Multiparty Session Types, Beyond Duality

Alceste Scalas
Imperial College London
alceste.scalas@imperial.ac.uk

Nobuko Yoshida
Imperial College London
n.yoshida@imperial.ac.uk

Multiparty Session Types (MPST) are a well-established typing discipline for message-passing processes interacting on *sessions* involving two or more participants. Session typing can ensure desirable properties: absence of communication errors and deadlocks, and protocol conformance. However, existing MPST works provide a *subject reduction* result that is arguably (and sometimes, surprisingly) restrictive: it only holds for typing contexts with strong *duality* constraints on the interactions between pairs of participants. Consequently, many "intuitively correct" examples cannot be typed and/or cannot be proved type-safe. We illustrate some of these examples, and discuss the reason for these limitations. Then, we present a novel MPST typing system that removes these restrictions.

1 Introduction

The *Multiparty Session Types (MPST)* framework [7] provides a well-established typing discipline for *processes* interacting on *multiparty sessions*. In a nutshell, it fosters a top-down approach where:

(1) a *global type* G formalises a *choreography* involving n interacting *roles*;

(2) G is *projected* onto a set of *(local) session types* $S_1, \ldots, S_n$ (one per role of G);

(3) $S_1, \ldots, S_n$ are assigned to *channels*, used by *MPST π-calculus processes* that are type-checked.

This ensures properties such as type safety and protocol conformance (i.e., all sent/received messages comply with the session types, and ultimately with their originating global type G). E.g., consider:

$$G = \mathtt{p}{\to}\mathtt{q}{:}\begin{Bmatrix} \mathtt{m1}(\mathrm{Int}).\mathtt{q}{\to}\mathtt{r}{:}\mathtt{m2}(\mathrm{Str}).\mathtt{r}{\to}\mathtt{p}{:}\mathtt{m3}(\mathrm{Bool}).\mathbf{end}, \\ \mathtt{stop}.\mathtt{q}{\to}\mathtt{r}{:}\mathtt{quit}.\mathbf{end} \end{Bmatrix} \tag{1}$$

The global type G involves three roles: $\mathtt{p}$, $\mathtt{q}$, $\mathtt{r}$. It says that $\mathtt{p}$ sends to $\mathtt{q}$ *either* a message $\mathtt{m1}$ (carrying an Integer) *or* $\mathtt{stop}$; in the first case, $\mathtt{q}$ sends $\mathtt{m2}$ to $\mathtt{r}$ (carrying a String), then $\mathtt{r}$ sends $\mathtt{m3}$ to $\mathtt{p}$ (carrying a Boolean), and the session **end**s; otherwise, in the second case, $\mathtt{q}$ sends $\mathtt{quit}$ to $\mathtt{r}$, and the session **end**s.

The *projections of G* describe the I/O actions expected from processes that play the roles in G:

$$S_{\mathtt{p}} = \mathtt{q}\bigoplus\begin{Bmatrix}\mathtt{m1}(\mathrm{Int}).\mathtt{r}\&\mathtt{m3}(\mathrm{Bool}), \\ \mathtt{stop}\end{Bmatrix} \quad S_{\mathtt{q}} = \mathtt{p}\&\begin{Bmatrix}\mathtt{m1}(\mathrm{Int}).\mathtt{r}\oplus\mathtt{m2}(\mathrm{Str}), \\ \mathtt{stop}.\mathtt{r}\oplus\mathtt{quit}\end{Bmatrix} \quad S_{\mathtt{r}} = \mathtt{q}\&\begin{Bmatrix}\mathtt{m2}(\mathrm{Str}).\mathtt{p}\oplus\mathtt{m3}(\mathrm{Bool}), \\ \mathtt{quit}\end{Bmatrix} \tag{2}$$

Here, $S_{\mathtt{p}}$, $S_{\mathtt{q}}$, $S_{\mathtt{r}}$ are the projections of G resp. onto $\mathtt{p}$, $\mathtt{q}$, $\mathtt{r}$. E.g., $S_{\mathtt{p}}$ is a session type that represents the behaviour of $\mathtt{p}$ in G: it must send ($\oplus$) to $\mathtt{q}$ either $\mathtt{m1}(\mathrm{Int})$ or $\mathtt{stop}$; in the first case, the channel is then used to receive ($\&$) message $\mathtt{m3}(\mathrm{Bool})$ from $\mathtt{r}$, and the session ends; otherwise, in the second case, the session ends. Below is an example of MPST process matching such types: s is a *multiparty session*, and $s[\mathtt{p}]$ (resp. $s[\mathtt{q}]$, $s[\mathtt{r}]$) is a *channel with role* used to interact on s while playing the role of $\mathtt{p}$ (resp. $\mathtt{q}$, $\mathtt{r}$).

$$(\nu s)\left(P_{\mathtt{p}} \mid P_{\mathtt{q}} \mid P_{\mathtt{r}}\right) \quad \text{where} \begin{cases} P_{\mathtt{p}} = s[\mathtt{p}][\mathtt{q}]\oplus\mathtt{stop} \\ P_{\mathtt{q}} = s[\mathtt{q}][\mathtt{p}]\&\{\mathtt{m1}(x).s[\mathtt{q}][\mathtt{r}]\oplus\mathtt{m2}(\texttt{"Hi"}),\, \mathtt{stop}.s[\mathtt{q}][\mathtt{r}]\oplus\mathtt{quit}\} \\ P_{\mathtt{r}} = s[\mathtt{r}][\mathtt{q}]\&\{\mathtt{m2}(y).s[\mathtt{r}][\mathtt{p}]\oplus\mathtt{m3}(\mathit{true}),\, \mathtt{quit}\} \end{cases} \tag{3}$$

Here, $(\nu s)\left(P_{\mathtt{p}} \mid P_{\mathtt{q}} \mid P_{\mathtt{r}}\right)$ is the parallel composition of processes $P_{\mathtt{p}}$, $P_{\mathtt{q}}$, $P_{\mathtt{r}}$ in the scope of s. In $P_{\mathtt{p}}$, "$s[\mathtt{p}][\mathtt{q}]\oplus\mathtt{stop}$" means: $s[\mathtt{p}]$ is used to send $\mathtt{stop}$ to $\mathtt{q}$. $P_{\mathtt{q}}$ uses $s[\mathtt{q}]$ to receive either $\mathtt{m1}$ or $\mathtt{stop}$ from $\mathtt{p}$; then, in the first case it uses $s[\mathtt{q}]$ to send $\mathtt{m2}$ to $\mathtt{r}$; in the second case, it uses $s[\mathtt{q}]$ to send $\mathtt{quit}$ to $\mathtt{r}$.

DRAFT 22nd March 2017

Using the standard MPST typing rules, we have the following typing derivation:

$$\dfrac{\forall \mathtt{p}' \in \mathrm{roles}(G) = \{\mathtt{p},\mathtt{q},\mathtt{r}\} : \; S_{\mathtt{p}'} = G\restriction\mathtt{p}' \qquad \dfrac{\dfrac{\dfrac{\vdots}{\vdash P_{\mathtt{p}} \rhd s[\mathtt{p}]{:}S_{\mathtt{p}}} \quad \dfrac{\vdots}{\vdash P_{\mathtt{q}} \rhd s[\mathtt{q}]{:}S_{\mathtt{q}}}}{\vdash P_{\mathtt{p}} \,|\, P_{\mathtt{q}} \rhd s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}}\;[\textsc{Std-}|] \qquad \dfrac{\vdots}{\vdash P_{\mathtt{r}} \rhd s[\mathtt{r}]{:}S_{\mathtt{r}}}}{\vdash P_{\mathtt{p}} \,|\, P_{\mathtt{q}} \,|\, P_{\mathtt{r}} \rhd s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}, s[\mathtt{r}]{:}S_{\mathtt{r}}}\;[\textsc{Std-}|]}{\vdash (\nu s{:}G)\left(P_{\mathtt{p}} \,|\, P_{\mathtt{q}} \,|\, P_{\mathtt{r}}\right) \rhd \varnothing}\;[\textsc{Std-}\nu] \tag{4}$$

Here, s has type G, because in the premise of $[\textsc{Std-}\nu]$, $s[\mathtt{p}]$ (resp. $s[\mathtt{q}]$, $s[\mathtt{r}]$) has type $G\restriction\mathtt{p}$ (resp. $G\restriction\mathtt{q}$, $G\restriction\mathtt{r}$), i.e., the projection of G onto $\mathtt{p}$ (resp. $\mathtt{q}$, $\mathtt{r}$). The parallel process $P_{\mathtt{p}} | P_{\mathtt{q}} | P_{\mathtt{r}}$ is typed by rule $[\textsc{Std-}|]$, that splits the typing context so that a channel endpoint is not used by two parallel sub-processes. In the omitted part of the derivation, $P_{\mathtt{p}}$, $P_{\mathtt{q}}$, $P_{\mathtt{r}}$ are typed: they use resp. $s[\mathtt{p}]$, $s[\mathtt{q}]$, $s[\mathtt{r}]$ abiding by $S_{\mathtt{p}}$, $S_{\mathtt{q}}$, $S_{\mathtt{r}}$.

Inter-role dependencies Note that after sending `stop` to $\mathtt{q}$, $P_{\mathtt{p}}$ above terminates without receiving messages from $\mathtt{r}$ via $s[\mathtt{p}]$. Indeed, according to $S_{\mathtt{p}}$, $P_{\mathtt{p}}$ must input a message from $\mathtt{r}$ only after sending `m1` (instead of `stop`) to $\mathtt{q}$. Now, if we take $P_{\mathtt{r}}$, we see that it could potentially try to send `m3` to $\mathtt{p}$ — and this would be an error. However, such an error does *not* occur: by observing the whole ensemble $P_{\mathtt{p}} | P_{\mathtt{q}} | P_{\mathtt{r}}$, we see that when $P_{\mathtt{q}}$ receives `stop` from $P_{\mathtt{q}}$, it "forwards" `quit` to $\mathtt{r}$ (as per $S_{\mathtt{q}}$); and when $P_{\mathtt{r}}$ receives `quit` from $\mathtt{q}$, it does not try to communicate with $\mathtt{p}$ (as per $S_{\mathtt{r}}$). Hence, $S_{\mathtt{p}}$ makes $P_{\mathtt{p}}$ "assume" that by sending a certain message to the process playing role $\mathtt{q}$, it will also influence the process playing $\mathtt{r}$: this is an implicit *inter-role communication dependency*, that can be more explicitly observed in G.

Subject reduction Type safety is based on the *subject reduction* property: by showing that typed processes can only reduce to typed processes, we obtain that no (untypeable) error configurations can be reached. Hence, we would expect that subject reduction holds for our typed example above — roughly:

$$\vdash P \rhd \Gamma \text{ and } P \to^* P' \quad \text{implies} \quad \exists\Gamma' : \; \vdash P' \rhd \Gamma' \quad (\text{where } P = P_{\mathtt{p}} | P_{\mathtt{q}} | P_{\mathtt{r}} \text{ and } \Gamma = s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}, s[\mathtt{r}]{:}S_{\mathtt{r}}) \tag{5}$$

But surprisingly, *this is not the case!* In MPST works (e.g., [3]), the subject reduction statement reads:

$$\vdash P \rhd \Gamma \; \underline{\textit{with } \Gamma \textit{ consistent}} \quad \text{and} \quad P \to^* P' \quad \text{implies} \quad \exists\Gamma' \; \underline{\textit{consistent}} \text{ such that } \vdash P' \rhd \Gamma' \tag{6}$$

$$\text{where } \Gamma \textit{ is consistent} \text{ iff } \forall \left\{ s'[\mathtt{p}']{:}S_{\mathtt{p}'} , s'[\mathtt{q}']{:}S_{\mathtt{q}'} \right\} \subseteq \Gamma : \; S_{\mathtt{p}'}\restriction\mathtt{q}' \text{ and } S_{\mathtt{q}'}\restriction\mathtt{p}' \text{ are } \textit{dual} \; (\text{Def. A.5, A.4}) \tag{7}$$

i.e., subject reduction is *only proved for* consistent *(also called* coherent*) typing contexts*. The reason is technical: the proof of MPST subject reduction needs an invariant that **(R1)** makes the typing context "safe" (e.g., if the type of $s'[\mathtt{p}']$ sends a message to $\mathtt{q}'$, then the type of $s'[\mathtt{q}']$ can input such a message from $\mathtt{p}'$), **(R2)** is preserved when processes communicate and reduce, and **(R3)** is preserved when typing contexts are (de)composed by parallel composition (cf. $[\textsc{Std-}|]$ in (4)). Consistency satisfies **(R1)**–**(R3)** by imposing the *duality* of the partial projections $S_{\mathtt{p}'}\restriction\mathtt{q}'$ and $S_{\mathtt{q}'}\restriction\mathtt{p}'$, for each pair $\left\{s'[\mathtt{p}']{:}S_{\mathtt{p}'}, s'[\mathtt{q}']{:}S_{\mathtt{q}'}\right\} \subseteq \Gamma$ (cf. (7)). In our example, the pair $s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}$ is consistent: the outputs of $S_{\mathtt{p}}$ to $\mathtt{q}$ are dual w.r.t. the inputs of $S_{\mathtt{q}}$ from $\mathtt{p}$. However, the pair $s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{r}]{:}S_{\mathtt{r}}$ is *not consistent*: $S_{\mathtt{p}}\restriction\mathtt{r}$ and $S_{\mathtt{r}}\restriction\mathtt{p}$ are undefined—precisely because the inputs/outputs of $S_{\mathtt{p}}/S_{\mathtt{r}}$ from/to $\mathtt{r}/\mathtt{p}$ depend on previous I/O with $\mathtt{q}$! (See Ex. A.6) Spotting "inconsistencies" is tricky: it requires examining *all* pairs of projections of a choreography; further, choreographies with inter-role dependencies are often "inconsistent"—even simple ones, like our G in (1); and without consistency, subject reduction (and type safety) is not guaranteed.

Contributions We present a novel MPST typing system (Def. 3.3) whose subject reduction property (Theorem 3.6) holds for a larger set of processes w.r.t. existing MPST literature. Our typing rules *(1)* record and exploit "global" typing information through an additional "typing rely context", *(2)* require *"live"* (but *not* necessarily consistent) typing contexts, *(3)* do *not* require the existence of a global type (unlike $[\textsc{Std-}\nu]$ in (4)), and *(4)* subsume the "standard" MPST typing rules as special cases. We claim that our "liveness" condition is a reasonable, semantically-grounded characterisation of "correct" multiparty interactions, amenable for "bottom-up" choreography synthesis. Proofs are available in §C.

$$c, d ::= x \mid s[\mathtt{p}] \qquad P, Q ::= c[\mathtt{q}] \oplus \mathtt{m}(d)\,.\,P \;\mid\; c[\mathtt{q}] \&_{i \in I} \{\mathtt{m}_i(x_i)\,.\,P_i\} \;\mid\; P \mid Q \;\mid\; (\nu s)P \;\mid\; \mathbf{def}\, X(\widetilde{x}) = P \,\mathbf{in}\, Q \;\mid\; X\langle \widetilde{c} \rangle \;\mid\; \mathbf{0}$$

$$\text{[R-COMM]} \quad s[\mathtt{p}][\mathtt{q}] \&_{i \in I} \{\mathtt{m}_i(x_i)\,.\,P_i\} \mid s[\mathtt{q}][\mathtt{p}] \oplus \mathtt{m}_k(s'[\mathtt{r}])\,.\,Q \;\to\; P_k\{s'[\mathtt{r}]/x_k\} \mid Q \quad (k \in I)$$

$$\text{[R-}X\text{]} \quad \mathbf{def}\, X(x_1, \ldots, x_n) = P \,\mathbf{in}\, (X\langle s_1[\mathtt{p}_1], \ldots, s_n[\mathtt{p}_n] \rangle \mid Q) \;\to\; \mathbf{def}\, X(x_1, \ldots, x_n) = P \,\mathbf{in}\, (P\{s_1[\mathtt{p}_1]/x_1\} \cdots \{s_n[\mathtt{p}_n]/x_n\} \mid Q)$$

$$\text{[R-|]} \;\; P \to P' \;\text{ implies }\; P \mid Q \to P' \mid Q \qquad\qquad \text{[R-}\nu\text{]} \;\; P \to P' \;\text{ implies }\; (\nu s)P \to (\nu s)P'$$

$$\text{[R-}\mathbf{def}\text{]} \;\; P \to P' \;\text{ implies }\; \mathbf{def}\, X(\widetilde{x}) = Q \,\mathbf{in}\, P \;\to\; \mathbf{def}\, X(\widetilde{x}) = Q \,\mathbf{in}\, P'$$

Figure 1: Standard MPST π-calculus. Top: syntax. Bottom: semantics, up-to congruence $\equiv$ (Def. A.7).

2 Multiparty Sessions: Standard Calculus, Types and Typing Contexts

We first provide a series of standard MPST definitions, adopting a notation roughly based on [3]. For simplicity, we omit basic values (e.g., strings, integers, booleans...). Our new theory is presented in § 3.

Definition 2.1 (MPST π-Calculus)**.** *MPST processes have the syntax and semantics shown in Fig. 1.*

A *channel c or d* can be a *variable* x, or a *channel with role* $s[\mathtt{p}]$, used to play the *role* $\mathtt{p}$ in *session* s, and send/receive messages to/from other roles in s. A *process P or Q* can be: a *selection* that uses c to send a message $\mathtt{m}$ with *payload d* to $\mathtt{q}$, and continues as P; a *branching* that uses c to receive from $\mathtt{q}$ a message $\mathtt{m}_i$ (for any $i \in I$), binds x_i with its payload, and continues as P_i; a *parallel composition* where P, Q run concurrently (and possibly communicate); a *delimitation* of the scope of s to P; a *definition* of X (with *parameters* $\widetilde{x}$) as P, with scope Q; a *call* of X with *actual parameters* $\widetilde{c}$; a *terminated process* $\mathbf{0}$.

As expected, our MPST typing system (§ 3) will assign *session types* (Def. 2.2) to channels (Def. 2.1).

Definition 2.2 (Multiparty Session Types)**.** Session types *(ranged over by S, T) have the syntax:*

$$S, T \;::=\; \mathtt{p} \&_{i \in I} \mathtt{m}_i(S_i).S_i' \;\mid\; \mathtt{p} \oplus_{i \in I} \mathtt{m}_i(S_i).S_i' \;\mid\; \mathbf{end} \;\mid\; \mu\mathbf{t}.S \;\mid\; \mathbf{t} \qquad \textit{with } I \neq \emptyset \textit{ and } \forall i \in I : \mathrm{fv}(S_i) = \emptyset$$

where $\mathtt{m}_i$ *range over pairwise distinct* message labels. *We define* $\equiv$ *as the* coinductive equivalence between type trees*, such that:* $\mu\mathbf{t}.S \equiv S\{\mu\mathbf{t}.S/\mathbf{t}\}$ *(i.e., a recursive type is equivalent to its unfolding).*

The *branching type* (or *external choice*) $\mathtt{p} \&_{i \in I} \mathtt{m}_i(S_i).S_i'$ says that a channel must be used to receive from $\mathtt{p}$ one input of the form $\mathtt{m}_i(S_i)$, for any $i \in I$ chosen by $\mathtt{p}$; then, the channel must be used following the *continuation type* S_i'. The *selection type* (or *internal choice*) $\mathtt{p} \oplus_{i \in I} \mathtt{m}_i(S_i).S_i'$, instead, requires to use a channel to perform one output $\mathtt{m}_i(S_i)$ towards $\mathtt{p}$, for some $i \in I$, and continue using the channel according to S_i'. $\mathbf{end}$ is a *terminated* channel allowing no further inputs/outputs. $\mu\mathbf{t}.S$ is a *recursive* session type: μ binds the *recursion variable* $\mathbf{t}$ in S. Recursive types are *contractive*: i.e., in $\mu\mathbf{t}.S$ we have $S \neq \mathbf{t}'$ ($\forall \mathbf{t}'$).

Remark 2.3. *For brevity, we often omit* $\mathbf{0}$/$\mathbf{end}$ *in processes/types, and* $\mathbf{end}$*-typed message payloads.*

Remark 2.4. *Unlike most MPST works, our processes and types coalesce branching/input and selection/output. This reduces the amount of notation, without harming expressiveness: a "pure" input/output can be encoded as a singleton branch/select prefix, while a "pure" branching/selection can be represented by letting each choice carry an* $\mathbf{end}$*-typed channel.*

We adopt a standard definition of *MPST typing context* (Def. 2.5), slightly adapting the reduction $\to$.

Definition 2.5 (Typing Context)**.** *Let* Γ *be a partial mapping defined as:* $\Gamma ::= \emptyset \mid \Gamma, x{:}S \mid \Gamma, s[\mathtt{p}]{:}S$, *with the* composition Γ_1, Γ_2 *defined iff* $\mathrm{dom}(\Gamma_1) \cap \mathrm{dom}(\Gamma_2) = \emptyset$. *We write* $\Gamma \equiv \Gamma'$ *iff* $\mathrm{dom}(\Gamma) = \mathrm{dom}(\Gamma')$ *and* $\forall c \in \mathrm{dom}(\Gamma) : \Gamma(c) \equiv \Gamma'(c)$. *We define* $\Gamma \setminus c$ *so that* $(\Gamma \setminus c)(d) = \Gamma(d)$ *iff* $d \neq c$ *(and is undefined otherwise).*

The typing contexts reduction relation $\to$ *is inductively defined as shown on the right (where* $\maltese$ *stands for either* $\&$ *or* $\oplus$*).*

$$\frac{k \in I \qquad S_k \equiv T_k}{s[\mathtt{p}]{:}\mathtt{q} \oplus_{i \in I} \mathtt{m}_i(S_i).S_i',\, s[\mathtt{q}]{:}\mathtt{p} \&_{i \in I \cup J} \mathtt{m}_i(T_i).T_i' \;\to\; s[\mathtt{p}]{:}S_k', s[\mathtt{q}]{:}T_k'}$$

$$\frac{k \in I}{x{:}\mathtt{p} \maltese_{i \in I} \mathtt{m}_i(S_i).S_i' \to x{:}S_k'} \qquad \frac{\Gamma, c{:}S\{\mu\mathbf{t}.S/\mathbf{t}\} \to \Gamma'}{\Gamma, c{:}\mu\mathbf{t}.S \to \Gamma'} \qquad \frac{\Gamma \to \Gamma'}{\Gamma, c{:}S \to \Gamma', c{:}S}$$

An MPST typing context Γ maps variables and channels with roles to session types. The reduction relation $\to$ allows $s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}$ to "interact", provided that *(1)* $S_{\mathtt{p}}$ is a selection towards $\mathtt{q}$, *(2)* $S_{\mathtt{q}}$ is a branching from $\mathtt{p}$, and *(3)* the message labels of $S_{\mathtt{q}}$ are a superset of those in $S_{\mathtt{p}}$, with equivalent carried types. Clause *(3)* implies that if $S_{\mathtt{p}}$ has an "unreceivable" output message, then $s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}$ is stuck. Note that *we allow $x{:}S$ to reduce autonomously*: we add this minor extension to simplify our work in §3.

3 Multiparty Session Typing, Beyond Duality and Consistency

We now introduce our new MPST typing system. We start with a *co*inductive notion of *typing context liveness* (Def. 3.1), based on the reductions of MPST typing contexts (Def. 2.5): intuitively, it ensures that each selection is matched by a compatible branching, and each branching can receive a value from a compatible selection.

Definition 3.1 (Liveness)**.** *The predicate* $\mathrm{live}(\Gamma)$ *(read "*Γ is live*") is the largest predicate such that:*

- [L-&] $\mathrm{live}(\Gamma, s[\mathtt{p}]{:}S)$ *with* $S = \mathtt{q}\&_{i\in I}\mathtt{m}_i(S_i).S'_i$ *implies* $\exists i \in I : \exists\Gamma' : \Gamma, s[\mathtt{p}]{:}S \to^* \Gamma', s[\mathtt{p}]{:}S'_i$ *;*
- [L-⊕] $\mathrm{live}(\Gamma, s[\mathtt{p}]{:}S)$ *with* $S = \mathtt{q}\oplus_{i\in I}\mathtt{m}_i(S_i).S'_i$ *implies* $\forall i \in I : \exists\Gamma' : \Gamma, s[\mathtt{p}]{:}S \to^* \Gamma', s[\mathtt{p}]{:}S'_i$ *;*
- [L-μ] $\mathrm{live}(\Gamma, s[\mathtt{p}]{:}\mu\mathbf{t}.S)$ *implies* $\mathrm{live}(\Gamma, s[\mathtt{p}]{:}S\{\mu\mathbf{t}.S/\mathbf{t}\})$ *;*
- [L-$\to$] $\mathrm{live}(\Gamma)$ *and* $\Gamma \to \Gamma'$ *implies* $\mathrm{live}(\Gamma')$ *.*

Assume some live typing context $\Gamma, s[\mathtt{p}]{:}S$. By clause [L-&] of Def.3.1, if S is an external choice, then Γ must be able to reduce until *some* branch of S is triggered. By clause [L-⊕], if S is an internal choice, then Γ must be able to reduce allowing to send *each* message of S. Clause [L-μ] says that if S is recursive, its unfolding must be live, too. Clause [L-$\to$] ensures that if a live context reduces, the reduct is live, too.

Notably, a live typing context never deadlocks: i.e., if it cannot further reduce, then all its entries are **end**-typed. The *vice versa* is not true: a deadlock-free typing context might not be live. Moreover, liveness and consistency are incomparable, i.e., one does not imply the other, and *vice versa*.

Example 3.2. *The typing context* $s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}, s[\mathtt{r}]{:}S_{\mathtt{r}}$ *in* (4) *is live, but not consistent. The context* $s[\mathtt{p}]{:}\mathtt{q}\&\mathtt{m}_1.\mathtt{r}\oplus\mathtt{m}_3, s[\mathtt{q}]{:}\mathtt{r}\&\mathtt{m}_2.\mathtt{p}\oplus\mathtt{m}_1, s[\mathtt{r}]{:}\mathtt{p}\&\mathtt{m}_3.\mathtt{q}\oplus\mathtt{m}_2$ *is consistent but not live (nor deadlock-free): its inputs/outputs occur in the wrong order, thus preventing reductions. Finally, the typing context* $s[\mathtt{p}]{:}\mu\mathbf{t}.\mathtt{q}\oplus\mathtt{m}_1.\mathbf{t}, s[\mathtt{q}]{:}\mu\mathbf{t}.\mathtt{p}\&\mathtt{m}_1.\mathbf{t}, s[\mathtt{r}]{:}\mathtt{p}\&\mathtt{m}_2$ *is deadlock-free but not live:* $s[\mathtt{p}]$ *and* $s[\mathtt{q}]$ *generate infinite reductions, but* $s[\mathtt{r}]$ *cannot possibly perform its input (thus violating clause* [L-&] *of Def. 3.1).*

Definition 3.3 (MPST Typing Judgement)**.** *Let* Θ *be a partial mapping defined as:* $\Theta ::= \emptyset \mid \Theta, X{:}\widetilde{S}$ *An* MPST typing judgement *has the form:* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$ *with* $\mathrm{live}(\Gamma_g, \Gamma_r)$, *and is inductively defined by the rules in Fig. 2. Above,* Γ_g *and* Γ_r *are called respectively* guarantee *and* rely contexts.

Above, Θ assigns an n-uple of types to each process variable X (one type per argument). Intuitively, the judgement $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$ says: given the definitions in Θ, P "guarantees" to use its channels *linearly* according to Γ_g, by "relying" on the assumption that other processes use their channels according to Γ_r. The additional Γ_r is the most visible departure from standard MPST typing rules (albeit *we subsume them*: see p.6). Note: $\mathrm{live}(\Gamma_g, \Gamma_r)$ implies that Γ_g, Γ_r must be defined, i.e., $\mathrm{dom}(\Gamma_g) \cap \mathrm{dom}(\Gamma_r) = \emptyset$ (Def. 2.5).

In Fig. 2, the first three rules define auxiliary judgements: $\mathrm{end}(\Gamma)$ holds if all entries of Γ are **end**-typed; $\Gamma \vdash c{:}S$ holds if Γ only contains one entry $c{:}S'$, with $S' \equiv S$; $\Theta \vdash X : S_1, \ldots, S_n$ holds if Θ maps X to an n-uple of types equivalent to $S_1, \ldots, S_n$. The typing rule [T-**0**] says that $\mathbf{0}$ is typed if all channels in Γ_g are **end**-typed. By [T-**def**], $\mathbf{def}\, X(\widetilde{x}) = P\, \mathbf{in}\, Q$ is typed if P uses the arguments $x_1, \ldots, x_n$ according to $S_1, \ldots, S_n$ (with empty rely context), and the latter is the type of X when typing Q. By [T-X], $X\langle\widetilde{c}\rangle$ is typed if the types of $\widetilde{c}$ match those of the formal parameters of X, and any unused channel (in Γ_{g0}) is

$$\frac{\forall c \in \mathrm{dom}(\Gamma) : \Gamma(c) \equiv \mathbf{end}}{\mathrm{end}(\Gamma)} \qquad \frac{\Gamma \equiv c{:}S}{\Gamma \vdash c{:}S} \qquad \frac{\Theta(X) \equiv S_1,\ldots,S_n}{\Theta \vdash X : S_1,\ldots,S_n} \qquad \frac{\mathrm{end}(\Gamma_g)}{\Theta \vdash \mathbf{0} \rhd \Gamma_g \lhd \Gamma_r}\;[\text{T-}\mathbf{0}]$$

$$\frac{\Theta \vdash P \rhd x_1{:}S_1,\ldots,x_n{:}S_n \lhd \varnothing \quad \Theta, X{:}S_1,\ldots,S_n \vdash Q \rhd \Gamma_g \lhd \Gamma_r}{\Theta \vdash \mathbf{def}\, X(x_1,\ldots,x_n) = P \,\mathbf{in}\, Q \rhd \Gamma_g \lhd \Gamma_r}\;[\text{T-}\mathbf{def}] \qquad \frac{\Theta \vdash X : S_1,\ldots,S_n \quad \mathrm{end}(\Gamma_{g0}) \quad \forall i \in 1..n \quad \Gamma_{gi} \vdash c_i{:}S_i}{\Theta \vdash X\langle c_1,\ldots,c_n\rangle \rhd \Gamma_{g0},\Gamma_{g1},\ldots,\Gamma_{gn} \lhd \Gamma_r}\;[\text{T-}X]$$

$$\frac{\Gamma = \{s[\mathtt{p}]{:}S_{\mathtt{p}}\}_{\mathtt{p}\in I} \qquad \Theta \vdash P \rhd \Gamma_g,\Gamma \lhd \Gamma_r}{\Theta \vdash (\nu s{:}\Gamma)P \rhd \Gamma_g \lhd \Gamma_r}\;[\text{T-}\nu] \qquad \frac{\Theta \vdash P_1 \rhd \Gamma_1 \lhd \Gamma_r,\Gamma_2 \qquad \Theta \vdash P_2 \rhd \Gamma_2 \lhd \Gamma_r,\Gamma_1}{\Theta \vdash P_1 \,|\, P_2 \rhd \Gamma_1,\Gamma_2 \lhd \Gamma_r}\;[\text{T-}|]$$

$$\frac{S \equiv \mathtt{q}\&_{i\in I}\mathtt{m}_i(S_i).S'_i \qquad \forall i \in I \qquad \forall \Gamma'_r,\Gamma''_r : \Gamma_r \to^* \Gamma'_r \text{ and } \Gamma'_r,c{:}S \to \Gamma''_r,c{:}S'_i \qquad \Theta \vdash P_i \rhd \Gamma_g, y_i{:}S_i, c{:}S'_i \lhd \Gamma''_r}{\Theta \vdash c[\mathtt{q}]\&_{i\in I\cup J}\{\mathtt{m}_i(y_i).P_i\} \rhd \Gamma_g, c{:}S \lhd \Gamma_r}\;[\text{T-}\&]$$

$$\frac{S \equiv \mathtt{q}\oplus_{i\in I}\mathtt{m}_i(S_i).S'_i \quad \exists k \in I \quad \Gamma_{gd} \vdash d{:}S_k \quad \forall \Gamma'_r,\Gamma''_r : \Gamma_r \to^* \Gamma'_r \text{ and } \Gamma'_r,c{:}S \to \Gamma''_r,c{:}S'_k \quad \Theta \vdash P \rhd \Gamma_g, c{:}S'_k \lhd \Gamma''_r,\Gamma_{gd}}{\Theta \vdash c[\mathtt{q}]\oplus\mathtt{m}_k(d).P \rhd \Gamma_{gd},\Gamma_g, c{:}S \lhd \Gamma_r}\;[\text{T-}\oplus]$$

Figure 2: Our multiparty session typing rules.

end-typed. By [T-ν], $(\nu s)P$ is typed if P can be typed by adding Γ (that annotates s, and only has s-based entries) to Γ_g; note that since $\mathrm{live}(\Gamma_g,\Gamma_r)$, the premise $\Theta \vdash P \rhd \Gamma_g,\Gamma \lhd \Gamma_r$ implies $\mathrm{live}(\Gamma)$. By [T-|], $P_1 | P_2$ is typed by splitting the guarantee context in the premises (similarly to [Std-|] in (4))—but crucially, we record the "lost" information in the rely context, i.e., P_1 is typed by relying on the guarantees of P_2, and *vice versa*: this allows the typing rules to preserve liveness across a typing derivation. By [T-&], $c[\mathtt{q}]\&_{i\in I\cup J}\{\mathtt{m}_i(y_i).P_i\}$ is typed if c has type S, where S is a (possibly recursive) external choice from $\mathtt{q}$, with message labels $\mathtt{m}_i$ (where $i \in I$; as usual, the process can have more labels $\mathtt{m}_j$ with $j \in J$, which are immaterial); further, if Γ_r reduces to Γ''_r while moving $c{:}S$ to $c{:}S'_i$ (for any $i \in I$), then $c{:}S'_i$ and Γ''_r must type the continuation P_i, with the received channel $y_i{:}S_i$ added to the guarantee context — i.e., P_i must use it correctly. By [T-$\oplus$], $c[\mathtt{q}]\oplus\mathtt{m}_k(d).P$ is typed if c has type S, where S is a (possibly recursive) internal choice towards $\mathtt{q}$, whose message labels include $\mathtt{m}_k$; further, if Γ_r reduces to Γ''_r while moving $c{:}S$ to $c{:}S'_k$, then $c{:}S'_k$ and Γ''_r must type the continuation P — but with the sent channel $d{:}S_k$ now transferred into the rely context: i.e., after sending d, P cannot use it, but relies on its correct usage by the recipient.

Example 3.4. *We now revise the typing derivation* (4) *in §1 using our rules from Def. 3.3:*

$$\frac{\Gamma = s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}, s[\mathtt{r}]{:}S_{\mathtt{r}} \qquad \dfrac{\dfrac{\mathbb{D}\left\{\dfrac{\vdots}{\varnothing \vdash P_{\mathtt{p}} \rhd s[\mathtt{p}]{:}S_{\mathtt{p}} \lhd s[\mathtt{q}]{:}S_{\mathtt{q}}, s[\mathtt{r}]{:}S_{\mathtt{r}}}\right. \qquad \dfrac{\vdots}{\varnothing \vdash P_{\mathtt{q}} \rhd s[\mathtt{q}]{:}S_{\mathtt{q}} \lhd s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{r}]{:}S_{\mathtt{r}}}}{\varnothing \vdash P_{\mathtt{p}} | P_{\mathtt{q}} \rhd s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}} \lhd s[\mathtt{r}]{:}S_{\mathtt{r}}}\;[\text{T-}|] \qquad \dfrac{\vdots}{\varnothing \vdash P_{\mathtt{r}} \rhd s[\mathtt{r}]{:}S_{\mathtt{r}} \lhd s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}}}{\varnothing \vdash P_{\mathtt{p}} | P_{\mathtt{q}} | P_{\mathtt{r}} \rhd s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}, s[\mathtt{r}]{:}S_{\mathtt{r}} \lhd \varnothing}\;[\text{T-}|]}{\varnothing \vdash (\nu s{:}\Gamma)(P_{\mathtt{p}} | P_{\mathtt{q}} | P_{\mathtt{r}}) \rhd \varnothing \lhd \varnothing}\;[\text{T-}\nu]$$

We have $\mathrm{live}(\Gamma)$ *(see Ex. 3.2), and each application of* [T-|] *propagates the entries of* Γ *in its premises' guarantee/rely contexts, throughout the derivation. Now, consider the sub-derivation* $\mathbb{D}$ *(top left):*

$$\mathbb{D}\left\{\frac{S_{\mathtt{p}} \equiv \mathtt{q}\oplus\left\{\begin{matrix}\mathtt{m1}(\mathrm{Int}).\mathtt{r}\&\mathtt{m3}(\mathrm{Bool}),\\ \mathtt{stop}.\mathbf{end}\end{matrix}\right\} \quad \begin{matrix}s[\mathtt{q}]{:}S_{\mathtt{q}}, s[\mathtt{r}]{:}S_{\mathtt{r}},\\ s[\mathtt{p}]{:}S_{\mathtt{p}}\end{matrix} \to \begin{matrix}s[\mathtt{q}]{:}\mathtt{r}\oplus\mathtt{quit}, s[\mathtt{r}]{:}S_{\mathtt{r}},\\ s[\mathtt{p}]{:}\mathbf{end}\end{matrix} \quad \dfrac{\mathrm{end}(s[\mathtt{p}]{:}\mathbf{end})}{\varnothing \vdash \mathbf{0} \rhd s[\mathtt{p}]{:}\mathbf{end} \lhd s[\mathtt{q}]{:}\mathtt{r}\oplus\mathtt{quit}, s[\mathtt{r}]{:}S_{\mathtt{r}}}\;[\text{T-}\mathbf{0}]}{\varnothing \vdash s[\mathtt{p}][\mathtt{q}]\oplus\mathtt{stop}.\mathbf{0} \rhd s[\mathtt{p}]{:}S_{\mathtt{p}} \lhd s[\mathtt{q}]{:}S_{\mathtt{q}}, s[\mathtt{r}]{:}S_{\mathtt{r}}}\;[\text{T-}\oplus]\right.$$

Here, the rely context $s[\mathtt{q}]{:}S_{\mathtt{q}}, s[\mathtt{r}]{:}S_{\mathtt{r}}$ *cannot reduce (i.e., in* [T-$\oplus$] *in Fig. 2, we have* $\Gamma' = \Gamma_r$*). By composing it with* $s[\mathtt{p}]{:}S_{\mathtt{p}}$ *(that outputs* $\mathtt{stop}$ *like the process) we get the new rely context* $s[\mathtt{q}]{:}\mathtt{r}\oplus\mathtt{quit}, s[\mathtt{r}]{:}S_{\mathtt{r}}$*: by* [T-$\oplus$], *it must type the continuation* $\mathbf{0}$. *Note: in the instance of* [T-$\mathbf{0}$], *the rely context* never interacts with $s[\mathtt{p}]$, *albeit* $\mathtt{p}$ *occurs in* $S_{\mathtt{r}}$*: this captures the inter-role dependency discussed in §1.*

Subject reduction (without surprises) Our typing rules enjoy the *Substitution Lemma 3.5* below: if P is typed with an S-typed variable x, and a Γ_r containing an S-typed channel with role $s[\mathtt{p}]$, then $P\{s[\mathtt{p}]/x\}$ can be typed by *(1)* removing $s[\mathtt{p}]$ from Γ_r, and *(2)* using it in place of x in the guarantee context.

Lemma 3.5. *Let* $\Theta \vdash P \rhd \Gamma_g, x{:}S \lhd \Gamma_r$ *such that* $\Gamma_r \vdash s[\mathtt{p}]{:}S$*. Then,* $\Theta \vdash P\{{}^{s[\mathtt{p}]}/_{x}\} \rhd \Gamma_g, s[\mathtt{p}]{:}S \lhd \Gamma_r \setminus s[\mathtt{p}]$.

Lemma 3.5 allows to prove the *Subject Reduction Theorem 3.6* below. "As expected" (cf. (5) in § 1), it holds for all typed processes. The only condition, by Def. 3.3, is that rely/guarantee contexts must be live (Def. 3.1): this invariant (that replaces consistency) satisfies the requirements **(R1)**–**(R3)** seen in § 1.

Theorem 3.6. *If* $\Theta \vdash P \rhd \Gamma_g \lhd \Gamma_r$ *and* $P \to^* P'$*, then* $\exists \Gamma'_g$ *such that* $\Gamma_g \to^* \Gamma'_g$ *and* $\Theta \vdash P' \rhd \Gamma'_g \lhd \Gamma_r$.

From "new" to (simpler) "old" rules Rules [T-&]/[T-⊕] (Fig. 2) can be demanding: the quantification "$\forall \Gamma'_r : \Gamma_r \to^* \Gamma'_r \ldots$ can yield many premises (albeit they are finite: by Def. 2.5, the relation $\to$ induces finite-state systems). Lemma 3.7 allows to reduce the number of premises that must be checked explicitly.

Lemma 3.7 (Rely context strengthening)**.** *If* $\Theta \vdash P \rhd \Gamma_g \lhd \Gamma_r$ *and* $\Gamma_r \to^* \Gamma'_r$*, then* $\Theta \vdash P \rhd \Gamma_g \lhd \Gamma'_r$.

Intuitively, Lemma 3.7 holds because if a context reduces, it becomes "more deterministic"; hence, if P is typed relying on Γ_r, it can also rely on its reductions. Moreover, Γ_r can be simplified using Prop. 3.8.

Proposition 3.8. $\Theta \vdash P \rhd \Gamma_g \lhd \Gamma_r, x{:}S$ *iff* $\Theta \vdash P \rhd \Gamma_g \lhd \Gamma_r$. *(Note: by Def. 2.1, x cannot be a channel with role)*

By Prop. 3.8, variables in Γ_r are immaterial—intuitively, because they do *not* influence liveness (Def. 3.1). Lemma 3.7 and Prop. 3.8 become particularly useful when typing a process that branches/selects from/to a variable—and (in the latter case) sends a variable. In this case, rules [T-&] and [T-⊕] become the standard MPST typing rules for branch/input and select/output (see [3]), plus an unused rely context:

$$\frac{S \equiv \mathtt{q}\,\&_{i\in I}\mathtt{m}_i(S_i).S'_i \quad \forall i\in I \quad \Theta \vdash P_i \rhd \Gamma_g, y_i{:}S_i, x{:}S'_i \lhd \Gamma_r}{\Theta \vdash x[\mathtt{q}]\,\&_{i\in I\cup J}\{\mathtt{m}_i(y_i)\,.\,P_i\} \rhd \Gamma_g, x{:}S \lhd \Gamma_r}\;[\text{T\&}x] \qquad \frac{S \equiv \mathtt{q}\oplus_{i\in I}\mathtt{m}_i(S_i).S'_i \quad k\in I \quad \Gamma_{gy} \vdash y{:}S_k \quad \Theta \vdash P \rhd \Gamma_g, x{:}S'_k \lhd \Gamma_r}{\Theta \vdash x[\mathtt{q}] \oplus \mathtt{m}_k(y)\,.\,P \rhd \Gamma_{gy}, \Gamma_g, x{:}S \lhd \Gamma_r}\;[\text{T}\oplus x]$$

This is because in [T-⊕], Prop. 3.8 allows to omit Γ_{gy} from the rely context of the rightmost premise. Further, since $x{:}S$ reduces autonomously (cf. Def. 2.5), in both [T-&] and [T-⊕] the quantification "$\forall \Gamma'_r, \Gamma''_r : \Gamma_r \to^* \Gamma'_r$ and $\Gamma'_r, x{:}S \to \Gamma''_r, x{:}S'_i$" amounts to "$\forall \Gamma'_r : \Gamma_r \to^* \Gamma'_r$ and $\Gamma'_r, x{:}S \to \Gamma'_r, x{:}S'_i$"; hence, by Lemma 3.7, checking the rightmost premise *once* with Γ_r is sufficient to know that it holds $\forall \Gamma'_r$ such that $\Gamma_r \to^* \Gamma'_r$.

4 Conclusions and Discussion

We presented a new MPST typing system (Def. 3.3) providing subject reduction (Theorem 3.6) for a *larger* set of processes w.r.t. existing works. We record "global" typing information in a "rely context", thus removing the *consistency/duality* restrictions of previous MPST works.

On inter-role dependencies For the first time, we prove type safety of processes implementing choreographies with delegation and complex inter-role dependencies. (For more examples and discussion, in addition to § 1 and Ex. 3.4, see Ex. A.6 and § B). In this respect, to the best of our knowledge, the only comparable work is [6]; however, its process calculus only supports *one* session, and the paper crucially exploits this restriction to type parallel compositions without "splitting" them (cf. Table 8, rule [T-SESS]) and track a global type along reductions. Hence, unlike the present work, [6] does not support multiple sessions and delegation — and extending it to support them appears challenging. Moreover, we argue that our approach is simpler, since it does not depend on global types and makes them orthogonal w.r.t. typing rules.

On global types and liveness Our type system does *not* assume a global type for each session: it only (implicitly) requires that its *local* types are live (Def. 3.1), by rule [T-ν] (Fig. 2). Def. 3.1 is inspired by the notion of "safety" for Communicating Finite-State Machines (CFSMs) [2] (no deadlocks, orphan messages, unspecified receptions) and the idea of "MPSTs as CFSMs" [5]—that is *not* captured by previous MPST works, due to their consistency clause. The synchronous typing context reduction (Def. 2.5) implies decidable liveness; we plan to study the asynchronous setting. When needed, a set of local types can be related to a choreography either via "top-down" projection (§ 1), or "bottom-up" synthesis [8]. Similarly to most previous MPST works, our approach ensures that a typed process $(\nu s)\,(|_{\mathtt{p}\in I} P_{\mathtt{p}})$, with each $P_{\mathtt{p}}$ only interacting on $s[\mathtt{p}]$, is deadlock-free—but does not guarantee deadlock freedom in presence multiple interleaved sessions [1, 4]: we leave this topic as future work.

Relation with "standard" MPST typing As shown above, our typing rules become the "standard" ones if processes only use *variables*. We think that, by distinguishing a "static" process syntax (with variables only) from a "runtime" syntax (with variables *and* channels with roles), we could combine the "standard" typing rules for the former, with our typing rules for the latter (as required by Theorem 3.6).

Encodability [9] proves that consistency (7) creates a strong link between MPST and linear π-calculus (with *binary* channels), allowing to encode the former into the latter, and naturally guiding MPST implementations. The encodability of our "consistency-free" MPST processes (and their implementation) is now an open problem. Also, typed behavioural congruences and bisimulations need to be investigated.

Acknowledgements We thank the anonymous reviewers for their valuable comments and suggestions.

References

[1] L. Bettini, M. Coppo, L. D'Antoni, M. De Luca, M. Dezani-Ciancaglini & N. Yoshida (2008): *Global Progress in Dynamically Interleaved Multiparty Sessions*. In: *CONCUR*, doi:10.1007/978-3-540-85361-9˙33.

[2] D. Brand & P. Zafiropulo (1983): *On Communicating Finite-State Machines*. *JACM* 30(2), doi:10.1145/322374.322380.

[3] M. Coppo, M. Dezani-Ciancaglini, L. Padovani & N. Yoshida (2015): *A Gentle Introduction to Multiparty Asynchronous Session Types*. doi:10.1007/978-3-319-18941-3_4.

[4] M. Coppo, M. Dezani-Ciancaglini, N. Yoshida & L. Padovani (2016): *Global Progress for Dynamically Interleaved Multiparty Sessions*. *Mathematical Structures in Computer Science* 26(2), doi:10.1017/S0960129514000188.

[5] P. Deniélou & N. Yoshida (2013): *Multiparty Compatibility in Communicating Automata: Characterisation and Synthesis of Global Session Types*. In: *ICALP*, doi:10.1007/978-3-642-39212-2_18.

[6] M. Dezani-Ciancaglini, S. Ghilezan, S. Jakšić, J. Pantović & N. Yoshida (2016): *Precise subtyping for synchronous multiparty sessions*. In: *PLACES 2015*, doi:10.4204/EPTCS.203.3.

[7] K. Honda, N. Yoshida & M. Carbone (2008): *Multiparty asynchronous session types*. In: *POPL*, doi:10.1145/1328438.1328472. Full version: Volume 63, Issue 1, March 2016 (9), pages 1-67, *JACM*.

[8] J. Lange, E. Tuosto & N. Yoshida (2015): *From Communicating Machines to Graphical Choreographies*. In: *POPL*, doi:10.1145/2676726.2676964.

[9] A. Scalas, O. Dardha, R. Hu & N. Yoshida (2017): *A Linear Decomposition of Multiparty Sessions for Safe Distributed Programming*. Technical Report 2017/2, Imperial College London. Available here.

A Auxiliary Definitions and Details

This appendix contains several definitions taken from MPST literature, and further explanations on the example discussed in § 1 (see Ex. A.6).

A.1 Global, Local and Partial Session Types

Definition A.1 (Global Types)**.** *The syntax of a* Global type G *is:*

$$G \;::=\; \mathtt{p}{\to}\mathtt{q}{:}\{\mathtt{m_i}(T_i)\,.\,G_i\}_{i\in I} \;\Big|\; \mu\mathbf{t}.G \;\Big|\; \mathbf{t} \;\Big|\; \mathbf{end} \qquad \textit{where } \mathtt{p}\neq\mathtt{q},\ I\neq\emptyset,\ \textit{and } \forall i\in I:\mathrm{fv}(T_i)=\emptyset$$

The *global and local type projections* in Def. A.2 and Def. A.5 below, with their respective *merging operators*, are based on [11, 10]. Compared with simpler projection/merging definitions (e.g., those in [3]), they allow for more global types to be projectable, and more typing contexts to be deemed consistent[1]. Intuitively, the projection and merging in Def. A.2 below allow to "reconstruct" the expected behaviour of a role in a choreography, by collecting and combining its internal/external choices along different execution paths.

Definition A.2 (Projection of a Global Type)**.** *The* projection of G onto role $\mathtt{p}$, *written* $G{\upharpoonright}\mathtt{p}$, *is:*

$$(\mathtt{q}{\to}\mathtt{r}{:}\{\mathtt{m_i}(T_i).G_i\}_{i\in I}){\upharpoonright}\mathtt{p} = \begin{cases}\mathtt{r}\oplus_{i\in I}\mathtt{m}_i(T_i).(G_i{\upharpoonright}\mathtt{p}) & \textit{if } \mathtt{p}=\mathtt{q}\\ \mathtt{q}\&_{i\in I}\mathtt{m}_i(T_i).(G_i{\upharpoonright}\mathtt{p}) & \textit{if } \mathtt{p}=\mathtt{r}\\ \bigsqcap_{i\in I}G_i{\upharpoonright}\mathtt{p} & \textit{if } \mathtt{q}\neq\mathtt{p}\neq\mathtt{r}\end{cases} \qquad (\mu\mathbf{t}.G){\upharpoonright}\mathtt{p} = \begin{cases}\mu\mathbf{t}.(G{\upharpoonright}\mathtt{p}) & \textit{if } G{\upharpoonright}\mathtt{p}\neq\mathbf{t}'\ (\forall\mathbf{t}')\\ \mathbf{end} & \textit{otherwise}\end{cases}$$
$$\mathbf{t}{\upharpoonright}\mathtt{p} = \mathbf{t} \qquad \mathbf{end}{\upharpoonright}\mathtt{p} = \mathbf{end}$$

where $\bigsqcap$ *is the* merge operator for (local) session types, *defined as:*

$$\mathtt{p}\&_{i\in I}\mathtt{m}_i(S_i).S'_i \sqcap \mathtt{p}\&_{j\in J}\mathtt{m}_j(S_j).T'_j = \mathtt{p}\&_{k\in I\cap J}\mathtt{m}_k(S_k).(S'_k\sqcap T'_k)\ \&\ \mathtt{p}\&_{i\in I\setminus J}\mathtt{m}_i(S_i).S'_i\ \&\ \mathtt{p}\&_{j\in J\setminus I}\mathtt{m}_j(S_j).T'_j$$
$$\mathtt{p}\oplus_{i\in I}\mathtt{m}_i(S_i).S'_i \sqcap \mathtt{p}\oplus_{i\in I}\mathtt{m}_i(S_i).S'_i = \mathtt{p}\oplus_{i\in I}\mathtt{m}_i(S_i).S'_i$$
$$\mu\mathbf{t}.S\sqcap\mu\mathbf{t}.T = \mu\mathbf{t}.(S\sqcap T) \qquad \mathbf{t}\sqcap\mathbf{t}=\mathbf{t} \qquad \mathbf{end}\sqcap\mathbf{end}=\mathbf{end}$$

A.2 Partial Session Types and Duality

A session type S can be further projected onto a role $\mathtt{p}$, to "isolate" the interactions of S involving $\mathtt{p}$ from those involving other roles (Def. A.5 below). The result is a *partial session type* (Def. A.3 below). Partial session types feature branches, selections and recursion — but *no role annotations*: therefore, they are similar to *binary* session types (except that their payload types are *multiparty*), and endow a notion of *duality* (Def. A.4 below). As shown in §1, these technicalities are necessary to define *consistency* (cf. (7)).

Definition A.3 (Partial Session Types)**.** Partial session types, *ranged over by* H, *have the syntax:*

$$H ::= \&_{i\in I}\mathtt{m}_i(S_i).H_i \mid \oplus_{i\in I}\mathtt{m}_i(S_i).H_i \mid \mathbf{end} \mid \mu\mathbf{t}.H \mid \mathbf{t} \qquad \textit{with } I\neq\emptyset \textit{ and } \forall i\in I:\mathrm{fv}(S_i)=\emptyset$$

where $\mathtt{m}_i$ *range over pairwise distinct* message labels.

Definition A.4 (Duality of Partial Session Types)**.** *The* dual of a partial type H, *written* $\overline{H}$, *is:*

$$\overline{\&_{i\in I}\mathtt{m}_i(S_i).H_i} = \oplus_{i\in I}\mathtt{m}_i(S_i).\overline{H_i} \qquad \overline{\oplus_{i\in I}\mathtt{m}_i(S_i).H_i} = \&_{i\in I}\mathtt{m}_i(S_i).\overline{H_i} \qquad \overline{\mu\mathbf{t}.H}=\mu\mathbf{t}.\overline{H} \qquad \overline{\mathbf{t}}=\mathbf{t} \qquad \overline{\mathbf{end}}=\mathbf{end}$$

We say that H and H' are dual *iff* $\overline{H}=H'$.

Definition A.5 (Partial Projection)**.** *The* projection of S onto $\mathtt{p}$, *written* $S{\upharpoonright}\mathtt{p}$, *is a partial type defined as:*

$$(\mathtt{q}\&_{i\in I}\mathtt{m}_i(S_i).S'_i){\upharpoonright}\mathtt{p} = \begin{cases}\&_{i\in I}\mathtt{m}_i(S_i).(S'_i{\upharpoonright}\mathtt{p}) & \textit{if } \mathtt{p}=\mathtt{q}\\ \bigsqcap_{i\in I}S'_i{\upharpoonright}\mathtt{p} & \textit{if } \mathtt{p}\neq\mathtt{q}\end{cases} \qquad (\mathtt{q}\oplus_{i\in I}\mathtt{m}_i(S_i).S'_i){\upharpoonright}\mathtt{p} = \begin{cases}\oplus_{i\in I}\mathtt{m}_i(S_i).(S'_i{\upharpoonright}\mathtt{p}) & \textit{if } \mathtt{p}=\mathtt{q}\\ \bigsqcap_{i\in I}S'_i{\upharpoonright}\mathtt{p} & \textit{if } \mathtt{p}\neq\mathtt{q}\end{cases}$$
$$(\mu\mathbf{t}.S){\upharpoonright}\mathtt{p} = \begin{cases}\mu\mathbf{t}.(S{\upharpoonright}\mathtt{p}) & \textit{if } S{\upharpoonright}\mathtt{p}\neq\mathbf{t}'\ (\forall\mathbf{t}')\\ \mathbf{end} & \textit{otherwise}\end{cases} \qquad \mathbf{t}{\upharpoonright}\mathtt{p}=\mathbf{t} \qquad \mathbf{end}{\upharpoonright}\mathtt{p}=\mathbf{end}$$

where $\bigsqcap$ *is the* merge operator for partial session types, *defined as:*

$$\&_{i\in I}\mathtt{m}_i(S_i).H_i \sqcap \&_{i\in I}\mathtt{m}_i(S_i).H'_i = \&_{i\in I}\mathtt{m}_i(S_i).(H_i\sqcap H'_i)$$
$$\oplus_{i\in I}\mathtt{m}_i(S_i).H_i \sqcap \oplus_{j\in J}\mathtt{m}_j(S_j).H'_j = \oplus_{k\in I\cap J}\mathtt{m}_k(S_k).(H_k\sqcap H'_k)\oplus\oplus_{i\in I\setminus J}\mathtt{m}_i(S_i).H_i\oplus\oplus_{j\in J\setminus I}\mathtt{m}_j(S_j).H'_j$$
$$\mu\mathbf{t}.H\sqcap\mu\mathbf{t}.H' = \mu\mathbf{t}.(H\sqcap H') \qquad \mathbf{t}\sqcap\mathbf{t}=\mathbf{t} \qquad \mathbf{end}\sqcap\mathbf{end}=\mathbf{end}$$

[1]These merging operators introduce a subtle discrepancy between consistency and subject reduction, as stated in (6). This does *not* impact our treatment, since we lift the consistency requirement. For more details, see [9, §2.1, "On Consistency"]).

Note that Def. A.5 allows to merge different *internal* choices, but *not* different *external* choices. In the mechanics of "standard" MPST typing systems, this guarantees that a consistent typing context never reduces to configurations where an output from $\mathtt{p}$ to $\mathtt{q}$ is not matched by a corresponding input.

Example A.6. *Take $S_{\mathtt{p}}$, $S_{\mathtt{q}}$ and $S_{\mathtt{r}}$ from* (2) *in § 1. By Def. A.5, these partial projections are defined:*

$$S_{\mathtt{p}}\upharpoonright\mathtt{q} = \bigoplus\{\mathtt{m1}(\mathrm{Int})\,,\,\mathtt{stop}\} \qquad S_{\mathtt{q}}\upharpoonright\mathtt{p} = \binampersand\{\mathtt{m1}(\mathrm{Int})\,,\,\mathtt{stop}\}$$

Moreover, by Def. A.4, we have $S_{\mathtt{p}}\upharpoonright\mathtt{q} = \overline{S_{\mathtt{q}}\upharpoonright\mathtt{p}}$ *— i.e., the projections are dual. Therefore, by* (7)*, the pair* $s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{q}]{:}S_{\mathtt{q}}$ *is consistent.*

Now, consider the following projections, that are not *defined:*

$$\begin{array}{rcccl} S_{\mathtt{p}}\upharpoonright\mathtt{r} & = & \mathtt{r}\&\mathtt{m3}(\mathrm{Bool})\upharpoonright\mathtt{r} \sqcap \mathbf{end}\upharpoonright\mathtt{r} & = & \&\mathtt{m3}(\mathrm{Bool}) \sqcap \mathbf{end} \;\; \textit{(undefined)} \\ S_{\mathtt{r}}\upharpoonright\mathtt{p} & = & \mathtt{p}\oplus\mathtt{m3}(\mathrm{Bool})\upharpoonright\mathtt{p} \sqcap \mathbf{end}\upharpoonright\mathtt{p} & = & \oplus\mathtt{m3}(\mathrm{Bool}) \sqcap \mathbf{end} \;\; \textit{(undefined)} \end{array}$$

and therefore, by (7)*, the pair* $s[\mathtt{p}]{:}S_{\mathtt{p}}, s[\mathtt{r}]{:}S_{\mathtt{r}}$ *is* not *consistent.*

The reason for $S_{\mathtt{p}}\upharpoonright\mathtt{r}$ (resp. $S_{\mathtt{r}}\upharpoonright\mathtt{p}$) being undefined is that the session type $S_{\mathtt{p}}$ (resp. $S_{\mathtt{r}}$) starts with an interaction with $\mathtt{q}$ — and thus, when projecting the type onto $\mathtt{r}$ (resp. $\mathtt{p}$), Def. A.5 requires to "skip" the interaction with $\mathtt{q}$, project the continuations onto $\mathtt{r}$ (resp. $\mathtt{p}$), and merge them. The merge operation between partial types, however, is quite restrictive: it does not *allow to combine an branching (resp. selection) with* **end**. *These restrictions are in place to rule out erroneous communications (e.g., outputs from $\mathtt{p}$ to $\mathtt{r}$ that are not matched by corresponding input capabilities) that could be otherwise accepted by the "standard" MPST typing rules. The drawback is that the interaction between $\mathtt{p}$ (resp. $\mathtt{r}$) and $\mathtt{q}$ can only have a limited influence on the interactions between $\mathtt{p}$ (resp. $\mathtt{r}$) and other roles. This curtails the variety of inter-role dependencies supported by the "standard" MPST theory: see § 1 and § B.*

A.3 MPST Process Calculus

Definition A.7 (MPST π-Calculus Congruence)**.** *Let us define the* process declaration D *as:*

$$D \;\; ::= \;\; X(\widetilde{x}) = P$$

so that process definitions in Fig. 1 can be written as $\mathbf{def}\, D\, \mathbf{in}\, Q$*. We write* $\mathrm{fc}(P)$ *for the* free channels with roles in P*,* $\mathrm{dpv}(D)$ *for the* process variables declared in D*,* $\mathrm{fpv}(D)$ *for the* free process variables in D*, and* $\mathrm{fpv}(P)$ *for the* free process variables in P*. We write* $s \notin \mathrm{fc}(P)$ *to mean that there does* not *exist a $\mathtt{p}$ such that $s[\mathtt{p}] \in \mathrm{fc}(P)$. We define the* structural congruence $\equiv$ *as the smallest relation such that:*

$$P|Q \equiv Q|P \qquad (P|Q)|R \equiv P|(Q|R) \qquad P|\mathbf{0} \equiv P \qquad (\nu s)\mathbf{0} \equiv \mathbf{0} \qquad (\nu s)(\nu s')P \equiv (\nu s')(\nu s)P$$

$$(\nu s)(P|Q) \equiv P|(\nu s)Q \;\; \textit{if } s \notin \mathrm{fc}(P) \qquad \mathbf{def}\,D\,\mathbf{in}\,\mathbf{0} \equiv \mathbf{0} \qquad \mathbf{def}\,D\,\mathbf{in}\,(\nu s)P \equiv (\nu s)(\mathbf{def}\,D\,\mathbf{in}\,P) \;\; \textit{if } s \notin \mathrm{fc}(P)$$

$$\mathbf{def}\,D\,\mathbf{in}\,(P|Q) \equiv (\mathbf{def}\,D\,\mathbf{in}\,P)|Q \;\; \textit{if } \mathrm{dpv}(D) \cap \mathrm{fpv}(Q) = \emptyset$$

$$\mathbf{def}\,D\,\mathbf{in}\,(\mathbf{def}\,D'\,\mathbf{in}\,P) \equiv \mathbf{def}\,D'\,\mathbf{in}\,(\mathbf{def}\,D\,\mathbf{in}\,P) \;\; \textit{if } (\mathrm{dpv}(D) \cup \mathrm{fpv}(D)) \cap \mathrm{dpv}(D') = (\mathrm{dpv}(D') \cup \mathrm{fpv}(D')) \cap \mathrm{dpv}(D) = \emptyset$$

B More "Correct" but Non-Consistent Examples

We now reprise the projection/merging/consistency restrictions discussed in § 1 and Ex. A.6. We present more examples of "correct" choreographies that, when projected, yield typing contexts that are *not consistent* according to (7) — and therefore, do *not* provide subject reduction guarantees under the "standard" MPST theory. However, *all these examples are accepted by our theory* (i.e., can provide subject reduction guarantees, by Theorem 3.6) because *they all yield* live *typing contexts* (by Def. 3.1).

B.1 Consistency vs. History-Dependent Branching

In Ex. A.6, consistency does *not* hold because the partial projections $S_{\mathtt{p}} \upharpoonright \mathtt{r}$ and $S_{\mathtt{r}} \upharpoonright \mathtt{p}$ are undefined, due to the non-mergeability of internal/external choice and **end** (as per Def. A.5). We now try to "adjust" G in (1) and make it consistent, by adding another communication between $\mathtt{r}$ and $\mathtt{p}$:

$$G' = \mathtt{p}{\to}\mathtt{q}{:}\begin{Bmatrix} \mathtt{m1}(\mathtt{Int}) . \mathtt{q}{\to}\mathtt{r}{:}\mathtt{m2}(\mathtt{Str}) . \mathtt{r}{\to}\mathtt{p}{:}\mathtt{m3}(\mathtt{Bool}) . \mathbf{end} , \\ \mathtt{stop} . \mathtt{q}{\to}\mathtt{r}{:}\mathtt{quit} . \mathtt{r}{\to}\mathtt{p}{:}\mathtt{m4}(\mathtt{Int}) . \mathbf{end} \end{Bmatrix}$$

The projections of G' onto $\mathtt{p}$, $\mathtt{q}$ and $\mathtt{r}$ are respectively:

$$S'_{\mathtt{p}} = \mathtt{q} \bigoplus \begin{Bmatrix} \mathtt{m1}(\mathtt{Int}) . \mathtt{r}\&\mathtt{m3}(\mathtt{Bool}) , \\ \mathtt{stop} . \mathtt{r}\&\mathtt{m4}(\mathtt{Int}) \end{Bmatrix} \quad S'_{\mathtt{q}} = \mathtt{p} \,\&\! \begin{Bmatrix} \mathtt{m1}(\mathtt{Int}) . \mathtt{r}\oplus\mathtt{m2}(\mathtt{Str}) , \\ \mathtt{stop} . \mathtt{r}\oplus\mathtt{quit} \end{Bmatrix} \quad S'_{\mathtt{r}} = \mathtt{q} \,\&\! \begin{Bmatrix} \mathtt{m2}(\mathtt{Str}) . \mathtt{p}\oplus\mathtt{m3}(\mathtt{Bool}) , \\ \mathtt{quit} . \mathtt{p}\oplus\mathtt{m4}(\mathtt{Int}) \end{Bmatrix}$$

Let us now examine the partial projections $S'_{\mathtt{p}} \upharpoonright \mathtt{r}$ and $S'_{\mathtt{r}} \upharpoonright \mathtt{p}$:

$$\begin{array}{lllll} S'_{\mathtt{p}} \upharpoonright \mathtt{r} & = & \mathtt{r}\&\mathtt{m3}(\mathtt{Bool}) \upharpoonright \mathtt{r} \sqcap \mathtt{r}\&\mathtt{m4}(\mathtt{Int}) \upharpoonright \mathtt{r} & = & \&\mathtt{m3}(\mathtt{Bool}) \sqcap \&\mathtt{m4}(\mathtt{Int}) \qquad \text{(undefined)} \\ S'_{\mathtt{r}} \upharpoonright \mathtt{p} & = & \mathtt{p}\oplus\mathtt{m3}(\mathtt{Bool}) \upharpoonright \mathtt{p} \sqcap \mathtt{p}\oplus\mathtt{m4}(\mathtt{Int}) \upharpoonright \mathtt{p} & = & \oplus\mathtt{m3}(\mathtt{Bool}) \sqcap \oplus\mathtt{m4}(\mathtt{Int}) = \bigoplus\{\mathtt{m3}(\mathtt{Bool}) , \mathtt{m4}(\mathtt{Int})\} \end{array}$$

As noted in the comment of Def. A.5, partial type merging does not allow to combine different external choices, thus making $S'_{\mathtt{p}} \upharpoonright \mathtt{r}$ undefined. Hence, external choices cannot depend on the outcome of previous interactions with other roles: i.e., $\mathtt{p}$ cannot wait for different messages from $\mathtt{r}$, depending on the interaction between $\mathtt{p}$ and $\mathtt{q}$. However, different *internal* choices can be merged: e.g., $\mathtt{r}$ can send a different message to $\mathtt{p}$, depending on the previous interaction between $\mathtt{r}$ and $\mathtt{q}$. Hence, $S'_{\mathtt{r}} \upharpoonright \mathtt{p}$ is defined.

B.2 Consistency vs. Recursion

Consider the global type: $G'' = \mu\mathbf{t}.\mathtt{p}{\to}\mathtt{q}{:}\begin{Bmatrix} \mathtt{m1}(\mathtt{Int}) . \mathtt{q}{\to}\mathtt{r}{:}\mathtt{m2}(\mathtt{Str}) . \mathtt{r}{\to}\mathtt{p}{:}\mathtt{m3}(\mathtt{Bool}) . \mathbf{t} , \\ \mathtt{stop} . \mathtt{q}{\to}\mathtt{r}{:}\mathtt{quit} . \mathtt{r}{\to}\mathtt{p}{:}\mathtt{m3}(\mathtt{Bool}) . \mathbf{end} \end{Bmatrix}$

The global type G'' above is again similar to G in (1) — but now, if $\mathtt{p}$ sends $\mathtt{m1}(\mathtt{Int})$ to $\mathtt{q}$, the choreography loops: in fact, $\mathtt{q}$ then sends $\mathtt{m2}(\mathtt{Str})$ to $\mathtt{r}$, which sends $\mathtt{m3}(\mathtt{Bool})$ to $\mathtt{p}$, that continues recursively. Instead, if $\mathtt{p}$ chooses to send $\mathtt{stop}$ to $\mathtt{q}$, the latter notifies $\mathtt{quit}$ to $\mathtt{r}$, which sends $\mathtt{m3}(\mathtt{Bool})$ to $\mathtt{p}$, and the choreography **end**s. The projections of G'' onto $\mathtt{p}$, $\mathtt{q}$ and $\mathtt{r}$ are respectively:

$$S''_{\mathtt{p}} = \mu\mathbf{t}.\mathtt{q} \bigoplus \begin{Bmatrix} \mathtt{m1}(\mathtt{Int}).\mathtt{r}\&\mathtt{m3}(\mathtt{Bool}).\mathbf{t} , \\ \mathtt{stop}.\mathtt{r}\&\mathtt{m3}(\mathtt{Bool}).\mathbf{end} \end{Bmatrix} \quad S''_{\mathtt{q}} = \mu\mathbf{t}.\mathtt{p} \,\&\! \begin{Bmatrix} \mathtt{m1}(\mathtt{Int}).\mathtt{r}\oplus\mathtt{m2}(\mathtt{Str}).\mathbf{t} , \\ \mathtt{stop}.\mathtt{r}\oplus\mathtt{quit}.\mathbf{end} \end{Bmatrix} \quad S''_{\mathtt{r}} = \mu\mathbf{t}.\mathtt{q} \,\&\! \begin{Bmatrix} \mathtt{m2}(\mathtt{Str}).\mathtt{p}\oplus\mathtt{m3}(\mathtt{Bool}).\mathbf{t} , \\ \mathtt{quit}.\mathtt{p}\oplus\mathtt{m3}(\mathtt{Bool}).\mathbf{end} \end{Bmatrix}$$

Note that in all cases, $\mathtt{r}$ sends to $\mathtt{p}$ the message $\mathtt{m3}(\mathtt{Bool})$: this circumvents the merging problems described in Ex. A.6 and § B.1. However, we stumble into another roadblock:

$$\begin{array}{lllll} S''_{\mathtt{p}} \upharpoonright \mathtt{r} & = & \mu\mathbf{t}.((\mathtt{r}\&\mathtt{m3}(\mathtt{Bool}) . \mathbf{t}) \upharpoonright \mathtt{r} \sqcap (\mathtt{r}\&\mathtt{m3}(\mathtt{Bool}) . \mathbf{end}) \upharpoonright \mathtt{r}) & = & \mu\mathbf{t}.(\&\mathtt{m3}(\mathtt{Bool}) . (\mathbf{t} \sqcap \mathbf{end})) \quad \text{(undefined)} \\ S''_{\mathtt{r}} \upharpoonright \mathtt{p} & = & \mu\mathbf{t}.((\mathtt{p}\oplus\mathtt{m3}(\mathtt{Bool}) . \mathbf{t}) \upharpoonright \mathtt{p} \sqcap (\mathtt{p}\oplus\mathtt{m3}(\mathtt{Bool}) . \mathbf{end}) \upharpoonright \mathtt{p}) & = & \mu\mathbf{t}.(\oplus\mathtt{m3}(\mathtt{Bool}) . (\mathbf{t} \sqcap \mathbf{end})) \quad \text{(undefined)} \end{array}$$

This is another case of (unsupported) inter-role dependency: in $S''_{\mathtt{p}}$ (resp. $S''_{\mathtt{r}}$), the interaction with $\mathtt{q}$ determines whether, after receiving (resp. sending) $\mathtt{m3}(\mathtt{Bool})$ from $\mathtt{r}$ (resp. to $\mathtt{p}$), the type will *(a)* loop on $\mathbf{t}$, and thus, keep interacting with $\mathtt{r}$ (resp. $\mathtt{p}$), or *(b)* **end** the session. Hence, the partial projection $S''_{\mathtt{p}} \upharpoonright \mathtt{r}$ (resp. $S''_{\mathtt{r}} \upharpoonright \mathtt{p}$) tries to merge $\mathbf{t}$ (from case *(a)*) with **end** (from case *(b)*) — but the operation is undefined.

Additional references for the Appendix

[10] P. Deniélou, N. Yoshida, A. Bejleri & R. Hu (2012): *Parameterised Multiparty Session Types*. *Logical Methods in Computer Science*, doi:10.2168/LMCS-8(4:6)2012.

[11] N. Yoshida, P. Deniélou, A. Bejleri & R. Hu (2010): *Parameterised Multiparty Session Types*. In: *FoSSaCS*, doi:10.1007/978-3-642-12032-9˙10.

C Proofs

Proposition C.1. *If* $\mathrm{live}(\Gamma)$ *and* $\Gamma \to^* \Gamma'$, *then* $\mathrm{live}(\Gamma')$.

Proof. By induction on the number of reductions in $\Gamma \to^* \Gamma'$. The base case (0 reductions) is trivial, as it implies $\Gamma' = \Gamma$. The inductive case ($n+1$ reductions) is proved by the induction hypothesis, and clause [L-→] of Def. 3.1 (liveness). □

Proposition C.2. *For all* $\Gamma, \Gamma', \Gamma_0$ *such that* $\mathrm{dom}(\Gamma) \cap \mathrm{dom}(\Gamma_0) = \varnothing$*:* $\Gamma \to \Gamma'$ *if and only if* $\Gamma, \Gamma_0 \to \Gamma', \Gamma_0$, *with* Γ_0 not *reducing.*

Proof. ($\Longrightarrow$) By induction on the size of Γ_0, and by Def. 2.5.
($\Longleftarrow$) By induction on the size of Γ_0, and by Def. 2.5. □

Proposition C.3. *For all* $\Gamma, \Gamma', \Gamma_0$ *such that* $\mathrm{dom}(\Gamma) \cap \mathrm{dom}(\Gamma_0) = \varnothing$*:* $\Gamma \overbrace{\to \cdots \to}^{n\ steps} \Gamma'$ *if and only if* $\Gamma, \Gamma_0 \overbrace{\to \cdots \to}^{n\ steps} \Gamma', \Gamma_0$, *with* Γ_0 not *reducing.*

Proof. By induction on the number of reductions, using (C.2) in the inductive case. □

Proposition C.4. *Assume* $\mathrm{live}(\Gamma_0)$. *Then,* $\Gamma, \Gamma_0 \to \Gamma'$ *implies* $\Gamma' = \Gamma'', \Gamma''_0$ *such that either:*

(a) $\Gamma'' = \Gamma$ *and* $\Gamma_0 \to \Gamma''_0$, *or*

(b) $\Gamma' \to \Gamma''$ *and* $\Gamma''_0 = \Gamma_0$.

Proof. By Def. 3.1, noticing that we must have $\mathrm{dom}(\Gamma) \cap \mathrm{dom}(\Gamma_0) = \varnothing$ (otherwise, Γ, Γ_0 would be undefined by Def. 2.5); and since Γ_0 is live, it cannot interact with Γ. □

Proposition C.5. *Assume* $\mathrm{live}(\Gamma_0)$. *Then,* $\Gamma, \Gamma_0 \overbrace{\to \cdots \to}^{n\ steps} \Gamma'$ *implies* $\Gamma' = \Gamma'', \Gamma''_0$ *such that, for some* $m \leq n$, $\Gamma \overbrace{\to \cdots \to}^{m\ steps} \Gamma''$ *and* $\Gamma_0 \overbrace{\to \cdots \to}^{n-m\ steps} \Gamma''_0$; *moreover,* $\Gamma, \Gamma_0 \overbrace{\to \cdots \to}^{m\ steps} \Gamma'', \Gamma_0$.

Proof. By induction on n, using Prop. C.4.The *"moreover…"* part of the statement is an immediate consequence of the first part. □

Proposition C.6. *Assume* $\mathrm{live}(\Gamma_0)$ *and* $\mathrm{dom}(\Gamma) \cap \mathrm{dom}(\Gamma_0) = \varnothing$. *Then,* $\mathrm{live}(\Gamma, \Gamma_0)$ *if and only if* $\mathrm{live}(\Gamma)$.

Proof. ($\Longrightarrow$) We show that the following predicate:

$$\mathbb{P} = \{\Gamma \mid \mathrm{live}(\Gamma, \Gamma_0)\} \tag{8}$$

satisfies the clauses of Def. 3.1 (liveness). We proceed by examining each $\Gamma' \in \mathbb{P}$, reminding that:

$$\mathrm{dom}(\Gamma) \cap \mathrm{dom}(\Gamma_0) = \varnothing \qquad \text{(by hypothesis)} \tag{9}$$

We have the following (non-mutually exclusive) cases:

- $\Gamma = \Gamma', s[\mathtt{p}]{:}S$ with $S = \mathtt{q}\oplus_{i\in I}\mathtt{m}_i(S_i).S'_i$. We have:

$$\begin{aligned}
&\mathrm{live}(\Gamma,\Gamma_0) \quad\text{and thus}\quad \mathrm{live}(\Gamma', s[\mathtt{p}]{:}S, \Gamma_0) && \text{(by (8))} && (10)\\
&\forall i\in I : \exists\Gamma'' : \Gamma', s[\mathtt{p}]{:}S, \Gamma_0 \to^* \Gamma'', s[\mathtt{p}]{:}S'_i, \Gamma_0 && \text{(by (10), [L-}\oplus\text{] and Prop. C.5)} && (11)\\
&\forall i\in I : \exists\Gamma'' : \Gamma', s[\mathtt{p}]{:}S \to^* \Gamma'', s[\mathtt{p}]{:}S'_i && \text{(by (11) and Prop. C.3)} && (12)
\end{aligned}$$

  and by the shape of Γ and (12), we conclude that Γ satisfies clause [L-$\oplus$];

- $\Gamma = \Gamma', s[\mathtt{p}]{:}S$ with $S = \mathtt{q}\&_{i\in I}\mathtt{m}_i(S_i).S'_i$. We have:

$$\begin{aligned}
&\mathrm{live}(\Gamma,\Gamma_0) \quad\text{and thus}\quad \mathrm{live}(\Gamma', s[\mathtt{p}]{:}S, \Gamma_0) && \text{(by (8))} && (13)\\
&\exists i\in I : \exists\Gamma'' : \Gamma', s[\mathtt{p}]{:}S, \Gamma_0 \to^* \Gamma'', s[\mathtt{p}]{:}S'_i, \Gamma_0 && \text{(by (13), [L-\&] and Prop. C.5)} && (14)\\
&\exists i\in I : \exists\Gamma'' : \Gamma', s[\mathtt{p}]{:}S \to^* \Gamma'', s[\mathtt{p}]{:}S'_i && \text{(by (14) and Prop. C.3)} && (15)
\end{aligned}$$

  and by the shape of Γ and (15), we conclude that Γ satisfies clause [L-&];

- $\Gamma = \Gamma', s[\mathtt{p}]{:}\mu\mathbf{t}.S$. We have:

$$\begin{aligned}
&\mathrm{live}(\Gamma,\Gamma_0) \quad\text{and thus}\quad \mathrm{live}(\Gamma', s[\mathtt{p}]{:}\mu\mathbf{t}.S, \Gamma_0) && \text{(by (8))} && (16)\\
&\mathrm{live}(\Gamma', s[\mathtt{p}]{:}S\{\mu\mathbf{t}.S/\mathbf{t}\}, \Gamma_0) && \text{(by (16) and [L-}\mu\text{])} && (17)\\
&\Gamma', s[\mathtt{p}]{:}S\{\mu\mathbf{t}.S/\mathbf{t}\} \in \mathbb{P} && \text{(by (17) and (8))} && (18)
\end{aligned}$$

  and by the shape of Γ and (18), we conclude that Γ satisfies clause [L-μ];

- $\Gamma \to \Gamma'$. We have:

$$\begin{aligned}
&\Gamma,\Gamma_0 \to \Gamma',\Gamma_0 && \text{(by (9) and Prop. C.2)} && (19)\\
&\mathrm{live}(\Gamma,\Gamma_0) && \text{(by (8))} && (20)\\
&\mathrm{live}(\Gamma',\Gamma_0) && \text{(by (20), (19) and [L-}\to\text{])} && (21)\\
&\Gamma' \in \mathbb{P} && \text{(by (21) and (8))} && (22)
\end{aligned}$$

  and by the hypothesis $\Gamma \to \Gamma'$ and (22), we conclude that Γ satisfies clause [L-$\to$].

We have proved that $\mathbb{P}$ satisfies all clauses of Def. 3.1 (liveness). Since live($\cdot$) is the largest predicate satisfying such clauses, we have that $\forall\Gamma : \Gamma\in\mathbb{P}$ implies live(Γ); and since $\forall\Gamma : \mathrm{live}(\Gamma,\Gamma_0)$ implies $\Gamma\in\mathbb{P}$, we conclude that $\forall\Gamma : \mathrm{live}(\Gamma,\Gamma_0)$ implies live(Γ).

($\Longleftarrow$) We show that the following predicate:

$$\mathbb{P} = \{\Gamma,\Gamma_0 \mid \mathrm{live}(\Gamma)\} \qquad (23)$$

satisfies the clauses of Def. 3.1 (liveness). We proceed similarly to the case above ($\Longrightarrow$): by examining each $\Gamma,\Gamma_0 \in \mathbb{P}$. Then, we know that live($\cdot$) is the largest predicate satisfying all clauses of Def. 3.1 (liveness); therefore, we have that $\forall\Gamma : \Gamma\in\mathbb{P}$ implies live(Γ); and since $\forall\Gamma : \mathrm{live}(\Gamma)$ implies $\Gamma,\Gamma_0 \in \mathbb{P}$, we conclude that $\forall\Gamma : \mathrm{live}(\Gamma)$ implies live(Γ,Γ_0). □

Proposition C.7. *If* end(Γ), *then* live(Γ).

Proof. By Def. 3.3, we know that $\forall c \in \mathrm{dom}(\Gamma) : \Gamma(c) = \mathbf{end}$: therefore, Γ (vacuously) satisfies clauses [L-&], [L-$\oplus$] and [L-μ] of Def. 3.1 (liveness). Furthermore, by Def. 2.5 (reductions of Γ), we have $\Gamma \not\to$: hence, Γ also (vacuously) satisfies clause [L-$\to$] of Def. 3.1 (liveness). We conclude live(Γ). □

Proposition C.8. $\mathrm{live}(\Gamma, x{:}S)$ *if and only if* $\mathrm{live}(\Gamma)$.

Proof. Straightforward by Def. 3.1 (liveness), noticing that $x{:}S$ is not relevant for the definition (which only considers channels with roles). □

Proposition 3.8. $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r, x{:}S$ *iff* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$. *(Note: by Def. 2.1, x cannot be a channel with role)*

Proof. By induction on the derivation of $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r, x{:}S$, using Prop. C.8 in the base cases. □

Corollary C.9. $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$ *iff* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r \setminus x$

Proof. If $x \notin \mathrm{dom}(\Gamma_r)$, then the statement holds vacuously. Otherwise, if $x \in \mathrm{dom}(\Gamma_r)$, it is a direct consequence of Prop. 3.8. □

Lemma 3.5. *Let* $\Theta \vdash P \triangleright \Gamma_g, x{:}S \triangleleft \Gamma_r$ *such that* $\Gamma_r \vdash s[\mathtt{p}]{:}S$. *Then,* $\Theta \vdash P\{s[\mathtt{p}]/x\} \triangleright \Gamma_g, s[\mathtt{p}]{:}S \triangleleft \Gamma_r \setminus s[\mathtt{p}]$.

Proof. Assume the hypotheses. We have:

$$\Theta \vdash P \triangleright \Gamma_g, x{:}S \triangleleft \Gamma_r \qquad \text{(by hypothesis)} \qquad (24)$$
$$\Gamma_r \vdash s[\mathtt{p}]{:}S \qquad \text{(by hypothesis)} \qquad (25)$$
$$\mathrm{live}(\Gamma_g, \Gamma_r) \qquad \text{(by hypothesis (well-formedness) and Prop. C.8)} \qquad (26)$$

Note that, in the judgement in the conclusion, the liveness requirement of the guarantee/rely contexts (Def. 3.3) is trivially satisfied:

$$\mathrm{live}(\Gamma_g, s[\mathtt{p}]{:}S, (\Gamma_r \setminus s[\mathtt{p}])) \qquad \text{(by (26), (25), Def. 3.3 and Def. 3.1)} \qquad (27)$$

We proceed by induction on the derivation of $\Theta \vdash P \triangleright \Gamma_g, x{:}S \triangleleft \Gamma_r$:

- base case [T-**0**]. We have:

$$P = \mathbf{0} \quad \text{and} \quad \mathrm{end}(\Gamma_g, x{:}S) \qquad \text{(from the rule conclusion and premises)} \qquad (28)$$
$$S = \mathbf{end} \qquad \text{(by (28) and Def. 3.3)} \qquad (29)$$
$$\mathrm{end}(\Gamma_g) \qquad \text{(by (28) and Def. 3.3)} \qquad (30)$$
$$\mathrm{end}(\Gamma_g, s[\mathtt{p}]{:}S) \qquad \text{(by (30), (29) and Def. 3.3)} \qquad (31)$$
$$P\{s[\mathtt{p}]/x\} \;=\; \mathbf{0}\{s[\mathtt{p}]/x\} \;=\; \mathbf{0} \qquad \text{(from (28))} \qquad (32)$$

$$\dfrac{\mathrm{end}(\Gamma_g, s[\mathtt{p}]{:}S)}{\Theta \vdash P\{s[\mathtt{p}]/x\} \triangleright \Gamma_g, s[\mathtt{p}]{:}S \triangleleft \Gamma_r \setminus s[\mathtt{p}]}\;[\text{T-}\mathbf{0}] \qquad \text{(by (31), (32) and (27))}$$

- base case [T-X]. We have:

$$\dfrac{\Theta \vdash X : S_1, \dots, S_n \quad \mathrm{end}(\Gamma_{g0}) \quad \forall i \in 1..n \quad \Gamma_{gi} \vdash c_i{:}S_i}{\Theta \vdash P \triangleright \Gamma_g, x{:}S \triangleleft \Gamma_r}\;[\text{T-}X] \qquad \text{(from the rule definition)} \qquad (33)$$
$$P = X\langle c_1, \dots, c_n\rangle \qquad \text{(from the conclusion of (33))} \qquad (34)$$
$$\Gamma_g, x{:}S = \Gamma_{g0}, \Gamma_{g1}, \dots, \Gamma_{gn} \qquad \text{(by (33))} \qquad (35)$$

Now, from (35) and by Def. 3.3, we have two cases:

- $\Gamma_{g0} \vdash x{:}S$. Then:

$$\Gamma_{g0} = x{:}S \text{ and } S \equiv \textbf{end} \qquad \text{(by (33) and Def. 3.3)} \quad (36)$$
$$\text{end}(s[\mathtt{p}]{:}S) \qquad \text{(by (36) and Def. 3.3)} \quad (37)$$
$$\forall i \in 1..n:\ c_i \neq x \qquad \text{(by (36), (35) and Def. 2.5)} \quad (38)$$
$$P\{s[\mathtt{p}]/x\} = P \qquad \text{(by (34) and (38))} \quad (39)$$

$$\dfrac{\Theta \vdash X : S_1,\ldots,S_n \quad \text{end}(s[\mathtt{p}]{:}S) \quad \forall i \in 1..n \quad \Gamma_{gi} \vdash c_i{:}S_i}{\Theta \vdash P\{s[\mathtt{p}]/x\} \triangleright \Gamma_g, s[\mathtt{p}]{:}S \triangleleft \Gamma_r \backslash s[\mathtt{p}]}\ [\text{T-}X] \qquad \text{(by (36), (37) (39))}$$

- $\Gamma_{gk} \vdash x{:}S$, for some $k \in 1..n$. Then:

$$\Gamma_{gk} \equiv x{:}S \qquad \text{(by (33))} \quad (40)$$
$$c_k = x \text{ and } \forall i \in 1..n:\ i \neq k \text{ implies } c_i \neq x \qquad \text{(by (35), (40) and Def. 2.5)} \quad (41)$$
$$P\{s[\mathtt{p}]/x\} = X\langle c_1,\ldots,c_{k-1}, s[\mathtt{p}], c_{k+1},\ldots,c_n\rangle \qquad \text{(by (34) and (41))} \quad (42)$$

$$\dfrac{\begin{array}{c}\Theta \vdash X : S_1,\ldots,S_n \quad \text{end}(\Gamma_{g0}) \quad s[\mathtt{p}]{:}S \vdash s[\mathtt{p}]{:}S \\ \forall i \in \{1..n\} \setminus \{k\} \quad \Gamma_{gi} \vdash c_i{:}S_i\end{array}}{\Theta \vdash P\{s[\mathtt{p}]/x\} \triangleright \Gamma_g, s[\mathtt{p}]{:}S \triangleleft \Gamma_r \backslash s[\mathtt{p}]}\ [\text{T-}X] \qquad \text{(by (40), (42))}$$

- inductive case [T-&]. In this case, P is a branching on some variable. We have two cases:
  (a) P is a branching on some $y \neq x$. Then, the statement holds by the induction hypothesis, and rule [T-&]; or
  (b) P is a branching on x, i.e., x is substituted by channel with role $s[\mathtt{p}]$.

  We detail the proof for the latter (and most interesting) case. We have:

$$P = x[\mathtt{q}] \&_{i \in I \cup J} \{\mathtt{m}_i(y_i) . P_i\} \ \wedge\ S \equiv \mathtt{q} \&_{i \in I} \mathtt{m}_i(S_i).S_i' \qquad \text{(from rule conclusion/premises)} \quad (43)$$
$$\Gamma_r \vdash s[\mathtt{p}]{:}S = s[\mathtt{p}]{:}\mathtt{q} \&_{i \in I} \mathtt{m}_i(S_i).S_i' \qquad \text{(by (25) and (43))} \quad (44)$$
$$\forall i \in I:\ \forall \Gamma_r', \Gamma_r'':\ \Gamma_r \to^* \Gamma_r' \text{ and } \Gamma_r', x{:}S \to \Gamma_r'', x{:}S_i'$$
$$\Gamma_r'' = \Gamma_r' \qquad \text{(by Def. 2.5)} \quad (45)$$
$$\Theta \vdash P_i \triangleright \Gamma_g, y_i{:}S_i, x{:}S_i' \triangleleft \Gamma_r' \qquad \text{(from the rule premises and (45))} \quad (46)$$
$$\forall i \in I:\ \forall \Gamma_r', \Gamma_r'' \text{ such that } \begin{cases} \Gamma_r \backslash s[\mathtt{p}] \to^* \Gamma_r' \text{ and} \\ \Gamma_r', s[\mathtt{p}]{:}S \to \Gamma_r'', s[\mathtt{p}]{:}S_i' \text{ and} \\ \Gamma_r'', s[\mathtt{p}]{:}S_i', x{:}S \to \Gamma_r'', s[\mathtt{p}]{:}S_i', x{:}S_i' : \end{cases}$$
$$\Theta \vdash P_i \triangleright \Gamma_g, y_i{:}S_i, x{:}S_i' \triangleleft \Gamma_r'', s[\mathtt{p}]{:}S_i' \qquad \text{(by (46))} \quad (47)$$
$$\Gamma_r'', s[\mathtt{p}]{:}S_i' \vdash s[\mathtt{p}]{:}S_i' \qquad \text{(by Def. 3.3)} \quad (48)$$
$$\Theta \vdash P_i\{s[\mathtt{p}]/x\} \triangleright \Gamma_g, y_i{:}S_i, s[\mathtt{p}]{:}S_i' \triangleleft \Gamma_r'' \qquad \text{(by (47), (48) and i.h.)} \quad (49)$$

$$\dfrac{\begin{array}{c} S \equiv \mathtt{q} \&_{i \in I} \mathtt{m}_i(S_i).S_i' \qquad \forall i \in I \\ \forall \Gamma_r', \Gamma_r'':\ \Gamma_r \backslash s[\mathtt{p}] \to^* \Gamma_r' \text{ and } \Gamma_r', s[\mathtt{p}]{:}S \to \Gamma_r'', s[\mathtt{p}]{:}S_i' \\ \Theta \vdash P_i\{s[\mathtt{p}]/x\} \triangleright \Gamma_g, y_i{:}S_i, s[\mathtt{p}]{:}S_i' \triangleleft \Gamma_r'' \end{array}}{\Theta \vdash s[\mathtt{p}][\mathtt{q}] \&_{i \in I \cup J} \{\mathtt{m}_i(y_i) . P_i\{s[\mathtt{p}]/x\}\} \triangleright \Gamma_g, s[\mathtt{p}]{:}S \triangleleft \Gamma_r \backslash s[\mathtt{p}]}\ [\text{T-}\&] \qquad \big(\text{by (43), (49)}\big)$$

- inductive case [T-⊕]. In this case, P is a selection on some variable. We have two cases:

(a) P is a selection on some $y \neq x$. Then, the statement holds by the induction hypothesis, and rule [T-⊕]; or

(b) P is a selection on x, i.e., x is substituted by channel with role $s[\mathtt{p}]$.

We detail the proof for the latter (and most interesting) case. We have:

$$
\begin{array}{llr}
\exists k \in I:\; P = x[\mathtt{q}] \oplus \mathtt{m}_k(d)\,.\,P \;\wedge\; S \equiv \mathtt{q} \bigoplus_{i\in I} \mathtt{m}_i(S_i).S'_i & \text{(from rule conclusion/premises)} & (50)\\
\quad \Gamma_g = \Gamma_{g0}, \Gamma_{gd} \;\text{ where }\; \Gamma_{gd} \vdash d{:}S_k & \text{(from rule conclusion/premises)} & (51)\\
\quad \Gamma_r \vdash s[\mathtt{p}]{:}S = s[\mathtt{p}]{:}\mathtt{q} \bigoplus_{i\in I} \mathtt{m}_i(S_i).S'_i & \text{(by (25) and (50))} & (52)\\
\quad \forall \Gamma'_r, \Gamma''_r : \Gamma_r \to^* \Gamma'_r \;\text{ and }\; \Gamma'_r, x{:}S \to \Gamma''_r, x{:}S'_k & & \\
\qquad \Gamma''_r = \Gamma'_r & \text{(by Def. 2.5)} & (53)\\
\qquad \Theta \vdash P \rhd \Gamma_{g0}, x{:}S'_k \lhd \Gamma'_r, \Gamma_{gd} & \text{(from the rule premises and (53))} & (54)
\end{array}
$$

$$
\forall \Gamma'_r, \Gamma''_r \;\text{ such that }\; \left\{\begin{array}{l}
\Gamma_r \backslash s[\mathtt{p}] \to^* \Gamma'_r \;\text{ and}\\
\Gamma'_r, s[\mathtt{p}]{:}S \to \Gamma''_r, s[\mathtt{p}]{:}S'_k \;\text{ and}\\
\Gamma''_r, s[\mathtt{p}]{:}S'_k, x{:}S \to \Gamma''_r, s[\mathtt{p}]{:}S'_k, x{:}S'_k :
\end{array}\right.
$$

$$
\begin{array}{llr}
\qquad \Theta \vdash P \rhd \Gamma_{g0}, x{:}S'_k \lhd \Gamma''_r, s[\mathtt{p}]{:}S'_k, \Gamma_{gd} & \text{(by (54))} & (55)\\
\qquad \Gamma''_r, s[\mathtt{p}]{:}S'_k, \Gamma_{gd} \vdash s[\mathtt{p}]{:}S'_k & \text{(by Def. 3.3)} & (56)\\
\qquad \Theta \vdash P\{^{s[\mathtt{p}]}/_{x}\} \rhd \Gamma_{g0}, s[\mathtt{p}]{:}S'_k \lhd \Gamma''_r, \Gamma_{gd} & \text{(by (55), (56) and i.h.)} & (57)
\end{array}
$$

$$
\dfrac{\begin{array}{c}
S \equiv \mathtt{q} \&_{i\in I} \mathtt{m}_i(S_i).S'_i \qquad \exists k \in I \qquad \Gamma_{gd} \vdash d{:}S_k\\
\forall \Gamma'_r, \Gamma''_r : \Gamma_r \backslash s[\mathtt{p}] \to^* \Gamma'_r \;\text{ and }\; \Gamma'_r, s[\mathtt{p}]{:}S \to \Gamma''_r, s[\mathtt{p}]{:}S'_k\\
\Theta \vdash P\{^{s[\mathtt{p}]}/_{x}\} \rhd \Gamma_{g0}, s[\mathtt{p}]{:}S'_k \lhd \Gamma''_r, \Gamma_{gd}
\end{array}}{\Theta \vdash\; s[\mathtt{p}][\mathtt{q}] \oplus \mathtt{m}_k(d)\,.\,P\{^{s[\mathtt{p}]}/_{x}\} \;\rhd \Gamma_g, s[\mathtt{p}]{:}S \lhd \Gamma_r \backslash s[\mathtt{p}]}\;\text{[T-⊕]} \qquad \big(\text{by (50), (57), (51)}\big)
$$

- inductive case [T-|]. We have:

$$
\begin{array}{llr}
\exists \Gamma_1, \Gamma_2 : \Gamma_g = \Gamma_1, \Gamma_2 & \text{(from the rule conclusion)} & (58)\\
\Theta \vdash P_1 \,|\, P_2 \;\rhd \Gamma_1, \Gamma_2, x{:}S \lhd \Gamma_r & \text{(from the rule conclusion and (58))} & (59)
\end{array}
$$

Now, in the rule premises, $x{:}S$ could be used to type either P_1 or P_2 — i.e., we have either:

$$
\begin{array}{lr}
\Theta \vdash P_1 \rhd \Gamma_1, x{:}S \lhd \Gamma_r, \Gamma_2 \quad\text{and}\quad \Theta \vdash P_2 \rhd \Gamma_2 \lhd \Gamma_r, \Gamma_1, x{:}S \quad\text{and}\quad x \notin \mathrm{fv}(P_2) & (60)\\
\text{or} & \\
\Theta \vdash P_1 \rhd \Gamma_1 \lhd \Gamma_r, \Gamma_2, x{:}S \quad\text{and}\quad \Theta \vdash P_2 \rhd \Gamma_2, x{:}S \lhd \Gamma_r, \Gamma_1 \quad\text{and}\quad x \notin \mathrm{fv}(P_1) & (61)
\end{array}
$$

We now proceed assuming (60) (the proof for (61) is symmetric). We have:

$$
\begin{array}{llr}
\Theta \vdash P_1\{^{s[\mathtt{p}]}/_{x}\} \rhd \Gamma_1, s[\mathtt{p}]{:}S \lhd (\Gamma_r \backslash s[\mathtt{p}]), \Gamma_2 & \text{(by (60), (25) and i.h.)} & (62)\\
\Theta \vdash P_2 \rhd \Gamma_2 \lhd (\Gamma_r \backslash s[\mathtt{p}]), \Gamma_1, s[\mathtt{p}]{:}S & \text{(by (60), (25) and Prop. 3.8)} & (63)\\
P_2 = P_2\{^{s[\mathtt{p}]}/_{x}\} & \text{(by (60), since } x \notin \mathrm{fv}(P_2)) & (64)
\end{array}
$$

$$\dfrac{\begin{array}{c}\Theta \vdash P_1\{s[\mathtt{p}]/x\} \triangleright \Gamma_1, s[\mathtt{p}]{:}S \triangleleft (\Gamma_r \backslash s[\mathtt{p}]), \Gamma_2 \\ \Theta \vdash P_2\{s[\mathtt{p}]/x\} \triangleright \Gamma_2 \triangleleft (\Gamma_r \backslash s[\mathtt{p}]), \Gamma_1, s[\mathtt{p}]{:}S\end{array}}{\Theta \vdash (P_1 \mid P_2)\{s[\mathtt{p}]/x\} \triangleright \Gamma_g, s[\mathtt{p}]{:}S \triangleleft \Gamma_r \backslash s[\mathtt{p}]}\ [\text{T-}|] \qquad \text{(by (62), (63), (64) and (58))} \qquad (65)$$

- inductive case [T-**def**]. We have:

$$\dfrac{\Theta \vdash Q \triangleright x_1{:}S_1, \ldots, x_n{:}S_n \triangleleft \varnothing \quad \Theta, X{:}S_1, \ldots, S_n \vdash R \triangleright \Gamma_g, x{:}S \triangleleft \Gamma_r}{\Theta \vdash P \triangleright \Gamma_g, x{:}S \triangleleft \Gamma_r}\ [\text{T-}\mathbf{def}] \qquad \text{(from rule def.)} \qquad (66)$$

$P \;=\; \mathbf{def}\, X(x_1, \ldots, x_n) = Q \,\mathbf{in}\, R$ (by (66)) (67)

$\mathrm{fv}(Q) \subseteq \{x_1, \ldots, x_n\}$ (by (66) (typing of Q)) (68)

$\mathrm{fv}(X(x_1, \ldots, x_n) = Q) \;=\; \varnothing$ (by (68)) (69)

$P\{s[\mathtt{p}]/x\} \;=\; \mathbf{def}\, X(x_1, \ldots, x_n) = Q \,\mathbf{in}\, (R\{s[\mathtt{p}]/x\})$ (by (67) and (69)) (70)

$\Theta, X{:}S_1, \ldots, S_n \vdash R\{s[\mathtt{p}]/x\} \triangleright \Gamma_g, s[\mathtt{p}]{:}S \triangleleft \Gamma_r \backslash s[\mathtt{p}]$ (by (66) (typing of R) and i.h.) (71)

$$\dfrac{\begin{array}{c}\Theta \vdash Q \triangleright x_1{:}S_1, \ldots, x_n{:}S_n \triangleleft \varnothing \\ \Theta, X{:}S_1, \ldots, S_n \vdash R\{s[\mathtt{p}]/x\} \triangleright \Gamma_g, s[\mathtt{p}]{:}S \triangleleft \Gamma_r \backslash s[\mathtt{p}]\end{array}}{\Theta \vdash P\{s[\mathtt{p}]/x\} \triangleright \Gamma_g, s[\mathtt{p}]{:}S \triangleleft \Gamma_r \backslash s[\mathtt{p}]}\ [\text{T-}\mathbf{def}] \qquad \text{(by (70) and (71))}$$

□

Proposition C.10. *If* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$ *and* $\Gamma'_g \equiv \Gamma_g$ *and* $\Gamma'_r \equiv \Gamma_r$*, then* $\Theta \vdash P \triangleright \Gamma'_g \triangleleft \Gamma'_r$.

Proof. Assuming $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$, we proceed by induction on its derivation. □

Lemma 3.7 (Rely context strengthening). *If* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$ *and* $\Gamma_r \rightarrow^* \Gamma'_r$*, then* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma'_r$.

Proof. By induction on the derivation of $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$, using Prop. C.1 in the base cases. □

Proposition C.11. *Assume* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$*. Take some* Γ'_r *such that* $\mathrm{live}(\Gamma'_r)$ *and* $(\mathrm{dom}(\Gamma_g) \cup \mathrm{dom}(\Gamma_r)) \cap \mathrm{dom}(\Gamma'_r) = \varnothing$*. Then,* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r, \Gamma'_r$.

Proof. By induction on the derivation of $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$. □

Proposition C.12. *Assume* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$*. Take some* Γ'_g *such that* $\mathrm{end}(\Gamma'_g)$ *and* $(\mathrm{dom}(\Gamma_g) \cup \mathrm{dom}(\Gamma_r)) \cap \mathrm{dom}(\Gamma'_g) = \varnothing$*. Then,* $\Theta \vdash P \triangleright \Gamma_g, \Gamma'_g \triangleleft \Gamma_r$.

Proof. By induction on the derivation of $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$. □

Proposition C.13. *Assume* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$*. If* $X \notin \mathrm{dom}(\Theta)$*, then* $\Theta, X{:}\widetilde{S} \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$.

Proof. By induction on the derivation of $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$. □

Theorem 3.6. *If* $\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r$ *and* $P \rightarrow^* P'$*, then* $\exists \Gamma'_g$ *such that* $\Gamma_g \rightarrow^* \Gamma'_g$ *and* $\Theta \vdash P' \triangleright \Gamma'_g \triangleleft \Gamma_r$.

Proof. Assume:

$$\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r \qquad\qquad \text{(by hypothesis)} \tag{72}$$

We first prove the following statement, for *one* reduction step of P:

$$P \to P' \quad \text{implies} \quad \exists \Gamma_g' \text{ such that } \Gamma_g \to^* \Gamma_g' \text{ and } \Theta \vdash P' \triangleright \Gamma_g' \triangleleft \Gamma_r \tag{73}$$

Assuming the transition $P \to P'$, we proceed by induction on its derivation.

- base case [R-Comm]. We have:

$$P \;=\; s[\mathtt{p}][\mathtt{q}] \&_{i\in I} \{\mathtt{m}_i(y_i)\,.\,P_i\} \mid s[\mathtt{q}][\mathtt{p}] \oplus \mathtt{m}_k(s'[\mathtt{r}])\,.\,Q \quad (k \in I) \qquad \text{(from the rule definition)} \tag{74}$$

$$P' \;=\; P_k\{{}^{s'[\mathtt{r}]}/_{y_k}\} \mid Q \qquad \text{(from the rule definition)} \tag{75}$$

$$\begin{array}{l}\exists \Gamma_1, \Gamma_2 : \\ \Gamma = \Gamma_1, \Gamma_2 \;\;\wedge\end{array} \dfrac{\begin{array}{c}\Theta \vdash s[\mathtt{p}][\mathtt{q}] \&_{i\in I} \{\mathtt{m}_i(y_i)\,.\,P_i\} \triangleright \Gamma_1 \triangleleft \Gamma_r, \Gamma_2 \\ \Theta \vdash s[\mathtt{q}][\mathtt{p}] \oplus \mathtt{m}_k(s'[\mathtt{r}])\,.\,Q \triangleright \Gamma_2 \triangleleft \Gamma_r, \Gamma_1\end{array}}{\Theta \vdash P \triangleright \Gamma \triangleleft \Gamma_r}\,[\text{T-}|] \qquad \text{(by (74), inv. of [T-|])} \tag{76}$$

$$\exists \Gamma_{r1} = \Gamma_r, \Gamma_2 \quad \text{and} \quad \exists \Gamma_{r2} = \Gamma_r, \Gamma_1 \qquad \text{(by (76), premises of [T-|])} \tag{77}$$

$$\begin{array}{c}\exists J : \varnothing \neq J \subseteq I \quad \exists \Gamma_{g1}, S :\; \Gamma_1 = \Gamma_{g1}, s[\mathtt{p}]{:}S \quad \text{such that} \\ \dfrac{\begin{array}{c} S \equiv \mathtt{q} \&_{i\in J} \mathtt{m}_i(S_i).S_i' \qquad \forall i \in J \qquad \forall \Gamma_r', \Gamma_r'' \\ \Gamma_{r1} \to^* \Gamma_r' \text{ and } \Gamma_r', s[\mathtt{p}]{:}S \to \Gamma_r'', s[\mathtt{p}]{:}S_i' \\ \text{implies} \\ \Theta \vdash P_i \triangleright \Gamma_{g1}, y_i{:}S_i, s[\mathtt{p}]{:}S_i' \triangleleft \Gamma_r'' \end{array}}{\Theta \vdash s[\mathtt{p}][\mathtt{q}] \&_{i\in I} \{\mathtt{m}_i(y_i)\,.\,P_i\} \triangleright \Gamma_1 \triangleleft \Gamma_{r1}}\,[\text{T-}\&] \end{array} \qquad \text{(by (76), inv. of [T-\&])} \tag{78}$$

$$\begin{array}{c}\exists I' \neq \varnothing \quad \exists \Gamma_{g2s'}, \Gamma_{g2}, T :\; \Gamma_2 = \Gamma_{g2s'}, \Gamma_{g2}, s[\mathtt{q}]{:}T \quad \text{such that} \\ \dfrac{\begin{array}{c} T \equiv \mathtt{p} \bigoplus_{i\in I'} \mathtt{m}_i(T_i).T_i' \qquad \exists k \in I' \qquad \Gamma_{g2s'} \vdash s'[\mathtt{r}]{:}T_k \qquad \forall \Gamma_r', \Gamma_r'' \\ \Gamma_{r2} \to^* \Gamma_r' \text{ and } \Gamma_r', s[\mathtt{q}]{:}T \to \Gamma_r'', s[\mathtt{q}]{:}T_k' \\ \text{implies} \\ \Theta \vdash Q \triangleright \Gamma_{g2}, s[\mathtt{q}]{:}T_k' \triangleleft \Gamma_r'', \Gamma_{g2s'} \end{array}}{\Theta \vdash s[\mathtt{q}][\mathtt{p}] \oplus \mathtt{m}_k(s'[\mathtt{r}])\,.\,Q \triangleright \Gamma_2 \triangleleft \Gamma_{r2}}\,[\text{T-}\oplus] \end{array} \qquad \text{(by (76), inv. [T-}\oplus\text{])} \tag{79}$$

$$s[\mathtt{q}]{:}\mathtt{p} \bigoplus_{i\in I'} \mathtt{m}_i(T_i).T_i' \;\equiv\; s[\mathtt{q}]{:}T \in \Gamma_2 \subsetneq \Gamma_{r1} \qquad \text{(by (79) and (77))} \tag{80}$$

$$s[\mathtt{p}]{:}\mathtt{q} \&_{i\in J} \mathtt{m}_i(S_i).S_i' \;\equiv\; s[\mathtt{p}]{:}S \in \Gamma_1 \subsetneq \Gamma_{r2} \qquad \text{(by (78) and (77))} \tag{81}$$

$$I' \subseteq J \quad \text{and therefore} \quad k \in J \qquad \text{(by (80), (77), (78), live}(\Gamma_1, \Gamma_{r1})\text{ and Def. 3.1)} \tag{82}$$

$$\forall i \in J :\; S_i \equiv T_i \qquad \text{(by (82), live}(\Gamma_1, \Gamma_{r1})\text{ and Def. 2.5)} \tag{83}$$

$$\Gamma_{g2s'} \vdash s'[\mathtt{r}]{:}S_k \qquad \text{(by (79) and (83))} \tag{84}$$

$$\Gamma_{r1} \vdash s'[\mathtt{r}]{:}S_k \qquad \text{(by (84), (79) and (77))} \tag{85}$$

$$\forall i \in J :\; s[\mathtt{p}]{:}S, s[\mathtt{q}]{:}T \;\to\equiv\; s[\mathtt{p}]{:}S_i', s[\mathtt{q}]{:}T_i \qquad \text{(by (82), (80), live}(\Gamma_1, \Gamma_{r1})\text{ and Def. 3.1)} \tag{86}$$

$$\exists \Gamma_{r1}' :\; \Gamma_{r1}, s[\mathtt{p}]{:}S \;\to\equiv\; \Gamma_{r1}', s[\mathtt{p}]{:}S_k', , s[\mathtt{q}]{:}T_k' \qquad \text{(by (80) and (86))} \tag{87}$$

$$\Theta \vdash P_k \triangleright \Gamma_{g1}, y_k{:}S_k, s[\mathtt{p}]{:}S_k' \triangleleft \Gamma_{r1}' \qquad \text{(by (87) and (78))} \tag{88}$$

$$\Gamma_{r1}' \vdash s'[\mathtt{r}]{:}S_k \tag{89}$$

$$\Theta \vdash P_k\{{}^{s'[\mathtt{r}]}/_{y_k}\} \triangleright \Gamma_{g1}, s'[\mathtt{r}]{:}S_k, s[\mathtt{p}]{:}S'_k \triangleleft \Gamma'_{r1} \setminus s'[\mathtt{r}] \qquad \text{(by (88), (89) and Lemma 3.5)} \quad (90)$$

$$\Gamma'_{r1} \setminus s'[\mathtt{r}] \;\equiv\; \Gamma_r, \Gamma_{g2}, s[\mathtt{q}]{:}T'_k \qquad \text{(by (77), (79) and (87))} \quad (91)$$

$$\exists \Gamma'_{r2} :\; \Gamma_{r2}, s[\mathtt{q}]{:}T \;\rightarrow_{\equiv}\; \Gamma'_{r2}, s[\mathtt{p}]{:}S'_k, s[\mathtt{q}]{:}T'_k \qquad \text{(by (81) and (86))} \quad (92)$$

$$\Theta \vdash Q \triangleright \Gamma_{g2}, s[\mathtt{q}]{:}T'_k \triangleleft \Gamma'_{r2}, \Gamma_{g2s'} \qquad \text{(by (92) and (79))} \quad (93)$$

$$\Gamma'_{r2}, \Gamma_{g2s'} \;\equiv\; \Gamma_r, \Gamma_{g1}, \Gamma_{g2s'}, s[\mathtt{p}]{:}S'_k \qquad \text{(by (77), (78), (92), (84) and Def. 3.3)} \quad (94)$$

$$\exists \Gamma' = \Gamma'_1, \Gamma'_2 \quad \text{such that:} \quad (95)$$

$$\Gamma'_1 \;=\; \Gamma_{g1}, \Gamma_{g2s'}, s[\mathtt{p}]{:}S'_k \quad (96)$$

$$\Gamma'_2 \;=\; \Gamma_{g2}, s[\mathtt{q}]{:}T'_k \quad (97)$$

$$\Gamma \rightarrow^* \Gamma' \qquad \text{(by (76), (78), (79), (96), (97))} \quad (98)$$

$$\dfrac{\begin{array}{l}\Theta \vdash P_k\{{}^{s'[\mathtt{r}]}/_{y_k}\} \triangleright \Gamma'_1 \triangleleft \Gamma_r, \Gamma'_2 \\ \Theta \vdash Q \triangleright \Gamma'_2 \triangleleft \Gamma_r, \Gamma'_1\end{array}}{\Theta \vdash\; P_k\{{}^{s'[\mathtt{r}]}/_{y_k}\} \mid Q \;\triangleright\; \Gamma' \;\triangleleft\; \Gamma_r}\,[\text{T-}|] \qquad \begin{pmatrix}\text{by (90), (96), (91), (97);} \\ \text{by (93), (97), (94), (96)}\end{pmatrix} \quad (99)$$

- base case [R-X]. We have:

$$P = \textbf{def}\, X(x_1, \ldots, x_n) = Q \textbf{ in } (X\langle s_1[\mathtt{p}_1], \ldots, s_1[\mathtt{p}_1]\rangle \mid R) \qquad \text{(from the rule definition)} \quad (100)$$

$$P' = \textbf{def}\, X(x_1, \ldots, x_n) = Q \textbf{ in } (Q\{{}^{s_1[\mathtt{p}_1]}/_{x_1}\}\cdots\{{}^{s_n[\mathtt{p}_n]}/_{x_n}\} \mid R) \qquad \text{(from the rule definition)} \quad (101)$$

$$\Gamma_g = \Gamma_{g*}, \Gamma_{gR} \quad \text{and} \quad \Gamma_{g*} = \Gamma_{g0}, \Gamma_{g1}, \ldots \Gamma_{gn} \quad \text{and} \quad \Theta' = \Theta, X{:}S_1, \ldots, S_n \quad \text{such that:} \quad (102)$$

$$\dfrac{\Theta \vdash Q \triangleright x_1{:}S_1, \ldots, x_n{:}S_n \triangleleft \varnothing \qquad \dfrac{\dfrac{\begin{array}{l}\Theta' \vdash X : S_1, \ldots, S_n \\ \text{end}(\Gamma_{g0}) \quad \forall i \in 1..n \quad \Gamma_{gi} \vdash s_i[\mathtt{p}_i]{:}S_i\end{array}}{\Theta' \vdash X\langle s_1[\mathtt{p}_1], \ldots, s_1[\mathtt{p}_1]\rangle \triangleright \Gamma_{g*} \triangleleft \Gamma_r, \Gamma_{gR}}\,[\text{T-}X] \qquad \Theta' \vdash R \triangleright \Gamma_{gR} \triangleleft \Gamma_r, \Gamma_{g*}}{\Theta' \vdash X\langle s_1[\mathtt{p}_1], \ldots, s_1[\mathtt{p}_1]\rangle \mid R \triangleright \Gamma_g \triangleleft \Gamma_r}\,[\text{T-}|]}{\Theta \vdash P \triangleright \Gamma_g \triangleleft \Gamma_r}\,[\text{T-}\textbf{def}]$$

$$\text{(by (100), (72), inv. [T-}\textbf{def}\text{],[T-}|\text{],[T-}X\text{])} \quad (103)$$

$$X \notin \text{dom}(\Theta) \qquad \text{(by (102))} \quad (104)$$

$$\text{live}(\Gamma_{g0}) \qquad \text{(by (103) and Prop. C.7)} \quad (105)$$

$$\text{live}(\Gamma_{g1}, \ldots, \Gamma_{gn}, \Gamma_r, \Gamma_{gR}) \qquad \text{(by (103), Def. 3.3, (105) and Prop. C.6)} \quad (106)$$

$$\text{live}((\Gamma_{g1}, \ldots, \Gamma_{gn}, \Gamma_r, \Gamma_{gR}) \setminus x_1, \ldots, x_n) \qquad \text{(by (106) and Prop. C.8 } \times n\text{)} \quad (107)$$

$$\text{live}(\Gamma_{g1}, \ldots, \Gamma_{gn}, ((\Gamma_r, \Gamma_{gR}) \setminus x_1, \ldots, x_n)) \qquad \text{(by (107), (103) and Def. 3.3)} \quad (108)$$

$$\Theta \vdash Q \triangleright x_1{:}S_1, \ldots, x_n{:}S_n \triangleleft \Gamma_{g1}, \ldots, \Gamma_{gn}, ((\Gamma_r, \Gamma_{gR}) \setminus x_1, \ldots, x_n) \qquad \text{((108) (103), and Prop. C.11)} \quad (109)$$

$$\Theta \vdash Q\{{}^{s_1[\mathtt{p}_1]}/_{x_1}\}\cdots\{{}^{s_n[\mathtt{p}_n]}/_{x_n}\} \triangleright \Gamma_{g1}, \ldots, \Gamma_{gn} \triangleleft (\Gamma_r, \Gamma_{gR}) \setminus x_1, \ldots, x_n \qquad \text{((109), (103), Lemma 3.5 } \times n\text{)} \quad (110)$$

$$\Theta \vdash Q\{{}^{s_1[\mathtt{p}_1]}/_{x_1}\}\cdots\{{}^{s_n[\mathtt{p}_n]}/_{x_n}\} \triangleright \Gamma_{g*} \triangleleft \Gamma_r, \Gamma_{gR} \qquad \text{((110), (102), (103), Prop. C.12, Cor. C.9 } \times n\text{)} \quad (111)$$

$$\Theta' \vdash Q\{{}^{s_1[\mathtt{p}_1]}/_{x_1}\}\cdots\{{}^{s_n[\mathtt{p}_n]}/_{x_n}\} \triangleright \Gamma_{g*} \triangleleft \Gamma_r, \Gamma_{gR} \qquad \text{(by (111), (104), (102) and Prop. C.13)} \quad (112)$$

$$\text{Let } \Gamma'_g = \Gamma_g = \Gamma_{g*}, \Gamma_{gR} = \Gamma_{g0}, \Gamma_{g1}, \ldots, \Gamma_{gn}, \Gamma_{gR} \quad (\text{hence, } \Gamma_g \rightarrow^* \Gamma'_g) \qquad \text{(by (102))} \quad (113)$$

$$\dfrac{\Theta \vdash Q \rhd x_1{:}S_1,\ldots,x_n{:}S_n \lhd \varnothing \qquad \dfrac{\Theta' \vdash Q\{^{s_1[\mathrm{p}_1]}/_{x_1}\}\cdots\{^{s_n[\mathrm{p}_n]}/_{x_n}\} \rhd \Gamma_{g*} \lhd \Gamma_r,\Gamma_{gR} \qquad \Theta' \vdash R \rhd \Gamma_{gR} \lhd \Gamma_r,\Gamma_{g*}}{\Theta' \vdash Q\{^{s_1[\mathrm{p}_1]}/_{x_1}\}\cdots\{^{s_n[\mathrm{p}_n]}/_{x_n}\} \mid R \rhd \Gamma'_g \lhd \Gamma_r}\,\text{[T-|]}}{\Theta \vdash P' \rhd \Gamma'_g \lhd \Gamma_r}\,\text{[T-}\textbf{def}\text{]}$$

(by (111), (113), (101))

- inductive case [R-|]. We have:

$$P = P_1 \mid P_2 \qquad \text{(from the rule conclusion)} \qquad (114)$$
$$\exists \Gamma_1,\Gamma_2 : \quad \Gamma_g = \Gamma_1,\Gamma_2 \qquad \text{(by (114) (only typable by [T-|]) and (72))} \qquad (115)$$
$$\Theta \vdash P_1 \rhd \Gamma_1 \lhd \Gamma_r,\Gamma_2 \qquad \text{(by (114), (115) and inversion of [T-|])} \qquad (116)$$
$$\Theta \vdash P_2 \rhd \Gamma_2 \lhd \Gamma_r,\Gamma_1 \qquad \text{(by (114), (115) and inversion of [T-|])} \qquad (117)$$

Now, $P \to P'$ implies either:

$$P_1 \to P'_1 \quad \text{and} \quad P' = P'_1 \mid P_2 \qquad \text{(by inversion of [R-|])} \qquad (118)$$

or

$$P_2 \to P'_2 \quad \text{and} \quad P' = P_1 \mid P'_2 \qquad \text{(by congruence} \equiv \text{and inversion of [R-|])} \qquad (119)$$

We proceed assuming (118) (the proof for (119) is symmetric).

$$\exists \Gamma'_1 : \; \Gamma_1 \to^* \Gamma'_1 \;\text{ and }\; \Theta \vdash P'_1 \rhd \Gamma'_1 \lhd \Gamma_r,\Gamma_2 \qquad \text{(by (116), (118) and i.h.)} \qquad (120)$$
$$\Gamma_1 \to^* \Gamma'_1 \qquad \text{(by (120))} \qquad (121)$$
$$\Gamma_r,\Gamma_1 \to^* \Gamma_r,\Gamma'_1 \qquad \text{(by (121) and Def. 2.5)} \qquad (122)$$
$$\Theta \vdash P_2 \rhd \Gamma_2 \lhd \Gamma_r,\Gamma'_1 \qquad \text{(by (117), (122) and Lemma 3.7)} \qquad (123)$$
$$\exists \Gamma'_g = \Gamma'_1,\Gamma_2 \;\text{ such that }\; \Gamma_g \to^* \Gamma'_g \qquad \text{(by (115), (120) and Def. 2.5)} \qquad (124)$$

$$\dfrac{\Theta \vdash P'_1 \rhd \Gamma'_1 \lhd \Gamma_r,\Gamma_2 \qquad \Theta \vdash P_2 \rhd \Gamma_2 \lhd \Gamma_r,\Gamma'_1}{\Theta \vdash P' \rhd \Gamma'_g \lhd \Gamma_r}\,\text{[T-|]}$$

(by (118), (120), (123), (124))

- inductive case [R-ν]. We have:

$$\exists Q : \; P = (\nu s)\, Q \qquad \text{(from the rule definition)} \qquad (125)$$
$$P \to P' \;\text{ implies }\; \exists Q' : Q \to Q' \;\text{ and }\; P' = (\nu s)\, Q' \qquad \text{(by (125) and inversion of [R-}\nu\text{])} \qquad (126)$$

$$\dfrac{\Gamma_s = \{c[\mathrm{p}]{:}S_{\mathrm{p}}\}_{\mathrm{p}\in I} \qquad \Theta \vdash Q \rhd \Gamma_g,\Gamma_s \lhd \Gamma_r}{\Theta \vdash (\nu s{:}\Gamma_s)\, Q \rhd \Gamma_g \lhd \Gamma_r}\,\text{[T-}\nu\text{]} \qquad \text{(by (125), only typable by [T-}\nu\text{])} \qquad (127)$$

$$\exists \Gamma''_g : \; \Gamma_g,\Gamma_s \to^* \Gamma''_g \;\text{ and }\; \Theta \vdash Q' \rhd \Gamma''_g \lhd \Gamma_r \qquad \text{(by (127) (premise of [T-}\nu\text{], (126) and i.h.)} \qquad (128)$$
$$\text{live}(\Gamma_s) \qquad \text{(by (127) (premise and conclusion of [T-}\nu\text{]) and Prop. C.6)} \qquad (129)$$
$$\exists \Gamma'_g,\Gamma'_s : \; \Gamma''_g = \Gamma'_g,\Gamma'_s \;\text{ and }\; \Gamma_g \to^* \Gamma'_g \;\text{ and }\; \Gamma_s \to^* \Gamma'_s \qquad \text{(by (129), (128) and Prop. C.5)} \qquad (130)$$
$$\Gamma_g,\Gamma_r \to^* \Gamma'_g,\Gamma_r \qquad \text{(by (130) and Def. 2.5)} \qquad (131)$$
$$\text{live}(\Gamma'_g,\Gamma_r) \qquad \text{(by (127) (conclusion of [T-}\nu\text{]), (131) and Prop. C.1)} \qquad (132)$$

$$\dfrac{\Gamma'_s = \{s[\mathrm{p}]{:}S'_{\mathrm{p}}\}_{\mathrm{p}\in I} \qquad \Theta \vdash Q' \rhd \Gamma'_g,\Gamma'_s \lhd \Gamma_r}{\Theta \vdash P' \rhd \Gamma'_g \lhd \Gamma_r}\,\text{[T-}\nu\text{]}$$

(by (126), (128), (130), (132))

- inductive case [R-**def**]. We have:

$$P \;=\; \textbf{def}\, X(\widetilde{x}) = Q \;\textbf{in}\; R \qquad \text{(from the rule definition)} \qquad (133)$$

$$P' \;=\; \textbf{def}\, X(\widetilde{x}) = Q \;\textbf{in}\; R' \quad \text{with} \quad R \to R' \qquad \text{(from the rule premises)} \qquad (134)$$

$$\dfrac{\Theta \vdash Q \rhd x_1{:}S_1,\ldots,x_n{:}S_n \lhd \varnothing \qquad \Theta, X{:}S_1,\ldots,S_n \vdash R \rhd \Gamma_g \lhd \Gamma_r}{\Theta \vdash P \rhd \Gamma_g \lhd \Gamma_r}\;[\text{T-}\textbf{def}] \qquad \text{(by (133), (72), inv. [T-}\textbf{def}\text{])} \qquad (135)$$

$$\exists \Gamma' : \Gamma \to^* \Gamma' \;\text{ and }\; \Theta, X{:}S_1,\ldots,S_n \vdash R' \rhd \Gamma'_g \lhd \Gamma_r \qquad \text{(by (135), (134) and i.h).} \qquad (136)$$

$$\dfrac{\Theta \vdash Q \rhd x_1{:}S_1,\ldots,x_n{:}S_n \lhd \varnothing \qquad \Theta, X{:}S_1,\ldots,S_n \vdash R' \rhd \Gamma'_g \lhd \Gamma_r}{\Theta \vdash P' \rhd \Gamma'_g \lhd \Gamma_r}\;[\text{T-}\textbf{def}] \qquad \text{(by (136) and (134))}$$

We have thus proved (73). Then, we prove the main statement by induction on the number of reductions in $P \to^* P'$: the base case is trivial (0 reductions, and thus $P = P'$); in the inductive case ($n+1$ reductions), we apply the induction hypothesis and (73). □