A Gentle Adventure Mechanising Message Passing Concurrency Systems

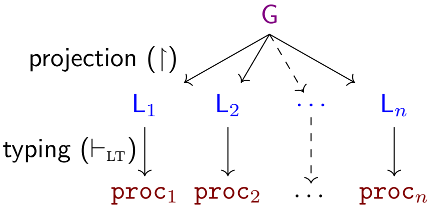
Formalising the Metatheory for smol-Zooid

David Castro-Perez, Francisco Ferreira, **Lorenzo Gheri**, and Nobuko Yoshida

Imperial College London

University of Kent

The MPST World, as We Know It



G

projection ($\upharpoonright$)

L_1 L_2 $\cdots$ L_n

typing ($\vdash_{\text{LT}}$)

$\texttt{proc}_1$ $\texttt{proc}_2$ $\cdots$ $\texttt{proc}_n$

K. Honda, N. Yoshida, and M. Carbone. Multiparty asynchronous session types. POPL '08

Zoid

$$\begin{tikzcd}
G \arrow[r, "\Re"] \arrow[d, "\upharpoonright"] & G^{c} \arrow[r, "\text{LTS}"] \arrow[d, "\upharpoonright^{c}"] & \text{global trace} \arrow[d, leftrightarrow, "="] \\
L \arrow[r, "\Re"] \arrow[d, "\vdash_{\mathrm{LT}}"] \arrow[ddr, dashed, dash, red, bend left, "\text{Well typed!}"'] & L^{c} \arrow[r, "\text{LTS}"] & \text{local trace} \\
\texttt{proc} \arrow[rr, "\text{LTS}"] \arrow[d, "\text{extraction}"'] \arrow[dr, leftrightarrow, "\text{DSL layer}"] & & \text{process trace} \arrow[u, "\text{erase}"'] \arrow[dl, dashed, dash, red, "\text{Certified semantics!}"] \\
\text{OCaml code} \arrow[r, dashed, dash, red, "\text{Extractable!}"] & \textbf{Zoid} &
\end{tikzcd}$$

D. Castro-Perez, F. Ferreira, L. Gheri, and N. Yoshida. Zoid: a DSL for certified multiparty computation: from mechanised metatheory to certified multiparty processes. PLDI 2021

Introducing the Metatheory of smol-Zooid Types

Simple, but significant multiparty session type metatheory!

$$\begin{tikzcd}
G \arrow[r, "\mathfrak{R}"] \arrow[d, "\upharpoonright"] & G^c \arrow[r, "\mathsf{LTS}"] \arrow[d, "\upharpoonright^c"] & \text{global trace} \arrow[d, leftrightarrow, "="] \\
L \arrow[r, "\mathfrak{R}"] & L^c \arrow[r, "\mathsf{LTS}"] & \text{local trace}
\end{tikzcd}$$

Embark on our Gentle Adventure!!! https://github.com/emtst/GentleAdventure

Formalisation of Global and Local Types

Inductively Defined Datatypes

$$\begin{aligned} \mathsf{G} \;&::= \mathtt{end} \\ &\mid X \\ &\mid \mu X.\mathsf{G} \\ &\mid \mathsf{p} \to \mathsf{q} :(\mathbb{S}).\mathsf{G} \end{aligned}$$

$$\begin{aligned} \mathsf{L} \;&:= \mathtt{end} \\ &\mid X \\ &\mid \mu X.\mathsf{L} \\ &\mid ![\mathsf{q}];(\mathbb{S}).\mathsf{L} \\ &\mid ?[\mathsf{p}];(\mathbb{S}).\mathsf{L} \end{aligned}$$

Coinductively Defined Datatypes

$$\begin{aligned} \mathsf{G}^{\mathsf{c}} \;::=\; &\mathtt{end}^{\mathsf{c}} \\ &\mid \mathsf{p} \to \mathsf{q} :(\mathbb{S}).\mathsf{G}^{\mathsf{c}} \\ &\mid \mathsf{p} \rightsquigarrow \mathsf{q} :(\mathbb{S}).\mathsf{G}^{\mathsf{c}} \end{aligned}$$

$$\begin{aligned} \mathsf{L}^{\mathsf{c}} \;::=\; &\mathtt{end}^{\mathsf{c}} \\ &\mid !^{\mathsf{c}}[\mathsf{p}];(\mathbb{S}).\mathsf{L}^{\mathsf{c}} \\ &\mid ?^{\mathsf{c}}[\mathsf{q}];(\mathbb{S}).\mathsf{L}^{\mathsf{c}} \end{aligned}$$

Formalisation of Global and Local Types

$$\begin{tikzcd}
\mathsf{G} = \mu X.\mathsf{p} \to \mathsf{q} :(\mathbb{S}).X \arrow[r, "\Re"] \arrow[d, "\upharpoonright"] & \mathsf{G}^{\mathsf{c}} = \mathsf{p} \to \mathsf{q} :(\mathbb{S}).\mathsf{G}^{\mathsf{c}} \arrow[d, "\upharpoonright^{c}"] \\
\begin{array}{l} \mathsf{G} \upharpoonright_{\mathsf{p}} = \mu X.![\mathsf{q}];(\mathbb{S}).X \\ \mathsf{G} \upharpoonright_{\mathsf{q}} = \mu X.?[\mathsf{p}];(\mathbb{S}).X \end{array} \arrow[r, "\Re"] & \begin{array}{l} \mathsf{L}^{\mathsf{c}}{}_{\mathsf{p}} = !^{\mathsf{c}}[\mathsf{q}];(\mathbb{S}).\mathsf{L}^{\mathsf{c}}{}_{\mathsf{p}} \\ \mathsf{L}^{\mathsf{c}}{}_{\mathsf{q}} = ?^{\mathsf{c}}[\mathsf{p}];(\mathbb{S}).\mathsf{L}^{\mathsf{c}}{}_{\mathsf{q}} \\ \text{with } \mathsf{G}^{\mathsf{c}} \upharpoonright^{\mathsf{c}} \mathsf{p}\ \mathsf{L}^{\mathsf{c}}{}_{\mathsf{p}} \\ \text{and } \mathsf{G}^{\mathsf{c}} \upharpoonright^{\mathsf{c}} \mathsf{q}\ \mathsf{L}^{\mathsf{c}}{}_{\mathsf{q}} \end{array}
\end{tikzcd}$$

Abandoning Inductive Datatypes

Theorem (Unravelling preserves projections)

Given G, L, G^{c} *and* L^{c}, *such that* (a) $\mathsf{G}{\upharpoonright}\mathsf{r} = \mathsf{L}$, (b) $\mathsf{G}\Re\mathsf{G}^{\mathsf{c}}$, *and* (c) $\mathsf{L}\Re\mathsf{L}^{\mathsf{c}}$, *then* $\mathsf{G}^{\mathsf{c}}\ {\upharpoonright}^{c}\mathsf{r}\ \mathsf{L}^{\mathsf{c}}$.

$$\begin{tikzcd}
\mathsf{G} \arrow[r, "\Re"] \arrow[d, "\upharpoonright"] & \mathsf{G}^{\mathsf{c}} \arrow[d, "\upharpoonright^{\mathsf{c}}"] \\
\mathsf{L} \arrow[r, "\Re"] & \mathsf{L}^{\mathsf{c}}
\end{tikzcd}$$

Proof.

By coinduction. :) □

The Paco Library for Coq: https://plv.mpi-sws.org/paco/

Type Semantics for Zooid

$$\begin{tikzcd}
G^{c} \arrow[r, "\mathsf{LTS}"] \arrow[d, "\upharpoonright^{c}"] & \text{global trace} \arrow[d, leftrightarrow, "="] \\
L^{c} \arrow[r, "\mathsf{LTS}"] & \text{local trace}
\end{tikzcd}$$

With Love, from p to q

p sends:

$$\begin{tikzcd}
\mathsf{p} \to \mathsf{q} :(S).\mathsf{G}^{\mathsf{c}} \arrow[r, "!\mathsf{pq}S"] \arrow[d, "\upharpoonright_{\mathsf{p}}"] & \mathsf{p} \rightsquigarrow \mathsf{q} :(S).\mathsf{G}^{\mathsf{c}} \arrow[r, "?\mathsf{qp}S"] \arrow[dr, "\upharpoonright_{\mathsf{p}}"] & \mathsf{G}^{\mathsf{c}} \arrow[d, "\upharpoonright_{\mathsf{p}}"] \\
!^{\mathsf{c}}[\mathsf{q}];(S).\mathsf{L}^{\mathsf{c}} \arrow[rr, "!\mathsf{pq}S"] & & \mathsf{L}^{\mathsf{c}}
\end{tikzcd}$$

q receives:

$$\begin{tikzcd}
\mathsf{p} \to \mathsf{q} :(S).\mathsf{G}^{\mathsf{c}} \arrow[r, "!\mathsf{pq}S"] \arrow[d, "\upharpoonright_{\mathsf{q}}"] & \mathsf{p} \rightsquigarrow \mathsf{q} :(S).\mathsf{G}^{\mathsf{c}} \arrow[r, "?\mathsf{qp}S"] \arrow[dl, "\upharpoonright_{\mathsf{q}}"] & \mathsf{G}^{\mathsf{c}} \arrow[d, "\upharpoonright_{\mathsf{q}}"] \\
?^{\mathsf{c}}[\mathsf{p}];(S).\mathsf{L}^{\mathsf{c}\prime} \arrow[rr, "?\mathsf{qp}S"] & & \mathsf{L}^{\mathsf{c}\prime}
\end{tikzcd}$$

Tools for our LTS

Actions. $!\mathsf{pq}\mathbb{S}$ and $?\mathsf{qp}\mathbb{S}$

(Local) Environments. E such that, $E(\mathsf{p}) = \mathsf{L^c}_\mathsf{p}$ where $\mathsf{G^c} \upharpoonright^{\mathsf{c}} \mathsf{p}\ \mathsf{L^c}_\mathsf{p}$

Queues and Queue Environments. Q, buffers for asynchronous communication.

$$!^{\mathsf{c}}[\mathsf{q}];(S).\mathsf{L^c} \xrightarrow{\quad\mathsf{step}\quad} \mathsf{L^c}$$

$$Q(\mathsf{p},\mathsf{q}) = [] \xrightarrow{\quad\mathsf{enqueue}\quad} Q(\mathsf{p},\mathsf{q}) = [S] \xrightarrow{\quad\mathsf{dequeue}\quad} Q(\mathsf{p},\mathsf{q}) = []$$

$$?^{\mathsf{c}}[\mathsf{p}];(S).\mathsf{L^{c\prime}} \xrightarrow{\quad\mathsf{step}\quad} \mathsf{L^{c\prime}}$$

Tools for our LTS

Actions. $!\mathsf{pq}\mathsf{S}$ and $?\mathsf{qp}\mathsf{S}$

(Local) Environments. E such that, $E(\mathsf{p}) = \mathsf{L^c}_{\mathsf{p}}$ where $\mathsf{G^c} \upharpoonright^{\mathsf{c}} \mathsf{p}\ \mathsf{L^c}_{\mathsf{p}}$

Queues and Queue Environments. Q, buffers for asynchronous communication.

$$!^{\mathsf{c}}[\mathsf{q}];(S).\mathsf{L^c} \xrightarrow{\quad\mathsf{step}\quad} \mathsf{L^c}$$

$$Q(\mathsf{p},\mathsf{q}) = [] \xrightarrow{\mathsf{enqueue}} Q(\mathsf{p},\mathsf{q}) = [S] \xrightarrow{\mathsf{dequeue}} Q(\mathsf{p},\mathsf{q}) = []$$

$$?^{\mathsf{c}}[\mathsf{p}];(S).\mathsf{L^{c\prime}} \xrightarrow{\quad\mathsf{step}\quad} \mathsf{L^{c\prime}}$$

$$\begin{tikzcd}
 & \mathsf{G^c} \arrow[dl, "\upharpoonright^{\mathsf{c}}"'] \arrow[dr, "\upharpoonright^{\mathsf{q}}"] & \\
E & & Q
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathsf{G^c} \arrow[d, "\upharpoonright\upharpoonright"] \\
(E, Q)
\end{tikzcd}$$

Theorems

Theorem (Step Soundness)

If $\mathsf{G}^{\mathsf{c}} \xrightarrow{a} \mathsf{G}^{\mathsf{c}\prime}$ *and* $\mathsf{G}^{\mathsf{c}} \upharpoonleft\upharpoonleft(E, Q)$*, there exist* E' *and* Q' *such that* $\mathsf{G}^{\mathsf{c}\prime} \upharpoonleft\upharpoonleft(E', Q')$ *and* $(E, Q) \xrightarrow{a} (E', Q')$*.*

Theorem (Step Completeness)

If $(E, Q) \xrightarrow{a} (E', Q')$ *and* $\mathsf{G}^{\mathsf{c}} \upharpoonleft\upharpoonleft(E, Q)$*, there exist* $\mathsf{G}^{\mathsf{c}\prime}$ *such that* $\mathsf{G}^{\mathsf{c}\prime} \upharpoonleft\upharpoonleft(E', Q')$ *and* $\mathsf{G}^{\mathsf{c}} \xrightarrow{a} \mathsf{G}^{\mathsf{c}\prime}$*.*

Theorem (Trace equivalence)

If $\mathsf{G}^{\mathsf{c}} \upharpoonleft\upharpoonleft(E, Q)$*, then* $tr^{g}t\mathsf{G}^{\mathsf{c}}$ *if and only if* $tr^{l}t(E, Q)$ *.*

Lemma, to give the flavour

$$\begin{tikzcd}
\mathsf{p} \to \mathsf{q} :(S).\mathsf{G}^{\mathsf{c}} \arrow[r, "!\mathsf{pqS}"] \arrow[d, "\upharpoonleft_{\mathsf{p}}"] & \mathsf{p} \rightsquigarrow \mathsf{q} :(S).\mathsf{G}^{\mathsf{c}} \arrow[dr, "\upharpoonleft_{\mathsf{p}}"] & \\
!^{\mathsf{c}}[\mathsf{q}];(S).\mathsf{L}^{\mathsf{c}} \arrow[rr, "!\mathsf{pqS}"] & & \mathsf{L}^{\mathsf{c}}
\end{tikzcd}$$

$\longrightarrow$ Coq!

$$\begin{tikzcd}
\mathsf{p} \to \mathsf{q} :(S).\mathsf{G}^{\mathsf{c}} \arrow[r, "!\mathsf{pqS}"] \arrow[d, "\upharpoonleft_{\mathsf{q}}"] & \mathsf{p} \rightsquigarrow \mathsf{q} :(S).\mathsf{G}^{\mathsf{c}} \arrow[dl, "\upharpoonleft_{\mathsf{q}}"] \\
?^{\mathsf{c}}[\mathsf{p}];(S).\mathsf{L}^{\mathsf{c}\prime} &
\end{tikzcd}$$

Our Adventurer Rests and Meditates

- Formal proofs are not easy! (But useful and fun!)
- Proof design is the key.
- Proof techniques are to be taken seriously: (co)induction, functions VS relations...

Our Adventurer Rests and Meditates

- Formal proofs are not easy! (But useful and fun!)
- Proof design is the key.
- Proof techniques are to be taken seriously: (co)induction, functions VS relations...

"You need to stay focused. Otherwise you miss the subtleties!" [1]

[1] Barney Greenway (Napalm Death), after suprising the audience with a blitz performance of "You Suffer".

Future

- Adding Features for Reasoning about Processes
- Certifying Existing Systems (e.g., integration with νScr)
- Moving Further towards Coinduction (e.g., Interaction Trees)
- Hoping for New People and Collaborations :)

Check out our material!

→ D. Castro-Perez, F. Ferreira, L. Gheri, and N. Yoshida. "Zooid: a DSL for certified multiparty computation: from mechanised metatheory to certified multiparty processes". PLDI 2021.
DOI: https://doi.org/10.1145/3453483.3454041
website: http://mrg.doc.ic.ac.uk/publications/zooid-paper/

→ This tutorial is available at https://github.com/emtst/GentleAdventure

Thank You!