Stripping multiple laters, one step at a time. Recovering intuitive specifications with the “step modality”!

Jonas Kastberg Hinrichsen, Aarhus University

May 23, 2023
Iris Workshop, MPI-SWS, Germany

Ghost theories and the later modality ($\triangleright$)

A significant strength of Iris is the ability to define higher-order abstract *ghost theories*

Ghost theories and the later modality ($\triangleright$)

A significant strength of Iris is the ability to define higher-order abstract *ghost theories*:

$$\textsc{HO-ghost-alloc} \quad \Rrightarrow \exists\gamma.\, \boxed{\bullet\, P}^{\gamma} * \boxed{\circ\, P}^{\gamma}$$

$$\textsc{HO-ghost-update} \quad \frac{\boxed{\bullet\, P}^{\gamma} \qquad \boxed{\circ\, Q}^{\gamma}}{\Rrightarrow \boxed{\bullet\, R}^{\gamma} * \boxed{\circ\, R}^{\gamma}}$$

$$\textsc{HO-ghost-agree} \quad \frac{\boxed{\bullet\, P}^{\gamma} \qquad \boxed{\circ\, Q}^{\gamma}}{\triangleright(P = Q)}$$

Ghost theories and the later modality (▷)

A significant strength of Iris is the ability to define higher-order abstract *ghost theories*:

$$\frac{\textsc{HO-ghost-alloc}}{\Rrightarrow \exists \gamma.\, \boxed{\bullet\, P}^{\gamma} * \boxed{\circ\, P}^{\gamma}} \qquad \textsc{HO-ghost-update}\ \frac{\boxed{\bullet\, P}^{\gamma} \quad \boxed{\circ\, Q}^{\gamma}}{\Rrightarrow \boxed{\bullet\, R}^{\gamma} * \boxed{\circ\, R}^{\gamma}} \qquad \textsc{HO-ghost-agree}\ \frac{\boxed{\bullet\, P}^{\gamma} \quad \boxed{\circ\, Q}^{\gamma}}{\triangleright(P = Q)}$$

The higher-order properties are facilitated via *step-indexing*, which incur laters (▷).

Ghost theories and the later modality ($\triangleright$)

A significant strength of Iris is the ability to define higher-order abstract *ghost theories*:

$$\textsc{HO-ghost-alloc}\quad \Rrightarrow \exists\gamma.\ \boxed{\bullet\, P}^{\gamma} * \boxed{\circ\, P}^{\gamma}$$

$$\textsc{HO-ghost-update}\quad \frac{\boxed{\bullet\, P}^{\gamma} \qquad \boxed{\circ\, Q}^{\gamma}}{\Rrightarrow \boxed{\bullet\, R}^{\gamma} * \boxed{\circ\, R}^{\gamma}}$$

$$\textsc{HO-ghost-agree}\quad \frac{\boxed{\bullet\, P}^{\gamma} \qquad \boxed{\circ\, Q}^{\gamma}}{\triangleright(P = Q)}$$

The higher-order properties are facilitated via *step-indexing*, which incur laters ($\triangleright$).
The laters ($\triangleright$) of higher-order ghost theories play a key role in regards to:

- **Expressivity:** What they can express and how they can be used in other proofs

Ghost theories and the later modality ($\triangleright$)

A significant strength of Iris is the ability to define higher-order abstract *ghost theories*:

$$\textsc{HO-ghost-alloc} \quad \dfrac{}{\Rrightarrow \exists \gamma.\, \boxed{\bullet\, P}^{\gamma} * \boxed{\circ\, P}^{\gamma}}$$

$$\textsc{HO-ghost-update} \quad \dfrac{\boxed{\bullet\, P}^{\gamma} \qquad \boxed{\circ\, Q}^{\gamma}}{\Rrightarrow \boxed{\bullet\, R}^{\gamma} * \boxed{\circ\, R}^{\gamma}}$$

$$\textsc{HO-ghost-agree} \quad \dfrac{\boxed{\bullet\, P}^{\gamma} \qquad \boxed{\circ\, Q}^{\gamma}}{\triangleright(P = Q)}$$

The higher-order properties are facilitated via *step-indexing*, which incur laters ($\triangleright$).
The laters ($\triangleright$) of higher-order ghost theories play a key role in regards to:

- **Expressivity:** What they can express and how they can be used in other proofs
- **Intuition:** How intuitive it is to use them and explain them to others

Ghost theories and the later modality ($\triangleright$)

A significant strength of Iris is the ability to define higher-order abstract *ghost theories*:

$$\textsc{HO-ghost-alloc} \quad \Rrightarrow \exists \gamma.\ \boxed{\bullet\, P}^{\gamma} * \boxed{\circ\, P}^{\gamma} \qquad \textsc{HO-ghost-update} \quad \frac{\boxed{\bullet\, P}^{\gamma} \qquad \boxed{\circ\, Q}^{\gamma}}{\Rrightarrow \boxed{\bullet\, R}^{\gamma} * \boxed{\circ\, R}^{\gamma}} \qquad \textsc{HO-ghost-agree} \quad \frac{\boxed{\bullet\, P}^{\gamma} \qquad \boxed{\circ\, Q}^{\gamma}}{\triangleright(P = Q)}$$

The higher-order properties are facilitated via *step-indexing*, which incur laters ($\triangleright$).
The laters ($\triangleright$) of higher-order ghost theories play a key role in regards to:

- **Expressivity:** What they can express and how they can be used in other proofs
- **Intuition:** How intuitive it is to use them and explain them to others

Recent developments of Iris extend later-based expressivity of ghost theories, but the intuition of using them arguably lacks behind.

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\Rrightarrow$) incur a later ($\triangleright$) per transition.

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

$\boxed{P_1}^{\gamma}$

e_1

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$\boxed{P_1}^{\gamma} \quad \Rrightarrow \quad \boxed{P_2}^{\gamma}$$

e_1

Logical Steps ($\triangleright$) vs Program Steps ($\leadsto$)

Traditionally, higher-order ghost theory transitions ($\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\leadsto$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$\boxed{P_1}^{\gamma} \quad \Rrightarrow \quad \boxed{P_2}^{\gamma}$$

$$\triangleright$$

$$e_1$$

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\mathrel{|\!\!\Rrightarrow}$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$
\begin{array}{ccc}
\boxed{P_1}^{\gamma} & \mathrel{|\!\!\Rrightarrow} & \boxed{P_2}^{\gamma} \\
 & \triangleright & \\
e_1 & \rightsquigarrow & e_2
\end{array}
$$

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\mid\!\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$
\begin{array}{ccccccc}
\boxed{P_1}^{\gamma} & \mid\!\Rrightarrow & \boxed{P_2}^{\gamma} & \mid\!\Rrightarrow & \cdots & \boxed{P_n}^{\gamma} \\
 & \triangleright & & \triangleright & \cdots & \\
e_1 & \rightsquigarrow & e_2 & \rightsquigarrow & \cdots & e_n
\end{array}
$$

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$\begin{array}{ccccccccc}
\boxed{P_1}^{\gamma} & \Rrightarrow & \boxed{P_2}^{\gamma} & \Rrightarrow & \cdots & \boxed{P_n}^{\gamma} & \Rrightarrow & \cdots \\
 & \triangleright & & \triangleright & \cdots & & \triangleright & \cdots \\
e_1 & \rightsquigarrow & e_2 & \rightsquigarrow & \cdots & e_n & \rightsquigarrow & \cdots
\end{array}$$

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\mathrel{|}\!\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$
\begin{array}{ccccccccc}
\boxed{P_1}^{\gamma} & \mathrel{|}\!\Rrightarrow & \boxed{P_2}^{\gamma} & \mathrel{|}\!\Rrightarrow & \cdots & \boxed{P_n}^{\gamma} & \mathrel{|}\!\Rrightarrow & \cdots \\
 & \triangleright & & \triangleright & \cdots & & \triangleright & \cdots \\
e_1 & \rightsquigarrow & e_2 & \rightsquigarrow & \cdots & e_n & \rightsquigarrow & \cdots
\end{array}
$$

Logical Steps ($\triangleright$) vs Program Steps ($\leadsto$)

Traditionally, higher-order ghost theory transitions ($\mid\!\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\leadsto$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$
\begin{array}{ccccccccc}
\boxed{P_1}^{\gamma} & \mid\!\Rrightarrow & \boxed{P_2}^{\gamma} & \mid\!\Rrightarrow & \cdots & \boxed{P_n}^{\gamma} & \mid\!\Rrightarrow & \cdots \\
 & \triangleright & & \triangleright & \cdots & & \triangleright & \cdots \\
e_1 & \leadsto & e_2 & \leadsto & \cdots & e_n & \leadsto & \cdots
\end{array}
$$

More complex higher-order ghost theories incur multiple laters per transition.

Logical Steps ($\triangleright$) vs Program Steps ($\leadsto$)

Traditionally, higher-order ghost theory transitions ($\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\leadsto$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$\begin{array}{ccccccccccc}
\boxed{P_1}^{\gamma} & \Rrightarrow & & & & \boxed{P_2}^{\gamma} & \Rrightarrow & \cdots & \boxed{P_n}^{\gamma} & \Rrightarrow & \cdots \\
& \triangleright & & \triangleright & \cdots & & \triangleright & \cdots & & \triangleright & \cdots \\
e_1 & \leadsto & e_2 & \leadsto & \cdots & e_n & \leadsto & \cdots & & &
\end{array}$$

More complex higher-order ghost theories incur multiple laters per transition.

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\mathrel{|}\!\!\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$
\begin{array}{ccccccccccc}
\boxed{P_1}^{\gamma} & \mathrel{|}\!\!\Rrightarrow & & & & \boxed{P_2}^{\gamma} & \mathrel{|}\!\!\Rrightarrow & \cdots & \boxed{P_n}^{\gamma} & \mathrel{|}\!\!\Rrightarrow & \cdots \\
& \triangleright & & \triangleright & \cdots & & \triangleright & \cdots & & \triangleright & \cdots \\
e_1 & \rightsquigarrow & e_2 & \rightsquigarrow & \cdots & e_n & \rightsquigarrow & \cdots & & &
\end{array}
$$

More complex higher-order ghost theories incur multiple laters per transition.
Recent developments allow stripping multiple laters per program step.

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$
\begin{array}{ccccccccccc}
\boxed{P_1}^{\gamma} & \Rrightarrow & & & & \boxed{P_2}^{\gamma} & \Rrightarrow & \cdots & \boxed{P_n}^{\gamma} & \Rrightarrow & \cdots \\
 & \triangleright & & \triangleright & \cdots & & \triangleright & \cdots & & \triangleright & \cdots \\
e_1 & \rightsquigarrow & & & & e_2 & \rightsquigarrow & \cdots & e_n & \rightsquigarrow & \cdots
\end{array}
$$

More complex higher-order ghost theories incur multiple laters per transition.
Recent developments allow stripping multiple laters per program step.

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\mid\!\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$\begin{array}{ccccccccccc}
\boxed{P_1}^{\gamma} & \mid\!\Rrightarrow & & & & \boxed{P_2}^{\gamma} & \mid\!\Rrightarrow & \cdots & \boxed{P_n}^{\gamma} & \mid\!\Rrightarrow & \cdots \\
& \triangleright & & \triangleright & \cdots & & \triangleright & \cdots & & \triangleright & \cdots \\
e_1 & \rightsquigarrow & & & & e_2 & \rightsquigarrow & \cdots & e_n & \rightsquigarrow & \cdots
\end{array}$$

More complex higher-order ghost theories incur multiple laters per transition.
Recent developments allow stripping multiple laters per program step.
While this recovers expressivity, intuition lacks behind.

Logical Steps ($\triangleright$) vs Program Steps ($\leadsto$)

Traditionally, higher-order ghost theory transitions ($\mid\!\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\leadsto$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$
\begin{array}{cccccccccc}
\boxed{P_1}^{\gamma} & \mid\!\Rrightarrow & & & & \boxed{P_2}^{\gamma} & \mid\!\Rrightarrow & \cdots & \boxed{P_n}^{\gamma} & \mid\!\Rrightarrow & \cdots \\
& \triangleright & & \triangleright & \cdots & & \triangleright & \cdots & & \triangleright & \cdots \\
e_1 & \leadsto & & & & e_2 & \leadsto & \cdots & e_n & \leadsto & \cdots
\end{array}
$$

More complex higher-order ghost theories incur multiple laters per transition.
Recent developments allow stripping multiple laters per program step.
While this recovers expressivity, intuition lacks behind.
Key Idea: Recover intuition of taking a physical step with a new **step modality**: $\mid\!\leadsto$

Logical Steps ($\triangleright$) vs Program Steps ($\leadsto$)

Traditionally, higher-order ghost theory transitions ($|\!\!\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\leadsto$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$
\begin{array}{cccccccc}
\boxed{P_1}^{\gamma} & |\!\!\Rrightarrow & \boxed{P_2}^{\gamma} & |\!\!\Rrightarrow & \cdots & \boxed{P_n}^{\gamma} & |\!\!\Rrightarrow & \cdots \\
 & |\!\!\leadsto & & |\!\!\leadsto & \cdots & & |\!\!\leadsto & \cdots \\
e_1 & \leadsto & e_2 & \leadsto & \cdots & e_n & \leadsto & \cdots
\end{array}
$$

More complex higher-order ghost theories incur multiple laters per transition.
Recent developments allow stripping multiple laters per program step.
While this recovers expressivity, intuition lacks behind.
Key Idea: Recover intuition of taking a physical step with a new **step modality**: $|\!\!\leadsto$

Logical Steps ($\triangleright$) vs Program Steps ($\rightsquigarrow$)

Traditionally, higher-order ghost theory transitions ($\mid\!\Rrightarrow$) incur a later ($\triangleright$) per transition.
We can strip one later ($\triangleright$) per program step ($\rightsquigarrow$).
Applying a higher-order ghost theory once per step is historically intuitive.

$$
\begin{array}{ccccccccc}
\boxed{P_1}^{\gamma} & \mid\!\Rrightarrow & \boxed{P_2}^{\gamma} & \mid\!\Rrightarrow & \cdots & \boxed{P_n}^{\gamma} & \mid\!\Rrightarrow & \cdots \\
 & \mid\!\rightsquigarrow & & \mid\!\rightsquigarrow & \cdots & & \mid\!\rightsquigarrow & \\
e_1 & \rightsquigarrow & e_2 & \rightsquigarrow & \cdots & e_n & \rightsquigarrow & \cdots
\end{array}
$$

More complex higher-order ghost theories incur multiple laters per transition.
Recent developments allow stripping multiple laters per program step.
While this recovers expressivity, intuition lacks behind.
Key Idea: Recover intuition of taking a physical step with a new **step modality**: $\mid\!\rightsquigarrow$

Contributions and outline of this talk

Recalling the ongoing story of adding and resolving more laters in ghost theories

- From the perspective of my work on the **Actris Ghost Theory**

Contributions and outline of this talk

Recalling the ongoing story of adding and resolving more laters in ghost theories

- From the perspective of my work on the **Actris Ghost Theory**

Introducing the **step modality**: $\mid\!\!\leadsto P$

- A scalable means of leveraging the ever-improving *expressivity* of later-stripping mechanisms, alongside *intuitive* specification and proof patterns

Contributions and outline of this talk

Recalling the ongoing story of adding and resolving more laters in ghost theories

- From the perspective of my work on the **Actris Ghost Theory**

Introducing the **step modality**: $|\!\!\leadsto P$

- A scalable means of leveraging the ever-improving *expressivity* of later-stripping mechanisms, alongside *intuitive* specification and proof patterns

Introducing the **Session Escrow Pattern**

- A specification pattern built derived via the Actris Ghost Theory for distributed systems, which was virtually inexpressible without the step modality

Contributions and outline of this talk

Recalling the ongoing story of adding and resolving more laters in ghost theories

- From the perspective of my work on the **Actris Ghost Theory**

Introducing the **step modality**: $\mid\!\!\rightsquigarrow P$

- A scalable means of leveraging the ever-improving *expressivity* of later-stripping mechanisms, alongside *intuitive* specification and proof patterns

Introducing the **Session Escrow Pattern**

- A specification pattern built derived via the Actris Ghost Theory for distributed systems, which was virtually inexpressible without the step modality

Everything in the talk is mechanised as a *shallow embedding* on top of Iris

A story told in multiple steps

The first step

$$\begin{array}{ccc} \boxed{P_1}^{\gamma} & \Mapsto & \boxed{P_2}^{\gamma} \\ & \rhd & \\ e_1 & \rightsquigarrow & e_2 \end{array}$$

The Actris (1.0) Ghost Theory [POPL'20]

A language-agnostic higher-order specification pattern for reasoning about reliable resource transfer between two participants

The Actris (1.0) Ghost Theory [POPL'20]

A language-agnostic higher-order specification pattern for reasoning about reliable resource transfer between two participants:

$$P, Q ::= \ \ldots \mid \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \ \vec{v_2} \mid \mathsf{prot\_own_l}\ \chi\ \mathit{prot} \mid \mathsf{prot\_own_r}\ \chi\ \mathit{prot} \mid \ldots$$

The Actris (1.0) Ghost Theory [POPL'20]

A language-agnostic higher-order specification pattern for reasoning about reliable resource transfer between two participants:

$$P, Q ::= \ \ldots \mid \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} \mid \mathsf{prot\_own_l}\ \chi\ \mathit{prot} \mid \mathsf{prot\_own_r}\ \chi\ \mathit{prot} \mid \ldots$$

$$\mathit{prot} ::= \mathbf{!}\,\vec{x}{:}\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot} \mid \mathbf{?}\vec{x}{:}\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot} \mid \mathbf{end}$$

The Actris (1.0) Ghost Theory [POPL'20]

A language-agnostic higher-order specification pattern for reasoning about reliable resource transfer between two participants:

$$P, Q ::= \ \ldots \mid \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \ \vec{v_2} \mid \mathsf{prot\_own_l}\ \chi\ \mathit{prot} \mid \mathsf{prot\_own_r}\ \chi\ \mathit{prot} \mid \ldots$$

$$\mathit{prot} ::= \mathbf{!}\,\vec{x}{:}\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot} \mid \mathbf{?}\vec{x}{:}\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot} \mid \mathbf{end}$$

$$\dfrac{\textsc{pre-proto-send-l} \\ \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \ \vec{v_2} \qquad \mathsf{prot\_own_l}\ \chi\ (\mathbf{!}\,\vec{x}{:}\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow \triangleright \mathsf{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own_l}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}$$

The Actris (1.0) Ghost Theory [POPL'20]

A language-agnostic higher-order specification pattern for reasoning about reliable resource transfer between two participants:

$$P, Q ::= \ \ldots \mid \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \ \vec{v_2} \mid \mathsf{prot\_own_l}\ \chi\ \mathit{prot} \mid \mathsf{prot\_own_r}\ \chi\ \mathit{prot} \mid \ldots$$

$$\mathit{prot} ::= \mathbf{!}\,\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot} \mid \mathbf{?}\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot} \mid \mathbf{end}$$

$$\frac{\mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \ \vec{v_2} \qquad \mathsf{prot\_own_l}\ \chi\ (\mathbf{!}\,\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow \triangleright \mathsf{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own_l}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}\ \textsc{pre-proto-send-l}$$

Expressivity: Can be applied once every step (as it incurs one later)

The Actris (1.0) Ghost Theory [POPL'20]

A language-agnostic higher-order specification pattern for reasoning about reliable resource transfer between two participants:

$$P, Q ::= \ \ldots \mid \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \ \vec{v_2} \mid \mathsf{prot\_own_l}\ \chi\ \mathit{prot} \mid \mathsf{prot\_own_r}\ \chi\ \mathit{prot} \mid \ldots$$

$$\mathit{prot} ::= \mathbf{!}\,\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot} \mid \mathbf{?}\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot} \mid \mathbf{end}$$

$$\dfrac{\text{PRE-PROTO-SEND-L} \\ \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \ \vec{v_2} \qquad \mathsf{prot\_own_l}\ \chi\ (\mathbf{!}\,\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow \triangleright \mathsf{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own_l}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}$$

Expressivity: Can be applied once every step (as it incurs one later)
Intuition: How do we explain the use of this pattern?

“Dont worry, just take a step”

“Dont worry, just take a step”

We can resolve a later every step

$$\frac{\{P\}\, e\, \{w.\, Q\}}{\{P * \triangleright R\}\, e\, \{w.\, Q * R\}}\ \textsc{Ht-later-frame}$$

“Dont worry, just take a step”

We can resolve a later every step, and we can always update ghost state

$$\frac{\{P\}\, e\, \{w.\, Q\}}{\{P * \triangleright R\}\, e\, \{w.\, Q * R\}}\ \textsc{Ht-later-frame}$$

$$\frac{P \Rrightarrow P' \qquad \{P'\}\, e\, \{w.\, Q'\} \qquad \forall w.\, Q' \Rrightarrow Q}{\{P\}\, e\, \{w.\, Q\}}\ \textsc{Ht-csq-vs}$$

"Dont worry, just take a step"

We can resolve a later every step, and we can always update ghost state

$$\textsc{Ht-later-frame}\quad \frac{\{P\}\, e\, \{w.\, Q\}}{\{P * \triangleright R\}\, e\, \{w.\, Q * R\}}$$

$$\textsc{Ht-csq-vs}\quad \frac{P \Rrightarrow P' \qquad \{P'\}\, e\, \{w.\, Q'\} \qquad \forall w.\, Q' \Rrightarrow Q}{\{P\}\, e\, \{w.\, Q\}}$$

A historically accepted intuition for step-indexed logics

But what if there are multiple laters?

$$
\begin{array}{cccccc}
\boxed{P_1}^{\gamma} & \mathrel{|}\!\!\Rrightarrow & & & & \boxed{P_2}^{\gamma} \\
& \triangleright & & \triangleright & \cdots & \\
e_1 & \rightsquigarrow & e_2 & \rightsquigarrow & \cdots & e_n
\end{array}
$$

Actris (2.0) Ghost Theory [LMCS'22]

An extension of Actris 1.0, that imposed an iterated number of laters in the send rule, relative to inbound buffer $\vec{v_2}$:

$$\frac{\textsc{proto-send-l} \quad \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (!\,\vec{x}:\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow \triangleright^{|\vec{v_2}|}\mathsf{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}$$

Actris (2.0) Ghost Theory [LMCS'22]

An extension of Actris 1.0, that imposed an iterated number of laters in the send rule, relative to inbound buffer $\vec{v_2}$:

$$\frac{\text{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \text{prot\_own}_\text{l}\ \chi\ (!\,\vec{x}:\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow \triangleright^{|\vec{v_2}|}\text{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \text{prot\_own}_\text{l}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}\ \text{PROTO-SEND-L}$$

Expressivity: Limited to sequentual programs and coarse-grained concurrency, as each ghost theory transition incurs multiple laters

Actris (2.0) Ghost Theory [LMCS'22]

An extension of Actris 1.0, that imposed an iterated number of laters in the send rule, relative to inbound buffer $\vec{v_2}$:

$$\frac{\textsc{proto-send-l} \qquad \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (\mathbf{!}\,\vec{x}:\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow \triangleright^{|\vec{v_2}|}\mathsf{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}$$

Expressivity: Limited to sequentual programs and coarse-grained concurrency, as each ghost theory transition incurs multiple laters
Intuition: How do we present this use???

One step at a time..

The need for skip instructions. The rules PROTO-SEND-L and PROTO-SEND-R from Figure 20 contain a number of later modalities ($\triangleright$) proportional to the other endpoint's buffer. As explained in § 9.3 these later modalities are the consequence of having to perform a number of case analyses on the subprotocol relation, which is defined using guarded recursion, and thus contains a later modality for each recursive unfolding.

To eliminate these later modalities, we instrument the code of the **`send`** function with the **`skipN`** (**`llength`** r) instruction, which performs a number of skips equal to the size of the other endpoint's buffer r. The **`skipN`** instruction has the following specification:

$$\{\triangleright^n P\}\,\texttt{skipN}\;n\,\{P\}$$

Instrumentation with skip instructions is used often in work on step-indexing, see *e.g.*, [SSB16; GST+20]. Instrumentation is needed because current step-indexed logics like Iris unify physical/program steps and logical steps, *i.e.*, for each physical/program step at most one later can be eliminated from the hypotheses. In recent work by Svendsen *et al.* [SSB16], Matsushita and Jourdan [MJ20], and Spies *et al.* [SGG+21] more liberal versions of step-indexing have been proposed, but none of these versions of step-indexing have been integrated into the main Coq development of Iris and HeapLang.

But what if you can only take one step?

$$\begin{array}{cccccc} \boxed{P_1}^{\gamma} & \Rrightarrow & & & & \boxed{P_2}^{\gamma} \\ & \triangleright & & \triangleright & \cdots & \\ e_1 & \rightsquigarrow & & & & e_2 \end{array}$$

But what if you can only take one step?

$$\begin{array}{cccccc} \boxed{P_1}^{\gamma} & \Rrightarrow & & & & \boxed{P_2}^{\gamma} \\ & \triangleright & & \triangleright & \cdots & \\ e_1 & \rightsquigarrow & & & & e_2 \end{array}$$

E.g. if you have to put your ghost state in an invariant:

$$\boxed{\exists \vec{v_1}, \vec{v_2}.\ \mathsf{prot\_ctx}\ \chi\ \ \vec{v_1}\ \ \vec{v_2} * \ldots}$$

But what if you can only take one step?

$$\begin{array}{llll} \boxed{P_1}^{\gamma} & \Rrightarrow & & \boxed{P_2}^{\gamma} \\ & \triangleright & \triangleright \quad \cdots & \\ e_1 & \rightsquigarrow & & e_2 \end{array}$$

E.g. if you have to put your ghost state in an invariant:

$$\boxed{\exists \vec{v_1}, \vec{v_2}.\ \mathsf{prot\_ctx}\ \chi\ \ \vec{v_1}\ \ \vec{v_2} * \ldots}$$

Invariants impose that we *must* strip all laters during a single step.

But what if you can only take one step?

$$\boxed{P_1}^{\gamma} \Rrightarrow \qquad \triangleright \qquad \triangleright \quad \cdots \qquad \boxed{P_2}^{\gamma}$$

$$e_1 \rightsquigarrow \qquad e_2$$

E.g. if you have to put your ghost state in an invariant:

$$\boxed{\exists \vec{v_1}, \vec{v_2}.\ \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots}$$

Invariants impose that we *must* strip all laters during a single step.

NB: Invariant mask details are omitted in this talk.

Stripping multiple laters during a single physical step

[Matsushita et. al, 2022] extends Iris to allow stripping laters corresponding to total number of steps taken at every step, tracked via *time receipts* ($⧖\ n$)

Stripping multiple laters during a single physical step

[Matsushita et. al, 2022] extends Iris to allow stripping laters corresponding to total number of steps taken at every step, tracked via *time receipts* ($⧖\ n$):

$$\textsc{Ht-time-get}\quad \frac{\{P * ⧖\ 0\}\, e\, \{\Phi\}}{\{P\}\, e\, \{\Phi\}} \qquad \textsc{Ht-time-incr}\quad \frac{\{P\}\, e\, \{w.\, Q\}}{\{P * ⧖\ n\}\, e\, \{w.\, Q * ⧖\ (n+1)\}} \qquad \textsc{Ht-time-frame'}\quad \frac{\{P\}\, e\, \{w.\, Q\}}{\left\{P * ⧖\ n * \triangleright^{(n+1)} R\right\}\, e\, \{w.\, Q * R\}}$$

Stripping multiple laters during a single physical step

[Matsushita et. al, 2022] extends Iris to allow stripping laters corresponding to total number of steps taken at every step, tracked via *time receipts* ($⧗\, n$):

$$\textsc{Ht-time-get}\quad \frac{\{P * ⧗\, 0\}\, e\, \{\Phi\}}{\{P\}\, e\, \{\Phi\}}$$

$$\textsc{Ht-time-incr}\quad \frac{\{P\}\, e\, \{w.\, Q\}}{\{P * ⧗\, n\}\, e\, \{w.\, Q * ⧗\, (n+1)\}}$$

$$\textsc{Ht-time-frame}\quad \frac{\{P\}\, e\, \{w.\, Q\}}{\left\{P * ⧗\, n * \models\!\!\Rrightarrow^{(n+1)} R\right\}\, e\, \{w.\, Q * R\}}$$

Stripping multiple laters during a single physical step

[Matsushita et. al, 2022] extends Iris to allow stripping laters corresponding to total number of steps taken at every step, tracked via *time receipts* ($\text{⧗}\ n$):

$$\textsc{Ht-time-get}\ \frac{\{P * \text{⧗}\ 0\}\ e\ \{\Phi\}}{\{P\}\ e\ \{\Phi\}} \qquad \textsc{Ht-time-incr}\ \frac{\{P\}\ e\ \{w.\ Q\}}{\{P * \text{⧗}\ n\}\ e\ \{w.\ Q * \text{⧗}\ (n+1)\}} \qquad \textsc{Ht-time-frame}\ \frac{\{P\}\ e\ \{w.\ Q\}}{\left\{P * \text{⧗}\ n * \models\!\!\!\Rrightarrow^{(n+1)} R\right\}\ e\ \{w.\ Q * R\}}$$

We can then track the relevant step count lower bounds in our invariant:

$$\mathtt{prot\_ctx}_{\text{⧗}}\ \chi\ \vec{v_1}\ \vec{v_2} \triangleq \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} * \text{⧗}\ |\vec{v_1}| * \text{⧗}\ |\vec{v_2}| \qquad \boxed{\exists \vec{v_1}, \vec{v_2}.\ \mathtt{prot\_ctx}_{\text{⧗}}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots}$$

Stripping multiple laters during a single physical step

[Matsushita et. al, 2022] extends Iris to allow stripping laters corresponding to total number of steps taken at every step, tracked via *time receipts* ($⧗\ n$):

$$\textsc{Ht-time-get}\ \frac{\{P * ⧗\ 0\}\ e\ \{\Phi\}}{\{P\}\ e\ \{\Phi\}} \qquad \textsc{Ht-time-incr}\ \frac{\{P\}\ e\ \{w.\ Q\}}{\{P * ⧗\ n\}\ e\ \{w.\ Q * ⧗\ (n+1)\}} \qquad \textsc{Ht-time-frame}\ \frac{\{P\}\ e\ \{w.\ Q\}}{\left\{P * ⧗\ n * |\!\Rightarrow\!\triangleright^{(n+1)} R\right\}\ e\ \{w.\ Q * R\}}$$

We can then track the relevant step count lower bounds in our invariant:

$$\mathtt{prot\_ctx}_{⧗}\ \chi\ \vec{v_1}\ \vec{v_2} \triangleq \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} * ⧗\ |\vec{v_1}| * ⧗\ |\vec{v_2}| \qquad \boxed{\exists \vec{v_1}, \vec{v_2}.\ \mathtt{prot\_ctx}_{⧗}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots}$$

Expressivity: Can use the send rule at every step, as we can strip all the necessary laters using Ht-step-frame, and update the step count lower bounds in tandem with our ghost state using Ht-step-incr!

Stripping multiple laters during a single physical step

[Matsushita et. al, 2022] extends Iris to allow stripping laters corresponding to total number of steps taken at every step, tracked via *time receipts* ($⧖\ n$):

$$\text{HT-TIME-GET}\quad \frac{\{P * ⧖\ 0\}\ e\ \{\Phi\}}{\{P\}\ e\ \{\Phi\}} \qquad \text{HT-TIME-INCR}\quad \frac{\{P\}\ e\ \{w.\ Q\}}{\{P * ⧖\ n\}\ e\ \{w.\ Q * ⧖\ (n+1)\}} \qquad \text{HT-TIME-FRAME}\quad \frac{\{P\}\ e\ \{w.\ Q\}}{\left\{P * ⧖\ n * \Rrightarrow^{(n+1)} R\right\}\ e\ \{w.\ Q * R\}}$$

We can then track the relevant step count lower bounds in our invariant:

$$\texttt{prot\_ctx}_{⧖}\ \chi\ \vec{v_1}\ \ \vec{v_2} \triangleq \text{prot\_ctx}\ \chi\ \vec{v_1}\ \ \vec{v_2} * ⧖\ |\vec{v_1}| * ⧖\ |\vec{v_2}| \qquad \boxed{\exists \vec{v_1}, \vec{v_2}.\ \texttt{prot\_ctx}_{⧖}\ \chi\ \vec{v_1}\ \ \vec{v_2} * \ldots}$$

Expressivity: Can use the send rule at every step, as we can strip all the necessary laters using Ht-step-frame, and update the step count lower bounds in tandem with our ghost state using Ht-step-incr!
Intuition: ...?

Well..

$$\frac{\{P * ⧖\, 0\}\, \langle ip;\, e\rangle\, \{\Phi\}}{\{P\}\, \langle ip;\, e\rangle\, \{\Phi\}}\ \textsc{Ht-step-get} \qquad \frac{\{P\}\, \langle ip;\, e\rangle\, \{w.\, Q\}}{\{P * ⧖\, n\}\, \langle ip;\, e\rangle\, \{w.\, Q * ⧖\, n+1\}}\ \textsc{Ht-step-incr}$$

$$\frac{\{P\}\, \langle ip;\, e\rangle\, \{w.\, Q\}}{\{P * ⧖\, n * \triangleright^n R\}\, \langle ip;\, e\rangle\, \{w.\, Q * R\}}\ \textsc{Ht-step-frame} \qquad \frac{⧖\, n}{⧖\, n * ⧖\, n}\ \textsc{step-dup}$$

Fig. 12. The mechanism for stripping multiple laters. We require e to be an atomic expression.

The shared logical context can then be captured as the following Iris invariant:

$$\begin{aligned}\exists Tl, Tr, Rl, Rr.\ &\mathsf{auth\_list}\ \chi_{\mathsf{Tl}}\ Tl * \mathsf{auth\_list}\ \chi_{\mathsf{Tr}}\ Tr * \mathsf{auth\_list}\ \chi_{\mathsf{Rl}}\ Rl * \mathsf{auth\_list}\ \chi_{\mathsf{Rr}}\ Rr * \\ &\mathsf{prot\_ctx}\ \chi_{\mathsf{chan}}\ (Tl - Rr)\ (Tr - Rl) * Rr \leq_{\mathsf{p}} Tl * Rl \leq_{\mathsf{p}} Tr * ⧖\, |Tl| * ⧖\, |Tr|\end{aligned}$$

Stripping multiple laters. In Iris, and thus Aneris, one can strip a later whenever a step of computation is taken. Conventionally the intuition is that one step equates stripping one later. However, recent discoveries [Matsushita et al. 2022; Mével et al. 2019; Spies et al. 2022] uncovered various methods for stripping *mutliple* laters per step. Based on these discoveries we extended Aneris with a similar, albeit more simplistic, mechanism as presented in Figure 12. The mechanism lets us strip multiple laters during one physical step, based on the amount of steps that has been taken thus far. The rule Ht-step-get lets us track a new lower bound of steps taken thus far $⧖\, 0$, and Ht-step-incr allows us to increase it by one, every time a step is taken. Crucially, the rule Ht-step-frame lets us frame resources under an amount of laters corresponding to the lower bound

So what is the problem?

The ghost theory stopped being a self-contained pattern

So what is the problem?

The ghost theory stopped being a self-contained pattern

- We *can* soundly apply the ghost theory once every step

So what is the problem?

The ghost theory stopped being a self-contained pattern

- We *can* soundly apply the ghost theory once every step
- But we need to manually apply the later-stripping mechanism

So what is the problem?

The ghost theory stopped being a self-contained pattern

- We *can* soundly apply the ghost theory once every step
- But we need to manually apply the later-stripping mechanism
  - To strip multiple laters

So what is the problem?

The ghost theory stopped being a self-contained pattern

- We *can* soundly apply the ghost theory once every step
- But we need to manually apply the later-stripping mechanism
  - To strip multiple laters
  - To keep the local step count lower bounds up to date

So what is the problem?

The ghost theory stopped being a self-contained pattern

- We *can* soundly apply the ghost theory once every step
- But we need to manually apply the later-stripping mechanism
  - To strip multiple laters
  - To keep the local step count lower bounds up to date

Key Idea: What if we can capture the idea of taking a step to abstract over the later-stripping mechanism?

So what is the problem?

The ghost theory stopped being a self-contained pattern

- We *can* soundly apply the ghost theory once every step
- But we need to manually apply the later-stripping mechanism
  - To strip multiple laters
  - To keep the local step count lower bounds up to date

Key Idea: What if we can capture the idea of taking a step to abstract over the later-stripping mechanism?

$$\begin{array}{ccccc} \boxed{P_1}^{\gamma} & \Rrightarrow & & & \boxed{P_2}^{\gamma} \\ & \triangleright & \triangleright & \cdots & \\ e_1 & \rightsquigarrow & & & e_2 \end{array}$$

So what is the problem?

The ghost theory stopped being a self-contained pattern

- We *can* soundly apply the ghost theory once every step
- But we need to manually apply the later-stripping mechanism
  - To strip multiple laters
  - To keep the local step count lower bounds up to date

Key Idea: What if we can capture the idea of taking a step to abstract over the later-stripping mechanism?

$$\begin{array}{ccc} \boxed{P_1}^{\gamma} & \Rrightarrow & \boxed{P_2}^{\gamma} \\ & \vdash\!\rightsquigarrow & \\ e_1 & \rightsquigarrow & e_2 \end{array}$$

Solution: Introducing the step modality!

$$\models\!\!\rightsquigarrow P$$

Solution: Introducing the step modality!

$$\models\!\!\rightsquigarrow P$$

Captures the semantics of *taking a program step*.

Solution: Introducing the step modality!

$$\mathrel{|}\!\!\leadsto P$$

Captures the semantics of *taking a program step*.
Recovers the intuition that we can get P *after taking a single step*:

$$\frac{\{P\}\, e\, \{w.\, Q\}}{\{P * \mathrel{|}\!\!\leadsto R\}\, e\, \{w.\, Q * R\}}\ \textsc{Ht-step-frame}$$

Solution: Introducing the step modality!

$$\models\!\!\rightsquigarrow P$$

Captures the semantics of *taking a program step*.
Recovers the intuition that we can get P *after taking a single step*:

$$\dfrac{\{P\}\, e\, \{w.\, Q\}}{\{P * {\models\!\!\rightsquigarrow} R\}\, e\, \{w.\, Q * R\}}\ \textsc{Ht-step-frame}$$

Intentionally mimics the semantics of the original later- and update-stripping rules:

$$\dfrac{\{P\}\, e\, \{w.\, Q\}}{\{P * \triangleright R\}\, e\, \{w.\, Q * R\}}\ \textsc{Ht-later-frame} \qquad \dfrac{P \Rrightarrow P' \qquad \{P'\}\, e\, \{w.\, Q'\} \qquad \forall w.\, Q' \Rrightarrow Q}{\{P\}\, e\, \{w.\, Q\}}\ \textsc{Ht-csq-vs}$$

Step-based Actris Ghost Theory

Instead of ghost theories that expose verification details (such as laters):

$$\dfrac{\textsc{proto-send-l}\qquad\qquad\qquad\qquad\qquad\qquad}{\mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \mathsf{prot\_own_l}\ \chi\ (\mathbf{!}\,\vec{x}:\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}$$

$$\overline{\models\!\!\!\Rrightarrow \triangleright^{|\vec{v_2}|}\,\mathsf{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own_l}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}$$

Step-based Actris Ghost Theory

Instead of ghost theories that expose verification details (such as laters):

$$\frac{\text{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \text{prot\_own}_{\mathsf{l}}\ \chi\ (\mathbf{!}\,\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow \triangleright^{|\vec{v_2}|}\ \text{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \text{prot\_own}_{\mathsf{l}}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}\ \textsc{proto-send-l}$$

We can now have high-level patterns that hide such details:

$$\frac{\triangleright\,\texttt{prot\_ctx}_{\rlap{\scriptscriptstyle\Sigma}}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \triangleright\,\text{prot\_own}_{\mathsf{l}}\ \chi\ (\mathbf{!}\,\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad \triangleright\,P[\vec{t}/\vec{x}])}{\mid\!\!\leadsto \texttt{prot\_ctx}_{\rlap{\scriptscriptstyle\Sigma}}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \text{prot\_own}_{\mathsf{l}}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}\ \textsc{Step-proto-send-l}$$

Step-based Actris Ghost Theory

Instead of ghost theories that expose verification details (such as laters):

$$\frac{\text{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \text{prot\_own}_{\mathsf{l}}\ \chi\ (!\,\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow \triangleright^{|\vec{v_2}|}\ \text{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \text{prot\_own}_{\mathsf{l}}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}\ \textsc{proto-send-l}$$

We can now have high-level patterns that hide such details:

$$\frac{\triangleright\,\texttt{prot\_ctx}_{\mathbb{X}}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \triangleright\,\text{prot\_own}_{\mathsf{l}}\ \chi\ (!\,\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad \triangleright\,P[\vec{t}/\vec{x}])}{\models\!\!\leadsto \texttt{prot\_ctx}_{\mathbb{X}}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \text{prot\_own}_{\mathsf{l}}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}\ \textsc{Step-proto-send-l}$$

Expressivity: Can be applied at every step!

Step-based Actris Ghost Theory

Instead of ghost theories that expose verification details (such as laters):

$$\frac{\text{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \text{prot\_own}_{\mathsf{l}}\ \chi\ (\mathbf{!}\,\vec{x}{:}\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow \triangleright^{|\vec{v_2}|}\ \text{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \text{prot\_own}_{\mathsf{l}}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}\ \textsc{proto-send-l}$$

We can now have high-level patterns that hide such details:

$$\frac{\triangleright\,\texttt{prot\_ctx}_{\mathbb{X}}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \triangleright\,\text{prot\_own}_{\mathsf{l}}\ \chi\ (\mathbf{!}\,\vec{x}{:}\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad \triangleright\,P[\vec{t}/\vec{x}])}{\models\!\!\rightsquigarrow \texttt{prot\_ctx}_{\mathbb{X}}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \text{prot\_own}_{\mathsf{l}}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}\ \textsc{Step-proto-send-l}$$

Expressivity: Can be applied at every step!
Intuition: How do we explain this specification to newcomers (and reviewers)?

“Dont worry, just take a step”

The Session Escrow Pattern [ICFP'23] (Conditionally Accepted)

Derived abstractions may hide the number of laters that needs to be stripped

The Session Escrow Pattern [ICFP'23] (Conditionally Accepted)

Derived abstractions may hide the number of laters that needs to be stripped:

$$P, Q ::= \ldots \mid \texttt{ses\_own}_{\texttt{l}}\ \chi\ n\ m\ \mathit{prot} \mid \texttt{ses\_own}_{\texttt{r}}\ \chi\ n\ m\ \mathit{prot} \mid \\ \texttt{ses\_idx}_{\texttt{l}}\ \chi\ i\ v \mid \texttt{ses\_idx}_{\texttt{r}}\ \chi\ i\ v \mid \ldots$$

The Session Escrow Pattern [ICFP'23] (Conditionally Accepted)

Derived abstractions may hide the number of laters that needs to be stripped:

$$P, Q ::= \ldots \mid \mathtt{ses\_own_l}\ \chi\ n\ m\ prot \mid \mathtt{ses\_own_r}\ \chi\ n\ m\ prot \mid \mathtt{ses\_idx_l}\ \chi\ i\ v \mid \mathtt{ses\_idx_r}\ \chi\ i\ v \mid \ldots$$

$$\mathtt{ses\_own_l}\ \chi\ n\ m\ prot \triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \mathtt{prot\_ctx}_{\chi}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots$$
$$\mathtt{ses\_own_r}\ \chi\ n\ m\ prot \triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \mathtt{prot\_ctx}_{\chi}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots$$

The Session Escrow Pattern [ICFP'23] (Conditionally Accepted)

Derived abstractions may hide the number of laters that needs to be stripped:

$$P, Q ::= \ldots \mid \texttt{ses\_own}_{\texttt{l}}\ \chi\ n\ m\ \mathit{prot} \mid \texttt{ses\_own}_{\texttt{r}}\ \chi\ n\ m\ \mathit{prot} \mid \\ \texttt{ses\_idx}_{\texttt{l}}\ \chi\ i\ v \mid \texttt{ses\_idx}_{\texttt{r}}\ \chi\ i\ v \mid \ldots$$

$$\texttt{ses\_own}_{\texttt{l}}\ \chi\ n\ m\ \mathit{prot} \triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \texttt{prot\_ctx}_{\texttt{x}}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots$$
$$\texttt{ses\_own}_{\texttt{r}}\ \chi\ n\ m\ \mathit{prot} \triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \texttt{prot\_ctx}_{\texttt{x}}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots$$

Ideal-session-escrow-send

$$\frac{\texttt{ses\_own}_{\texttt{l}}\ \chi\ n\ m\ (\mathbf{!}\,(\vec{x}:\vec{\tau})\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\texttt{ses\_own}_{\texttt{l}}\ \chi\ (n+1)\ m\ (\mathit{prot}[\vec{t}/\vec{x}]) * \texttt{ses\_idx}_{\texttt{l}}\ \chi\ n\ (v[\vec{t}/\vec{x}])}$$

The Session Escrow Pattern [ICFP'23] (Conditionally Accepted)

Derived abstractions may hide the number of laters that needs to be stripped:

$$\begin{array}{c} P, Q ::= \ldots \mid \mathtt{ses\_own_l}\ \chi\ n\ m\ \mathit{prot} \mid \mathtt{ses\_own_r}\ \chi\ n\ m\ \mathit{prot} \mid \\ \mathtt{ses\_idx_l}\ \chi\ i\ v \mid \mathtt{ses\_idx_r}\ \chi\ i\ v \mid \ldots \end{array}$$

$$\begin{aligned} \mathtt{ses\_own_l}\ \chi\ n\ m\ \mathit{prot} &\triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \mathtt{prot\_ctx}_{\text{⧖}}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots \\ \mathtt{ses\_own_r}\ \chi\ n\ m\ \mathit{prot} &\triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \mathtt{prot\_ctx}_{\text{⧖}}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots \end{aligned}$$

Naive-session-escrow-send

$$\frac{\mathtt{ses\_own_l}\ \chi\ n\ m\ (\mathbf{!}\,(\vec{x}\!:\!\vec{\tau})\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow \triangleright^{???}\ \mathtt{ses\_own_l}\ \chi\ (n+1)\ m\ (\mathit{prot}[\vec{t}/\vec{x}]) * \mathtt{ses\_idx_l}\ \chi\ n\ (v[\vec{t}/\vec{x}])}$$

The Session Escrow Pattern [ICFP'23] (Conditionally Accepted)

Derived abstractions may hide the number of laters that needs to be stripped:

$$P, Q ::= \ldots \mid \mathtt{ses\_own_l}\ \chi\ n\ m\ prot \mid \mathtt{ses\_own_r}\ \chi\ n\ m\ prot \mid \\ \mathtt{ses\_idx_l}\ \chi\ i\ v \mid \mathtt{ses\_idx_r}\ \chi\ i\ v \mid \ldots$$

$$\mathtt{ses\_own_l}\ \chi\ n\ m\ prot \triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \mathtt{prot\_ctx}_{\chi}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots$$
$$\mathtt{ses\_own_r}\ \chi\ n\ m\ prot \triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \mathtt{prot\_ctx}_{\chi}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots$$

SESSION-ESCROW-SEND-L

$$\frac{\mathtt{ses\_own_l}\ \chi\ n\ m\ (\mathbf{!}\,(\vec{x}:\vec{\tau})\,\langle v\rangle\{P\}.\, prot) \qquad P[\vec{t}/\vec{x}]}{\mid\!\rightsquigarrow \mathtt{ses\_own_l}\ \chi\ (n+1)\ m\ (prot[\vec{t}/\vec{x}]) * \mathtt{ses\_idx_l}\ \chi\ n\ (v[\vec{t}/\vec{x}])}$$

The Session Escrow Pattern [ICFP'23] (Conditionally Accepted)

Derived abstractions may hide the number of laters that needs to be stripped:

$$P, Q ::= \ldots \mid \texttt{ses\_own}_{\texttt{l}}\ \chi\ n\ m\ prot \mid \texttt{ses\_own}_{\texttt{r}}\ \chi\ n\ m\ prot \mid \\ \texttt{ses\_idx}_{\texttt{l}}\ \chi\ i\ v \mid \texttt{ses\_idx}_{\texttt{r}}\ \chi\ i\ v \mid \ldots$$

$$\texttt{ses\_own}_{\texttt{l}}\ \chi\ n\ m\ prot \triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \texttt{prot\_ctx}_{\chi}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots$$
$$\texttt{ses\_own}_{\texttt{r}}\ \chi\ n\ m\ prot \triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \texttt{prot\_ctx}_{\chi}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots$$

SESSION-ESCROW-SEND-L

$$\frac{\texttt{ses\_own}_{\texttt{l}}\ \chi\ n\ m\ (\mathbf{!}\,(\vec{x}:\vec{\tau})\,\langle v\rangle\{P\}.\,prot) \qquad P[\vec{t}/\vec{x}]}{\models\!\!\rightsquigarrow \texttt{ses\_own}_{\texttt{l}}\ \chi\ (n+1)\ m\ (prot[\vec{t}/\vec{x}]) * \texttt{ses\_idx}_{\texttt{l}}\ \chi\ n\ (v[\vec{t}/\vec{x}])}$$

This ghost theory is virtually *inexpressible* without the step modality

The Session Escrow Pattern [ICFP'23] (Conditionally Accepted)

Derived abstractions may hide the number of laters that needs to be stripped:

$$P, Q ::= \ldots \mid \texttt{ses\_own}_{\texttt{l}}\ \chi\ n\ m\ prot \mid \texttt{ses\_own}_{\texttt{r}}\ \chi\ n\ m\ prot \mid \texttt{ses\_idx}_{\texttt{l}}\ \chi\ i\ v \mid \texttt{ses\_idx}_{\texttt{r}}\ \chi\ i\ v \mid \ldots$$

$$\texttt{ses\_own}_{\texttt{l}}\ \chi\ n\ m\ prot \triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \texttt{prot\_ctx}_{\texttt{X}}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots$$

$$\texttt{ses\_own}_{\texttt{r}}\ \chi\ n\ m\ prot \triangleq \boxed{\exists \vec{v_1}, \vec{v_2}.\, \texttt{prot\_ctx}_{\texttt{X}}\ \chi\ \vec{v_1}\ \vec{v_2} * \ldots} * \ldots$$

$$\frac{\texttt{ses\_own}_{\texttt{l}}\ \chi\ n\ m\ (\mathbf{!}\,(\vec{x}:\vec{\tau})\,\langle v\rangle\{P\}.\, prot) \qquad P[\vec{t}/\vec{x}]}{\mid\!\leadsto \texttt{ses\_own}_{\texttt{l}}\ \chi\ (n+1)\ m\ (prot[\vec{t}/\vec{x}]) * \texttt{ses\_idx}_{\texttt{l}}\ \chi\ n\ (v[\vec{t}/\vec{x}])}\ \textsc{session-escrow-send-l}$$

This ghost theory is virtually *inexpressible* without the step modality

- Inexpressible without explicitly mentioning the later-stripping mechanism

So how do we derive step-based ghost theories?

Step modality proof interface

The step modality admits a step-based version of the time receipt mechanism:

$$\textsc{Step-time-get}\quad \dfrac{}{\models_{\leadsto} ⧖\, 0} \qquad \textsc{Step-time-incr}\quad \dfrac{⧖\, n}{\models_{\leadsto} ⧖\, (n+1)} \qquad \textsc{Step-time-frame}\quad \dfrac{⧖\, n \qquad \models\!\!\rhd\!\!\Rightarrow^{(n+1)} P}{\models_{\leadsto} P}$$

Step modality proof interface

The step modality admits a step-based version of the time receipt mechanism:

$$\textsc{Step-time-get}\quad \dfrac{}{\models\!\!\leadsto ⧖\, 0} \qquad \textsc{Step-time-incr}\quad \dfrac{⧖\, n}{\models\!\!\leadsto ⧖\, (n+1)} \qquad \textsc{Step-time-frame}\quad \dfrac{⧖\, n \qquad \models\!\!\!\Rrightarrow\!\!\triangleright\!\!\Rrightarrow^{(n+1)} P}{\models\!\!\leadsto P}$$

Additionally, the step modality enjoys a mix of the later and update modality rules:

$$\textsc{Step-intro}\quad \dfrac{P}{\models\!\!\leadsto P} \qquad \textsc{Step-mono}\quad \dfrac{P \vdash Q}{\models\!\!\leadsto P \vdash \models\!\!\leadsto Q} \qquad \textsc{Step-sep-comm}\quad \dfrac{\models\!\!\leadsto P \qquad \models\!\!\leadsto Q}{\models\!\!\leadsto P * Q}$$

$$\textsc{Step-upd}\quad \dfrac{\models\!\!\!\Rrightarrow \models\!\!\leadsto \models\!\!\!\Rrightarrow P}{\models\!\!\leadsto P} \qquad \textsc{Step-later}\quad \dfrac{\triangleright P}{\models\!\!\leadsto P}$$

Vertical modularity of the step modality

The step modality properties lets us derive step-based ghost theories on top of the time receipt later-stripping mechanism

Vertical modularity of the step modality

The step modality properties lets us derive step-based ghost theories on top of the time receipt later-stripping mechanism:

$$\dfrac{\overset{\textsc{Step-time-frame}}{⧗\, n \qquad \mathrel{|\!\!\Rrightarrow\!\!\triangleright}^{(n+1)} P}}{\mathrel{|\!\!\rightsquigarrow} P} \quad \wedge \quad \dfrac{\overset{\textsc{Step-time-incr}}{⧗\, n}}{\mathrel{|\!\!\rightsquigarrow} ⧗\,(n+1)}$$

Vertical modularity of the step modality

The step modality properties lets us derive step-based ghost theories on top of the time receipt later-stripping mechanism:

$$\frac{\textsc{Step-time-frame}\quad ⧖\, n \qquad \Rrightarrow\!\!\triangleright^{(n+1)} P}{\mid\!\leadsto P} \;\wedge\; \frac{\textsc{Step-time-incr}\quad ⧖\, n}{\mid\!\leadsto ⧖\,(n+1)} \;\wedge$$

$$\frac{\textsc{proto-send-l}\quad \mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (\mathbf{!}\,\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,prot) \qquad P[\vec{t}/\vec{x}]}{\Rrightarrow\!\triangleright^{|\vec{v_2}|}\ \mathsf{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (prot[\vec{t}/\vec{x}])}$$

Vertical modularity of the step modality

The step modality properties lets us derive step-based ghost theories on top of the time receipt later-stripping mechanism:

$$\frac{\overset{\textsc{Step-time-frame}}{\mathrm{⧖}\, n \qquad \Rrightarrow^{(n+1)} P}}{\mathrel{|\!\!\leadsto} P} \quad\wedge\quad \frac{\overset{\textsc{Step-time-incr}}{\mathrm{⧖}\, n}}{\mathrel{|\!\!\leadsto} \mathrm{⧖}\,(n+1)} \quad\wedge$$

$$\frac{\overset{\textsc{proto-send-l}}{\mathsf{prot\_ctx}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (\mathbf{!}\,\vec{x}:\vec{\tau}\,\langle v\rangle\{P\}.\, prot) \qquad P[\vec{t}/\vec{x}]}}{\Rrightarrow\triangleright^{|\vec{v_2}|}\,\mathsf{prot\_ctx}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (prot[\vec{t}/\vec{x}])} \quad\Rightarrow$$

$$\frac{\overset{\textsc{Step-proto-send-l}}{\triangleright\,\texttt{prot\_ctx}_{⧖}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \triangleright\,\mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (\mathbf{!}\,\vec{x}:\vec{\tau}\,\langle v\rangle\{P\}.\, prot) \qquad \triangleright\, P[\vec{t}/\vec{x}])}}{\mathrel{|\!\!\leadsto} \texttt{prot\_ctx}_{⧖}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (prot[\vec{t}/\vec{x}])}$$

Vertical modularity of the step modality

. . . and even each other!

Vertical modularity of the step modality

... and even each other!:

$$\frac{\textsc{Step-proto-send-l}\quad \triangleright\, \mathtt{prot\_ctx}_{\chi}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \triangleright\, \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (\mathbf{!}\,\vec{x}\!:\!\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad \triangleright\, P[\vec{t}/\vec{x}]}{\mid\!\rightsquigarrow \mathtt{prot\_ctx}_{\chi}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])}$$

Vertical modularity of the step modality

... and even each other!:

$$\dfrac{\textsc{Step-proto-send-l}\qquad \triangleright\,\texttt{prot\_ctx}_{\chi}\ \chi\ \vec{v_1}\ \vec{v_2} \qquad \triangleright\,\mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (!\,\vec{x}{:}\vec{\tau}\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad \triangleright\,P[\vec{t}/\vec{x}])}{\mid\!\rightsquigarrow \texttt{prot\_ctx}_{\chi}\ \chi\ (\vec{v_1}\cdot[v[\vec{t}/\vec{x}]])\ \vec{v_2} * \mathsf{prot\_own}_{\mathsf{l}}\ \chi\ (\mathit{prot}[\vec{t}/\vec{x}])} \quad \Rightarrow$$

$$\dfrac{\textsc{session-escrow-send}\qquad \texttt{ses\_own}\ \chi\ s\ n\ m\ (!\,(\vec{x}{:}\vec{\tau})\,\langle v\rangle\{P\}.\,\mathit{prot}) \qquad P[\vec{t}/\vec{x}]}{\mid\!\rightsquigarrow \texttt{ses\_own}\ \chi\ s\ (n+1)\ m\ (\mathit{prot}[\vec{t}/\vec{x}]) * \texttt{ses\_idx}\ \chi\ s\ n\ (v[\vec{t}/\vec{x}])}$$

So how does it work?

So how does it work?

Well, one approach is...

The (simplified) step modality definition

Designed as a “frame” around the Hoare triple (/ weakest precondition):

$$\mid\!\!\leadsto P \triangleq \forall n.\ \text{⧖}_\bullet\, n \Rrightarrow (\text{⧖}_\bullet\, n * (\models\!\triangleright\!\!\Rightarrow^{(n+1)} \text{⧖}_\bullet\,(n+1) \Rrightarrow \text{⧖}_\bullet\,(n+1) * P))$$

The (simplified) step modality definition

Designed as a "frame" around the Hoare triple (/ weakest precondition):

$$\models_{\rightsquigarrow} P \triangleq \forall n.\, ⧗_\bullet\, n \Rrightarrow (⧗_\bullet\, n * (\models\!\!\Rrightarrow^{(n+1)} ⧗_\bullet\, (n+1) \Rrightarrow ⧗_\bullet\, (n+1) * P))$$

It reads as follows:

- Get the total step count ($⧗_\bullet\, n$)

The (simplified) step modality definition

Designed as a “frame” around the Hoare triple (/ weakest precondition):

$$\mid\!\leadsto P \triangleq \forall n.\, \text{⧗}_\bullet\, n \Rrightarrow (\text{⧗}_\bullet\, n * (\models\!\!\Rrightarrow^{(n+1)} \text{⧗}_\bullet\, (n+1) \Rrightarrow \text{⧗}_\bullet\, (n+1) * P))$$

It reads as follows:

- Get the total step count ($\text{⧗}_\bullet\, n$)
- Take a ghost step (in which n may be approximated with any local $\text{⧗}\ m$)

The (simplified) step modality definition

Designed as a "frame" around the Hoare triple (/ weakest precondition):

$$\models\!\!\rightsquigarrow P \triangleq \forall n.\, ⧗_\bullet\, n \Rrightarrow (⧗_\bullet\, n * ({\models\!\!\Rrightarrow\!\!\triangleright})^{(n+1)}\, ⧗_\bullet\, (n+1) \Rrightarrow ⧗_\bullet\, (n+1) * P))$$

It reads as follows:

- Get the total step count ($⧗_\bullet\, n$)
- Take a ghost step (in which n may be approximated with any local $⧗\, m$)
- Give back the total step count ($⧗_\bullet\, n$)

The (simplified) step modality definition

Designed as a “frame” around the Hoare triple (/ weakest precondition):

$$\mathrel{|\!\!\leadsto} P \triangleq \forall n.\, \text{⧖}_\bullet\, n \Rrightarrow (\text{⧖}_\bullet\, n * (\mathrel{|\!\!\Rrightarrow\!\!\triangleright}^{(n+1)} \text{⧖}_\bullet\, (n+1) \Rrightarrow \text{⧖}_\bullet\, (n+1) * P))$$

It reads as follows:

- Get the total step count ($\text{⧖}_\bullet\, n$)
- Take a ghost step (in which n may be approximated with any local $\text{⧖}\, m$)
- Give back the total step count ($\text{⧖}_\bullet\, n$)
- Strip laters (and ghost steps) relative to the total step count ($\mathrel{|\!\!\Rrightarrow\!\!\triangleright}^{(n+1)}$)

The (simplified) step modality definition

Designed as a “frame” around the Hoare triple (/ weakest precondition):

$$\mathrel{|\!\!\rightsquigarrow} P \triangleq \forall n.\ ⧗_{\bullet}\ n \Rrightarrow (⧗_{\bullet}\ n * ({\models\!\triangleright\!\!\Rrightarrow}^{(n+1)}\ ⧗_{\bullet}\ (n+1) \Rrightarrow ⧗_{\bullet}\ (n+1) * P))$$

It reads as follows:

- Get the total step count ($⧗_{\bullet}\ n$)
- Take a ghost step (in which n may be approximated with any local $⧗\ m$)
- Give back the total step count ($⧗_{\bullet}\ n$)
- Strip laters (and ghost steps) relative to the total step count (${\models\!\triangleright\!\!\Rrightarrow}^{(n+1)}$)
- Get the updated total step count, as a step has been taken ($⧗_{\bullet}\ (n+1)$)

The (simplified) step modality definition

Designed as a "frame" around the Hoare triple (/ weakest precondition):

$$\mid\!\leadsto P \triangleq \forall n.\ \text{⧖}_\bullet\ n \Rrightarrow (\text{⧖}_\bullet\ n * (\models\!\triangleright\!\Rrightarrow^{(n+1)} \text{⧖}_\bullet\ (n+1) \Rrightarrow \text{⧖}_\bullet\ (n+1) * P))$$

It reads as follows:

- Get the total step count ($\text{⧖}_\bullet\ n$)
- Take a ghost step (in which n may be approximated with any local $\text{⧖}\ m$)
- Give back the total step count ($\text{⧖}_\bullet\ n$)
- Strip laters (and ghost steps) relative to the total step count ($\models\!\triangleright\!\Rrightarrow^{(n+1)}$)
- Get the updated total step count, as a step has been taken ($\text{⧖}_\bullet\ (n+1)$)
- Take a ghost step (where local state can be updated $\text{⧖}\ m \Rrightarrow \text{⧖}\ (m+1)$)

The (simplified) step modality definition

Designed as a "frame" around the Hoare triple (/ weakest precondition):

$$\models\!\rightsquigarrow P \triangleq \forall n.\, ⧖_{\bullet}\, n \Rrightarrow (⧖_{\bullet}\, n * ({\models\!\Rrightarrow}^{(n+1)}\, ⧖_{\bullet}\, (n+1) \Rrightarrow ⧖_{\bullet}\, (n+1) * P))$$

It reads as follows:

- Get the total step count ($⧖_{\bullet}\, n$)
- Take a ghost step (in which n may be approximated with any local $⧖\, m$)
- Give back the total step count ($⧖_{\bullet}\, n$)
- Strip laters (and ghost steps) relative to the total step count (${\models\!\Rrightarrow}^{(n+1)}$)
- Get the updated total step count, as a step has been taken ($⧖_{\bullet}\, (n+1)$)
- Take a ghost step (where local state can be updated $⧖\, m \Rrightarrow ⧖\, (m+1)$)
- Give back the updated total step count ($⧖_{\bullet}\, (n+1)$)

The (simplified) step modality definition

Designed as a “frame” around the Hoare triple (/ weakest precondition):

$$\mid\!\leadsto P \triangleq \forall n.\ ⧗_{\bullet}\ n \Rrightarrow (⧗_{\bullet}\ n * (\mid\!\Rrightarrow\!\triangleright^{(n+1)}\ ⧗_{\bullet}\ (n+1) \Rrightarrow ⧗_{\bullet}\ (n+1) * P))$$

It reads as follows:

- Get the total step count ($⧗_{\bullet}\ n$)
- Take a ghost step (in which n may be approximated with any local $⧗\ m$)
- Give back the total step count ($⧗_{\bullet}\ n$)
- Strip laters (and ghost steps) relative to the total step count ($\mid\!\Rrightarrow\!\triangleright^{(n+1)}$)
- Get the updated total step count, as a step has been taken ($⧗_{\bullet}\ (n+1)$)
- Take a ghost step (where local state can be updated $⧗\ m \Rrightarrow ⧗\ (m+1)$)
- Give back the updated total step count ($⧗_{\bullet}\ (n+1)$)
- Prove the goal P

The step modality definition

The actual modality is language-generic, and is defined in terms of the language-parametric state interpretation: $S\ \sigma\ n\ \kappa s\ nt$

The step modality definition

The actual modality is language-generic, and is defined in terms of the language-parametric state interpretation: $S\ \sigma\ n\ \kappa s\ nt$

$$
\begin{aligned}
\mid\!\!\rightsquigarrow P \triangleq\ & \forall \sigma_1, n, \kappa, \kappa s, nt.\\
& \quad S\ \sigma_1\ n\ (\kappa \cdot \kappa s)\ nt \Rrightarrow\\
& \qquad (S\ \sigma_1\ n\ (\kappa \cdot \kappa s)\ nt\ *\\
& \qquad\quad \models\!\!\Rrightarrow^{((n_\triangleright\ n)+1)} \forall \sigma_2, nt'.\\
& \qquad\qquad S\ \sigma_2\ (n+1)\ \kappa s\ (nt' \cdot nt) \Rrightarrow\\
& \qquad\qquad\quad S\ \sigma_2\ (n+1)\ \kappa s\ (nt' \cdot nt) * P))
\end{aligned}
$$

The step modality definition

The actual modality is language-generic, and is defined in terms of the language-parametric state interpretation: $S\ \sigma\ n\ \kappa s\ nt$

$$
\begin{aligned}
\mid\!\rightsquigarrow P \triangleq\ & \forall \sigma_1, n, \kappa, \kappa s, nt. \\
& \quad S\ \sigma_1\ n\ (\kappa \cdot \kappa s)\ nt \Rrightarrow \\
& \qquad (S\ \sigma_1\ n\ (\kappa \cdot \kappa s)\ nt\ * \\
& \qquad\quad \mid\!\Rrightarrow\!\triangleright^{((n_\triangleright\ n)+1)} \forall \sigma_2, nt'. \\
& \qquad\qquad S\ \sigma_2\ (n+1)\ \kappa s\ (nt' \cdot nt) \Rrightarrow \\
& \qquad\qquad\quad S\ \sigma_2\ (n+1)\ \kappa s\ (nt' \cdot nt) * P))
\end{aligned}
$$

Language "primitive" rules should then be proven for the later-stripping mechanism of choice, e.g. the ones for time receipts (where $S\ \sigma\ n\ \kappa s\ nt \triangleq \mathbin{⧗}_\bullet\ n * \ldots$):

$$
\frac{\textsc{Step-time-get}}{\mid\!\rightsquigarrow \mathbin{⧗}\ 0}
\qquad
\frac{\mathbin{⧗}\ n}{\mid\!\rightsquigarrow \mathbin{⧗}\ (n+1)}\ \textsc{Step-time-incr}
\qquad
\frac{\mathbin{⧗}\ n \qquad \mid\!\Rrightarrow\!\triangleright^{(n+1)} P}{\mid\!\rightsquigarrow P}\ \textsc{Step-time-frame}
$$

A work in progress

A work in progress

The definition and interface is *not* final

A work in progress

The definition and interface is *not* final

However, everything in this talk has been fully mechanised using it.

Mechanisation:

`https://gitlab.mpi-sws.org/iris/iris/-/merge_requests/887`

`https://gitlab.mpi-sws.org/iris/actris/-/merge_requests/30`

Some reflections on the step modality

What if the modality abstracts away necessary details? (e.g. specific number of laters)

- The purpose of the step modality is to allow user-friendly specifications.
- One should retain the stronger specifications alongside the step-based ones.

Some reflections on the step modality

What if the modality abstracts away necessary details? (e.g. specific number of laters)

- The purpose of the step modality is to allow user-friendly specifications.
- One should retain the stronger specifications alongside the step-based ones.

Will we just end up with nested step modalities?: $\mid\!\!\rightsquigarrow^n P$

- Only if one actually *must* take multiple steps.

Some reflections on the step modality

What if the modality abstracts away necessary details? (e.g. specific number of laters)

- The purpose of the step modality is to allow user-friendly specifications.
- One should retain the stronger specifications alongside the step-based ones.

Will we just end up with nested step modalities?: $\mid\!\!\leadsto^n P$

- Only if one actually *must* take multiple steps.

How is the step modality related to later credits?

- They solve different problems, and can be used together

Some reflections on the step modality

What if the modality abstracts away necessary details? (e.g. specific number of laters)

- The purpose of the step modality is to allow user-friendly specifications.
- One should retain the stronger specifications alongside the step-based ones.

Will we just end up with nested step modalities?: $\mid\!\!\rightsquigarrow^n P$

- Only if one actually *must* take multiple steps.

How is the step modality related to later credits?

- They solve different problems, and can be used together

What about rules that didn't need to take a step? (like allocation rules)

- There is an associated (omitted) "pre-step" modality

Some reflections on the step modality

What if the modality abstracts away necessary details? (e.g. specific number of laters)

- The purpose of the step modality is to allow user-friendly specifications.
- One should retain the stronger specifications alongside the step-based ones.

Will we just end up with nested step modalities?: $\mid\!\!\rightsquigarrow^n P$

- Only if one actually *must* take multiple steps.

How is the step modality related to later credits?

- They solve different problems, and can be used together

What about rules that didn't need to take a step? (like allocation rules)

- There is an associated (omitted) "pre-step" modality

Is the step modality obsolete in Transfinite Iris?

- Pretty much, yeah

Future work

Better support for invariant masks

- Currently does not have nice modality rules

Future work

Better support for invariant masks

- Currently does not have nice modality rules

Consider language-agnostic later-stripping mechanisms

- Currently later-stripping rules are proven per language

Future work

Better support for invariant masks

- Currently does not have nice modality rules

Consider language-agnostic later-stripping mechanisms

- Currently later-stripping rules are proven per language

Better Iris proofmode support

- Currently works for non-viewshift updates

Future work

Better support for invariant masks

- Currently does not have nice modality rules

Consider language-agnostic later-stripping mechanisms

- Currently later-stripping rules are proven per language

Better Iris proofmode support

- Currently works for non-viewshift updates

Consider use cases besides later-stripping

- Abstract access to state interpretation may be beneficial for other problems

Future work

Better support for invariant masks

- Currently does not have nice modality rules

Consider language-agnostic later-stripping mechanisms

- Currently later-stripping rules are proven per language

Better Iris proofmode support

- Currently works for non-viewshift updates

Consider use cases besides later-stripping

- Abstract access to state interpretation may be beneficial for other problems

Looking for feedback!

- Suggestions for improvements, usefulness, etc.

STEP-QUESTIONS?

$$\frac{\textsf{Question}}{\mid\!\rightsquigarrow \textsf{Answer}}$$