Modular Multiparty Sessions with Mixed Choice

Franco Barbanera[1], Mariangiola Dezani[2]

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

ICE - June 20, 2025, Lille

OVERVIEW

- Simple MultiParty Sessions: a typing approach to multiparty session types;
- Extending SMPS with mixed-choice;
- Taming sessions with mixed-choice: a type system for modular sessions.

OVERVIEW

- Simple MultiParty Sessions: a typing approach to multiparty session types;
- Extending SMPS with mixed-choice;
- Taming sessions with mixed-choice: a type system for modular sessions.

OVERVIEW

- Simple MultiParty Sessions: a typing approach to multiparty session types;
- Extending SMPS with mixed-choice;
- Taming sessions with mixed-choice: a type system for modular sessions.

OVERVIEW

- Simple MultiParty Sessions: a typing approach to multiparty session types;
- Extending SMPS with mixed-choice;
- Taming sessions with mixed-choice: a type system for modular sessions.

Description/Verification of concurrent systems

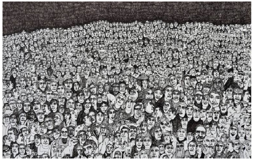
Global description

faithful/property-preserving

Implementation

Choreographic formalisms for description/verification of systems

Global description



Local (single component) behaviours

Two particular approaches to the Global-Local relationship

MPST: MultiParty Session Types (Honda,Yoshida et al.)

SMPS: Simple MultiParty Sessions (Dezani et al.)

Focusing on the essential

- No Delegation;
- Syncronous communications.

The SMPS and MPST approaches to Global $\leftrightarrow$ Local

	SMPS "verification oriented"	MPST "correct-by-design oriented"
Global description		
$\updownarrow$		
Local behaviours		

The SMPS and MPST approaches to Global $\leftrightarrow$ Local

	SMPS "verification oriented"	MPST "correct-by-design oriented"
Global description	**Global Type**	**Global Type**
$\updownarrow$		
Local behaviours		

The SMPS and MPST approaches to Global $\leftrightarrow$ Local

	SMPS “verification oriented”	MPST “correct-by-design oriented”
Global description	**Global Type**	**Global Type**
Local behaviours	**abstract Processes**	**Local Types**

The SMPS and MPST approaches to Global $\leftrightarrow$ Local

	SMPS “verification oriented”	MPST “correct-by-design oriented”
Global description	**Global Type**	**Global Type**
	Typing: $\vdash$	
Local behaviours	**abstract Processes**	**Local Types**

The SMPS and MPST approaches to Global $\leftrightarrow$ Local

	SMPS "verification oriented"	MPST "correct-by-design oriented"
Global description	**Global Type**	**Global Type**
$\updownarrow$	**Typing:** $\vdash$	**Projection:** $\upharpoonright$
Local behaviours	**abstract Processes**	**Local Types**

The SMPS and MPST approaches to Global $\leftrightarrow$ Local

	SMPS “verification oriented”		MPST “correct-by-design oriented”
Global description	**Global Type**		**Global Type**
	Typing: $\vdash$		**Projection:** $\upharpoonright$
Local behaviours	**abstract Processes**	$\approx$	**Local Types**

SMPS

The Calculus of Multiparty Sessions (SMPS local views)

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

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

The Calculus of Multiparty Sessions (SMPS local views)

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

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

The Calculus of Multiparty Sessions (SMPS local views)

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

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

The Calculus of Multiparty Sessions (SMPS local views)

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

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

The Calculus of Multiparty Sessions (SMPS local views)

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 type discipline for multiparty sessions

A type discipline for multiparty sessions

Global Types

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

A type discipline for multiparty sessions

Global Types

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

Typing Rules

A type discipline for multiparty sessions

Global Types

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

Typing Rules

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

A type discipline for multiparty sessions

Global Types

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

Typing Rules

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

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

SMPS with Mixed-Choice

More expressive behaviours with Mixed Choice

Processes

$$P ::=^{coind} \mathbf{0} \mid \Sigma_{i \in I} \pi_i . P_i \qquad \pi ::= \mathsf{p}?\lambda \mid \mathsf{p}!\lambda$$

Multiparty Sessions

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

(synchronous) Operational Semantics

$$\frac{}{\mathsf{p}[\mathsf{q}!\lambda.P + P'] \parallel \mathsf{q}[\mathsf{p}?\lambda.Q + Q'] \parallel \mathbb{M} \xrightarrow{\mathsf{p}\lambda\mathsf{q}} \mathsf{p}[P] \parallel \mathsf{q}[Q] \parallel \mathbb{M}}$$

More expressive behaviours with Mixed Choice

Processes

$$P ::=^{coind} \mathbf{0} \mid \Sigma_{i \in I} \pi_i . P_i \qquad \pi ::= \mathsf{p}?\lambda \mid \mathsf{p}!\lambda$$

Multiparty Sessions

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

(synchronous) Operational Semantics

$$\frac{}{\mathsf{p}[\mathsf{q}!\lambda.P + P'] \parallel \mathsf{q}[\mathsf{p}?\lambda.Q + Q'] \parallel \mathbb{M} \xrightarrow{\mathsf{p}\lambda\mathsf{q}} \mathsf{p}[P] \parallel \mathsf{q}[Q] \parallel \mathbb{M}}$$

More expressive behaviours with Mixed Choice

Processes

$$P ::=^{coind} \mathbf{0} \;\mid\; \Sigma_{i\in I}\pi_i.P_i \qquad\qquad \pi \;::=\; \mathsf{p}?\lambda \;\mid\; \mathsf{p}!\lambda$$

Multiparty Sessions

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

(synchronous) Operational Semantics

$$\dfrac{}{\mathsf{p}[\mathsf{q}!\lambda.P + P'] \parallel \mathsf{q}[\mathsf{p}?\lambda.Q + Q'] \parallel \mathbb{M} \xrightarrow{\mathsf{p}\lambda\mathsf{q}} \mathsf{p}[P] \parallel \mathsf{q}[Q] \parallel \mathbb{M}}$$

More expressive behaviours with Mixed Choice

Processes

$$P ::=^{coind} \mathbf{0} \;\mid\; \Sigma_{i\in I}\pi_i.P_i \qquad\qquad \pi \;::=\; \mathsf{p}?\lambda \;\mid\; \mathsf{p}!\lambda$$

Multiparty Sessions

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

(synchronous) Operational Semantics

$$\frac{}{\mathsf{p}[\mathsf{q}!\lambda.P + P'] \parallel \mathsf{q}[\mathsf{p}?\lambda.Q + Q'] \parallel \mathbb{M} \xrightarrow{\mathsf{p}\lambda\mathsf{q}} \mathsf{p}[P] \parallel \mathsf{q}[Q] \parallel \mathbb{M}}$$

More expressive behaviours with Mixed Choice

Processes

$$P ::=^{coind} \mathbf{0} \;\mid\; \Sigma_{i \in I} \pi_i . P_i \qquad\qquad \pi \;::=\; \mathsf{p}?\lambda \;\mid\; \mathsf{p}!\lambda$$

Multiparty Sessions

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

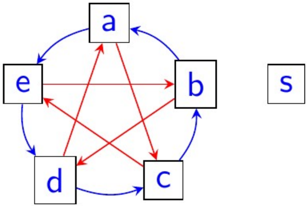
(synchronous) Operational Semantics

$$\frac{}{\mathsf{p}[\mathsf{q}!\lambda.P + P'] \parallel \mathsf{q}[\mathsf{p}?\lambda.Q + Q'] \parallel \mathbb{M} \xrightarrow{\mathsf{p}\lambda\mathsf{q}} \mathsf{p}[P] \parallel \mathsf{q}[Q] \parallel \mathbb{M}}$$

Try and discipline that!

The election session

The election session



$$\mathbb{E} = \mathsf{a}[P_\mathsf{a}] \parallel \mathsf{b}[P_\mathsf{b}] \parallel \mathsf{c}[P_\mathsf{c}] \parallel \mathsf{d}[P_\mathsf{d}] \parallel \mathsf{e}[P_\mathsf{e}] \parallel \mathsf{s}[P_\mathsf{s}]$$

$$\begin{aligned} \text{where} \quad P_\mathsf{a} \quad &= \quad \mathsf{e}!\mathit{leader} \\ &\quad +\mathsf{b}?\mathit{leader}.(\mathsf{c}!\mathit{leader} + \mathsf{d}?\mathit{leader}.\mathsf{s}!\mathit{elect}) \\ &\quad +\mathsf{s}?\mathit{del} \end{aligned}$$

and (with some abuse of notation)

$$P_\mathsf{s} \quad = \quad \Sigma_{\mathsf{x}\in\{\mathsf{a},\mathsf{b},\mathsf{c},\mathsf{d},\mathsf{e}\}}\left(\mathsf{x}?\mathit{elect}.\Sigma_{\mathsf{x}\in\{\mathsf{a},\mathsf{b},\mathsf{c},\mathsf{d},\mathsf{e}\}}\mathsf{x}!\mathit{del}\right)$$

A simple mixed-choice version of standard SMPS typing

$$\frac{\mathsf{G}_i \vdash \mathsf{p}[P_i] \parallel \mathsf{q}[Q_i] \parallel \mathbb{M}}{\mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i \in I} \vdash \mathsf{p}[\mathsf{q}!\{\lambda_i.P_i\}_{i \in I}] \parallel \mathsf{q}[\mathsf{p}?\{\lambda_j.Q_j\}_{j \in J}] \parallel \mathbb{M}}$$

A simple mixed-choice version of standard SMPS typing

$$\frac{\mathsf{G}_i \vdash \mathsf{p}[P_i] \parallel \mathsf{q}[Q_i] \parallel \mathbb{M}}{\mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i \in I} \vdash \mathsf{p}[\mathsf{q}!\{\lambda_i.P_i\}_{i \in I}] \parallel \mathsf{q}[\mathsf{p}?\{\lambda_j.Q_j\}_{j \in J}] \parallel \mathbb{M}}$$

essentially corresponds to

$$\frac{\mathsf{G}_i \vdash \mathbb{M}_i \qquad \mathbb{M} \xrightarrow{\Lambda_i} \mathbb{M}_i \quad \{\Lambda_i\}_{i \in I} = \text{ reductions involving } \mathsf{p} \in \mathbb{M}}{\Sigma_{i \in I} \Lambda_i.\mathsf{G}_i \vdash \mathbb{M}}$$

A simple mixed-choice version of standard SMPS typing

$$\frac{\mathsf{G}_i \vdash \mathsf{p}[P_i] \parallel \mathsf{q}[Q_i] \parallel \mathbb{M}}{\mathsf{p} \to \mathsf{q} : \{\lambda_i.\mathsf{G}_i\}_{i\in I} \vdash \mathsf{p}[\mathsf{q}!\{\lambda_i.P_i\}_{i\in I}] \parallel \mathsf{q}[\mathsf{p}?\{\lambda_j.Q_j\}_{j\in J}] \parallel \mathbb{M}}$$

essentially corresponds to

$$\frac{\mathsf{G}_i \vdash \mathbb{M}_i \qquad \mathbb{M} \xrightarrow{\Lambda_i} \mathbb{M}_i \quad \{\Lambda_i\}_{i\in I} = \text{ reductions involving } \mathsf{p} \in \mathbb{M}}{\Sigma_{i\in I}\Lambda_i.\mathsf{G}_i \vdash \mathbb{M}}$$

NO WAY! $\Sigma_{i\in I}\Lambda_i.\mathsf{G}_i$ would not capture the overall behaviour of the Election Session.

$$\frac{\mathsf{G}_i \vdash \mathbb{M}_i \qquad \mathbb{M} \xrightarrow{\Lambda_i} \mathbb{M}_i}{\Sigma_{i \in I} \Lambda_i.\mathsf{G}_i \vdash \mathbb{M}}$$

$$\frac{\mathsf{G}_i \vdash \mathbb{M}_i \qquad \mathbb{M} \xrightarrow{\Lambda_i} \mathbb{M}_i}{\Sigma_{i \in I} \Lambda_i . \mathsf{G}_i \vdash \mathbb{M}}$$

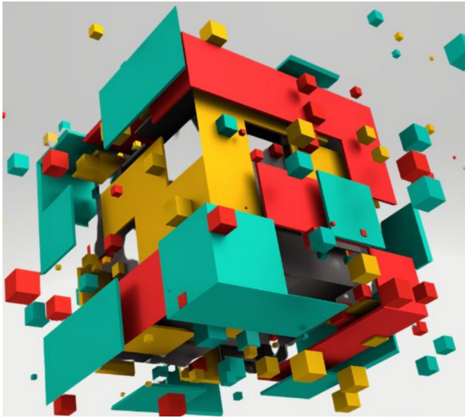
It would work for the Election Session

$$\frac{\mathsf{G}_i \vdash \mathbb{M}_i \qquad \mathbb{M} \xrightarrow{\Lambda_i} \mathbb{M}_i}{\Sigma_{i \in I} \Lambda_i . \mathsf{G}_i \vdash \mathbb{M}}$$

It would work for the Election Session

BUT

A global type would not "factorise" anything.
It would correspond to the complete reduction-tree.



MODULARISATION

Complex (software) systems can be decomposed – to some extent – into smaller, loosely coupled modules.

MODULARISATION

Complex (software) systems can be decomposed – to some extent – into smaller, loosely coupled modules.

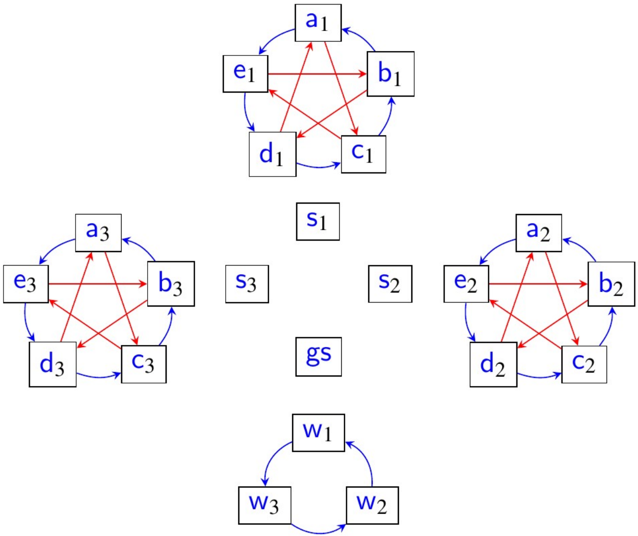
MODULARISATION

Complex (software) systems can be decomposed – to some extent – into smaller, loosely coupled modules.

"Divide and conquer"

Modular Election

Modular Election



Modular Election

$$\mathbb{E}^{gl} = \mathbb{E}_1 \parallel \mathbb{E}_2 \parallel \mathbb{E}_3 \parallel \mathbb{G}$$

where, for $1 \leq i \leq 3$,

$$\mathbb{E}_i = \mathsf{a}_i[P_{\mathsf{a}_i}] \parallel \mathsf{b}_i[P_{\mathsf{b}_i}] \parallel \mathsf{c}_i[P_{\mathsf{c}_i}] \parallel \mathsf{d}_i[P_{\mathsf{d}_i}] \parallel \mathsf{e}_i[P_{\mathsf{e}_i}] \parallel \mathsf{s}_i[P_{\mathsf{s}_i}]$$

$$\begin{aligned}
\text{with} \quad P_{\mathsf{a}_i} &= \mathsf{e}_i!\mathit{leader} \\
&\quad + \mathsf{b}_i?\mathit{leader}.\big(\mathsf{c}_i!\mathit{leader} + \mathsf{d}_i?\mathit{leader}.\mathsf{s}_i!\mathit{elect}.(\mathsf{s}_i?\mathit{gleader} + \mathsf{s}_i?\mathit{no})\big) \\
&\quad + \mathsf{s}_i?\mathit{del} \\
\text{and} \quad P_{\mathsf{s}_i} &= \Sigma_{\mathsf{x}\in\{\mathsf{a}_i,\mathsf{b}_i,\mathsf{c}_i,\mathsf{d}_i,\mathsf{e}_i\}}\mathsf{x}?\mathit{elect}.(\mathsf{gs}?\mathit{gleader}.\mathsf{x}!\mathit{gleader}.Q_i + \mathsf{gs}?\mathit{no}.\mathsf{x}!\mathit{no}.Q_i) \\
&\quad \text{with } Q_i = \Sigma_{\mathsf{x}\in\{\mathsf{a}_i,\mathsf{b}_i,\mathsf{c}_i,\mathsf{d}_i,\mathsf{e}_i\}}\mathsf{x}!\mathit{del}
\end{aligned}$$

$$\mathbb{G} = \mathsf{w}_1[P_{\mathsf{w}_1}] \parallel \mathsf{w}_2[P_{\mathsf{w}_2}] \parallel \mathsf{w}_3[P_{\mathsf{w}3}] \parallel \mathsf{gs}[P_{\mathsf{gs}}]$$

$$\begin{aligned}
\text{with} \quad P_{\mathsf{w}_i} &= \mathsf{w}_{i+2}!\mathit{leader} \\
&\quad + \mathsf{w}_{i+1}?\mathit{leader}.\mathsf{gs}!\mathit{gleader} \\
&\quad + \mathsf{gs}?\mathit{del} \\
\text{and} \quad P_{\mathsf{gs}} &= \Sigma_{i\in\{1,2,3\}}\mathsf{w}_i?\mathit{gleader}.\mathsf{s}_i!\mathit{gleader}.\mathsf{s}_{i+1}!\mathit{no}.\mathsf{s}_{i+2}!\mathit{no}.(\Sigma_{i\in\{1,2,3\}}\mathsf{w}_i!\mathit{del})
\end{aligned}$$

Modular Multiparty Sessions

Partitionable sessions such that

- Unrestricted mixed-choice can be used inside the modules;
- Inter-module communication through **connectors**;
- Connectors can interact using **one-to-one** mixed choice.

Modular Multiparty Sessions

Partitionable sessions such that

- Unrestricted mixed-choice can be used inside the modules;
- Inter-module communication through **connectors**;
- Connectors can interact using **one-to-one** mixed choice.

Modular Multiparty Sessions

Partitionable sessions such that

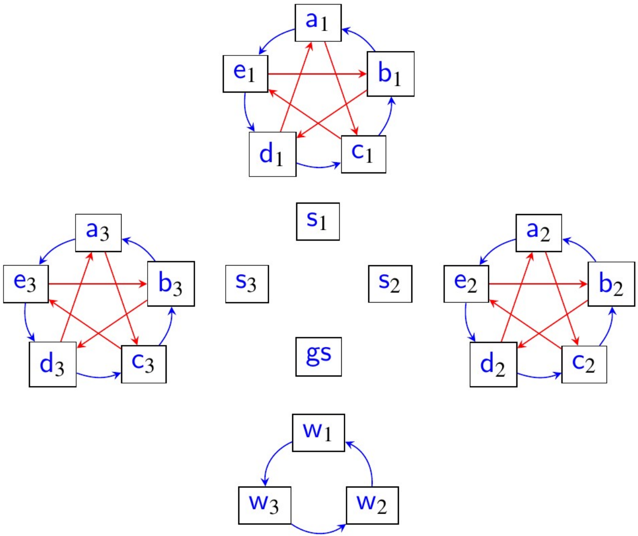
- Unrestricted mixed-choice can be used inside the modules;
- Inter-module communication through **connectors**;
- Connectors can interact using **one-to-one** mixed choice.

Modular Multiparty Sessions

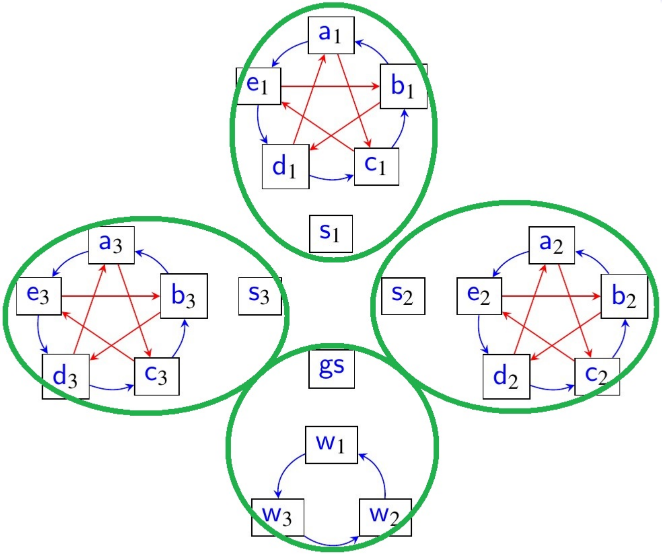
Partitionable sessions such that

- Unrestricted mixed-choice can be used inside the modules;
- Inter-module communication through **connectors**;
- Connectors can interact using **one-to-one** mixed choice.

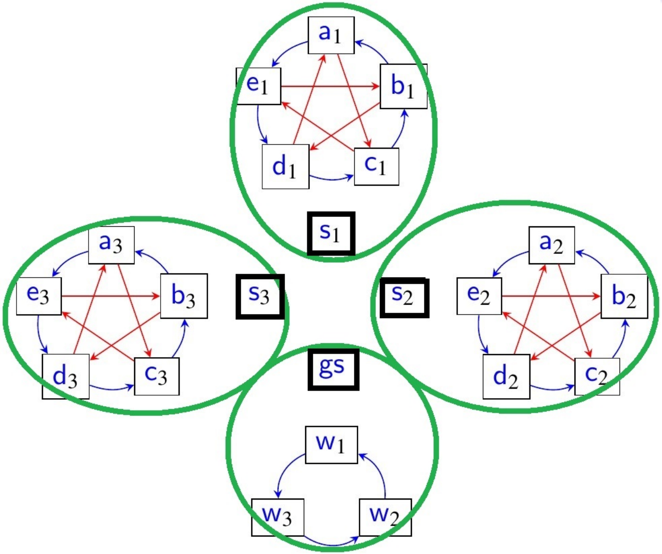
Modular Election



Modular Election



Modular Election



Modular Election

$$\mathbb{E}^{gl} \;=\; \mathbb{E}_1 \parallel \mathbb{E}_2 \parallel \mathbb{E}_3 \parallel \mathbb{G}$$

where, for $1 \leq i \leq 3$,

$$
\begin{array}{lrcl}
 & \mathbb{E}_i & = & \mathsf{a}_i[P_{\mathsf{a}_i}] \parallel \mathsf{b}_i[P_{\mathsf{b}_i}] \parallel \mathsf{c}_i[P_{\mathsf{c}_i}] \parallel \mathsf{d}_i[P_{\mathsf{d}_i}] \parallel \mathsf{e}_i[P_{\mathsf{e}_i}] \parallel \mathsf{s}_i[P_{\mathsf{s}_i}] \\
\text{with} & P_{\mathsf{a}_i} & = & \quad \mathsf{e}_i!\mathit{leader} \\
 & & & + \mathsf{b}_i?\mathit{leader}.\big(\mathsf{c}_i!\mathit{leader} + \mathsf{d}_i?\mathit{leader}.\mathsf{s}_i!\mathit{elect}.(\mathsf{s}_i?\mathit{gleader} + \mathsf{s}_i?\mathit{no})\big) \\
 & & & + \mathsf{s}_i?\mathit{del} \\
\text{and} & P_{\mathsf{s}_i} & = & \Sigma_{\mathsf{x}\in\{\mathsf{a}_i,\mathsf{b}_i,\mathsf{c}_i,\mathsf{d}_i,\mathsf{e}_i\}}\mathsf{x}?\mathit{elect}.(\mathsf{gs}?\mathit{gleader}.\mathsf{x}!\mathit{gleader}.Q_i + \mathsf{gs}?\mathit{no}.\mathsf{x}!\mathit{no}.Q_i) \\
 & & & \text{with } Q_i = \Sigma_{\mathsf{x}\in\{\mathsf{a}_i,\mathsf{b}_i,\mathsf{c}_i,\mathsf{d}_i,\mathsf{e}_i\}}\mathsf{x}!\mathit{del}
\end{array}
$$

$$
\begin{array}{lrcl}
 & \mathbb{G} & = & \mathsf{w}_1[P_{\mathsf{w}_1}] \parallel \mathsf{w}_2[P_{\mathsf{w}_2}] \parallel \mathsf{w}_3[P_{\mathsf{w}3}] \parallel \mathsf{gs}[P_{\mathsf{gs}}] \\
\text{with} & P_{\mathsf{w}_i} & = & \quad \mathsf{w}_{i+2}!\mathit{leader} \\
 & & & + \mathsf{w}_{i+1}?\mathit{leader}.\mathsf{gs}!\mathit{gleader} \\
 & & & + \mathsf{gs}?\mathit{del} \\
\text{and} & P_{\mathsf{gs}} & = & \Sigma_{i\in\{1,2,3\}}\mathsf{w}_i?\mathit{gleader}.\mathsf{s}_i!\mathit{gleader}.\mathsf{s}_{i+1}!\mathit{no}.\mathsf{s}_{i+2}!\mathit{no}.(\Sigma_{i\in\{1,2,3\}}\mathsf{w}_i!\mathit{del})
\end{array}
$$

Modular Election

$$\mathbb{E}^{gl} = \mathbb{E}_1 \| \mathbb{E}_2 \| \mathbb{E}_3 \| \mathbb{G}$$

where, for $1 \leq i \leq 3$,

$$\mathbb{E}_i = \mathsf{a}_i[P_{\mathsf{a}_i}] \| \mathsf{b}_i[P_{\mathsf{b}_i}] \| \mathsf{c}_i[P_{\mathsf{c}_i}] \| \mathsf{d}_i[P_{\mathsf{d}_i}] \| \mathsf{e}_i[P_{\mathsf{e}_i}] \| \mathsf{s}_i[P_{\mathsf{s}_i}]$$

$$\begin{aligned}
\text{with} \quad P_{\mathsf{a}_i} &= \mathsf{e}_i!\mathit{leader} \\
&\quad + \mathsf{b}_i?\mathit{leader}.\big(\mathsf{c}_i!\mathit{leader} + \mathsf{d}_i?\mathit{leader}.\mathsf{s}_i!\mathit{elect}.(\mathsf{s}_i?\mathit{gleader} + \mathsf{s}_i?\mathit{no})\big) \\
&\quad + \mathsf{s}_i?\mathit{del} \\
\text{and} \quad P_{\mathsf{s}_i} &= \Sigma_{\mathsf{x} \in \{\mathsf{a}_i, \mathsf{b}_i, \mathsf{c}_i, \mathsf{d}_i, \mathsf{e}_i\}} \mathsf{x}?\mathit{elect}.(\mathsf{gs}?\mathit{gleader}.\mathsf{x}!\mathit{gleader}.Q_i + \mathsf{gs}?\mathit{no}.\mathsf{x}!\mathit{no}.Q_i) \\
&\quad \text{with } Q_i = \Sigma_{\mathsf{x} \in \{\mathsf{a}_i, \mathsf{b}_i, \mathsf{c}_i, \mathsf{d}_i, \mathsf{e}_i\}} \mathsf{x}!\mathit{del}
\end{aligned}$$

$$\mathbb{G} = \mathsf{w}_1[P_{\mathsf{w}_1}] \| \mathsf{w}_2[P_{\mathsf{w}_2}] \| \mathsf{w}_3[P_{\mathsf{w}3}] \| \mathsf{gs}[P_{\mathsf{gs}}]$$

$$\begin{aligned}
\text{with} \quad P_{\mathsf{w}_i} &= \mathsf{w}_{i+2}!\mathit{leader} \\
&\quad + \mathsf{w}_{i+1}?\mathit{leader}.\mathsf{gs}!\mathit{gleader} \\
&\quad + \mathsf{gs}?\mathit{del} \\
\text{and} \quad P_{\mathsf{gs}} &= \Sigma_{i \in \{1,2,3\}} \mathsf{w}_i?\mathit{gleader}.\mathsf{s}_i!\mathit{gleader}.\mathsf{s}_{i+1}!\mathit{no}.\mathsf{s}_{i+2}!\mathit{no}.(\Sigma_{i \in \{1,2,3\}} \mathsf{w}_i!\mathit{del})
\end{aligned}$$

Modular Election

$$\mathbb{E}^{gl} = \mathbb{E}_1 \| \mathbb{E}_2 \| \mathbb{E}_3 \| \mathbb{G}$$

where, for $1 \leq i \leq 3$,

$$
\begin{aligned}
\mathbb{E}_i &= \mathsf{a}_i[P_{\mathsf{a}_i}] \| \mathsf{b}_i[P_{\mathsf{b}_i}] \| \mathsf{c}_i[P_{\mathsf{c}_i}] \| \mathsf{d}_i[P_{\mathsf{d}_i}] \| \mathsf{e}_i[P_{\mathsf{e}_i}] \; \| \boxed{\mathsf{s}_i[P_{\mathsf{s}_i}]} \\
\text{with} \quad P_{\mathsf{a}_i} &= \quad \mathsf{e}_i!\mathit{leader} \\
&+ \mathsf{b}_i?\mathit{leader}.\big(\mathsf{c}_i!\mathit{leader} + \mathsf{d}_i?\mathit{leader}.\mathsf{s}_i!\mathit{elect}.(\mathsf{s}_i?\mathit{gleader} + \mathsf{s}_i?\mathit{no})\big) \\
&+ \mathsf{s}_i?\mathit{del} \\
\text{and} \quad P_{\mathsf{s}_i} &= \Sigma_{\mathsf{x} \in \{\mathsf{a}_i,\mathsf{b}_i,\mathsf{c}_i,\mathsf{d}_i,\mathsf{e}_i\}} \mathsf{x}?\mathit{elect}.(\mathsf{gs}?\mathit{gleader}.\mathsf{x}!\mathit{gleader}.Q_i + \mathsf{gs}?\mathit{no}.\mathsf{x}!\mathit{no}.Q_i) \\
&\text{with } Q_i = \Sigma_{\mathsf{x} \in \{\mathsf{a}_i,\mathsf{b}_i,\mathsf{c}_i,\mathsf{d}_i,\mathsf{e}_i\}} \mathsf{x}!\mathit{del}
\end{aligned}
$$

$$
\begin{aligned}
\mathbb{G} &= \mathsf{w}_1[P_{\mathsf{w}_1}] \| \mathsf{w}_2[P_{\mathsf{w}_2}] \| \mathsf{w}_3[P_{\mathsf{w}3}] \; \| \boxed{\mathsf{gs}[P_{\mathsf{gs}}]} \\
\text{with} \quad P_{\mathsf{w}_i} &= \quad \mathsf{w}_{i+2}!\mathit{leader} \\
&+ \mathsf{w}_{i+1}?\mathit{leader}.\mathsf{gs}!\mathit{gleader} \\
&+ \mathsf{gs}?\mathit{del} \\
\text{and} \quad P_{\mathsf{gs}} &= \Sigma_{i \in \{1,2,3\}} \mathsf{w}_i?\mathit{gleader}.\mathsf{s}_i!\mathit{gleader}.\mathsf{s}_{i+1}!\mathit{no}.\mathsf{s}_{i+2}!\mathit{no}.(\Sigma_{i \in \{1,2,3\}} \mathsf{w}_i!\mathit{del})
\end{aligned}
$$

Modular typing

$$\frac{\mathsf{G}_i \vdash \mathbb{M}_i \qquad \mathbb{M} \xrightarrow{\Lambda_i} \mathbb{M}_i}{\Sigma_{i \in I} \Lambda_i . \mathsf{G}_i \vdash \mathbb{M}}$$

Modular typing

$$\frac{\mathsf{G}_i \vdash \mathbb{M}_i \qquad \mathbb{M} \xrightarrow{\Lambda_i} \mathbb{M}_i \quad \{\Lambda_i\}_{i \in I} = \text{ reductions involving } \mathsf{p} \in \mathbb{M}}{\Sigma_{i \in I} \Lambda_i . \mathsf{G}_i \vdash \mathbb{M}}$$

Modular typing

$$\dfrac{G_i \vdash^{\mathcal{P}} \mathbb{M}_i \qquad \mathbb{M} \xrightarrow{\Lambda_i} \mathbb{M}_i \qquad \{\Lambda_i\}_{i\in I} = \text{ reductions in a module of modularisation } \mathcal{P}}{\Sigma_{i\in I}\Lambda_i.G_i \vdash^{\mathcal{P}} \mathbb{M}}$$

Some properties of modularised typing

- Modularisation is preserved by reduction;
- Typability does depend **neither** on the module one starts with
  nor on the chosen modularisation.

Some properties of modularised typing

- Modularisation is preserved by reduction;
- Typability does depend **neither** on the module one starts with
  nor on the chosen modularisation.

Some properties of modularised typing

- Modularisation is preserved by reduction;
- Typability does depend **neither** on the module one starts with
  nor on the chosen modularisation.

What do we get by typing?

- Subject Reduction;
- Session Fidelity;
- Lock Freedom.

What do we get by typing?

- Subject Reduction;
- Session Fidelity;
- Lock Freedom.

What do we get by typing?

- Subject Reduction;
- Session Fidelity;
- Lock Freedom.

What do we get by typing?

- Subject Reduction;
- Session Fidelity;
- Lock Freedom.

Subject Reduction

If $\mathsf{G} \vdash^{\mathscr{P}} \mathbb{M}$ *and* $\mathbb{M} \xrightarrow{\Lambda} \mathbb{M}'$, *then* $\mathsf{G}' \vdash^{\mathscr{P}} \mathbb{M}'$ *and* $\mathsf{G} \xrightarrow{\Lambda} \mathsf{G}'$ *for some* G'.

Subject Reduction

$$\textit{If } \mathsf{G} \vdash^{\mathscr{P}} \mathbb{M} \textit{ and } \mathbb{M} \xrightarrow{\Lambda} \mathbb{M}', \textit{ then } \mathsf{G}' \vdash^{\mathscr{P}} \mathbb{M}' \textit{ and } \mathsf{G} \xrightarrow{\Lambda} \mathsf{G}' \textit{ for some } \mathsf{G}'.$$

Global Types LTS - coinductive [Dezani et al.]

$$[\text{E-COMM}] \frac{}{\Sigma_{i\in I}\Lambda_i.\mathsf{G}_i \xrightarrow{\Lambda_j} \mathsf{G}_j} \quad j \in I$$

$$[\text{I-COMM}] \frac{\mathsf{G}_i \xrightarrow{\Lambda} \mathsf{G}_i' \quad \Lambda \in cap(\mathsf{G}_i) \quad \mathsf{prt}(\Lambda) \cap \mathsf{prt}(\Lambda_i) = \emptyset \quad \forall i \in I}{\Sigma_{i\in I}\Lambda_i.\mathsf{G}_i \xrightarrow{\Lambda} \Sigma_{i\in I}\Lambda_i.\mathsf{G}_i'}$$

Session Fidelity

If $\mathsf{G} \vdash \mathbb{M}$ *and* $\mathsf{G} \xrightarrow{\Lambda} \mathsf{G}'$, *then* $\mathbb{M} \xrightarrow{\Lambda} \mathbb{M}'$ *and* $\mathsf{G}' \vdash \mathbb{M}'$ *for some* $\mathbb{M}'$.

Lock Freedom

If $\mathbb{M}$ is typable, then $\mathbb{M}$ is lock free.

Lock Freedom

If $\mathbb{M}$ is typable, then $\mathbb{M}$ is lock free.

Definition (Lock-freedom)

A session $\mathbb{M}$ is lock free *if $\mathbb{M} \xrightarrow{\sigma} \mathbb{M}'$ with σ finite and $\mathsf{p} \in \mathsf{prt}(\mathbb{M}')$ imply $\mathbb{M}' \xrightarrow{\sigma' \cdot \Lambda}$ for some σ' and Λ such that $\mathsf{p} \notin \mathsf{prt}(\sigma')$ and $\mathsf{p} \in \mathsf{prt}(\Lambda)$.*

Ongoing work

- Typing modular sessions with mixed choice and **asynchronous** communications
- Inspiration for conditions enabling to compose systems of Communicating Finite State Machines (mixed-choice harmful for that)
- What about projection?

Ongoing work

- Typing modular sessions with mixed choice and **asynchronous** communications
- Inspiration for conditions enabling to compose systems of Communicating Finite State Machines (mixed-choice harmful for that)
- What about projection?

Ongoing work

- Typing modular sessions with mixed choice and **asynchronous** communications
- Inspiration for conditions enabling to compose systems of Communicating Finite State Machines (mixed-choice harmful for that)
- What about projection?

Ongoing work

- Typing modular sessions with mixed choice and **asynchronous** communications
- Inspiration for conditions enabling to compose systems of Communicating Finite State Machines (mixed-choice harmful for that)
- What about projection?


Thank
you