Partially Typed Multiparty Sessions

Franco Barbanera[1], Mariangiola Dezani [2]

[1] University of Catania
[2] University of Torino

ICE - June 19, 2023, Lisbon

OVERVIEW

- Choreographies and MPSTs: a type assignment approach;
- Lock-freedom: egualitarism is not for system components;
- ICE'23: A MPST type assignment for classist Lock-freedom.

OVERVIEW

- Choreographies and MPSTs: a type assignment approach;
- Lock-freedom: egualitarism is not for system components;
- ICE'23: A MPST type assignment for classist Lock-freedom.

OVERVIEW

- Choreographies and MPSTs: a type assignment approach;
- Lock-freedom: egualitarism is not for system components;
- ICE'23: A MPST type assignment for classist Lock-freedom.

OVERVIEW

- Choreographies and MPSTs: a type assignment approach;
- Lock-freedom: egualitarism is not for system components;
- ICE'23: A MPST type assignment for classist Lock-freedom.

MPTS

MPTS

MultiParty Session Types (MPST):

a body of coreographic formalisms

MPTS

MultiParty Session Types (MPST):

a body of coreographic formalisms

where

two distinct but related views of a concurrent systems do coexist:

MPTS

MultiParty Session Types (MPST):

a body of coreographic formalisms

where

two distinct but related views of a concurrent systems do coexist:

global view: overall behaviour of the system formalised using the notion of Global Type

MPTS

MultiParty Session Types (MPST):

a body of coreographic formalisms

where

two distinct but related views of a concurrent systems do coexist:

global view: overall behaviour of the system formalised using the notion of Global Type

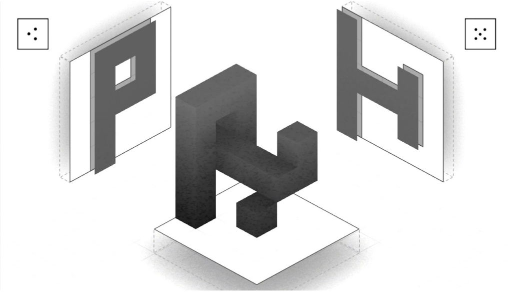
local view: behaviours of the single components in suitable process algebras

MPST approaches

Top-down MPST: communication protocols are explicity described as global types and, subsequently, by projecting them, local types are obtained for implementation.

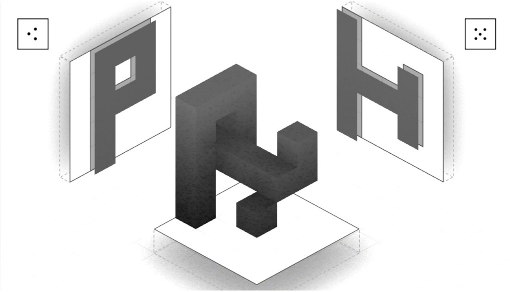
MPST approaches

Top-down MPST: communication protocols are explicity described as global types and, subsequently, by projecting them, local types are obtained for implementation.



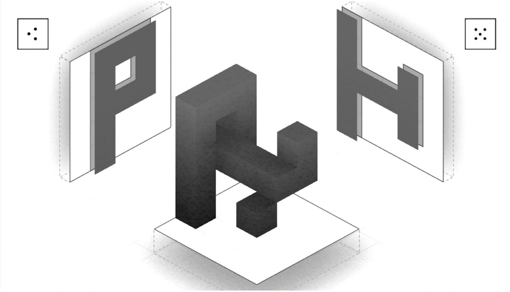
MPST approaches

Top-down MPST: communication protocols are explicity described as global types and, subsequently, by projecting them, local types are obtained for implementation.



MPST approaches

Top-down MPST: communication protocols are explicity described as global types and, subsequently, by projecting them, local types are obtained for implementation.



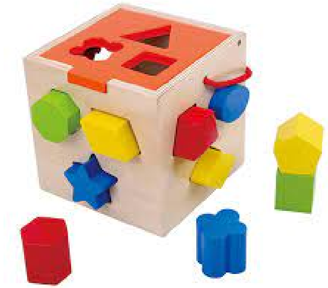
`Systems obtained by projecting (well-formed) global types enjoy good communication properties`

MPST approaches

Bottom-up MPST: no projection is used and local behaviours are checked against global types by means of a type assignment system.

MPST approaches

Bottom-up MPST: no projection is used and local behaviours are checked against global types by means of a type assignment system.

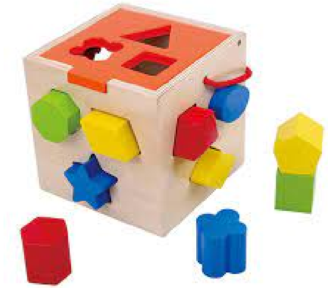


MPST approaches

Bottom-up MPST: no projection is used and local behaviours are checked against global types by means of a type assignment system.

MPST approaches

Bottom-up MPST: no projection is used and local behaviours are checked against global types by means of a type assignment system.



`Systems typable with (well-formed) global types enjoy good communication properties`

A “bottom-up” MPST

A “bottom-up” MPST

Calculus of Sessions and its type system

[B.,Dezani et al.FACS’22]

A "bottom-up" MPST

Calculus of Sessions and its type system

[B.,Dezani et al.FACS'22]

Processes

$$P ::=^{coind} \mathbf{0} \mid \mathsf{p}!\{\lambda_i.P_i\}_{i\in I} \mid \mathsf{p}?\{\lambda_i.P_i\}_{i\in I}$$

A "bottom-up" MPST

Calculus of Sessions and its type system

[`B.,Dezani et al.FACS'22`]

Processes

$$P ::=^{coind} \mathbf{0} \mid \mathsf{p}!\{\lambda_i.P_i\}_{i\in I} \mid \mathsf{p}?\{\lambda_i.P_i\}_{i\in I}$$

Multiparty Sessions

$$\mathbb{M} = \mathsf{p}_1[P_1] \parallel \cdots \parallel \mathsf{p}_n[P_n]$$

A “bottom-up” MPST

Calculus of Sessions and its type system

[B.,Dezani et al.FACS’22]

Processes

$$P ::=^{coind} \mathbf{0} \mid \mathsf{p}!\{\lambda_i.P_i\}_{i\in I} \mid \mathsf{p}?\{\lambda_i.P_i\}_{i\in I}$$

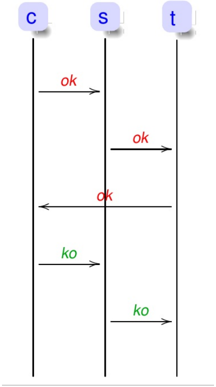
Multiparty Sessions

$$\mathbb{M} = \mathsf{p}_1[P_1] \parallel \cdots \parallel \mathsf{p}_n[P_n]$$

(synchronous) Operational Semantics

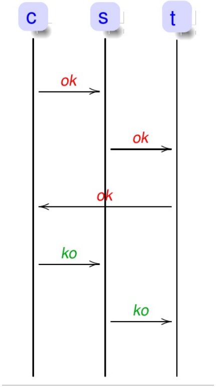
$$\frac{\ell \in I \subseteq J}{\mathsf{p}[\mathsf{q}!\{\lambda_i.P_i\}_{i\in I}] \parallel \mathsf{q}[\mathsf{p}?\{\lambda_j.Q_j\}_{j\in J}] \parallel \mathbb{M} \xrightarrow{\mathsf{p}\lambda_\ell\mathsf{q}} \mathsf{p}[P_\ell] \parallel \mathsf{q}[Q_\ell] \parallel \mathbb{M}}$$

A session example: carol, sam and tom



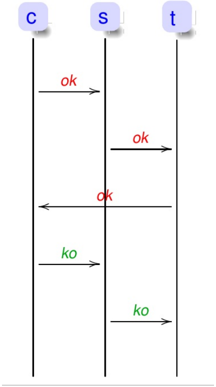
$c[s!\{\text{OK}.t?\text{OK}, \text{KO}\}] \parallel s[c?\{\text{OK}.t!\text{OK}, \text{KO}.t!\text{KO}\}] \parallel t[s?\{\text{OK}.c!\text{OK}, \text{KO}\}]$

A session example: carol, sam and tom



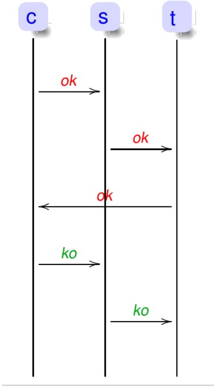
$$c[s!\{\textsc{ok}.t?\textsc{ok}, \textsc{ko}\}] \parallel s[c?\{\textsc{ok}.t!\textsc{ok}, \textsc{ko}.t!\textsc{ko}\}] \parallel t[s?\{\textsc{ok}.c!\textsc{ok}, \textsc{ko}\}]$$

A session example: carol, sam and tom



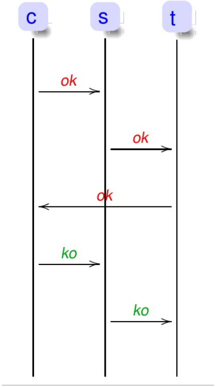
$$\mathsf{c}[\mathsf{s}!\{\textsc{ok}.\mathsf{t}?\textsc{ok}, \textsc{ko}\}] \parallel \mathsf{s}[\mathsf{c}?\{\textsc{ok}.\mathsf{t}!\textsc{ok}, \textsc{ko}.\mathsf{t}!\textsc{ko}\}] \parallel \mathsf{t}[\mathsf{s}?\{\textsc{ok}.\mathsf{c}!\textsc{ok}, \textsc{ko}\}]$$

A session example: carol, sam and tom



$c[s!\{\textsc{ok}.t?\textsc{ok}, \textsc{ko}\}] \parallel s[c?\{\textsc{ok}.t!\textsc{ok}, \textsc{ko}.t!\textsc{ko}\}] \parallel t[s?\{\textsc{ok}.c!\textsc{ok}, \textsc{ko}\}]$

A session example: carol, sam and tom



$$c[s!\{\textsc{ok}.t?\textsc{ok},\ \textsc{ko}\}] \parallel s[c?\{\textsc{ok}.t!\textsc{ok},\ \textsc{ko}.t!\textsc{ko}\}] \parallel t[s?\{\textsc{ok}.c!\textsc{ok},\ \textsc{ko}\}]$$

A session example: carol, sam and tom

Reduction example

$$\mathsf{c}[\mathsf{s}!\{\textsc{ok}.\mathsf{t}?\textsc{ok},\textsc{ko}\}] \parallel \mathsf{s}[\mathsf{c}?\{\textsc{ok}.\mathsf{t}!\textsc{ok},\textsc{ko}.\mathsf{t}!\textsc{ko}\}] \parallel \mathsf{t}[\mathsf{s}?\{\textsc{ok}.\mathsf{c}!\textsc{ok},\textsc{ko}\}]$$

A session example: carol, sam and tom

Reduction example

$$\mathsf{c}[\mathsf{s}!\{\textsc{ok}.\mathsf{t}?\textsc{ok},\textsc{ko}\}] \parallel \mathsf{s}[\mathsf{c}?\{\textsc{ok}.\mathsf{t}!\textsc{ok},\textsc{ko}.\mathsf{t}!\textsc{ko}\}] \parallel \mathsf{t}[\mathsf{s}?\{\textsc{ok}.\mathsf{c}!\textsc{ok},\textsc{ko}\}]$$

$$\Big\downarrow \; \mathsf{c}\textsc{ok}\mathsf{s}$$

$$\mathsf{c}[\mathsf{t}?\textsc{ok}] \parallel \mathsf{s}[\mathsf{t}!\textsc{ok}] \parallel \mathsf{t}[\mathsf{s}?\{\textsc{ok}.\mathsf{c}!\textsc{ok},\textsc{ko}\}]$$

A session example: carol, sam and tom

Reduction example

$$
\begin{array}{c}
\mathsf{c}[\mathsf{s}!\{\textsc{ok}.\mathsf{t}?\textsc{ok},\textsc{ko}\}] \parallel \mathsf{s}[\mathsf{c}?\{\textsc{ok}.\mathsf{t}!\textsc{ok},\textsc{ko}.\mathsf{t}!\textsc{ko}\}] \parallel \mathsf{t}[\mathsf{s}?\{\textsc{ok}.\mathsf{c}!\textsc{ok},\textsc{ko}\}] \\
\Big\downarrow{\scriptstyle \mathsf{c}\textsc{ok}\mathsf{s}} \\
\mathsf{c}[\mathsf{t}?\textsc{ok}] \parallel \mathsf{s}[\mathsf{t}!\textsc{ok}] \parallel \mathsf{t}[\mathsf{s}?\{\textsc{ok}.\mathsf{c}!\textsc{ok},\textsc{ko}\}] \\
\Big\downarrow{\scriptstyle \mathsf{s}\textsc{ok}\mathsf{t}} \\
\mathsf{c}[\mathsf{t}?\textsc{ok}] \parallel \mathsf{s}[\mathbf{0}] \parallel \mathsf{t}[\mathsf{c}!\textsc{ok}]
\end{array}
$$

A session example: carol, sam and tom

Reduction example

$$c[s!\{\mathrm{OK}.t?\mathrm{OK},\mathrm{KO}\}] \parallel s[c?\{\mathrm{OK}.t!\mathrm{OK},\mathrm{KO}.t!\mathrm{KO}\}] \parallel t[s?\{\mathrm{OK}.c!\mathrm{OK},\mathrm{KO}\}]$$

$$\Big\downarrow \; c\,\mathrm{OK}\,s$$

$$c[t?\mathrm{OK}] \parallel s[t!\mathrm{OK}] \parallel t[s?\{\mathrm{OK}.c!\mathrm{OK},\mathrm{KO}\}]$$

$$\Big\downarrow \; s\,\mathrm{OK}\,t$$

$$c[t?\mathrm{OK}] \parallel s[\mathbf{0}] \parallel t[c!\mathrm{OK}]$$

$$\Big\downarrow \; t\,\mathrm{OK}\,c$$

$$c[\mathbf{0}] \parallel s[\mathbf{0}] \parallel t[\mathbf{0}]$$

A session example: carol, sam and tom

Reduction example

$$\mathsf{c}[\mathsf{s}!\{\text{OK}.\mathsf{t}?\text{OK},\text{KO}\}] \parallel \mathsf{s}[\mathsf{c}?\{\text{OK}.\mathsf{t}!\text{OK},\text{KO}.\mathsf{t}!\text{KO}\}] \parallel \mathsf{t}[\mathsf{s}?\{\text{OK}.\mathsf{c}!\text{OK},\text{KO}\}]$$

A session example: carol, sam and tom

Reduction example

$$\mathsf{c}[\mathsf{s}!\{\textsc{ok}.\mathsf{t}?\textsc{ok},\textsc{ko}\}] \parallel \mathsf{s}[\mathsf{c}?\{\textsc{ok}.\mathsf{t}!\textsc{ok},\textsc{ko}.\mathsf{t}!\textsc{ko}\}] \parallel \mathsf{t}[\mathsf{s}?\{\textsc{ok}.\mathsf{c}!\textsc{ok},\textsc{ko}\}]$$

$$\Big\downarrow \mathsf{c}\textsc{ko}\mathsf{s}$$

$$\mathsf{c}[\mathbf{0}] \parallel \mathsf{s}[\mathsf{t}!\textsc{ko}] \parallel \mathsf{t}[\mathsf{s}?\{\textsc{ok}.\mathsf{c}!\textsc{ok},\textsc{ko}\}]$$

A session example: carol, sam and tom

Reduction example

$$\mathsf{c}[\mathsf{s}!\{\textsc{ok}.\mathsf{t}?\textsc{ok},\textsc{ko}\}] \parallel \mathsf{s}[\mathsf{c}?\{\textsc{ok}.\mathsf{t}!\textsc{ok},\textsc{ko}.\mathsf{t}!\textsc{ko}\}] \parallel \mathsf{t}[\mathsf{s}?\{\textsc{ok}.\mathsf{c}!\textsc{ok},\textsc{ko}\}]$$

$$\Big\downarrow \mathsf{c}\textsc{ko}\mathsf{s}$$

$$\mathsf{c}[\mathbf{0}] \parallel \mathsf{s}[\mathsf{t}!\textsc{ko}] \parallel \mathsf{t}[\mathsf{s}?\{\textsc{ok}.\mathsf{c}!\textsc{ok},\textsc{ko}\}]$$

$$\Big\downarrow \mathsf{s}\textsc{ko}\mathsf{t}$$

$$\mathsf{c}[\mathbf{0}] \parallel \mathsf{s}[\mathbf{0}] \parallel \mathsf{t}[\mathbf{0}]$$

Type system for the session calculus

Type system for the session calculus

Global Types

$$\mathsf{G} ::=^{coind} \texttt{End} \mid \mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i \in I}$$

Type system for the session calculus

Global Types

$$\mathsf{G} ::=^{coind} \mathtt{End} \;\mid\; \mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i \in I}$$

A global type for the example

$$\mathsf{c} \to \mathsf{s} : \{\textsc{ok}.\mathsf{s} \to \mathsf{t} : \textsc{ok}.\mathsf{t} \to \mathsf{c} : \textsc{ok}, \mathsf{ko}.\mathsf{s} \to \mathsf{t} : \mathsf{ko}\}$$

Type system for the session calculus

Global Types

$$\mathsf{G} ::=^{coind} \mathtt{End} \mid \mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i \in I}$$

Typing Rules

Type system for the session calculus

Global Types

$$\mathsf{G} ::=^{coind} \mathtt{End} \mid \mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i \in I}$$

Typing Rules

$$\frac{}{\mathtt{End} \vdash \mathsf{p}[\mathbf{0}]}$$

Type system for the session calculus

Global Types

$$\mathsf{G} ::=^{coind} \mathtt{End} \;\mid\; \mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i \in I}$$

Typing Rules

$$\frac{}{\mathtt{End} \vdash \mathsf{p}[\mathbf{0}]}$$

$$\frac{\mathsf{G}_i \vdash \mathsf{p}[P_i] \parallel \mathsf{q}[Q_i] \parallel \mathbb{M} \quad \mathsf{prt}(\mathsf{G}_i) \setminus \{\mathsf{p}, \mathsf{q}\} = \mathsf{prt}(\mathbb{M}) \quad \forall i \in I}{\mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i \in I} \vdash \mathsf{p}[\mathsf{q}!\{\lambda_i.P_i\}_{i \in I}] \parallel \mathsf{q}[\mathsf{p}?\{\lambda_j.Q_j\}_{j \in J}] \parallel \mathbb{M}} \; I \subseteq J$$

Type system for the session calculus

Example of type derivation

$$\texttt{End} \vdash \mathsf{c}[\mathbf{0}] \parallel \mathsf{s}[\mathbf{0}] \parallel \mathsf{t}[\mathbf{0}]$$

Type system for the session calculus

Example of type derivation

$$\dfrac{\texttt{End} \vdash \mathsf{c}[\mathbf{0}] \parallel \mathsf{s}[\mathbf{0}] \parallel \mathsf{t}[\mathbf{0}]}{\mathsf{t} \to \mathsf{c}{:}\textsc{ok} \vdash \mathsf{c}[\mathsf{t}?\textsc{ok}] \parallel \mathsf{s}[\mathbf{0}] \parallel \mathsf{t}[\mathsf{c}!\textsc{ok}]}$$

Type system for the session calculus

Example of type derivation

$$\dfrac{\dfrac{\mathtt{End} \vdash c[\mathbf{0}] \parallel s[\mathbf{0}] \parallel t[\mathbf{0}]}{t \to c{:}\textsc{ok} \vdash c[t?\textsc{ok}] \parallel s[\mathbf{0}] \parallel t[c!\textsc{ok}]}}{s \to t{:}\textsc{ok}.t \to c{:}\textsc{ok} \vdash c[t?\textsc{ok}] \parallel s[t!\textsc{ok}] \parallel t[s?\{\textsc{ok}.c!\textsc{ok}, ko\}]}$$

Type system for the session calculus

Example of type derivation

$$\dfrac{\dfrac{\texttt{End} \vdash \mathsf{c}[\mathbf{0}] \parallel \mathsf{s}[\mathbf{0}] \parallel \mathsf{t}[\mathbf{0}]}{\mathsf{t} \to \mathsf{c}{:}\mathsf{ok} \vdash \mathsf{c}[\mathsf{t}?\mathsf{ok}] \parallel \mathsf{s}[\mathbf{0}] \parallel \mathsf{t}[\mathsf{c}!\mathsf{ok}]}}{\mathsf{s} \to \mathsf{t}{:}\mathsf{ok}.\mathsf{t} \to \mathsf{c}{:}\mathsf{ok} \vdash \mathsf{c}[\mathsf{t}?\mathsf{ok}] \parallel \mathsf{s}[\mathsf{t}!\mathsf{ok}] \parallel \mathsf{t}[\mathsf{s}?\{\mathsf{ok}.\mathsf{c}!\mathsf{ok}, \mathsf{ko}\}]} \qquad \dfrac{\texttt{End} \vdash \mathsf{c}[\mathbf{0}] \parallel \mathsf{s}[\mathbf{0}] \parallel \mathsf{t}[\mathbf{0}]}{\mathsf{s} \to \mathsf{t}{:}\mathsf{ko} \vdash \mathsf{c}[\mathbf{0}] \parallel \mathsf{s}[\mathsf{t}!\mathsf{ko}] \parallel \mathsf{t}[\mathsf{s}?\{\mathsf{ok}.\mathsf{c}!\mathsf{ok}, \mathsf{ko}\}]}$$

Example of type derivation

$$
\dfrac{
\dfrac{
\dfrac{\mathtt{End} \vdash c[\mathbf{0}] \parallel s[\mathbf{0}] \parallel t[\mathbf{0}]}
{t \to c{:}ok \vdash c[t?ok] \parallel s[\mathbf{0}] \parallel t[c!ok]}
}{s \to t{:}ok.t \to c{:}ok \vdash c[t?ok] \parallel s[t!ok] \parallel t[s?\{ok.c!ok,ko\}]}
\qquad
\dfrac{\mathtt{End} \vdash c[\mathbf{0}] \parallel s[\mathbf{0}] \parallel t[\mathbf{0}]}
{s \to t{:}ko \vdash c[\mathbf{0}] \parallel s[t!ko] \parallel t[s?\{ok.c!ok,ko\}]}
}{c \to s : \{ok.s \to pt : ok.t \to c : ok, ko.s \to t : ko\} \vdash c[s!\{ok.t?ok,ko\}] \parallel s[c?\{ok.t!ok,ko.t!ko\}] \parallel t[s?\{ok.c!ok,ko\}]}
$$

What do we get by typing?

Definition (p-Lock)

$\mathbb{M}'$ is a p-lock if the participant p is willing to progress in $\mathbb{M}'$ but cannot do that in any continuation of $\mathbb{M}'$.

What do we get by typing?

Definition (Lock-freedom)

$\mathbb{M}$ is lock-free if, for each participant p,

$$\mathbb{M} \to^* \mathbb{M}' \text{ implies } \mathbb{M}' \text{ is not a p-lock}$$

What do we get by typing?

Theorem (B.,Dezani et al.FACS'22)

If $\mathbb{M}$ is typable with a well-formed (bounded) global type, then $\mathbb{M}$ is lock-free.

Universal Declaration of system-Components' Rights

- **Article 1** All system components must be developed equal in dignity and rights. They are endowed with communication capabilities and should interact towards one another in a spirit of cooperation.
- **Article 2** Any system component is entitled to all the good communication properties like lock freedom, without distinction of any kind, such as programming paradigm, language or interaction model.
- **Article 3** [...]

Universal Declaration of system-Components' Rights

- **Article 1** All system components must be developed equal in dignity and rights. They are endowed with communication capabilities and should interact towards one another in a spirit of cooperation.
- **Article 2** Any system component is entitled to all the good communication properties like lock freedom, without distinction of any kind, such as programming paradigm, language or interaction model.
- **Article 3** [...]

Universal Declaration of system-Components' Rights

- **Article 1** All system components must be developed equal in dignity and rights. They are endowed with communication capabilities and should interact towards one another in a spirit of cooperation.
- **Article 2** Any system component is entitled to all the good communication properties like lock freedom, without distinction of any kind, such as programming paradigm, language or interaction model.
- **Article 3** [...]

Universal Declaration of system-Components' Rights

- **Article 1** All system components must be developed equal in dignity and rights. They are endowed with communication capabilities and should interact towards one another in a spirit of cooperation.
- **Article 2** Any system component is entitled to all the good communication properties like lock freedom, without distinction of any kind, such as programming paradigm, language or interaction model.
- **Article 3** [...]

Universal Declaration of system-Components' Rights

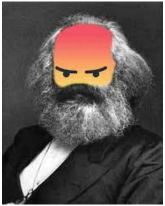
- **Article 1** All system components ~~must be developed equal in dignity and rights. They~~ are endowed with communication capabilities and should interact towards one another in a spirit of cooperation.
- **Article 2** Any system component is entitled to all the good communication properties like lock freedom, without distinction of any kind, such as programming paradigm, language or interaction model.
- **Article 3** [...]

Universal Declaration of system-Components' Rights

- **Article 1** All system components ~~must be developed equal in dignity and rights. They~~ are endowed with communication capabilities and should interact towards one another in a spirit of cooperation.
- **Article 2** Any system component is entitled to ~~all the good communication properties like lock freedom, without~~ no distinction of any kind, such as programming paradigm, language or interaction model.
- **Article 3** [...]

Universal Declaration of system-Components' Rights

- **Article 1** All system components ~~must be developed equal in dignity and rights. They~~ are endowed with communication capabilities and should interact towards one another in a spirit of cooperation.
- **Article 2** Any system component is entitled to ~~all the good communication properties like lock freedom, without~~ no distinction of any kind, such as programming paradigm, language or interaction model.
- **Article 3** [...]



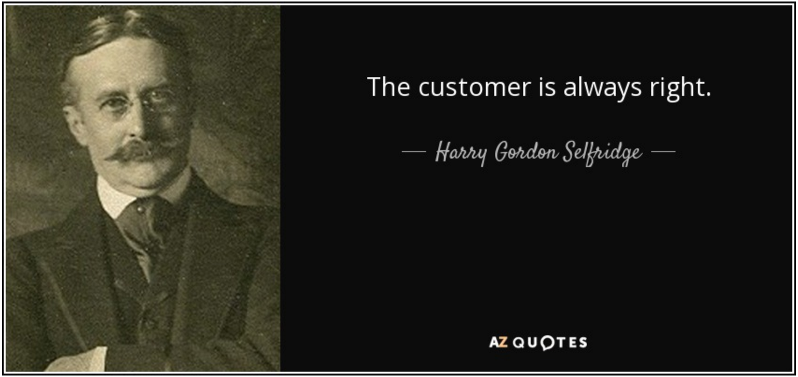
Non egalitarian systems

Non egalitarian systems

Any client-server setting is biased:

Non egalitarian systems

Any client-server setting is biased:



A non egalitarian system

A buyer can keep on ADDing goods - sold by a seller - in his shopping cart an unbounded number of times, until he decides to BUY the shopping cart's content. In the latter case, the seller informs the carrier for the SHIPment.

A non egalitarian system

A buyer can keep on ADDing goods - sold by a seller - in his shopping cart an unbounded number of times, until he decides to BUY the shopping cart's content. In the latter case, the seller informs the carrier for the SHIPment.

$$\mathbb{M} = \mathsf{b}[B] \parallel \mathsf{s}[S] \parallel \mathsf{c}[C]$$

where
$B = \mathsf{s}!\{\text{ADD}.B,\ \text{BUY}\}$
$S = \mathsf{b}?\{\text{ADD}.S,\ \text{BUY}.\mathsf{c}!\text{SHIP}\}$
$C = \mathsf{s}?\text{SHIP}.C$

A non egalitarian system

A buyer can keep on ADDing goods - sold by a seller - in his shopping cart an unbounded number of times, until he decides to BUY the shopping cart's content. In the latter case, the seller informs the carrier for the SHIPment.

$$\mathbb{M} = \mathsf{b}[B] \parallel \mathsf{s}[S] \parallel \mathsf{c}[C]$$

where

$B = \mathsf{s}!\{\textsc{add}.B,\ \textsc{buy}\}$

$S = \mathsf{b}?\{\textsc{add}.S,\ \textsc{buy}.\mathsf{c}!\textsc{ship}\}$

$C = \mathsf{s}?\textsc{ship}.C$

Not typable. In fact it is not c-lock free

$$\mathbb{M} \rightarrow^* \mathsf{b}[\mathbf{0}] \parallel \mathsf{s}[\mathbf{0}] \parallel \mathsf{c}[C]$$

A non egalitarian system

A buyer can keep on $\textsc{add}$ing goods - sold by a seller - in his shopping cart an unbounded number of times, until he decides to $\textsc{buy}$ the shopping cart's content. In the latter case, the seller informs the carrier for the $\textsc{ship}$ment.

$$\mathbb{M} = \mathsf{b}[B] \parallel \mathsf{s}[S] \parallel \mathsf{c}[C]$$

where

$B = \mathsf{s}!\{\textsc{add}.B,\ \textsc{buy}\}$
$S = \mathsf{b}?\{\textsc{add}.S,\ \textsc{buy}.\mathsf{c}!\textsc{ship}\}$
$C = \mathsf{s}?\textsc{ship}.C$

Not typable. In fact it is not c-lock free

$$\mathbb{M} \rightarrow^* \mathsf{b}[\mathbf{0}] \parallel \mathsf{s}[\mathbf{0}] \parallel \mathsf{c}[C]$$

Do we actually care about s and c?

Typing non egalitarian systems

We are interested in b's lock-freedom, not s's and c's

ICE'23: A type system such that if

$$G \vdash_{\{s,c\}} \mathbb{M}$$

then lock-freedom ensured only for participants other than s and c.
For our example this is possible for

$$G = b \to s{:}\{\textsc{add}.\,G,\ \textsc{buy}\}$$

Typing non egalitarian systems

We are interested in b's lock-freedom, not s's and c's

ICE'23: A type system such that if

$$\mathsf{G} \vdash_{\{\mathsf{s},\mathsf{c}\}} \mathbb{M}$$

then lock-freedom ensured only for participants other than s and c.
For our example this is possible for

$$\mathsf{G} = \mathsf{b} \to \mathsf{s}{:}\{\textsc{add}.\,\mathsf{G},\ \textsc{buy}\}$$

Typing non egalitarian systems

We are interested in b's lock-freedom, not s's and c's

ICE'23: A type system such that if

$$\mathsf{G} \vdash_{\{\mathsf{s},\mathsf{c}\}} \mathbb{M}$$

then lock-freedom ensured only for participants other than s and c.
For our example this is possible for

$$\mathsf{G} = \mathsf{b} \to \mathsf{s}{:}\{\textsc{add}.\,\mathsf{G},\ \textsc{buy}\}$$

Typing non egalitarian systems

$$[\text{End}] \quad \frac{}{\text{End} \vdash_{\emptyset} \mathsf{p}[\mathbf{0}]} \; [\text{End}]$$

$$\frac{\mathsf{G}_i \vdash_{\mathcal{P}_i} \mathsf{p}[P_i] \parallel \mathsf{q}[Q_i] \parallel \mathbb{M} \qquad (\mathsf{prt}(\mathsf{G}_i) \cup \mathcal{P}_i) \setminus \{\mathsf{p}, \mathsf{q}\} = \mathsf{prt}(\mathbb{M}) \quad \forall i \in I}{\mathsf{G} \vdash_{\mathcal{P}} \mathsf{p}[\mathsf{q}!\{\lambda_i.P_i\}_{i \in I}] \parallel \mathsf{q}[\mathsf{p}?\{\lambda_j.Q_j\}_{j \in J}] \parallel \mathbb{M}} \quad \begin{array}{l} \mathsf{G} = \mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i \in I} \\ \mathsf{G} \text{ is bounded} \\ \mathcal{P} = \bigcup_{i \in I} \mathcal{P}_i \\ I \subseteq J \end{array}$$

$$[\text{Weak}] \quad \frac{\mathsf{G} \vdash_{\mathcal{P}_1} \mathbb{M}_1}{\mathsf{G} \vdash_{\mathcal{P}_1 \cup \mathcal{P}_2} \mathbb{M}_1 \parallel \mathbb{M}_2} \quad \mathcal{P}_2 = \mathsf{prt}(\mathbb{M}_2) \neq \emptyset$$

Typing non egalitarian systems

$$[\texttt{End}] \quad \dfrac{}{\texttt{End} \vdash_{\emptyset} \mathsf{p}[\mathbf{0}]} \; [\texttt{End}]$$

$$\dfrac{\begin{array}{c}\mathsf{G}_i \vdash_{\mathcal{P}_i} \mathsf{p}[P_i] \parallel \mathsf{q}[Q_i] \parallel \mathbb{M} \\ (\mathsf{prt}(\mathsf{G}_i) \cup \mathcal{P}_i) \setminus \{\mathsf{p}, \mathsf{q}\} = \mathsf{prt}(\mathbb{M}) \quad \forall i \in I\end{array}}{\mathsf{G} \vdash_{\mathcal{P}} \mathsf{p}[\mathsf{q}!\{\lambda_i.P_i\}_{i \in I}] \parallel \mathsf{q}[\mathsf{p}?\{\lambda_j.Q_j\}_{j \in J}] \parallel \mathbb{M}} \; \begin{array}{l}\mathsf{G} = \mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i \in I} \\ \mathsf{G} \text{ is bounded} \\ \mathcal{P} = \bigcup_{i \in I} \mathcal{P}_i \\ I \subseteq J\end{array}$$

$$[\textsc{Weak}] \quad \dfrac{\mathsf{G} \vdash_{\mathcal{P}_1} \mathbb{M}_1}{\mathsf{G} \vdash_{\mathcal{P}_1 \cup \mathcal{P}_2} \mathbb{M}_1 \parallel \mathbb{M}_2} \; \mathcal{P}_2 = \mathsf{prt}(\mathbb{M}_2) \neq \emptyset$$

Typing non egalitarian systems

$$[\text{End}] \quad \frac{}{\text{End} \vdash_\emptyset \mathsf{p}[\mathbf{0}]} \; [\text{End}]$$

$$\frac{\mathsf{G}_i \vdash_{\mathcal{P}_i} \mathsf{p}[P_i] \parallel \mathsf{q}[Q_i] \parallel \mathbb{M} \qquad (\mathsf{prt}(\mathsf{G}_i) \cup \mathcal{P}_i) \setminus \{\mathsf{p},\mathsf{q}\} = \mathsf{prt}(\mathbb{M}) \quad \forall i \in I}{\mathsf{G} \vdash_{\mathcal{P}} \mathsf{p}[\mathsf{q}!\{\lambda_i.P_i\}_{i\in I}] \parallel \mathsf{q}[\mathsf{p}?\{\lambda_j.Q_j\}_{j\in J}] \parallel \mathbb{M}}$$

$\mathsf{G} = \mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i\in I}$
G is bounded
$\mathcal{P} = \bigcup_{i\in I} \mathcal{P}_i$
$I \subseteq J$

$$[\text{Weak}] \quad \frac{\mathsf{G} \vdash_{\mathcal{P}_1} \mathbb{M}_1}{\mathsf{G} \vdash_{\mathcal{P}_1 \cup \mathcal{P}_2} \mathbb{M}_1 \parallel \mathbb{M}_2} \quad \mathcal{P}_2 = \mathsf{prt}(\mathbb{M}_2) \neq \emptyset$$

Typing non egalitarian systems

$$[\text{End}] \quad \overline{\overline{\text{End} \vdash_{\emptyset} \mathsf{p}[\mathbf{0}]}} \; [\text{End}]$$

$$\frac{\mathsf{G}_i \vdash_{\mathcal{P}_i} \mathsf{p}[P_i] \parallel \mathsf{q}[Q_i] \parallel \mathbb{M} \qquad (\mathsf{prt}(\mathsf{G}_i) \cup \mathcal{P}_i) \setminus \{\mathsf{p}, \mathsf{q}\} = \mathsf{prt}(\mathbb{M}) \quad \forall i \in I}{\mathsf{G} \vdash_{\mathcal{P}} \mathsf{p}[\mathsf{q}!\{\lambda_i.P_i\}_{i\in I}] \parallel \mathsf{q}[\mathsf{p}?\{\lambda_j.Q_j\}_{j\in J}] \parallel \mathbb{M}}$$

$\mathsf{G} = \mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i\in I}$
G is bounded
$\mathcal{P} = \bigcup_{i\in I} \mathcal{P}_i$
$I \subseteq J$

$$[\text{Weak}] \quad \frac{\mathsf{G} \vdash_{\mathcal{P}_1} \mathbb{M}_1}{\mathsf{G} \vdash_{\mathcal{P}_1 \cup \mathcal{P}_2} \mathbb{M}_1 \parallel \mathbb{M}_2} \quad \mathcal{P}_2 = \mathsf{prt}(\mathbb{M}_2) \neq \emptyset$$

Typing non egalitarian systems

Theorem (Classist lock-freedom)

If $\mathsf{G} \vdash_{\mathcal{P}} \mathbb{M}$ *then* $\mathbb{M}$ *is* p*-lock free only if* $\mathsf{p} \notin \mathcal{P}$

Typing non egalitarian systems

$$\mathcal{D} \quad = \quad \dfrac{\mathcal{D} \qquad \dfrac{\dfrac{}{\mathbf{End} \vdash_{\emptyset} \mathsf{b}[\mathbf{0}]}\ [\mathrm{End}]}{\mathbf{End} \vdash_{\{\mathsf{s},\mathsf{c}\}} \mathsf{b}[\mathbf{0}] \parallel \mathsf{s}[\mathsf{c}!\textsc{ship}] \parallel \mathsf{c}[C]}\ [\textsc{weak}]}{\mathsf{G} \vdash_{\{\mathsf{s},\mathsf{c}\}} \mathsf{b}[B] \parallel \mathsf{s}[\mathsf{b}?\{\textsc{add}.S,\ \textsc{pay}.\mathsf{c}!\textsc{ship}\}] \parallel \mathsf{c}[\mathsf{s}?\textsc{ship}.C]}$$

where $\mathsf{G} = \mathsf{b} \to \mathsf{s}{:}\{\textsc{add}.\,\mathsf{G},\ \textsc{buy}\}$

$B = \mathsf{s}!\{\textsc{add}.B,\ \textsc{pay}\}$
$S = \mathsf{b}?\{\textsc{add}.S,\ \textsc{buy}.\mathsf{c}!\textsc{ship}\}$
$C = \mathsf{s}?\textsc{ship}.C$

Hence the Buyer-Seller-Carrier system is b-lock free

Typing non egalitarian systems

$$\mathcal{D} \quad = \quad \cfrac{\mathcal{D} \qquad \cfrac{\cfrac{}{\mathbf{End} \vdash_{\emptyset} \mathsf{b}[\mathbf{0}]}\,[\mathrm{End}]}{\mathbf{End} \vdash_{\{\mathsf{s},\mathsf{c}\}} \mathsf{b}[\mathbf{0}] \parallel \mathsf{s}[\mathsf{c}!\textsc{ship}] \parallel \mathsf{c}[C]}\,[\textsc{weak}]}{\mathsf{G} \vdash_{\{\mathsf{s},\mathsf{c}\}} \mathsf{b}[B] \parallel \mathsf{s}[\mathsf{b}?\{\textsc{add}.S,\ \textsc{pay}.\mathsf{c}!\textsc{ship}\}] \parallel \mathsf{c}[\mathsf{s}?\textsc{ship}.C]}$$

where $\mathsf{G} = \mathsf{b} \to \mathsf{s}{:}\{\textsc{add}.\,\mathsf{G},\ \textsc{buy}\}$

$B = \mathsf{s}!\{\textsc{add}.B,\ \textsc{pay}\}$
$S = \mathsf{b}?\{\textsc{add}.S,\ \textsc{buy}.\mathsf{c}!\textsc{ship}\}$
$C = \mathsf{s}?\textsc{ship}.C$

Hence the Buyer-Seller-Carrier system is b-lock free

Typing non egalitarian systems

$$\mathcal{D} \quad = \quad \dfrac{\mathcal{D} \qquad \dfrac{\dfrac{}{\mathbf{End} \vdash_{\emptyset} \mathsf{b}[\mathbf{0}]}\,[\mathbf{End}]}{\mathbf{End} \vdash_{\{\mathsf{s},\mathsf{c}\}} \mathsf{b}[\mathbf{0}] \parallel \mathsf{s}[\mathsf{c}!\textsc{ship}] \parallel \mathsf{c}[C]}\,[\textsc{weak}]}{\mathsf{G} \vdash_{\{\mathsf{s},\mathsf{c}\}} \mathsf{b}[B] \parallel \mathsf{s}[\mathsf{b}?\{\textsc{add}.S,\ \textsc{pay}.\mathsf{c}!\textsc{ship}\}] \parallel \mathsf{c}[\mathsf{s}?\textsc{ship}.C]}$$

where $\mathsf{G} = \mathsf{b} \to \mathsf{s}{:}\{\textsc{add}.\,\mathsf{G},\ \textsc{buy}\}$

$$B = \mathsf{s}!\{\textsc{add}.B,\ \textsc{pay}\}$$
$$S = \mathsf{b}?\{\textsc{add}.S,\ \textsc{buy}.\mathsf{c}!\textsc{ship}\}$$
$$C = \mathsf{s}?\textsc{ship}.C$$

Hence the Buyer-Seller-Carrier system is b-lock free

Ongoing work

Ongoing work

- Overshooting:

Ongoing work

- Overshooting: Well-formedness condition for global types does actually ensure more than needed (strong p-lock freedom)

Ongoing work

- Overshooting: Well-formedness condition for global types does actually ensure more than needed (strong p-lock freedom)
- Typing non egualitarian asynchronous systems.

Ongoing work

- Overshooting: Well-formedness condition for global types does actually ensure more than needed (strong p-lock freedom)
- Typing non egualitarian asynchronous systems.