R* c1 c2 R*

- Supports a notion of **logical atomicity**
  - extends reasoning principles usually reserved for atomic code to code that **appears** to be atomic
  - we can **define** logical atomicity in Iris

Part 1
Iris

Invariants

- An invariant is a property that holds of some piece of shared state at **all** times

$$\frac{\{\triangleright R * P\}\ e\ \{\triangleright R * Q\}_{\mathcal{E}} \qquad e \text{ atomic}}{\left\{\boxed{R}^{\iota} * P\right\}\ e\ \left\{Q\right\}_{\mathcal{E} \uplus \{\iota\}}}$$

The set of invariants that we may open

Invariants

- An invariant is a property that holds of some piece of shared state at **all** times

$$\frac{\{\triangleright R * P\}\ e\ \{\triangleright R * Q\}_{\mathcal{E}} \qquad e \text{ atomic}}{\left\{\boxed{R}^{\iota} * P\right\}\ e\ \left\{Q\right\}_{\mathcal{E} \uplus \{\iota\}}}$$

There exists a shared invariant that owns *R*

The set of invariants that we may open

Invariants

- An invariant is a property that holds of some piece of shared state at **all** times

We open the invariant and take ownership of R

$$\frac{\{\triangleright R * P\}\ e\ \{\triangleright R * Q\}_{\mathcal{E}} \qquad e\ \text{atomic}}{\left\{\boxed{R}^{\iota} * P\right\}\ e\ \left\{Q\right\}_{\mathcal{E} \uplus \{\iota\}}}$$

There exists a shared invariant that owns *R*

The set of invariants that we may open

Invariants

- An invariant is a property that holds of some piece of shared state at **all** times

We open the invariant and take ownership of R

To close the invariant, we must relinquish ownership of R

$$\frac{\{\triangleright R * P\}\ e\ \{\triangleright R * Q\}_{\mathcal{E}} \qquad e \text{ atomic}}{\left\{\boxed{R}^{\iota} * P\right\}\ e\ \left\{Q\right\}_{\mathcal{E} \uplus \{\iota\}}}$$

There exists a shared invariant that owns *R*

The set of invariants that we may open

Invariants

Higher-order separation logic + **Impredicative Invariants** + **Monoids**

- Introduces a circularity in the model
- Modelled using standard metric-based techniques (ModuRes library in Coq)

Monoids

- Iris is parameterised by a notion of ghost resources
- Ghost resources consists of
  - **Information** about the current ghost state
  - **Rights** to update ghost state
- We use monoids to model ghost resources

Monoids

- Ghost resource $\boxed{m}$ asserts ownership of m fragment

- Ghost resources can be split arbitrarily

$$\boxed{m_1 \cdot m_2} \Leftrightarrow \boxed{m_1} * \boxed{m_2}$$

- and support frame-preserving updates

$$\frac{\forall a_f.\ (a \cdot a_f)\downarrow \Rightarrow (b \cdot a_f)\downarrow}{\boxed{a} \Rrightarrow \boxed{b}}$$

Part 2
Recovering existing reasoning principles

Deriving small-footprint specifications

- **Example**: recovering small-footprint specifications from large-footprint specifications

- Same idea as in Superficially Substructural Types (ICFP12) and Fictional Separation Logic (ESOP12)

A λ-calculus with channels

- We instantiate Iris with a λ-calculus with channels

$$e ::= \ldots \mid \textbf{newch} \mid \textbf{send}(e, e) \mid \textbf{tryrecv}(e) \mid \textbf{fork}(e)$$

- with the following per-thread reduction semantics

$$C[c \mapsto M]; \textbf{send}(c, v) \to C[c \mapsto M \uplus \{v\}]; ()$$
$$C[c \mapsto \emptyset]; \textbf{tryrecv}(c) \to C[c \mapsto \emptyset]; \textbf{none}$$
$$C[c \mapsto M \uplus \{v\}]; \textbf{tryrecv}(c) \to C[c \mapsto M]; \textbf{some}(v)$$

Large-footprint specs

- Reduction relation lifts directly to large-footprint specs
- The reduction

$$C[c \mapsto M]; \textbf{send}(c, v) \to C[c \mapsto M \uplus \{v\}]; ()$$

yields the following axiom

$$\{\lfloor C[c \mapsto M] \rfloor\}\ \textbf{send}(c, v)\ \{r.\ r = () \wedge \lfloor C[c \mapsto M \uplus \{v\}] \rfloor\}$$

Asserts exclusive ownership of entire physical state

Small-footprint specs

- Large-footprint spec requires global reasoning

$$\{\lfloor C[c \mapsto M] \rfloor\}\ \textbf{send}(c, v)\ \{r.\ r = () \wedge \lfloor C[c \mapsto M \uplus \{v\}] \rfloor\}$$

- **Goal:** Derive small-footprint specification that only mentions channels affected by each operation

Small-footprint specs

- **Idea**
    - Introduce appropriate channel ghost resources
    - Introduce an invariant that owns the physical state (so that it can be shared) and ties ghost resources to physical state
- Extends to a general construction

Channel-local monoid

- **Goal:** ghost channels resources that support exclusive ownership of individual channels

- Use partial channel "heaps"

$$|\textsc{Net}| = Chan \overset{\text{fin}}{\rightharpoonup} MsgBag$$
$$f \cdot g = f \cup g, \qquad \text{if } \mathrm{dom}(f) \cap \mathrm{dom}(g) = \emptyset$$

- $\boxed{[c \mapsto M]}$ asserts exclusive ownership of ghost channel c and that contains messages M

Authoritative monoid

- **Goal:** a monoid with
  - An authoritative element $m\bullet$ that asserts that the current ghost state is exactly m
  - A partial element $m\circ$ that asserts ownership of an m fragment of the authoritative state
- s.t. all fragments combine to the authoritative state

Deriving a channel-local specification

$$\{c \prec M\}$$

$$\textbf{send}(c, m)$$

$$\{c \prec M \uplus \{m\}\}$$

Deriving a channel-local specification

Channel resource asserts ownership of corresponding fragment:

$$c \prec M \triangleq \boxed{[c \mapsto M]}^{\circ}$$

$$\{c \prec M\}$$

$$\textbf{send}(c, m)$$

$$\{c \prec M \uplus \{m\}\}$$

Deriving a channel-local specification

Invariant: the physical state is authoritative ghost state

$$\exists C.\ \boxed{C\bullet} * \lfloor C \rfloor$$

Channel resource asserts ownership of corresponding fragment:

$$c \prec M \triangleq \boxed{[c \mapsto M]\circ}$$

$\{c \prec M\}$

send(c, m)

$\{c \prec M \uplus \{m\}\}$

Deriving a channel-local specification

Invariant: the physical state is authoritative ghost state

$$\exists C.\ \boxed{C\bullet} * \lfloor C \rfloor$$

Channel resource asserts ownership of corresponding fragment:

$$c \prec M \triangleq \boxed{[c \mapsto M]\circ}$$

$\{c \prec M\}$

$\{\boxed{[c \mapsto M]\circ} * \qquad\qquad\qquad\qquad\}$

send(c, m)

$\{c \prec M \uplus \{m\}\}$

Deriving a channel-local specification

Invariant: the physical state is authoritative ghost state

$$\exists C.\ \boxed{C^{\bullet}} * \lfloor C \rfloor$$

Channel resource asserts ownership of corresponding fragment:

$$c \prec M \triangleq \boxed{[c \mapsto M]^{\circ}}$$

$$\{c \prec M\}$$

$$\{\boxed{[c \mapsto M]^{\circ}} * \boxed{C^{\bullet}} * \lfloor C \rfloor\}$$

$$\textbf{send}(c, m)$$

$$\{c \prec M \uplus \{m\}\}$$

Deriving a channel-local specification

Invariant: the physical state is authoritative ghost state

$$\exists C.\ \boxed{C\bullet} * \lfloor C \rfloor$$

Channel resource asserts ownership of corresponding fragment:

$$c \prec M \triangleq \boxed{[c \mapsto M]\circ}$$

$$\{c \prec M\}$$

$$\{\boxed{[c \mapsto M]\circ} * \boxed{C\bullet} * \lfloor C \rfloor\}$$

$\color{blue}{\textbf{send}(c, m)}$

$$\{ \qquad\qquad\qquad\qquad * \lfloor C[c \mapsto C(c) \uplus \{m\}] \rfloor\}$$

$$\{c \prec M \uplus \{m\}\}$$

Deriving a channel-local specification

Invariant: the physical state is authoritative ghost state

$$\exists C.\ \boxed{C\bullet} * \lfloor C \rfloor$$

Channel resource asserts ownership of corresponding fragment:

$$c \prec M \triangleq \boxed{[c \mapsto M]\circ}$$

$$\{c \prec M\}$$

$$\{\boxed{[c \mapsto M]\circ} * \boxed{C\bullet} * \lfloor C \rfloor\}$$

$$\textbf{send}(c, m)$$

$$\{\boxed{[c \mapsto M]\circ} * \boxed{C\bullet} * \lfloor C[c \mapsto C(c) \uplus \{m\}] \rfloor\}$$

$$\{c \prec M \uplus \{m\}\}$$

Deriving a channel-local specification

Invariant: the physical state is authoritative ghost state

$$\exists C.\ \boxed{C^{\bullet}} * \lfloor C \rfloor$$

Channel resource asserts ownership of corresponding fragment:

$$c \prec M \triangleq \boxed{[c \mapsto M]^{\circ}}$$

$$\{c \prec M\}$$

$$\{\boxed{[c \mapsto M]^{\circ}} * \boxed{C^{\bullet}} * \lfloor C \rfloor\}$$

$\textbf{send}(c, m)$

$$\{\boxed{[c \mapsto M]^{\circ}} * \boxed{C^{\bullet}} * \lfloor \overbrace{C[c \mapsto C(c) \uplus \{m\}]}^{C'} \rfloor\}$$

$$\{c \prec M \uplus \{m\}\}$$

Deriving a channel-local specification

Invariant: the physical state is authoritative ghost state

$$\exists C.\ \boxed{C\bullet} * \lfloor C \rfloor$$

Channel resource asserts ownership of corresponding fragment:

$$c \prec M \triangleq \boxed{[c \mapsto M]\circ}$$

$$\{c \prec M\}$$

$$\{\boxed{[c \mapsto M]\circ} * \boxed{C\bullet} * \lfloor C \rfloor\}$$

$\textbf{send}(c, m)$

$$\{\boxed{[c \mapsto M]\circ} * \boxed{C\bullet} * \lfloor \overbrace{C[c \mapsto C(c) \uplus \{m\}]}^{C'} \rfloor\}$$

$$\{\boxed{[c \mapsto M \uplus \{m\}]\circ} * \boxed{C'\bullet} * \lfloor C' \rfloor\}$$

$$\{c \prec M \uplus \{m\}\}$$

Deriving small-footprint specifications

- Channel monoid encodes small-footprint channel resources
- Invariant relates ghost and physical state using authoritative monoid to allow ownership of channel fragments

Recovering existing reasoning techniques

- We saw how to recover reasoning principles from Superficially Substructural Types and Fictional Separation

- One can also recover reasoning principles from CaReSL and iCAP through a encoding of STSs as monoids

Part 3
Logical atomicity

Logical atomicity

- In part 2 we used the invariant rule to access the shared physical resource

$$\frac{\{\triangleright R * P\}\ e\ \{\triangleright R * Q\}_{\mathcal{E}} \qquad e \text{ atomic}}{\left\{\boxed{R}^{\iota} * P\right\}\ e\ \left\{Q\right\}_{\mathcal{E} \uplus \{\iota\}}}$$

- This rule only applies to **atomic** expressions

- Iris allows us to extend this reasoning principle to **logically atomic** code

Logical atomicity

- In part 2 we used the invariant rule to access the shared physical resource

$$\frac{\{\triangleright R * P\}\; e\; \{\triangleright R * Q\}_{\mathcal{E}} \qquad e \text{ atomic}}{\left\{\boxed{R}^{\iota} * P\right\}\; e}$$

- This rule only applies to **ato**

We can **define** logically atomic triples

$$\langle P \rangle\; e\; \langle Q \rangle$$

- Iris allows us to extend this reasoning principle to **logically atomic** code

Logical atomicity

- **Example:** a blocking receive operation

$$\textbf{recv} \triangleq \textbf{rec}\ recv(c).\ \textbf{let}\ v = \textbf{tryrecv}(c)\ \textbf{in}$$
$$\textbf{case}\ v\ \textbf{of}\ \textbf{none} => recv(c) \mid \textbf{some}(m) => m$$

- Spins (without side effects) until a msg is received
- The linearisation point is the first successful **tryrecv**

Logical atomicity

- **Ideas**
  - Let clients reason about the state immediately before and after the linearisation point
  - Let clients open invariants **around** the linearisation point

Logical atomicity

- **Ideas**

  Parameterise our specifications with view shifts

  - Let clients reason about the state immediately before and after the linearisation point
  - Let clients open invariants **around** the linearisation point

  Let view shifts open and close invariants

Mask-changing view shifts

- Index view shifts with the set of invariants enabled before and after the view shift

$$P \;{}^{\mathcal{E}_1}\!\!\Rrightarrow^{\mathcal{E}_2}\; Q$$

- **Asserts**
    - that we can update the instrumented state from P to Q without changing the physical state
    - where the invariants in $\mathcal{E}_1$ are enabled before the view shift
    - and the invariants in $\mathcal{E}_2$ are enabled after the view shift

Mask-changing view shifts

- We can change the invariant mask around atomic expressions, provided we restore it again

$$\frac{e \text{ atomic} \qquad P \;{}^{\{\iota\}}\!\Rrightarrow^{\emptyset}\; P' \qquad \{P'\}\; e\; \{v.\, Q'\}_{\emptyset} \qquad \forall v.\, Q' \;{}^{\emptyset}\!\Rrightarrow^{\{\iota\}}\; Q}{\{P\}\; e\; \{v.\, Q\}_{\{\iota\}}}$$

- We can open and close invariants using view shifts

$$\boxed{P}^{\iota} \;{}^{\{\iota\}}\!\Rrightarrow^{\emptyset}\; \triangleright P \qquad\qquad \boxed{P}^{\iota} * \triangleright P \;{}^{\emptyset}\!\Rrightarrow^{\{\iota\}}\; \top$$

Logical atomicity

- **Idea:** Let clients open and close invariants around linearisation point and update instrumented state

$$
\begin{aligned}
\langle P \rangle\ e\ \langle Q \rangle_{\mathcal{E}} \approx \forall R_p, R_q, \mathcal{E}_R.\ & \mathcal{E} \cap \mathcal{E}_R = \emptyset\ \wedge \\
& (R_p \Lleftarrow\!\!\Rrightarrow^{-\mathcal{E}_R} P) \wedge (Q \Rrightarrow_{-\mathcal{E}_R} R_q) \\
& \Rightarrow\ \{R_p\}\ e\ \{R_q\}
\end{aligned}
$$

- This allows us to open invariants around logically atomic code

$$
\frac{\langle \triangleright R * P \rangle\ e\ \langle \triangleright R * Q \rangle_{\mathcal{E}}}{\left\langle \boxed{R}^{\iota} * P \right\rangle\ e\ \left\langle Q \right\rangle_{\mathcal{E} \uplus \{\iota\}}}
$$

Logical atomicity

- **Idea:** Let clients open and close invariants around linearisation point and update instrumented state

$$\langle P \rangle \; e \; \langle Q \rangle_{\mathcal{E}} \approx \forall R_p, R_q, \mathcal{E}_R.\; \mathcal{E} \cap \mathcal{E}_R = \emptyset \;\wedge$$
$$(R_p \Lleftarrow\!\!\Rrightarrow^{-\mathcal{E}_R} P) \wedge (Q \Rrightarrow_{-\mathcal{E}_R} R_q)$$
$$\Rightarrow \; \{R_p\} \; e \; \{R_q\}$$

- This allows us to open invariants around logically atomic code

$$\frac{\langle \triangleright R * P \rangle \; e \; \langle \triangleright R * Q \rangle_{\mathcal{E}}}{\left\langle \boxed{R}^{\iota} * P \right\rangle \; e \; \left\langle Q \right\rangle_{\mathcal{E} \uplus \{\iota\}}}$$

From the client's point of view it looks like we have access to the invariant R for the duration of e.

Logical atomicity

- **Idea:** Let clients open and c
linearisation point and update instru

$$\langle P\rangle\ e\ \langle Q\rangle_{\mathcal{E}} \approx \forall R_p, R_q, \mathcal{E}_R.\ \mathcal{E}\cap\mathcal{E}_R=\emptyset \wedge$$
$$(R_p \Lleftarrow\!\!\Rrightarrow^{-\mathcal{E}_R} P) \wedge (Q \Rrightarrow_{-\mathcal{E}_R} R_q)$$
$$\Rightarrow\ \{R_p\}\ e\ \{R_q\}$$

From the module's point of view we only access the invariant in the linearisation point.

- This allows us to open invariants around logically atomic code

$$\frac{\langle \triangleright R * P\rangle\ e\ \langle \triangleright R * Q\rangle_{\mathcal{E}}}{\left\langle \boxed{R}^{\iota} * P\right\rangle\ e\ \left\langle Q\right\rangle_{\mathcal{E}\uplus\{\iota\}}}$$

From the client's point of view it looks like we have access to the invariant R for the duration of e.

Case study

logically atomic

- elimination stack
- mutable references as channels
- message passing blocking receive

physically atomic

- small-footprint specifications
- λ-calculus with asynchronous message passing

Logical atomicity

- Logical atomicity is not built into Iris, but Iris is sufficiently expressive that we can **define** it in Iris.

Conclusions

- Iris is
  - simpler than previous logics
  - can encode reasoning principles from previous logics
  - and can do some fancy new stuff (logical atomicity)
- **Monoids and invariants are all you need**