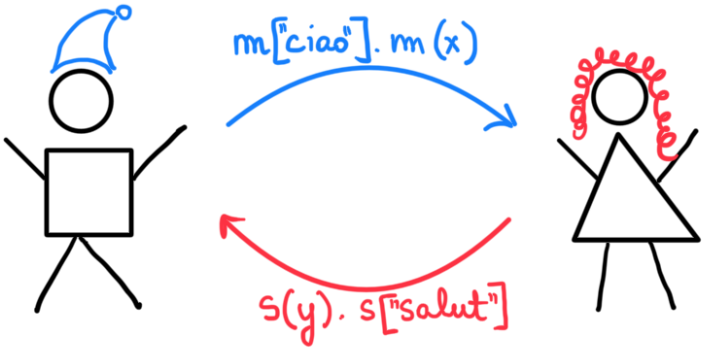
SESSION TYPES

Lecture 3 : Multiparty

MGS 2024

Matteo Acclavio ⊗ Sonia Marin



Multiparty session types "generalize" binary session types to the case of more than two participants.

The central idea is to introduce global types, which describe multiparty communication at a high level and provide a way to check protocol compliance

It was originally inspired by the design of an industrial language for protocol description.

This requires us to slightly change our perspective on what we have done so far.

Indeed we have looked at process terms (implementation) and questioned whether they could be typed, i.e. corresponded to any protocol description

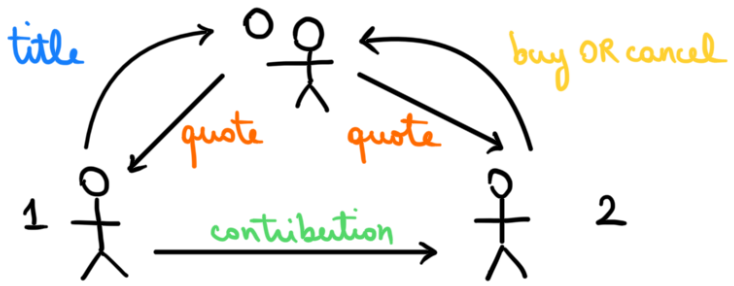
bottom-up approach

The alternative is to look at session types as specifications of the concurrent protocol we are interested in and check whether a given process term implements the expected communication.

top-down approach

The Two-Buyer Example

Two buyers B1 and B2 attempt to buy a book together from a seller S
- B1 sends the title of a book to S
- S sends a quote to both B1 and B2
- B1 sends to B2 the amount they wish to contribute
- B2 informs S of their choice, that is
  - either buy : B2 sends to S the address to post the book and ends transaction
  - or cancel : directly ends transaction



Let us try to devise binary session types to describe this protocol using 3 channels $x_0 x_1$, $y_1 y_2$, $z_2 z_0$.

$x_0 : S_0 = (\text{string}) \triangleright [\text{nat}] \triangleright \text{wait}$ $\quad x_1 : S_0^{\perp} = [\text{string}] \triangleright (\text{nat}) \triangleright \text{close}$

$y_1 : S_1 = [\text{nat}] \triangleright \text{wait}$ $\quad y_2 : S_1^{\perp} = (\text{nat}) \triangleright \text{close}$

$z_2 : S_2 = (\text{nat}) \triangleright \oplus\{\text{buy} : [\text{string}] \triangleright \text{close}, \text{cancel} : \text{close}\}$

$z_0 : S_2^{\perp} = [\text{nat}] \triangleright \&\{\text{buy} : (\text{string}) \triangleright \text{wait}, \text{cancel} : \text{wait}\}$

This would ideally specify the two-buyer protocol but it does not !

For example, it lets us implement both :

$P_{b1} = x_1[\text{title}] . x(\text{quote}) . y_1[\text{amount}] \ldots$

and $P_{b1}^{*} = y_1[\text{amount}] . x_1[\text{title}] . x_1(\text{quote}) \ldots$

↑ does not follow the high-level protocol description

The idea is to add a second layer of typing to have more control on the implementation of protocols
(and for example rule out P_{b1}^{*})

Multiparty session calculus

Base sets: expression variables denoted by $x, y, \ldots$
constant values $c \ldots$ ← nat, string ...
expressions $e ::= x \mid c \mid e + e$
labels $k, l \ldots$
session participants $p, q, \ldots$

Processes: We will simplify the syntax a lot in order to focus on the key aspects of multiparty

$P ::=$ inact

$\mid p[q, e].P$ ← p sends the value obtained by evaluating e to q and continues as P

$\mid p(q, x).P$ ← p receives a value from q and uses it in place of x when continuing as P

$\mid p \triangleright \{q, l : P_l\}_{l \in L}$ ← p offers a choice to q between different P_l to continue as

$\mid p \triangleleft q, k : P$ ← p selects a label from q and continues as P

We define a multiparty session as a parallel composition of pairs of participants and processes

$$M = p_1 \circ P_1 \mid \dots \mid p_n \circ P_n \quad =: \prod_{i \le n} p_i \circ P_i$$

with the intuition that process P_i plays the role of participant p_i and can interact with other processes playing other roles in M

A multiparty session is well formed if:

- all the participants are different, i.e., $p_i \neq p_j \quad \forall i \neq j$
- the only active participant in P_i is p_i
  i.e. only occurences of $p_i[\_]$, $p_i(\_)$, $p_i \triangleright \_$, $p_i \triangleleft \_$

You can think of a multiparty session as generalizing the binary session $(\nu pq)(P \mid Q)$ where $fv(P) = p$ and $fv(Q) = q$ to something like $(\nu p_1 \dots p_n)(P_1 \mid \dots \mid P_n)$ with $fv(P_i) = p_i$

Structural Congruence for multiparty sessions

$$\frac{P \equiv Q}{p \circ P \mid M \equiv p \circ Q \mid M} \qquad M \mid M' \equiv M' \mid M \qquad (M \mid M') \mid M'' \equiv M \mid (M' \mid M'')$$

Operational semantics for multiparty sessions

$$p \circ p[q, e].P \mid q \circ q(p, x).Q \mid M \longrightarrow p \circ P \mid q \circ Q[c/x] \mid M \quad \text{if } e \downarrow c$$

$$p \circ p \triangleright \{q, \ell : P_\ell\}_{\ell \in L} \mid q \circ q \triangleleft p, k : Q \mid M \longrightarrow p \circ P_k \mid q \circ Q \mid M \quad \text{if } k \in L$$

$$\frac{M_1 \equiv M_1' \qquad M_1 \longrightarrow M_2 \qquad M_2 \equiv M_2'}{M_1' \longrightarrow M_2'}$$

Multiparty Type System

Global types provide the high level specification of the communication

$G ::=$ end — termination of session

exchange $\mid p \to q : [T].\ G$ ← p sends a value of type T to participants q and then interactions happen as G intends

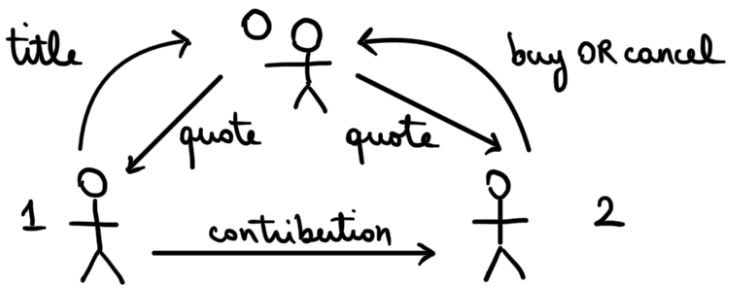
choice $\mid p \to q : \{l : G_l\}_{l \in L}$ ← p selects one of the labels in L from the options offered by q if l is chosen, interactions happen as G_l intends

The set of participants of G is $pt(G)$ defined as

$$pt(\text{end}) = \emptyset \qquad pt(p \to q : [T].\ G) = \{p, q\} \cup pt(G)$$

$$pt(p \to q : \{l : G_l\}_{l \in L}) = \{p, q\} \cup pt(G_l)$$

Example :



$$G = 1 \to 0 : [\text{string}].\ 0 \to 1 : [\text{nat}].\ 0 \to 2 : [\text{nat}].$$
$$1 \to 2 : [\text{nat}].\ 2 \to 0 : \{\text{buy} : 2 \to 0 : [\text{string}].\text{end},$$
$$\text{cancel} : \text{end}\}$$

Local session types describe the behaviour of a single participant in a multiparty session.

They are defined almost exactly as binary session types except they record the destination participant.

$$
\begin{aligned}
S ::= \quad & \mathrm{end} \\
& \mid [p, T] \blacktriangleleft S \quad \mid (p, T) \blacktriangleleft S \\
& \mid \&\{p, l : S_l\}_{l \in L} \quad \mid \oplus\{p, l : S_l\}_{l \in L}
\end{aligned}
$$

The projection of a global type onto a participant r is defined as

$$\mathrm{end}\,|_r = \mathrm{end}$$

$$
(p \to q : [T].\, G)\,|_r = \begin{cases} [q, T] \blacktriangleright G|_r & \text{if } r = p \\ (p, T) \blacktriangleright G|_r & \text{if } r = q \\ G|_r & \text{o/w} \end{cases}
$$

$$
(p \to q : \{l : G_l\}\,|_r = \begin{cases} \oplus\{q, l : G_l|_r\} & \text{if } r = p \\ \&\{p, l : G|_r\} & \text{if } r = q \\ G_k|_r & \text{where } k \in L \\ & \text{if } G_l|_r = G_{l'}|_r \text{ for all } l, l' \in L \end{cases}
$$

Example:

$$
\begin{aligned}
G = \; & 1 \to 0 : [\mathrm{string}].\; 0 \to 1 : [\mathrm{nat}].\; 0 \to 2 : [\mathrm{nat}]. \\
& 1 \to 2 : [\mathrm{nat}] \cdot\; 2 \to 0 : \left\{ \begin{array}{l} \mathrm{buy} : 2 \to 0 : [\mathrm{string}].\mathrm{end}, \\ \mathrm{cancel} : \mathrm{end} \end{array} \right\}
\end{aligned}
$$

$$G|_1 = [0, \mathrm{string}] \blacktriangleright (0, \mathrm{nat}) \blacktriangleright [2, \mathrm{nat}] \blacktriangleright \mathrm{end}$$

Note that there exist global types that cannot be projected on ALL their participants.

Type system

We say that a multiparty session is **well typed** if there exists a global type G such that $\vdash M : G$

$$\frac{\left\{ p_i : G|_{p_i} \vdash P_i \right\}_{i \in I}}{\vdash \prod_{i \leq n} p_i \circ P_i \; : \; G} \text{(MRES)} \qquad pt(G) \subseteq \{p_i\}_{i \in I}$$

$$\frac{\text{only } p : \text{end} \in \Gamma}{\Gamma \vdash \text{inact}} \text{(INACT)} \qquad \frac{\Gamma, y : T, p : S \vdash P}{\Gamma, p : (q, T) \blacktriangleleft S \vdash p\,(q, y).P} \text{(RECV)}$$

$$\frac{\Gamma \Vdash e : T \qquad \Gamma, p : S \vdash P}{\Gamma, p : [q, T] \blacktriangleleft S \vdash p\,[q, e].P} \text{(SEND)}$$

$$\frac{\left\{ \Gamma, p : S_\ell \vdash P_\ell \right\}_{\ell \in L}}{\Gamma, p : \&\{q, \ell : S_\ell\}_{\ell \in L} \vdash p \triangleright \{q, \ell : P_\ell\}_{\ell \in L}} \text{(BRA)}$$

$$\frac{\Gamma, p : S_k \vdash P}{\Gamma, x : \oplus\{q, \ell : S_\ell\}_{\ell \in L} \vdash p \triangleleft q, k : P} \text{(SEL)}_{k \in L}$$

Properties

Subject Reduction : If $\vdash M : G$ and $M \longrightarrow M'$
Then there exists G' such that $\vdash M' : G'$

where G' is structurally obtained from G as $G \Rightarrow G'$.

Progress: If $\vdash M : G$ then either $M \equiv (p \circ \text{inact})$
or there exists M' such that $M \longrightarrow M'$

As a consequence we get that well typed multiparty sessions **never get stuck**.

References:

today
- Yoshida and Gheri (2019)
  A very gentle introduction to multiparty session types
- Coppo et al. (2015)
  A gentle introduction to multiparty asynchronous session types

first paper
- Honda, Yoshida and Carbone (2008)
  Multiparty asynchronous session types

further reading
- Yoshida and Scalas (2019)
  Less is more: multiparty session types revisited