CrossMark

Multiparty session types as coherence proofs

Marco Carbone[1] · **Fabrizio Montesi[2]** iD ·
Carsten Schürmann[1] · **Nobuko Yoshida[3]**

Abstract We propose a Curry–Howard correspondence between a language for programming multiparty sessions and a generalisation of Classical Linear Logic (CLL). In this framework, propositions correspond to the local behaviour of a participant in a multiparty session type, proofs to processes, and proof normalisation to executing communications. Our key contribution is generalising duality, from CLL, to a new notion of n-ary compatibility, called *coherence*. Building on coherence as a principle of compositionality, we generalise the cut rule of CLL to a new rule for composing many processes communicating in a multiparty session. We prove the soundness of our model by showing the admissibility of our new rule, which entails deadlock-freedom via our correspondence.

1 Introduction

Session types are protocols for communications in concurrent systems [16,26]. A recent line of work investigates Curry–Howard correspondences between the type theory of session types and linear logic, where proofs correspond to processes, propositions to types, and proof normalisation to communications [6,28]. An important consequence of such correspondences is that several notions that usually require complex additional definitions and proofs, e.g.,

✉ Fabrizio Montesi
fmontesi@imada.sdu.dk

Marco Carbone
carbonem@itu.dk

Carsten Schürmann
carsten@itu.dk

Nobuko Yoshida
n.yoshida@imperial.ac.uk

[1] IT University of Copenhagen, Rued Langgaards Vej 7, 2700 Copenhagen, Denmark

[2] University of Southern Denmark, Campusvej 55, 5230 Odense, Denmark

[3] Imperial College London, 180 Queen's Gate, London SW7 2AZ, UK

dependency relations for deadlock-freedom [12,23], follow for free from the theory of linear logic, yielding a succinct formulation of the formal foundations of sessions.

The aforementioned correspondences cover only session types with exactly two participants, called *binary session types*. In practice, however, protocols often describe the behaviour of multiple participants [25]. *Multiparty Session Types (MPSTs)* have been proposed to capture such protocols, by matching the communications enacted by many participants with a global scenario [17]. Unfortunately, MPSTs are more involved than binary session types, since they include complex analyses on the structure of protocols and a mapping from *global types*, which describe multiparty protocols, to *local types*, which describe the local behaviour of each single participant. So far, it has been unclear whether a succinct logical formulation of MPSTs can be developed, as done for binary session types. Therefore, we ask:

Can we design a proof theory for reasoning about multiparty sessions?

A positive answer to our question would lead to a clearer understanding of the principles that underpin multiparty session programming. The main challenge lies in the foundational notion of duality found in linear logic, which, in a Curry–Howard interpretation of propositions as types, checks whether the session types of two respective participants are compatible. It is an open question how to generalise the notion of type duality to that of "multiparty compatibility" found in MPSTs, which allows to compose an arbitrary number of participants [14,17,20]. Therefore, differently from previous work, we are in a situation where the existing logic does not provide us with natural tools for dealing with the types we desire to capture.

The **main contribution** of this work is the development of Multiparty Classical Processes (MCP), a proof theory for reasoning on synchronous multiparty communications. The key aspect of MCP is that it generalises Classical Linear Logic (CLL) [15], by building on a new notion of type compatibility, called *coherence*, that replaces duality. Using MCP, we can provide a concise reconstruction of the foundations of MPSTs. In the following, we outline our investigation:

- *Coherence* We start by formalising a language for *local types* and *global types* (Sect. 3, Types). As in MPSTs, a local type denotes the I/O actions of a single participant in a session, whereas a global type denotes the desired interactions among all participants in a session. We then present *coherence*, a proof system for determining whether a set of local types follow the scenario denoted by a global type (Sect. 3, Coherence). We prove the adequacy of coherence by showing that global types are proof terms for coherence proofs (Sect. 3, Fig. 2); equivalences between coherence proofs correspond to the equivalences between global types originally formulated with an auxiliary definition in [8] (Sect. 3, Proposition 1); and, the coherence proof system yields projection and extraction procedures from global types to local types and vice versa (Sect. 3, Proposition 2 and Proposition 3). Finally, we show that coherence generalises the notion of duality in CLL (Sect. 3, Proposition 4). Our extraction procedure is the first not requiring auxiliary conditions (e.g., dependency relations as in [19]) and capturing nested protocols [13].

- *Multiparty classical processes* We present Multiparty Classical Processes (MCP), a proof theory that is in a Curry–Howard correspondence with a language for synchronous multiparty sessions (Sect. 4). The key aspect of MCP is using coherence as a new principle for compositionality in order to generalise the standard cut rule of linear logic, by allowing an arbitrary number of proofs to be composed (Sect. 4, Fig. 6). Such a generalisation gives us a consavative extension of the binary cut rule of Classical Linear Logic (CLL) (Sect. 7). From the proof theory of MCP, we derive logically-founded notions of structural equivalences and reductions for multiparty processes (Sect. 4, Figs. 7 and 8). Driven by the correspondence between processes and proofs, we show that: communications among

processes always follow their session types (Sect. 5, Theorem 4); and, communications never get stuck (Sect. 5, Corollary 1), improving on previous techniques for analysing progress in multiparty sessions (Sect. 8).

2 Preview

We give an informal introduction to MCP with the 2-buyer protocol [17], where two buyers buy a book together from a seller. This can be described by the following global type:

$$
\begin{aligned}
&1.\ \mathsf{B1} \to \mathsf{S} : \langle \mathbf{str} \rangle;\ \mathsf{S} \to \mathsf{B1} : \langle \mathbf{int} \rangle;\ \mathsf{S} \to \mathsf{B2} : \langle \mathbf{int} \rangle;\ \mathsf{B1} \to \mathsf{B2} : \langle \mathbf{int} \rangle; \\
&2.\ \mathsf{B2} \to \mathsf{S} : \&(\mathsf{B2} \to \mathsf{S} : \langle \mathbf{addr} \rangle;\ \mathsf{end},\ \ \mathsf{end})
\end{aligned}
\tag{1}
$$

Above, $\mathsf{B1}$ (the first buyer), $\mathsf{B2}$ (the second buyer) and S (the seller) are *roles*. In Line 1, $\mathsf{B1}$ sends the book title to S, then S sends a quote to $\mathsf{B1}$ and $\mathsf{B2}$. At this point, $\mathsf{B1}$ sends to $\mathsf{B2}$ the fraction of the price it wishes to pay. In Line 2, $\mathsf{B2}$ communicates to S whether ($\&$) to proceed with the purchase and, if so, also an address for the delivery.

In multiparty session types, each role in a global type is implemented by a different process. For example, the following three programs implement the roles in (1):

$$
\begin{aligned}
\text{Buyer1} &\overset{\text{def}}{=} \overline{x}^{\,\mathsf{B1\,S}}(title);\ x^{\mathsf{B1\,S}}(quote);\ \overline{x}^{\,\mathsf{B1\,B2}}(contr) \\
\text{Buyer2} &\overset{\text{def}}{=} x^{\mathsf{B2\,S}}(quote);\ x^{\mathsf{B2\,B1}}(contr);\ \left(x^{\mathsf{B2\,S}}.\mathsf{inl};\ \overline{x}^{\,\mathsf{B2\,S}}(addr) + x^{\mathsf{B2\,S}}.\mathsf{inr} \right) \\
\text{Seller} &\overset{\text{def}}{=} x^{\mathsf{S\,B1}}(title);\ \overline{x}^{\,\mathsf{S\,B1}}(quote);\ \overline{x}^{\,\mathsf{S\,B2}}(quote);\ x^{\mathsf{S\,B2}}\&.\mathsf{case}\left(x^{\mathsf{S\,B2}}(addr),\ \ \mathbf{0} \right)
\end{aligned}
$$

The three processes above are defined in the π-calculus with (synchronous) multi-party sessions [12,18], and communicate using the session (or channel) x. In term Buyer1, $\overline{x}^{\,\mathsf{B1\,S}}(title)$ means "as role $\mathsf{B1}$, send the book $title$ over channel x to the process implementing role S"; $x^{\mathsf{B1\,S}}(quote)$ means "as role $\mathsf{B1}$, receive a quote over channel x from the process implementing role S"; finally, $\overline{x}^{\,\mathsf{B1\,B2}}(contr)$ means "as role $\mathsf{B1}$, send to the process implementing role $\mathsf{B2}$, over channel x, the amount the first buyer is willing to contribute with". Note that Buyer2 makes a choice after receiving the contribution from Buyer1, i.e., it either accepts or rejects the purchase by respectively selecting the left or right branch of the case construct in the code of Seller.

Following the approach in [28], we can type channel x using CLL propositions (differently from [28], we use $\invamp$ to type outputs and $\otimes$ to type inputs, see Sect. 8):

$$
\begin{aligned}
\text{usage of } x \text{ in Buyer1}: &\quad \mathbf{str} \invamp \mathbf{int} \otimes \mathbf{int} \invamp \mathsf{end} \\
\text{usage of } x \text{ in Buyer2}: &\quad \mathbf{int} \otimes \mathbf{int} \otimes ((\mathbf{addr} \invamp \mathsf{end}) \oplus \mathsf{end}) \\
\text{usage of } x \text{ in Seller}: &\quad \mathbf{str} \otimes \mathbf{int} \invamp \mathbf{int} \invamp ((\mathbf{addr} \otimes \mathsf{end}) \& \mathsf{end})
\end{aligned}
\tag{2}
$$

Above, each proposition states how x is used by each process. For instance, Buyer1 outputs ($\invamp$) a string, receives ($\otimes$) an integer, sends another integer and finally terminates (end).

CLL cannot compose our three processes using the above specifications, since its composition rule Cut can only compose two processes, which communicate over the same channel x with compatible binary session types A and $A^{\perp}$:

$$
\frac{P \vdash \Delta, x : A \qquad Q \vdash \Delta', x : A^{\perp}}{(\boldsymbol{\nu} x : A)\,(P \mid Q) \vdash \Delta, \Delta'} \ \mathsf{Cut}
$$

Using the same channel among our three processes is essential for tracking the dependencies expressed by the global type in (1): for example, we need to ensure that Seller sends a quote to Buyer2 only after it has received a request for a book from Buyer1. Such constraints cannot

be tracked by binary session types [17]. To overcome this issue, we annotate each connective in propositions with roles. For example, the type of x for Buyer1 would become:

annotated usage of x in Buyer1 : $\mathbf{str} \,\otimes^{\mathsf{S}}\, \mathbf{int} \,\otimes^{\mathsf{S}}\, \mathbf{int} \,\otimes^{\mathsf{B2}}\, \mathsf{end}$

annotated usage of x in Buyer2 : $\mathbf{int} \,\otimes^{\mathsf{S}}\, \mathbf{int} \,\otimes^{\mathsf{B1}}\, ((\mathbf{addr} \,\otimes^{\mathsf{S}}\, \mathsf{end}) \oplus^{\mathsf{S}}\, \mathsf{end})$ (3)

annotated usage of x in Seller : $\mathbf{str} \,\otimes^{\mathsf{B1}}\, \mathbf{int} \,\otimes^{\mathsf{B1}}\, \mathbf{int} \,\otimes^{\mathsf{B2}}\, ((\mathbf{addr} \,\otimes^{\mathsf{B2}}\, \mathsf{end}) \,\&^{\mathsf{B2}}\, \mathsf{end})$

Annotations identify the dual role of each action, e.g., the usage for Buyer1 now reads: send a string to S ($\otimes^{\mathsf{S}}$); receive an integer from S ($\otimes^{\mathsf{S}}$); send an integer to $\mathsf{B2}$ ($\otimes^{\mathsf{B2}}$); and, terminate (end). We can then reformulate Cut as:

$$\frac{P_i \vdash \Gamma_i, x^{p_i} : A_i \quad G \vDash \{p_i : A_i\}_i}{(\boldsymbol{\nu} x : G)\left(\prod_i P_i\right) \vdash \{\Gamma_i\}_i} \ \text{MCut}$$

In our new *multiparty* cut rule MCut, if some processes P_i use session x as role p_i (denoted x^{p_i}), each according to some respective types A_i, and such types coherently follow a global type specification G (formalised by the judgement $G \vDash \{p_i : A_i\}_i$), then we can compose them in parallel within the scope of session x, written $(\boldsymbol{\nu} x : G)(P_1 \mid \ldots \mid P_n)$. In our example, for i ranging from 1 to 3, $\{p_i : A_i\}_i$ would correspond to the types in (3), where p_1, p_2 and p_3 would be, respectively, Buyer1, Buyer2 and Seller. In Sect. 6, we will show that such types coherently follow the global type given in (1).

MCP goes beyond the original multiparty session types [17], capturing also multicasting and nested protocols [12,13]. For example, we can enhance the 2-buyer protocol as:

1. $\mathsf{B1} \rightarrow \mathsf{S} : \langle \mathbf{str} \rangle; \ \mathsf{S} \rightarrow \mathsf{B1}, \mathsf{B2} : \langle \mathbf{int} \rangle; \ \mathsf{B1} \rightarrow \mathsf{B2} : \langle \mathbf{int} \rangle;$

2. $\mathsf{B2} \!\rightarrow\! \mathsf{B1}, \mathsf{S} : \&\begin{pmatrix} \mathsf{B2} \!\rightarrow\! \mathsf{S} : \langle \mathbf{addr} \rangle; \ \mathsf{end}, \\ \mathsf{B1} \!\rightarrow\! \mathsf{S} : \langle G_{\mathsf{sub}} \rangle; \mathsf{B1} \!\rightarrow\! \mathsf{S} : \langle \mathbf{str} \rangle; \mathsf{B2} \!\rightarrow\! \mathsf{S} : \langle \mathbf{str} \rangle; \mathsf{end} \end{pmatrix}$ (4)

Above, S multicasts the price to both $\mathsf{B1}$ and $\mathsf{B2}$; and $\mathsf{B2}$ multicasts its decision to $\mathsf{B1}$ and S. We have also updated the right branch of the choice using a nested protocol G_{sub}, which is private to $\mathsf{B1}$ and S, where $\mathsf{B1}$ tells S whether it wants to purchase the product alone:

$$G_{\mathsf{sub}} = \mathsf{B1} \!\rightarrow\! \mathsf{S} : \&\big(\mathsf{B1} \!\rightarrow\! \mathsf{S} : \langle \mathbf{addr} \rangle; \ \mathsf{end}, \quad \mathsf{B1} \!\rightarrow\! \mathsf{S} : \langle \mathbf{str} \rangle; \mathsf{end}\big)$$

In MCP, nested protocols can proceed in parallel to their originating protocols. For example, the last two communications, where $\mathsf{B1}$ and $\mathsf{B2}$ inform S of their respective reasons for not completing the purchase, can be executed in parallel to G_{sub}. We will formalise this in Sect. 5.

3 Coherence

We give a proof-theoretical reconstruction of coherence, from [17]. Our theory generalises duality, from CLL, to checking the compatibility of multiple types. We define coherence as a proof system for deriving sets of (compatible) *local types*, which describe the local behaviours of participants in a multiparty session. Global types are proof terms for coherence proofs, yielding a correspondence between sets of compatible local types and their global descriptions.

Types The syntax of local and global types is given in Fig. 1, where p, q range over a set of *roles*. Global types are highlighted, to distinguish them as proof terms. Highlighting is also used in our syntax of local types, to show the difference with CLL. We will adopt the same convention in Sect. 4 when we present more terms. A local type A describes the local

$$A, B, \ldots ::= \quad 1 \quad (\textit{unit for } \otimes) \qquad | \qquad \bot \quad (\textit{unit for } \invamp)$$
$$| \quad A \invamp^{\tilde{p}} B \quad (\textit{send A to } \tilde{p}, \textit{ then B}) \quad | \quad A \otimes^{p} B \quad (\textit{receive A from p, then B})$$
$$| \quad A \oplus^{\tilde{p}} B \quad (\textit{select A or B in } \tilde{p}) \quad | \quad A \&^{p} B \quad (\textit{offer A or B to p})$$
$$| \quad !A \quad (\textit{replication}) \qquad | \quad ?A \quad (\textit{server})$$

$$G ::= p \rightarrow \tilde{q} : \langle G' \rangle; G \mid p \rightarrow \tilde{q} : \&(G_1, G_2) \mid ?p \rightarrow !\tilde{q} : \langle G \rangle \mid \mathsf{end}^{p\tilde{q}}$$

Fig. 1 Local types $(A, B, \ldots)$ and global types (G)

behaviour of a role in a session. Types 1 and $\bot$ denote session termination, respectively representing the request and the acceptance for closing a session (which were informally abstracted by end in our previous examples). A type $A \invamp^{\tilde{p}} B$ denotes a multicast output of a session with type A to roles $\tilde{p}$, with a continuation B. A type $A \otimes^{p} B$ represents an input of a session with type A from role p, with continuation B. Types $A \oplus^{\tilde{p}} B$ and $A \&^{p} B$ denote, respectively, the output of a choice between the continuations A and B to roles $\tilde{p}$ and the input of a choice from role p. The replicated type $!A$ offers behaviour A as many times as requested. Finally, type $?A$ requests the execution of a replicated type and proceeds as A.

A global type G describes the behaviour of many participants. In the interaction $p \rightarrow \tilde{q} : \langle G' \rangle; G$, role p sends to roles $\tilde{q}$ a message to create a new session of type G', and then the protocol proceeds as G. In $p \rightarrow \tilde{q} : \&(G_1, G_2)$, role p communicates to roles $\tilde{q}$ its choice of either branch G_1 or G_2. A type $?p \rightarrow !\tilde{q} : \langle G \rangle$ denotes that role p may ask roles $\tilde{q}$ to execute G many times. Finally, in $\mathsf{end}^{p\tilde{q}}$, role p asks roles $\tilde{q}$ to terminate the session (for brevity, we often write end).

Judgements A *role typing* $p : A$ states that role p behaves as specified by type A. Our *judgements* for coherence have the form $G \vDash p_1 : A_1, \ldots, p_n : A_n$ which reads as "the types $A_1, \ldots, A_n$ of the respective roles $p_1, \ldots, p_n$ (assumed to be pairwise distinct) are compatible and follow the global type G". We use Θ to range over sets of role typings, and make the standard assumption that we can write $\Theta, p : A$ only if a role typing for p does not appear in Θ. Given some roles $\tilde{p}$, we use the notation $\{p_i : A_i\}_i$ to denote the set of role typings $p_1 : A_1, \ldots, p_n : A_n$, assuming $\tilde{p} = p_1, \ldots, p_n$ and i ranging from 1 to n. Given G, we say that G is *valid* if there exists Θ such that $G \vDash \Theta$. Conversely, given Θ, we say that Θ is *coherent* if there exists G such that $G \vDash \Theta$. Intuitively, the validity and coherency correspond to syntactic well-formedness or projectability in [17].

We report the rules for deriving coherence judgements in Fig. 2. Rule $\otimes \invamp$ matches the output type from role p to roles $\tilde{q}$ with the input types of roles $\tilde{q}$, whenever (i) the types for the newly created session are coherent and (ii) the types of all continuations are also coherent. Rule $\oplus \&$ checks that both possibilities in a choice are coherent, where all roles participating in the communication are allowed to have different behaviour and the other roles are not (a

$$\frac{G \vDash \Theta, p{:}B, \{q_i{:}D_i\}_i \quad G' \vDash p{:}A, \{q_i{:}C_i\}_i}{p \rightarrow \tilde{q} : \langle G' \rangle; G \quad \vDash \quad \Theta, \; p{:}A \invamp^{\tilde{q}} B, \{q_i{:}C_i \otimes^{p} D_i\}_i} \otimes\invamp \qquad \frac{}{\mathsf{end}^{p\tilde{q}} \vDash p{:}\bot, q_1{:}1, \ldots, q_n{:}1} 1\bot$$

$$\frac{G_1 \vDash \Theta, p{:}A, \{q_i{:}C_i\}_i \quad G_2 \vDash \Theta, p{:}B, \{q_i{:}D_i\}_i}{p \rightarrow \tilde{q} : \&(G_1, G_2) \vDash \Theta, p{:}A \oplus^{\tilde{q}} B, \{q_i{:}C_i \&^{p} D_i\}_i} \oplus\& \qquad \frac{G \vDash p{:}A, \{q_i{:}B_i\}_i}{?p \rightarrow !\tilde{q} : \langle G \rangle \vDash p{:}?A, \{q_i{:}!B_i\}_i} !?$$

Fig. 2 Coherence

multicast generalisation of [17]). In rule !?, we check that a client requests the creation of a coherent session only from replicated services. Finally, rule $1\perp$ checks that all participants agree on the termination of a protocol. As in CLL, we interpret type 1 as a terminated process and $\perp$ as a process that has terminated its behaviour in a session and proceeds with other sessions. Therefore, we read rule $1\perp$ as "a protocol terminates when one participant waits (type $\perp$) for the termination of all the others (type 1), which execute in parallel". This design choice simplifies our development; we discuss a generalisation in Sect. 8.

For example, $p \rightarrow \tilde{q} : \&(\mathsf{end}^{r\tilde{q}}, \mathsf{end}^{p\tilde{s}})$ is not valid since end does not have correct participant annotation. $p \rightarrow \tilde{q} : \&(p \rightarrow \tilde{r} : \langle \mathsf{end}^{r\tilde{q}} \rangle, \mathsf{end}^{p\tilde{q}})$ is not valid either since a global type in the right branch does not contain the participants p and q_i (hence it does not match with rule $\oplus\&$).

Example 1 (*2-Buyer Protocol*) We can revisit the local types for the 2-buyer protocol in Sect. 2 (1), where now data types are abstracted by 1's and $\perp$'s.

$$
\begin{aligned}
A &\stackrel{\text{def}}{=} \perp \wp \mathsf{S} 1 \otimes^{\mathsf{S}} \perp \wp^{\mathsf{B2}} \perp \\
B &\stackrel{\text{def}}{=} 1 \otimes^{\mathsf{S}} 1 \otimes^{\mathsf{B1}} ((\perp \wp^{\mathsf{S}} 1) \oplus^{\mathsf{S}} 1) \\
C &\stackrel{\text{def}}{=} 1 \otimes^{\mathsf{B1}} \perp \wp^{\mathsf{B1}} \perp \wp^{\mathsf{B2}} \Big((1 \otimes^{\mathsf{B2}} \perp) \&^{\mathsf{B2}} 1 \Big)
\end{aligned}
$$

Let G be the global type in (1) with end instead of data types; then, $G \vDash \mathsf{B1} : A, \mathsf{B2} : B, \mathsf{S} : C$.

3.1 Properties of coherence

Swapping Immediately from our correspondence between global types and coherence proofs, we can reconstruct the standard notion of swapping $\simeq^{\mathsf{g}}$ for global types from [8]. Intuitively, two communications involving different roles can always be swapped, capturing the fact that separate roles execute concurrently. For example, the following coherence proof (for p, q, r, s different):

$$
\frac{\dfrac{G \vDash \Theta, p\colon A', q\colon B', r\colon C', s\colon D \quad G'' \vDash \bar{\Theta}, r\colon C, s\colon D}{r \rightarrow s : \langle G'' \rangle; G \vDash \Theta, p\colon A', q\colon B', r\colon C \wp^{s} C', s\colon D \otimes^{r} D'} \otimes\wp \quad G' \vDash p\colon A, q\colon B}{p \rightarrow q : \langle G' \rangle; r \rightarrow s : \langle G'' \rangle; G \vDash \Theta, p\colon A \wp^{q} A', q\colon B \otimes^{p} B', r\colon C \wp^{s} C', s\colon D \wp^{r} D',} \otimes\wp
$$

is equivalent to ($\simeq^{\mathsf{g}}$)

$$
\frac{\dfrac{G \vDash \Theta, p\colon A', q\colon B', r\colon C', s\colon D \quad G' \vDash \tilde{\Theta}, p\colon A, q\colon B}{p \rightarrow q : \langle G' \rangle; G \vDash \Theta, p\colon A \wp^{q} A', q\colon B \otimes^{p} B', r\colon C', s\colon D'} \otimes\wp \quad G'' \vDash r\colon C, s\colon D}{r \rightarrow s : \langle G'' \rangle; p \rightarrow q : \langle G' \rangle; G \vDash \Theta, p\colon A \wp^{q} A', q\colon B \otimes^{p} B', r\colon C \wp^{s} C', s\colon D \wp^{r} D'} \otimes\wp
$$

Such equivalence proves that a global type of the form $p \rightarrow q : \langle G' \rangle; r \rightarrow s : \langle G'' \rangle; G$ is equivalent to $r \rightarrow s : \langle G'' \rangle; p \rightarrow q : \langle G' \rangle; G$. Formally:

Definition 1 (*Swapping congruence $\simeq^{\mathsf{g}}$*) The swapping congruence $\simeq^{\mathsf{g}}$ is the smallest congruence satisfying the rules in Fig. 3.

In general, two global types are proof terms for the same set of local typings if and only if they are equivalent. To prove this, we first need to introduce two auxiliary lemmas.

Lemma 1 *Let $G \vDash \Theta$. Then:*

- $\Theta = \Theta', p\colon A \wp^{\tilde{q}} B, \{q_i\colon C_i \otimes^{p} D_i\}_i$ *implies that the proof for deriving G contains an application of rule $\otimes\wp$ that introduces $p\colon A \wp^{\tilde{q}} B, \{q_i\colon C_i \otimes^{p} D_i\}_i$;*

$$\frac{\{p, \tilde{q}\} \cap \{r, \tilde{s}\} = \emptyset}{p \rightarrow \tilde{q} : \langle G \rangle; r \rightarrow \tilde{s} : \langle G' \rangle; G'' \simeq^{\mathsf{g}} r \rightarrow \tilde{s} : \langle G' \rangle; p \rightarrow \tilde{q} : \langle G \rangle; G''} \ (\rightarrow\rightarrow)$$

$$\frac{\{p, \tilde{q}\} \cap \{r, \tilde{s}\} = \emptyset}{p \rightarrow \tilde{q} : \&(r \rightarrow \tilde{s} : \langle G \rangle; G_1, r \rightarrow \tilde{s} : \langle G \rangle; G_2) \simeq^{\mathsf{g}} r \rightarrow \tilde{s} : \langle G \rangle; p \rightarrow \tilde{q} : \&(G_1, G_2)} \ (\rightarrow\oplus)$$

$$\frac{\{p, \tilde{q}\} \cap \{r, \tilde{s}\} = \emptyset}{\begin{array}{c} p \rightarrow \tilde{q} : \&(r \rightarrow \tilde{s} : \&(G_1, G_2), r \rightarrow \tilde{s} : \&(G_3, G_4)) \\ \simeq^{\mathsf{g}} \\ r \rightarrow \tilde{s} : \&(p \rightarrow \tilde{q} : \&(G_1, G_3), p \rightarrow \tilde{q} : \&(G_2, G_4)) \end{array}} \ (\oplus\oplus)$$

Fig. 3 Swapping relation $\simeq^{\mathsf{g}}$ for global types

- $\Theta = \Theta'$, $p: A \oplus^{\tilde{q}} B$, $\{q_i : C_i \&^p D_i\}_i$ implies that the proof for deriving G contains an application of rule $\oplus\&$ that introduces $p: A \oplus^{\tilde{q}} B$, $\{q_i : C_i \&^p D_i\}_i$;
- $\Theta = \Theta'$, $p:?A$, $\{q_i :!B_i\}_i$ implies that the proof for deriving G contains an application of rule $!?$ that introduces $p:?A$, $\{q_i :!B_i\}_i$.

Proof The thesis follows immediately from the definition of the rules for coherence, since there are no elimination rules and, e.g., rule $\otimes\otimes$ is the only one that can introduce the propositions that we are interested in. The same argument holds for the other cases. □

In the following Lemma, we consider a weight metric that will be useful later for reasoning inductively on coherence proofs. The reason for introducing this metric is that the swapping rule $(\rightarrow \oplus)$ from Fig. 3 does not preserve the size of the proof that it transforms: if we apply it from left to right, the communications at the beginning of both branches of the choice gets conflated to a single one; if we apply it from right to left, the communication before the choice gets duplicated in the branches of the choice. Even the height of the derivation tree for the proof changes when we apply this rule. However, we can define a weight metric that allows us to see that having a (swappable) communication before a choice is equivalent to having two instances of it in the two branches of the same choice. Formally, the weight $\mathsf{w}(G)$ of a global type G is a natural number; the function w is inductively defined as:

$$\mathsf{w}(\mathsf{end}^{p\tilde{q}}) = 2 \qquad \mathsf{w}(?p \rightarrow !\tilde{q} : \langle G \rangle) = \mathsf{w}(G) + 1$$
$$\mathsf{w}(p \rightarrow \tilde{q} : \&(G_1, G_2)) = \mathsf{w}(G_1) + \mathsf{w}(G_2)$$
$$\mathsf{w}(p \rightarrow \tilde{q} : \langle G' \rangle; G) = \mathsf{w}(G') \cdot \mathsf{w}(G)$$

A type $?p \rightarrow !\tilde{q} : \langle G \rangle$ has the weight of G plus 1. A choice weighs as the sum of its two branches. A communication $p \rightarrow \tilde{q} : \langle G' \rangle; G$ weighs as the product of the respective weights of G' and G. In other words, looking at the correspondence with coherence proofs, $\otimes$ corresponds to product (of weights) and $\oplus$ to sum (of weights). In the base case, the term $\mathsf{end}^{p\tilde{q}}$ weighs 2, since we must ensure that the function is strictly monotonic on the inductive definition of a term G.

Lemma 2 *Let G_1 be a valid global type. Then:*

- *$G_1 \vDash \Theta$, $p: A \otimes^{\tilde{q}} B$, $\{q_i : C_i \otimes^p D_i\}_i$ implies that there exists $G_2 = p \rightarrow \tilde{q} : \langle G_2'' \rangle; G_2'$ such that $G_1 \simeq^{\mathsf{g}} G_2$, $\mathsf{w}(G_1) = \mathsf{w}(G_2)$, and $G_2 \vDash \Theta$, $p: A \otimes^{\tilde{q}} B$, $\{q_i : C_i \otimes^p D_i\}_i$;*
- *$G_1 \vDash \Theta$, $p: A \oplus^{\tilde{q}} B$, $\{q_i : C_i \&^p D_i\}_i$ implies that there exists $G_2 = p \rightarrow \tilde{q} : \&(G_2', G_2'')$ such that $G_1 \simeq^{\mathsf{g}} G_2$, $\mathsf{w}(G_1) = \mathsf{w}(G_2)$, and $G_2 \vDash \Theta$, $p: A \oplus^{\tilde{q}} B$, $\{q_i : C_i \&^p D_i\}_i$;*
- *$G_1 \vDash \Theta$, $p:?A$, $\{q_i :!B_i\}_i$ implies that there exists $G_2 =?p \rightarrow !\tilde{q} : \langle G_2' \rangle$ such that $G_1 \simeq^{\mathsf{g}} G_2$, $\mathsf{w}(G_1) = \mathsf{w}(G_2)$, and $G_2 \vDash \Theta$, $p:?A$, $\{q_i :!B_i\}_i$.*

Proof The proof is by induction on the derivation of G_1. We focus on the first implication; the others follow by similar reasoning. By Lemma 1, we know that the proof for deriving G_1 must contain an application of rule $\otimes\!\mathbin{\bindnasrepma}$ that introduces $p\colon A\mathbin{\bindnasrepma}^{\tilde{q}} B$, $\{q_i\colon C_i \otimes^p D_i\}_i$.

We proceed by cases on the last applied rule in the derivation:

- **Case** $1\bot$ (Base case) This case is not applicable since there must be an application of $\otimes\!\mathbin{\bindnasrepma}$ (by Lemma 1).
- **Case** $\otimes\!\mathbin{\bindnasrepma}$ introducing $p\colon A\mathbin{\bindnasrepma}^{\tilde{q}} B$, $\{q_i\colon C_i \otimes^p D_i\}_i$:

$$\frac{G_1' \vDash \Theta,\; p\colon B,\; \{q_i\colon D_i\}_i \quad G_1'' \vDash p\colon A,\; \{q_i\colon C_i\}_i}{p \mathrel{-\!\!>} \tilde{q} : \langle G_1''\rangle;\, G_1' \;\vDash\; \Theta,\;\; p\colon A\mathbin{\bindnasrepma}^{\tilde{q}} B,\; \{q_i\colon C_i \otimes^p D_i\}_i} \otimes\!\mathbin{\bindnasrepma}$$

The thesis follows trivially since $G_1 = G_2$.

- **Case** $\otimes\!\mathbin{\bindnasrepma}$ not introducing $p\colon A\mathbin{\bindnasrepma}^{\tilde{q}} B$, $\{q_i\colon C_i \otimes^p D_i\}_i$:

$$\frac{G_1' \vDash \Theta',\; \begin{array}{l} r\colon F,\; \{s_j\colon I_j\}_j,\; p\colon A\mathbin{\bindnasrepma}^{\tilde{q}} B, \\ \{q_i\colon C_i \otimes^p D_i\}_i \end{array} \quad G_1'' \vDash r\colon E,\; \{s_i\colon H_i\}_i}{r \mathrel{-\!\!>} \tilde{s} : \langle G_1''\rangle;\, G_1' \;\vDash\; \begin{array}{l} \Theta',\; r\colon E\mathbin{\bindnasrepma}^{\tilde{s}} F,\; \{s_j\colon H_j \otimes^r I_j\}_j, \\ p\colon A\mathbin{\bindnasrepma}^{\tilde{q}} B,\; \{q_i\colon C_i \otimes^p D_i\}_i \end{array}} \otimes\!\mathbin{\bindnasrepma}$$

By induction hypothesis, we know that we can rewrite this proof as:

$$\frac{\mathcal{F} \qquad G_1'' \vDash r\colon E,\; \{s_i\colon H_i\}_i}{r \mathrel{-\!\!>} \tilde{s} : \langle G_1''\rangle;\, p \mathrel{-\!\!>} \tilde{q} : \langle G_2''\rangle;\, G_2' \;\vDash\; \begin{array}{l} \Theta',\; r\colon E\mathbin{\bindnasrepma}^{\tilde{s}} F,\; \{s_j\colon H_j \otimes^r I_j\}_j, \\ p\colon A\mathbin{\bindnasrepma}^{\tilde{q}} B,\; \{q_i\colon C_i \otimes^p D_i\}_i \end{array}} \otimes\!\mathbin{\bindnasrepma}$$

where

$$\mathcal{F} = \frac{G_2' \vDash \Theta',\; r\colon F,\; \{s_j\colon I_j\}_j,\; p\colon B,\; \{q_i\colon D_i\}_i \quad G_2'' \vDash p\colon A,\; \{q_i\colon C_i\}_i}{p \mathrel{-\!\!>} \tilde{q} : \langle G_2''\rangle;\, G_2' \;\vDash\; \begin{array}{l} \Theta',\; r\colon F,\; \{s_j\colon I_j\}_j, \\ p\colon A\mathbin{\bindnasrepma}^{\tilde{q}} B,\; \{q_i\colon C_i \otimes^p D_i\}_i \end{array}} \otimes\!\mathbin{\bindnasrepma}$$

such that $G_1' \simeq^{\mathsf{g}} p \mathrel{-\!\!>} \tilde{q} : \langle G_2''\rangle;\, G_2'$ and $\mathsf{w}(G_1') = \mathsf{w}(p \mathrel{-\!\!>} \tilde{q} : \langle G_2''\rangle;\, G_2')$. Hence, by the fact that the swapping relation $\simeq^{\mathsf{g}}$ is a congruence, we have $r \mathrel{-\!\!>} \tilde{s} : \langle G_1''\rangle;\, G_1' \simeq^{\mathsf{g}} r \mathrel{-\!\!>} \tilde{s} : \langle G_1''\rangle;\, p \mathrel{-\!\!>} \tilde{q} : \langle G_2''\rangle;\, G_2'$. Moreover, applying rule $(\to\to)$ from the definition of $\simeq^{\mathsf{g}}$, we obtain:

$$\frac{\mathcal{F}' \qquad G_2'' \vDash p\colon A,\; \{q_i\colon C_i\}_i}{p \mathrel{-\!\!>} \tilde{q} : \langle G_2''\rangle;\, r \mathrel{-\!\!>} \tilde{s} : \langle G_1''\rangle;\, G_2' \;\vDash\; \begin{array}{l} \Theta',\; r\colon E\mathbin{\bindnasrepma}^{\tilde{s}} F,\; \{s_j\colon H_j \otimes^r I_j\}_j, \\ p\colon A\mathbin{\bindnasrepma}^{\tilde{q}} B,\; \{q_i\colon C_i \otimes^p D_i\}_i \end{array}} \otimes\!\mathbin{\bindnasrepma}$$

where

$$\mathcal{F}' = \frac{G_2' \vDash \Theta',\; r\colon F,\; \{s_j\colon I_j\}_j,\; p\colon B,\; \{q_i\colon D_i\}_i \quad G_1'' \vDash r\colon E,\; \{s_i\colon H_i\}_i}{r \mathrel{-\!\!>} \tilde{s} : \langle G_1''\rangle;\, G_2' \vDash \Theta',\; r\colon F,\; \{s_j\colon I_j\}_j,\; p\colon A\mathbin{\bindnasrepma}^{\tilde{q}} B,\; \{q_i\colon C_i \otimes^p D_i\}_i} \otimes\!\mathbin{\bindnasrepma}$$

such that $r \mathrel{-\!\!>} \tilde{s} : \langle G_1''\rangle;\, G_1' \simeq^{\mathsf{g}} p \mathrel{-\!\!>} \tilde{q} : \langle G_2''\rangle;\, r \mathrel{-\!\!>} \tilde{s} : \langle G_1''\rangle;\, G_2'$ and $\mathsf{w}(r \mathrel{-\!\!>} \tilde{s} : \langle G_1''\rangle;\, G_1') = \mathsf{w}(p \mathrel{-\!\!>} \tilde{q} : \langle G_2''\rangle;\, r \mathrel{-\!\!>} \tilde{s} : \langle G_1''\rangle;\, G_2')$.

- The cases for $\oplus\&$ and $!?$ are similar to the previous one. For case $\oplus\&$, we may have to apply the transformation $(\to \oplus)$. As anticipated, although this transformation changes the size of a proof, it does not change its weight (which is what we need to prove here). This

follows from simple distribution of multiplication over addition. Rule ($\rightarrow \oplus$) states (we report it verbatim, since our argument holds independently from this particular proof):

$$p \rightarrow \tilde{q} : \&(r \rightarrow \tilde{s} : \langle G \rangle; G_1, r \rightarrow \tilde{s} : \langle G \rangle; G_2) \simeq^g r \rightarrow \tilde{s} : \langle G \rangle; p \rightarrow \tilde{q} : \&(G_1, G_2)$$

We can easily verify that weight remains unaffected:

$$
\begin{array}{ll}
\mathsf{w}(\, p \rightarrow \tilde{q} : \&(r \rightarrow \tilde{s} : \langle G \rangle; G_1, r \rightarrow \tilde{s} : \langle G \rangle; G_2)\,) = & \textit{(by definition of } \mathsf{w}) \\
\mathsf{w}(r \rightarrow \tilde{s} : \langle G \rangle; G_1\,) + \mathsf{w}(r \rightarrow \tilde{s} : \langle G \rangle; G_2\,) = & \textit{(by definition of } \mathsf{w}) \\
\big(\mathsf{w}(G) \cdot \mathsf{w}(G_1)\big) + \big(\mathsf{w}(G) \cdot \mathsf{w}(G_2)\big) = & \textit{(by distribution)} \\
\mathsf{w}(G) \cdot \big(\mathsf{w}(G_1) + \mathsf{w}(G_2)\big) = & \textit{(by definition of } \mathsf{w}) \\
\mathsf{w}(r \rightarrow \tilde{s} : \langle G \rangle; p \rightarrow \tilde{q} : \&(G_1, G_2)\,) &
\end{array}
$$

$\square$

Proposition 1 (Swapping) *Let $G_1 \vDash \Theta$. Then, $G_1 \simeq^g G_2$ if and only if $G_2 \vDash \Theta$.*

Proof (only if direction) For each rule defining $G_1 \simeq^g G_2$, we can expand G_1 to its corresponding coherence proofs and commute its last applied rule to obtain a proof for G_2 with the same Θ. In each case, the two proofs prove the same Θ. We report the representative case of ($\rightarrow \oplus$). We have that:

$$G_1 = p \rightarrow \tilde{q} : \&(r \rightarrow \tilde{s} : \langle G_1'' \rangle; G_1', r \rightarrow \tilde{s} : \langle G_1'' \rangle; G_2')$$
$$G_2 = r \rightarrow \tilde{s} : \langle G_1'' \rangle; p \rightarrow \tilde{q} : \&(G_1', G_2')$$

$$\frac{\{p, \tilde{q}\} \cap \{r, \tilde{s}\} = \emptyset}{\begin{array}{c} p \rightarrow \tilde{q} : \&(r \rightarrow \tilde{s} : \langle G_1'' \rangle; G_1', r \rightarrow \tilde{s} : \langle G_1'' \rangle; G_2') \\ \simeq^g r \rightarrow \tilde{s} : \langle G_1'' \rangle; p \rightarrow \tilde{q} : \&(G_1', G_2') \end{array}} \ (\rightarrow \oplus)$$

The proof for G_1 is:

$$\frac{\mathcal{D} \qquad\qquad \mathcal{E}}{p \rightarrow \tilde{q} : \&(r \rightarrow \tilde{s} : \langle G_1'' \rangle; G_1', r \rightarrow \tilde{s} : \langle G_1'' \rangle; G_2') \vDash \begin{array}{l} \Theta', r : E \otimes^{\tilde{s}} F, \{s_j : H_j \otimes^r I_j\}_j, \\ p : A \oplus^{\tilde{q}} B, \{q_i : C_i \&^p D_i\}_i \end{array}} \ \oplus\&$$

where

$$\mathcal{D} = \frac{G_1' \vDash \Theta', r : F, \{s_j : I_j\}_j, p : A, \{q_i : C_i\}_i \qquad G_1'' \vDash r : E, \{s_j : H_j\}_j}{r \rightarrow \tilde{s} : \langle G_1'' \rangle; G_1' \vDash \Theta', r : E \otimes^{\tilde{s}} F, \{s_j : H_j \otimes^r I_j\}_j, p : A, \{q_i : C_i\}_i} \ \otimes \otimes$$

and

$$\mathcal{E} = \frac{G_2' \vDash \Theta', r : F, \{s_j : I_j\}_j, p : B, \{q_i : D_i\}_i \qquad G_1'' \vDash r : E, \{s_j : H_j\}_j}{r \rightarrow \tilde{s} : \langle G_1'' \rangle; G_2' \vDash \Theta', r : E \otimes^{\tilde{s}} F, \{s_j : H_j \otimes^r I_j\}_j, p : B, \{q_i : D_i\}_i} \ \otimes \otimes$$

We can rewrite the proof above to obtain one for G_2, proving the thesis:

$$\frac{\mathcal{F} \qquad\qquad G_1'' \vDash r : E, \{s_j : H_j\}_j}{r \rightarrow \tilde{s} : \langle G_1'' \rangle; p \rightarrow \tilde{q} : \&(G_1', G_2') \vDash \begin{array}{l} \Theta', r : E \otimes^{\tilde{s}} F, \{s_j : H_j \otimes^r I_j\}_j, \\ p : A \oplus^{\tilde{q}} B, \{q_i : C_i \&^p D_i\}_i \end{array}} \ \otimes \otimes$$

where

$$\mathcal{F} = \frac{G_1' \vDash \begin{array}{l} \Theta', r : F, \\ \{s_j : I_j\}_j, p : A, \{q_i : C_i\}_i \end{array} \qquad G_2' \vDash \begin{array}{l} \Theta', r : F, \\ \{s_j : I_j\}_j, p : B, \{q_i : D_i\}_i \end{array}}{p \rightarrow \tilde{q} : \&(G_1', G_2') \vDash \Theta', r : F, \{s_j : I_j\}_j, p : A \oplus^{\tilde{q}} B, \{q_i : C_i \&^p D_i\}_i} \ \oplus\&$$

$\square$

Proof (*if direction*) We prove that $G_1 \vDash \Theta$ and $G_2 \vDash \Theta$ imply $G_1 \simeq^g G_2$. We proceed by mutual induction on the weights of the proof derivations of G_1 and G_2. For the base case where G_1 is derived by rule $1\bot$, then G_2 must be derived in the same way, hence the two proofs are the same and the thesis follows. We now move to the inductive cases, looking first at the last applied rule used to derive G_1.

- **Case $\otimes\bindnasrepma$.** Here we have that $G_1 = p \rightarrow \tilde{q} : \langle G_1' \rangle; G_1''$ such that:

$$\frac{G_1' \vDash \Theta', \ p\colon B, \ \{q_i\colon D_i\}_i \quad G_1'' \vDash p\colon A, \ \{q_i\colon C_i\}_i}{p \rightarrow \tilde{q} : \langle G_1'' \rangle; G_1' \ \vDash \ \Theta', \ p\colon A\bindnasrepma^{\tilde{q}} B, \ \{q_i\colon C_i \otimes^p D_i\}_i} \ \otimes\bindnasrepma$$

If G_2 ends with an application of $\otimes\bindnasrepma$ that introduces the same principal formulas, we have that $G_2 = p \rightarrow \tilde{q} : \langle G_2'' \rangle; G_2'$. Since G_2 has the same typing of G_1, from rule $\otimes\bindnasrepma$ we know that also G_2'' and G_2' have the same typings of G_1'' and G_1' respectively, because the rule directs precisely the distribution of roles and types by looking at the role annotations. The thesis now follows directly by induction hypothesis (since the proofs for the premises are smaller). Otherwise, by Lemma 1 we know that we can apply Lemma 2 to obtain a G_3 such that: $G_2 \simeq^g G_3$; $G_3 \vDash \Theta', \ p\colon A\bindnasrepma^{\tilde{q}} B, \ \{q_i\colon C_i \otimes^p D_i\}_i$; $G_3 = p \rightarrow \tilde{q} : \langle G_3'' \rangle; G_3'$; and, the weight of the derivation of G_3 is the same as that of the derivation of G_2. By the correspondence between global types and coherence proofs, we know that:

$$\frac{G_3' \vDash \Theta', \ p\colon B, \ \{q_i\colon D_i\}_i \quad G_3'' \vDash p\colon A, \ \{q_i\colon C_i\}_i}{p \rightarrow \tilde{q} : \langle G_3'' \rangle; G_3' \ \vDash \ \Theta', \ p\colon A\bindnasrepma^{\tilde{q}} B, \ \{q_i\colon C_i \otimes^p D_i\}_i} \ \otimes\bindnasrepma$$

The thesis now follows by induction hypothesis on G_3'' and G_3'.
- **Case $\oplus\&$.** This case is similar to that for $\otimes\bindnasrepma$.
- **Case !?.** Here we have that $G_1 = ?p \rightarrow !\tilde{q} : \langle G_1' \rangle$ such that:

$$\frac{G_1' \vDash p\colon A, \{q_i\colon B_i\}_i}{?p \rightarrow !\tilde{q} : \langle G_1' \rangle \vDash p\colon{?}A, \{q_i\colon{!}B_i\}_i} \ !?$$

By hypothesis, we know that $G_2 \vDash p\colon{?}A, \{q_i\colon{!}B_i\}_i$. That means that G_2 has !? as last applied rule:

$$\frac{G_2' \vDash p\colon A, \{q_i\colon B_i\}_i}{?p \rightarrow !\tilde{q} : \langle G_2' \rangle \vDash p\colon{?}A, \{q_i\colon{!}B_i\}_i} \ !?$$

The thesis now follows by induction hypothesis. □

Projection and extraction The hallmark of the theory of multiparty session types is projection: developers can write protocols as global types, and then automatically project a global type onto a set of local types that can be used to modularly verify the behaviour of each participant. As there is only one possible rule application for each production in the syntax of global types, we can construct an algorithm that traverses the structure of G:

Proposition 2 (Projection) *Given G, we can compute in linear time Θ (if it exists) such that $G \vDash \Theta$.*

We can also use coherence for the inverse procedure, i.e., the extraction of a global type from a set of local typings Θ. If Θ is coherent, we can just apply the first applicable coherence rule, noting that the sizes of the local types in the premises always get smaller:

Proposition 3 (Extraction) *Given Θ, we can compute G (if it exists) such that $G \vDash \Theta$.*

Example 2 In the 2-buyer protocol, $G \vDash \mathsf{B1}: A, \mathsf{B2}: B, \mathsf{S}: C$ implies: (i) we can infer A, B and C from G (proposition 2) and (ii) we can extract G from $\mathsf{B1}: A, \mathsf{B2}: B, \mathsf{S}: C$ (proposition 3). This observation follows directly from the proof coherence, which we describe now. Three applications of the axiom $1\bot$ rule yield:

$$\mathsf{end} \vDash \mathsf{B1}: \bot, \mathsf{S}: 1 \tag{5}$$

$$\mathsf{end} \vDash \mathsf{B2}: 1, \mathsf{S}: 1 \tag{6}$$

$$\mathsf{end} \vDash \mathsf{B1}: \bot, \mathsf{B2}: 1, \mathsf{S}: 1 \tag{7}$$

We omit the annotations to end for better readability. Next, we combine (5) and (6) using $\otimes \mathbin{\rotatebox[origin=c]{180}{\&}}$.

$$\mathsf{B2} \rightarrow \mathsf{S}: \langle \mathsf{end} \rangle; \mathsf{end} \vDash \mathsf{B2}: \bot \mathbin{\rotatebox[origin=c]{180}{\&}}^{\mathsf{S}} 1, \mathsf{S}: 1 \otimes^{\mathsf{B2}} \bot \tag{8}$$

Next, we combine (7) and (8) using $\oplus \&$ to obtain

$$\vDash \mathsf{B1}: \bot, \mathsf{B2}: \left((\bot \mathbin{\rotatebox[origin=c]{180}{\&}}^{\mathsf{S}} 1) \oplus^{\mathsf{S}} 1 \right), \mathsf{S}: ((1 \otimes^{\mathsf{B2}} \bot) \&^{\mathsf{B2}} 1)$$

The corresponding global type is

$$\mathsf{S} \rightarrow \mathsf{B2}: \&(\mathsf{B2} \rightarrow \mathsf{S}: \langle \mathsf{end} \rangle; \mathsf{end}, \mathsf{end}).$$

Another application of $\otimes \mathbin{\rotatebox[origin=c]{180}{\&}}$ with an axiom application in the right premiss

$$\mathsf{end} \vDash \mathsf{B1}: \bot, \mathsf{S}: 1$$

yields

$$\vDash \mathsf{B1}: \bot \mathbin{\rotatebox[origin=c]{180}{\&}}^{\mathsf{B2}} \bot, \mathsf{B2}: 1 \otimes^{\mathsf{B1}} \left((\bot \mathbin{\rotatebox[origin=c]{180}{\&}}^{\mathsf{S}} 1) \oplus^{\mathsf{S}} 1 \right), \mathsf{S}: ((1 \otimes^{\mathsf{B2}} \bot) \&^{\mathsf{B2}} 1) \tag{9}$$

with global type

$$\mathsf{B2} \rightarrow \mathsf{B1}: \langle \mathsf{end} \rangle; \mathsf{S} \rightarrow \mathsf{B2}: \&(\mathsf{B2} \rightarrow \mathsf{S}: \langle \mathsf{end} \rangle; \mathsf{end}, \mathsf{end}).$$

We repeat this operation three more times using rule $\otimes \mathbin{\rotatebox[origin=c]{180}{\&}}$ on (9) with the appropriate instances of the axiom rule as right premisses and obtain the coherence proof of

$$\vDash \mathsf{B1}: A, \mathsf{B2}: B, \mathsf{S}: C$$

with the appropriate global type:

$$\begin{aligned}
&\mathsf{B1} \rightarrow \mathsf{S}: \langle \mathsf{end} \rangle; \\
&\mathsf{B1} \rightarrow \mathsf{S}: \langle \mathsf{end} \rangle; \\
&\mathsf{B2} \rightarrow \mathsf{S}: \langle \mathsf{end} \rangle; \\
&\mathsf{B2} \rightarrow \mathsf{B1}: \langle \mathsf{end} \rangle; \\
&\mathsf{S} \rightarrow \mathsf{B2}: \&(\mathsf{B2} \rightarrow \mathsf{S}: \langle \mathsf{end} \rangle; \mathsf{end}, \mathsf{end}).
\end{aligned}$$

Global reductions We define reductions for global types, denoted $\tilde{G} \rightsquigarrow \tilde{G}'$, where $\tilde{G}$ is a multiset $\{G_1, \ldots, G_n\}$. *Global type reductions are just a convention* (recalling [8]), which we use in Sect. 5 to concisely formalise how processes follow their protocols. Formally, $\rightsquigarrow$ is the smallest relation satisfying the rules in Fig. 4. Rule $\mathsf{g}_{\otimes \mathbin{\rotatebox[origin=c]{180}{\&}}}$ models a communication that creates a new session of type G', which will then proceed in parallel to the continuation G. Rule $\mathsf{g}_{1\bot}$ models session termination. Rules $\mathsf{g}_{\oplus\&1}$ and $\mathsf{g}_{\oplus\&2}$ model the execution of a choice. In rules $\mathsf{g}_{!?}$, $\mathsf{g}_{!C}$ and $\mathsf{g}_{!W}$, a replicated protocol can be respectively executed exactly

$$(\mathsf{g}_{\otimes\mathscr{B}}) \ \{ \ p \to \tilde{q} : \langle G' \rangle; G \ \} \ \rightsquigarrow \ \{ \ G, G' \ \}$$
$$(\mathsf{g}_{\oplus\&1}) \ \{ \ p \to \tilde{q} : \&(G_1, G_2) \ \} \rightsquigarrow \{ \ G_1 \ \}$$
$$(\mathsf{g}_{\oplus\&2}) \ \{ \ p \to \tilde{q} : \&(G_1, G_2) \ \} \rightsquigarrow \{ \ G_2 \ \}$$
$$(\mathsf{g}_{!?}) \ \{ \ ?p \to !\tilde{q} : \langle G \rangle \ \} \ \rightsquigarrow \ \{ \ G \ \}$$
$$(\mathsf{g}_{!C}) \ \{ \ ?p \to !\tilde{q} : \langle G \rangle \ \} \rightsquigarrow \{ \ ?p \to !\tilde{q} : \langle G \rangle, ?p \to !\tilde{q} : \langle G \rangle \ \}$$
$$(\mathsf{g}_{!W}) \ \{ \ ?p \to !\tilde{q} : \langle G \rangle \ \} \rightsquigarrow \emptyset$$
$$(\mathsf{g}_{1\perp}) \ \{ \ \mathsf{end}^{p\tilde{q}} \ \} \rightsquigarrow \emptyset$$
$$(\mathsf{g}_{\mathsf{ctx}}) \ \tilde{G}_1 \rightsquigarrow \tilde{G}_2 \ \Rightarrow \ \tilde{G}, \tilde{G}_1 \rightsquigarrow \tilde{G}, \tilde{G}_2$$
$$(\mathsf{g}_{\mathsf{eq}}) \ \tilde{G}_0 \simeq^{\mathsf{g}} \tilde{G}_1 \quad \tilde{G}_1 \rightsquigarrow \tilde{G}_2 \quad \tilde{G}_2 \simeq^{\mathsf{g}} \tilde{G}_3 \ \Rightarrow \ \tilde{G}_0 \rightsquigarrow \tilde{G}_3$$

Fig. 4 Global types, reduction semantics

once, multiple, or zero times. Rule $\mathsf{g}_{\mathsf{ctx}}$ lifts the behaviour of a protocol to a multiset of protocols executing concurrently. We abuse the notation $\tilde{G}, \tilde{G}'$ to indicate the union of the two multisets $\tilde{G}$ and $\tilde{G}'$. Finally, rule g_{eq} allows for swappings in a global type. In this rule, $\tilde{G} \simeq^{\mathsf{g}} \tilde{G}'$ is the point-wise extension of the swapping relation $\simeq^{\mathsf{g}}$ to multisets. Formally, $\tilde{G} \simeq^{\mathsf{g}} \tilde{G}'$ if and only if $\tilde{G} = \{G_1, \ldots, G_n\}$, $\tilde{G}' = \{G'_1, \ldots, G'_n\}$, and $G_i \simeq^{\mathsf{g}} G'_i$ for all $i \in [1, n]$. Our semantics preserves validity. Below we write that $\tilde{G}$ is valid if all the G_i in $\tilde{G}$ are valid.

Theorem 1 (Coherence preservation) *If $\tilde{G}$ is valid and $\tilde{G} \rightsquigarrow \tilde{G}'$, then $\tilde{G}'$ is valid.*

Remark 1 Rule $\mathsf{g}_{!?}$ can be derived from rules $\mathsf{g}_{!C}$ and $\mathsf{g}_{!W}$. Including it simplifies our presentation, since each global type reduction corresponds to a communication in MCP (Sect. 5).

Coherence as generalised duality Coherence is a generalisation of duality (from CLL [15]): in the degenerate case of a session with two participants, the two notions coincide. We recall the definition of duality $X^\perp$, defined inductively on the syntax of linear logic propositions:

$$(X \otimes Y)^\perp = X^\perp \mathscr{B} Y^\perp \qquad (X \mathscr{B} Y)^\perp = X^\perp \otimes Y^\perp$$
$$1^\perp = \perp \qquad\qquad \perp^\perp = 1$$
$$(X \oplus Y)^\perp = X^\perp \& Y^\perp \qquad (X \& Y)^\perp = X^\perp \oplus Y^\perp$$
$$(!X)^\perp = ?X^\perp \qquad\qquad (?X)^\perp = !X^\perp$$

We define a partial encoding $[\![\cdot]\!]$ from local types into linear logic propositions:

$$[\![1]\!] = 1 \quad [\![\perp]\!] = \perp \quad [\![!A]\!] = ![\![A]\!] \quad [\![?A]\!] = ?[\![A]\!] \quad [\![A \otimes^p B]\!] = [\![A]\!] \otimes [\![B]\!]$$
$$[\![A \mathscr{B}^q B]\!] = [\![A]\!] \mathscr{B} [\![B]\!] \quad [\![A \oplus^p B]\!] = [\![A]\!] \oplus [\![B]\!] \quad [\![A \&^p B]\!] = [\![A]\!] \& [\![B]\!]$$

The encoding $[\![\cdot]\!]$ is defined only when $\mathscr{B}$ and $\oplus$ are annotated with a single role. We get:

Proposition 4 (Coherence as duality) *Let A, B be propositions where all subterms of the form $C \mathscr{B}^{\tilde{p}} D$ or $C \oplus^{\tilde{p}} D$ are such that $\tilde{p} = q$ for some q. Then, $[\![A]\!] = [\![B]\!]^\perp$ if and only if there exists G such that $G \vDash p : A, q : B$.*

Observe that, in Proposition 4, the G corresponding to the coherence proof for the validity of $p : A, q : B$ is necessarily unique, since coherence is deterministic in the case of two propositions – the structures of the propositions force the order in which the rules must be applied.

$$P, Q, R ::= \quad \overline{x}^{p\tilde{q}}(y); P \quad \text{(send)} \quad | \quad x^{pq}(y); (P \mid Q) \quad \text{(recv)}$$

$$| \quad x^{p\tilde{q}}.\mathsf{inl}; P \quad \text{(left sel)} \quad | \quad x^{p\tilde{q}}.\mathsf{inr}; P \quad \text{(right sel)}$$

$$| \quad x^{pq}.\mathsf{case}(P, Q) \quad \text{(case)} \quad | \quad (\boldsymbol{\nu}x{:}G)\left(\prod_i P_i\right) \quad \text{(res)}$$

$$| \quad \mathsf{close}\, x^p \quad \text{(close)} \quad | \quad \mathsf{wait}\, x^p; P \quad \text{(wait)}$$

$$| \quad !x^p(y); P \quad \text{(service)} \quad | \quad ?x^p(y); P \quad \text{(client)}$$

$$| \quad P + Q \quad \text{(choice)}$$

Fig. 5 MCP, syntax of processes

4 Multiparty classical processes

In this section, we present Multiparty Classical Processes (MCP). MCP captures dependencies among actions performed by different participants in a multiparty session, whereas, in previous work, actions among different pairs of participants must be independent [6,28]. We use the synchronous semantics from [18] for a simplicity of the presentation.

Environments Let Γ, Δ range over typing environments:

$$\Gamma, \Delta \; ::= \; \cdot \quad | \quad \Gamma, x^p : A$$

Intuitively, $x^p : A$ means that role p in session x follows behaviour A. We write $?\Delta$ whenever Δ contains only types of the form $?A$, and write $\Delta, x^p : A$ only when x^p does not appear in Δ.

Processes We report the syntax of processes in Fig. 5. In MCP, both input and output names are bound, as in [28]. Term *(send)* creates a new session y and sends it, as role p, to the processes respectively playing roles $\tilde{q}$ in session x; then, the process proceeds as P. The dual operation *(recv)* receives, as role p in session x, a fresh session y from the process playing role q; the process then proceeds as the parallel composition of P (dedicated to session y) and Q (dedicated to continuing session x). Similarly, terms *(left sel)* and *(right sel)* multicast a selection of a left or right branch respectively to the processes playing roles $\tilde{q}$ in session x, as role p. A selection is received by term *(case)*, which offers the two selectable branches. Terms *(close)* and *(wait)* terminate a session. Term *(choice)* is the standard non-deterministic choice. In a restriction *(res)*, x is bound in the processes P_i; we use the standard type annotation (as in [28]) to show the relation between the semantics of processes and global types in Sect. 5. In term $x^{pq}(y); (P \mid Q)$, y is bound in P but not in Q. In terms $\overline{x}^{p\tilde{q}}(y); P$, $!x^p(y); P$, and $?x^p(y); P$, y is bound in P.

Judgements Judgements in MCP have the form $P \vdash x_1^{p_1} : A_1, \ldots, x_n^{p_n} : A_n$, meaning that process P implements roles p_i in the respective session x_i with behaviour A_i.

Rules We report the rules of MCP in Fig. 6. Intuitively, a process is typed with local types; then, we use coherence to check that the local types of composed processes (rule MCut) coherently implement a global type. All rules are defined up to context exchange.

Rule MCut is central: it extends the Cut of CLL to composing in parallel an arbitrary number of P_i that communicate using session x. The rule checks that the composition of the respective local behaviours of the composed processes is coherent ($G \vDash \{p_i : A_i\}_i$). In the conclusion, $\{\Gamma_i\}_i$ is the disjoint union of all Γ_i in the premise.

$$\frac{P \vdash \Gamma, y^p : A \quad Q \vdash \Delta, x^p : B}{x^{pq}(y); (P \mid Q) \vdash \Gamma, \Delta, x^p : A \otimes^q B} \ \otimes \qquad \frac{P \vdash \Gamma, y^p : A, x^p : B}{\overline{x}^{p\tilde{q}}(y); P \vdash \Gamma, x^p : A \otimes^{\tilde{q}} B} \ \otimes$$

$$\frac{P \vdash \Gamma \quad Q \vdash \Gamma}{P + Q \vdash \Gamma} \ + \qquad \frac{P \vdash \Gamma, x^p : A \quad Q \vdash \Gamma, x^p : B}{x^{pq}.\mathsf{case}(P, Q) \vdash \Gamma, x^p : A \&^q B} \ \&$$

$$\frac{\{P_i \vdash \Gamma_i, x^{p_i} : A_i\}_i \quad G \vDash \{p_i : A_i\}_i}{(\boldsymbol{\nu} x : G)\left(\prod_i P_i\right) \vdash \{\Gamma_i\}_i} \ \mathsf{MCut}$$

$$\frac{P \vdash \Gamma, x^p : A}{x^{p\tilde{q}}.\mathsf{inl}; P \vdash \Gamma, x^p : A \oplus^{\tilde{q}} B} \ \oplus_1 \qquad \frac{P \vdash \Gamma, x^p : B}{x^{p\tilde{q}}.\mathsf{inr}; P \vdash \Gamma, x^p : A \oplus^{\tilde{q}} B} \ \oplus_2$$

$$\frac{}{\mathsf{close}\, x^p \vdash x^p : 1} \ 1 \qquad \frac{P \vdash \Gamma}{\mathsf{wait}\, x^p; P \vdash \Gamma, x^p : \perp} \ \perp$$

$$\frac{P \vdash ?\Gamma, y^p : A}{!x^p(y); P \vdash ?\Gamma, x^p : \,!A} \ ! \qquad \frac{P \vdash \Gamma, y^p : A}{?x^p(y); P \vdash \Gamma, x^p : \,?A} \ ?$$

$$\frac{P \vdash \Gamma}{P \vdash \Gamma, x^p : \,?A} \ \mathsf{Weaken} \qquad \frac{P \vdash \Gamma, y^p : \,?A, z^p : \,?A}{P[x/y][x/z] \vdash \Gamma, x^p : \,?A} \ \mathsf{Contract}$$

Fig. 6 MCP, typing rules

Rule $\otimes$ types an input $x^{pq}(y); (P \mid Q)$, where the subprocess P plays role p with behaviour A in the received multiparty session y; session x then proceeds by following behaviour B for role p in Q. Observe that the $\otimes$ is annotated with the role q that p wishes to receive from. The multicast output $\overline{x}^{p\tilde{q}}(y); P$ in rule $\otimes$ creates a new session y and sends it, as role p in session x, to roles $\tilde{q}$. The new session y is used by P as role p with type A, assuming that the other processes receiving it implement the other roles (this assumption is checked by coherence in MCut, when processes are composed). We discuss in Sect. 8 how to relax the constraint that the role p played in session y is the same.

Rules $\oplus_1$ and $\oplus_2$ type, respectively, the multicast of a left and right selection, by checking that the process continuation follows the expected local type. Similarly, rule $\&$ types a branching by checking that the continuations implement the respective expected local types.

Rule $+$ types the nondeterministic process $P + Q$, by checking that both P and Q implement the same local behaviours. Observe that P and Q may still be substantially different, since they may (i) perform different selections on some sessions (as rules $\oplus_1$ and $\oplus_2$ can yield the same typing), and (ii) have different inner compositions of processes whose types have been hidden by rule MCut.

Rules 1 and $\perp$ type, respectively, the request and the acceptance for closing a multiparty session. Rules ! and ? type, respectively, the replicated offering of a service and its repeated usage (a client). Since a service typed by ! may be used multiple times, we require that its continuation does not use any linear behaviour ($?\Delta$). Rules Weaken and Contract type, respectively, the absence of clients or the presence of multiple clients. In rule Contract, sessions y and z are contracted into a single session x with a standard name substitution, provided that they have the same type $?A$.

5 Semantics

In this section, we demonstrate the consistency of MCP, by establishing a cut-elimination result that yields an operational semantics and important properties, e.g., deadlock-freedom.

5.1 Structural equivalences as commuting conversions

MCP supports *commuting conversions*, permutations of applications of MCut that maintain the validity of judgements. As an example, consider the following proof equivalence ($\equiv$):

$$\dfrac{\dfrac{P \vdash \Delta, y^p : A, x^p : B, z^r : C}{\overline{x}^{p\tilde{q}}(y); P \vdash \Delta, x^p : A \otimes^{\tilde{q}} B, z^r : C} \;\otimes \quad Q_i \vdash \Gamma_i, z^{s_i} : D_i \quad G \vDash r : C, \{s_i : D_i\}_i}{(\boldsymbol{\nu}z : G)\left(\overline{x}^{p\tilde{q}}(y); P \mid \prod_i Q_i\right) \vdash \{\Gamma_i\}_i, \Delta, x^p : A \otimes^{\tilde{q}} B} \; \text{MCut}$$

$$\equiv$$

$$\dfrac{\dfrac{P \vdash \Delta, y^p : A, x^p : B, z^r : C \quad Q_i \vdash \Gamma_i, z^{s_i} : D_i \quad G \vDash r : C, \{s_i : D_i\}_i}{(\boldsymbol{\nu}z : G)\left(P \mid \prod_i Q_i\right) \vdash \{\Gamma_i\}_i, \Delta, y^p : A, x^p : B} \; \text{MCut}}{\overline{x}^{p\tilde{q}}(y); (\boldsymbol{\nu}z : G)\left(P \mid \prod_i Q_i\right) \vdash \{\Gamma_i\}_i, \Delta, x^p : A \otimes^{\tilde{q}} B} \; \otimes$$

Above, an output is moved out of a restriction of a different session (or in it, reading in the other direction), as in [28]. In this example, the output process is the first in the parallel under the restriction; in general, this is not always the case since the process may be any of those in the parallel composition. In order to represent equivalences independently of the position of processes in a parallel, we use process contexts [24]. A context, denoted by $\mathcal{C}$, is a parallel composition with a hole: $\mathcal{C}[\cdot] ::= \cdot \quad | \quad \mathcal{C}[\cdot] \mid P \quad | \quad P \mid \mathcal{C}[\cdot]$. All equivalences are reported in Fig. 7. The equivalence κ_{par} permutes processes in a parallel, since the premises of rule MCut can be in any order. In κ_{cut}, we can swap two restrictions, which corresponds to swapping two applications of rule MCut. The equivalence $\kappa_{\otimes}$ shows that a restriction can always be swapped with an output on a different session. Similarly, the equivalence $\kappa_{\otimes}$ swaps a restriction with an input, requiring that the restricted name (z in this case) occurs free in P. In the case of $\oplus$, we have two equivalences, corresponding to the right and left selection respectively. For $\kappa_{\&}$, we can move a restriction to each branch of a case construct, also duplicating the context $\mathcal{C}$. Equivalences $\kappa_!$ and $\kappa_?$ allow to swap a restriction with a service and a client respectively. Finally, $\kappa_{\perp}$ is the case for $\mathsf{wait}\, x^p$. There is no equivalence for the process $\mathsf{close}\, x^p$ since it is only typable with the axiom 1.

$$
\begin{array}{rll}
(\kappa_{\mathsf{par}}) & (\boldsymbol{\nu}z : G)\left(\prod_{i \in \tilde{k}} P_i\right) \equiv (\boldsymbol{\nu}z : G)\left(\prod_{j \in \tilde{k}'} P_j\right) & (\tilde{k} \text{ permutation of } \tilde{k}') \\[4pt]
(\kappa_{\mathsf{cut}}) & (\boldsymbol{\nu}x : G)\left(\mathcal{C}\left[(\boldsymbol{\nu}y : G')\, \mathcal{C}'[P]\right]\right) \equiv (\boldsymbol{\nu}y : G')\left(\mathcal{C}'\left[(\boldsymbol{\nu}x : G)\, \mathcal{C}[P]\right]\right) & (x, y \in \mathsf{fn}(P), \\
& & x, y \text{ not free in } \mathcal{C}[\cdot]) \\[4pt]
(\kappa_{\otimes}) & (\boldsymbol{\nu}z : G)\left(\mathcal{C}\left[\overline{x}^{p\tilde{q}}(y); P\right]\right) \equiv \overline{x}^{p\tilde{q}}(y); (\boldsymbol{\nu}z : G)\, \mathcal{C}[P] & \\[4pt]
(\kappa_{\otimes}) & (\boldsymbol{\nu}z : G)\left(\mathcal{C}\left[x^{pq}(y); (P \mid Q)\right]\right) \equiv x^{pq}(y); (P \mid (\boldsymbol{\nu}z : G)\,(\mathcal{C}[Q])) & (z \notin \mathsf{fn}(P)) \\[4pt]
(\kappa_{\oplus 1}) & (\boldsymbol{\nu}z : G)\, \mathcal{C}[x^{pq}.\mathsf{inl}; P] \equiv x^{pq}.\mathsf{inl}; (\boldsymbol{\nu}z : G)\, \mathcal{C}[P] & \\[4pt]
(\kappa_{\oplus 2}) & (\boldsymbol{\nu}z : G)\, \mathcal{C}[x^{pq}.\mathsf{inr}; P] \equiv x^{pq}.\mathsf{inr}; (\boldsymbol{\nu}z : G)\, \mathcal{C}[P] & \\[4pt]
(\kappa_{\&}) & (\boldsymbol{\nu}z : G)\, \mathcal{C}[x^{pq}.\mathsf{case}(P, Q)] \equiv x^{pq}.\mathsf{case}((\boldsymbol{\nu}z : G)\, \mathcal{C}[P], (\boldsymbol{\nu}z : G)\, \mathcal{C}[Q]) & \\[4pt]
(\kappa_!) & (\boldsymbol{\nu}z : G)\, \mathcal{C}[!x^p(y); P] \equiv !x^p(y); (\boldsymbol{\nu}z : G)\, \mathcal{C}[P] & \\[4pt]
(\kappa_?) & (\boldsymbol{\nu}z : G)\, \mathcal{C}[?x^p(y); P] \equiv ?x^p(y); (\boldsymbol{\nu}z : G)\, \mathcal{C}[P] & \\[4pt]
(\kappa_{\perp}) & (\boldsymbol{\nu}z : G)\, \mathcal{C}[\mathsf{wait}\, x^p; P] \equiv \mathsf{wait}\, x^p; (\boldsymbol{\nu}z : G)\, \mathcal{C}[P] &
\end{array}
$$

Fig. 7 MCP, structural equivalences

$(\beta_{\otimes\mathbin{\rotatebox[origin=c]{180}{\&}}})$ $\quad (\boldsymbol{\nu}x\!:\!p\!\operatorname{->}\!\tilde{q} : \langle G'\rangle;G)\Big(\prod_i x^{q_i\,p}(y);(P_i \mid Q_i) \mid \overline{x}^{\,p\tilde{q}}(y);R \mid \prod_j P_j\Big)$

$$\rightarrow \quad (\boldsymbol{\nu}y\!:\!G')\Big(\prod_i P_i \mid (\boldsymbol{\nu}x\!:\!G)\,(\prod_i Q_i \mid R \mid \prod_j P_j)\Big)$$

$(\beta_{\oplus\&1})$ $\quad (\boldsymbol{\nu}x\!:\!p\!\operatorname{->}\!\tilde{q}\!: \&(G_1,G_2))\,\big(x^{p\tilde{q}}.\mathsf{inl};P \mid \prod_i x^{q_i p}.\mathsf{case}(Q_i,R_i) \mid \prod_j P_j\big)$

$$\rightarrow \quad (\boldsymbol{\nu}x\!:\!G_1)\,\Big(P \mid \prod_i Q_i \mid \prod_j P_j\Big)$$

$(\beta_{\oplus\&2})$ $\quad (\boldsymbol{\nu}x\!:\!p\!\operatorname{->}\!\tilde{q}\!: \&(G_1,G_2))\,\big(x^{p\tilde{q}}.\mathsf{inr};P \mid \prod_i x^{q_i p}.\mathsf{case}(Q_i,R_i) \mid \prod_j P_j\big)$

$$\rightarrow \quad (\boldsymbol{\nu}x\!:\!G_2)\,\Big(P \mid \prod_i R_i \mid \prod_j P_j\Big)$$

$(\beta_{!?})$ $\quad (\boldsymbol{\nu}x\!:\!?p\operatorname{->}!\tilde{q} : \langle G\rangle)\,\Big(?x^p(y);P \mid \prod_i !x^{q_i}(y);Q_i\Big) \;\rightarrow\; (\boldsymbol{\nu}y\!:\!G)\,\Big(P \mid \prod_i Q_i\Big)$

$(\beta_{!W})$ $\quad (\boldsymbol{\nu}x\!:\!?p\operatorname{->}!\tilde{q} : \langle G\rangle)\,\Big(\prod_i !x^{q_i}(y);Q_i \mid P\Big) \;\rightarrow\; P \qquad \text{if } x \notin \mathsf{fn}(P)$

$(\beta_{!C})$ $\quad (\boldsymbol{\nu}x\!:\!?p\operatorname{->}!\tilde{q} : \langle G\rangle)\,\Big(\prod_i !x^{q_i}(w);Q_i \mid P[x/y][x/z]\Big)$

$$\rightarrow \quad (\boldsymbol{\nu}y\!:\!?p\operatorname{->}!\tilde{q} : \langle G\rangle)\,\left(\begin{array}{l}\prod_i !y^{q_i}(w);Q_i \mid \\ (\boldsymbol{\nu}z\!:\!?p\operatorname{->}!\tilde{q} : \langle G\rangle)\,(\prod_i !z^{q_i}(w);Q_i \mid P)\end{array}\right)$$

$(\beta_{1\perp})$ $\quad (\boldsymbol{\nu}x\!:\!\mathsf{end}^{p\tilde{q}})\,\big(\mathsf{wait}\,x^p;P \mid \prod_i \mathsf{close}\,x^{q_i}\big) \;\rightarrow\; P$

(β_{+}) $\quad (\boldsymbol{\nu}x\!:\!G)\,\Big((P_1 + P_2) \mid \prod_i Q_i\Big) \;\rightarrow\; (\boldsymbol{\nu}x\!:\!G)\,\Big(P_j \mid \prod_i Q_i\Big) \qquad j \in \{1,2\}$

Fig. 8 MCP, cut reductions

5.2 Process reductions as MCut reductions

As for equivalences, we use our proof theory to derive reductions for processes, given in Fig. 8. The full proof derivations of such reductions are given in the Appendix. In the reduction $\beta_{\otimes\mathbin{\rotatebox[origin=c]{180}{\&}}}$, the output from role p to roles $\tilde{q}$ on session x is matched with the inputs at such roles, creating a new session y, following the global type of x. Reductions $\beta_{\oplus\&1}$ and $\beta_{\oplus\&2}$ capture the left and right multicast selection of a branching, respectively. In $\beta_{!?}$, a set of services with a single client is reduced to the composition of the bodies of such services with that of the client; the type $?p \operatorname{->} !\tilde{q} : \langle G\rangle$ of x is correspondingly reduced to G. Reduction $\beta_{!W}$ garbage collects a set of unused services. In $\beta_{!C}$, instead, a set of services is replicated to handle multiple clients. Finally, reduction $\beta_{1\perp}$ terminates a session x.

5.3 Properties

In the remainder, we abuse the notation $P \rightarrow P'$ to refer to process reductions closed up to our structural equivalence $\equiv$, as in standard process calculi. We restrict $P \rightarrow P'$ to be a top-level reduction, i.e., we do not allow reductions of sub-terms in P. This does not introduce any loss of generality, as in [28].

Processes and types Since both equivalences and reductions are derived from judgement-preserving proof transformations, we immediately obtain the following two properties:

Theorem 2 (Subject congruence) *$P \vdash \Delta$ and $P \equiv Q$ imply that $Q \vdash \Delta$.*

Theorem 3 (Subject reduction) *$P \vdash \Delta$ and $P \to Q$ imply that $Q \vdash \Delta$.*

In Fig. 8, global type annotations should not be mistaken for a requirement of our reductions; they are rather a guarantee given by our proof theory: if a process is reducible, then its sessions are surely typed with the respective global types reported in the rule. We use this property to reconstruct the result of session fidelity from multiparty session types [17]. In the following, $\mathsf{gt}(P)$ denotes the multiset of global types used in the restrictions inside P.

Theorem 4 (Session fidelity) *$P \vdash \Delta$ and $P \to P'$ imply that either $\mathsf{gt}(P) \leadsto \mathsf{gt}(P')$ or $\mathsf{gt}(P) \simeq^{\mathsf{g}} \mathsf{gt}(P')$.*

Proof (*Sketch*) First, we observe that we can disregard structural equivalences ($\equiv$) without any loss of generality, because $\equiv$ does not change the global types in P. We now proceed by cases on the reduction applied to P, from Fig. 8. For all such cases, we observe that the global types involved in the reduction are transformed according to the rules for the semantics of global types. □

Deadlock freedom Processes in MCP are guaranteed to be deadlock free. We use the standard methodology from [6,28]. First, we prove that the MCut rule in MCP is admissible:

Theorem 5 (MCut Admissibility) *$P_i \vdash \Gamma_i, x^{p_i} : A_i$, for $i \in [1, n]$, and $G \vDash \{p_i : A_i\}_i$ imply that there exists Q such that $Q \vdash \{\Gamma_i\}_i$.*

Proof By induction on the sizes of the proofs for $P_i \vdash \Delta_i, x^{p_i} : A_i$ and the formulae A_i. If a reduction from Fig. 8 is applicable, then we apply it. For all such reductions, we can observe that the size of the proof and/or the formulae decrease in the right-hand side, and therefore the thesis follows by induction hypothesis. Otherwise, we can apply one of the commuting conversions from Fig. 7. In this case, the proof gets smaller while the formulae stay the same.

Our case coverage is complete, because when a commuting conversion cannot be applied we can always apply a reduction. In fact, commuting conversions cannot be applied only if all proofs for P_i end with an application of a rule with principal variable x. But if that is the case, then by coherence we have that there must be at least two proofs for P_k and P_j that have compatible types and can be reduced. □

The admissibility of MCut gives us a methodology for removing cuts from a proof, corresponding to executing communications in a process until all restrictions are eliminated. However, the indiscriminate application of cut reductions inside of proofs allows for executing communications under input prefixes. This is not in line with the standard operational formulation of process calculi, where this kind of reductions are usually not allowed. Therefore, it is also useful to prove that all restrictions that appear at the top-level can be eliminated without reducing prefixed sub-terms. Below, we say that P is a restriction if it is of the form $(\nu x : G)(\prod_i P_i)$, and we write $\to^+$ for one or more applications of $\to$.

Corollary 1 (Deadlock freedom) *$P \vdash \Delta$ and P is a restriction imply $P \to^+ Q$ for some Q that is not a restriction.*

Proof The proof follows the same structure as that presented in [28] for the calculus CP, only generalised from the Cut rule in Classical Linear Logic to rule MCut in MCP.

Since P is a restriction, the last applied rule in the proof of P is MCut. We now proceed by cases on the last applied rules of the premises of such MCut. If one of the premises is

itself an MCut, we recursively eliminate it. Otherwise, either: all premises are logical rules that act on the restriction variable, thus we can apply a reduction from Fig. 8; or, at least one premise is a logical rule that acts on a variable other than the restriction variable, thus we can apply a commuting conversion from Fig. 7. □

6 The 2-buyer protocol example

We now formalise the 2-buyer protocol from Sect. 2 and expand it further.

Processes and types Roles $\mathsf{B1}$, $\mathsf{B2}$ and S are implemented as the processes:

$$\overline{x}^{\mathsf{B1\,S}}(title);\ \mathsf{wait}\,title^{\mathsf{B1}};\ x^{\mathsf{B1\,S}}(quote);\left(\begin{matrix}\mathsf{close}\,quote^{\mathsf{B1}} & \mathsf{|}\\ \overline{x}^{\mathsf{B1\,B2}}(contrib);\ \mathsf{wait}\,contrib^{\mathsf{B1}};\ \mathsf{wait}\,x^{\mathsf{B1}};\ \mathsf{close}\,z^{\mathsf{Z}}\end{matrix}\right)$$

$$x^{\mathsf{B2\,S}}(quote);\ \left(\mathsf{close}\,quote^{\mathsf{B2}}\ \mathsf{|}\ x^{\mathsf{B2\,B1}}(contrib);\ \left(\mathsf{close}\,contrib^{\mathsf{B2}}\ \mathsf{|}\ P_{B2}\right)\right)$$

$$x^{\mathsf{S\,B1}}(title);\ \left(\mathsf{close}\,title^{\mathsf{S}}\ \mathsf{|}\ \overline{x}^{\ \mathsf{S\,B1}}(quote);\ \mathsf{wait}\,quote^{\mathsf{S}};\ \overline{x}^{\ \mathsf{S\,B2}}(quote);\ \mathsf{wait}\,quote^{\mathsf{S}};\ P_S)\right)$$

The first process is the first buyer Buyer1. In the second process, the second buyer Buyer2, subterm P_{B2} implements the choice of whether to accept or reject the purchase:

$$\left(x^{\mathsf{B2\,S}}.\mathsf{inl};\ \overline{x}^{\ \mathsf{B2\,S}}(addr);\ \mathsf{wait}\,addr^{\mathsf{S}};\ \mathsf{close}\,x^{\mathsf{B2}}\right)\ +\ \left(x^{\mathsf{B2\,S}}.\mathsf{inr};\ \mathsf{close}\,x^{\mathsf{B2}}\right)$$

Finally, in the third process, the implementation of the seller, P_S is the process:

$$x^{\mathsf{S\,B2}}.\mathsf{case}\left(x^{\mathsf{S\,B2}}(addr);\ \left(\mathsf{close}\,addr^{\mathsf{S}}\ \mathsf{|}\ \mathsf{close}\,x^{\mathsf{S}}\right),\quad \mathsf{close}\,x^{\mathsf{S}}\right)$$

At the level of types, the local types in Example 1 from Sect. 3 can be used to type the three processes above: Buyer1 $\vdash x^{\mathsf{B1}}: A, z^{\mathsf{Z}}: 1$, Buyer2 $\vdash x^{\mathsf{B2}}: B$ and Seller $\vdash x^{\mathsf{S}}: C$. If we apply our new cut rule, we obtain $(\nu x : G)$ (Buyer1 $\mathsf{|}$ Buyer2 $\mathsf{|}$ Seller) $\vdash z^{\mathsf{Z}}: 1$ where the global type G, corresponding to equation (1) in Sect. 2, is such that $G \models \mathsf{B1}: A, \mathsf{B2}: B$, $\mathsf{S}: C$.

Nested multiparty sessions We can extend the example above by implementing the global type (4) in Sect. 2, where the first buyer creates a sub-session with the seller if the second buyer decides not to contribute to the purchase. Below, we give an excerpt of the new seller:

$$\begin{matrix}...\overline{x}^{\mathsf{S\,B1,B2}}(quote);\ \mathsf{wait}\,quote^{\mathsf{S}};\\ x^{\mathsf{S\,B2}}.\mathsf{case}\left(...,x^{\mathsf{S\,B1}}(y);\left(\begin{matrix}P_{\mathsf{sub}} & \mathsf{|}\\ x^{\mathsf{S\,B1}}(why);\left(\mathsf{close}\,why^{\mathsf{S}}\,\mathsf{|}\,x^{\mathsf{S\,B2}}(why);...\right)\end{matrix}\right)\right)\end{matrix}$$

where $P_{\mathsf{sub}} = y^{\mathsf{S\,B1}}.\mathsf{case}\left(\begin{matrix}y^{\mathsf{S\,B1}}(addr);\ (\mathsf{close}\,addr^{\mathsf{S}}\ \mathsf{|}\ \mathsf{close}\,y^{\mathsf{S}}),\\ y^{\mathsf{S\,B1}}(why);\ (\mathsf{close}\,why^{\mathsf{S}}\ \mathsf{|}\ \mathsf{close}\,y^{\mathsf{S}})\end{matrix}\right)$. Hence, the type of channel x, from the seller's viewpoint, becomes:[1]

$$1\otimes^{\mathsf{B1}}\perp\mathbin{\rotatebox[origin=c]{180}{\&}}^{\mathsf{B1,B2}}\left((1\otimes^{\mathsf{B2}}1)\ \&^{\mathsf{B2}}\ \left(((1\otimes^{\mathsf{B1}}1)\ \&^{\mathsf{B1}}\ (1\otimes^{\mathsf{B1}}1))\otimes^{\mathsf{B1}}1\otimes^{\mathsf{B1}}1\otimes^{\mathsf{B2}}1\right)\right)$$

We can then use coherence to infer the global type (4) in Sect. 2.

---

[1] $1\otimes B21$ or $1\otimes B2\perp$?

$$\frac{\vdash \Gamma, A \quad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B} \otimes \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \otimes B} \,\otimes$$

$$\frac{\vdash \Gamma, A \quad \vdash \Delta, A^{\perp}}{\vdash \Gamma, \Delta} \;\; \text{Cut}$$

$$\frac{\vdash \Gamma, A}{\vdash \Gamma, A \oplus B} \oplus_1 \qquad \frac{\vdash \Gamma, B}{\vdash \Gamma, A \oplus B} \oplus_2 \qquad \frac{\vdash \Gamma, A \quad \vdash \Gamma, B}{\vdash \Gamma, A \& B} \;\&$$

$$\frac{}{\vdash 1} \; 1 \qquad \frac{\vdash \Gamma}{\vdash \Gamma, \perp} \; \perp \qquad \frac{\vdash \Gamma}{\vdash \Gamma, ?A} \; \text{Weaken}$$

$$\frac{\vdash ?\Gamma, A}{\vdash ?\Gamma, !A} \; ! \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, ?A} \; ? \qquad \frac{\vdash \Gamma, ?A, ?A}{\vdash \Gamma, ?A} \; \text{Contract}$$

Fig. 9 Classical Linear Logic (CLL)

Services We extend the example to support multiple clients on a replicated session a:

$$(\nu a :?\mathsf{B1} \rightarrow !\mathsf{B2}, \mathsf{S} : \langle G \rangle) \, (\text{ Buyers } | \; !a^{\mathsf{B2}}(x); \text{Buyer2} \; | \; !a^{\mathsf{S}}(x); \text{Seller })$$

where Buyers is the process: $(\nu z : \mathsf{end}) \, (?a^{\mathsf{B1}}(x); \text{Buyer1} \; | \; ?a^{\mathsf{B1}}(x); \text{Buyer1}')$. Process Buyer1$'$ initially behaves as Buyer1, but we replaced the term $\mathsf{close}\, z^{\mathsf{Z}}$ with the term $\mathsf{wait}\, z^{\mathsf{Z}}; \mathsf{close}\, w^{\mathsf{W}}$. By applying $\beta_{!C}$ once, $\beta_{!?}$ twice, and commuting conversions, the process above can be reduced to the parallel composition of two sessions that follow the 2-buyer protocol:

$$(\nu x : G) \, \Big(\text{Buyer2} \; | \; \text{Seller} \; | \; (\nu z : \mathsf{end}) \, \big(\text{Buyer1} \; | \; (\nu x : G) \, (\text{Buyer2} \; | \; \text{Seller} \; | \; \text{Buyer1'}) \big) \Big)$$

7 Relation to linear logic

In this section, we elaborate on the relationship between MCP and Classical Linear Logic (CLL). For the reader's convenience, the rules defining CLL are reported in Fig. 9. With a slight abuse of notation, we consider CLL without the axiom for initial sequents $\vdash A, A^{\perp}$ and the rule for the additive unit $\top$. We did not consider these rules in the development of MCP, since they do not contribute to our main aim of capturing multiparty session types but rather are interesting orthogonal extensions that we leave for future work. For example, adding the axiom has been shown to be useful in capturing polymorphism [28]. In the rest of this section, we also do not consider rule $+$ in MCP since it does not add any expressivity to the underlying proof theory.

The rules of MCP and CLL look very similar. Thus, it is natural to ask whether there is any difference in the judgements that can be derived in the two systems. Interestingly, we find out that the sets of derivable judgements in the two systems are identical. Recall the encoding $\llbracket \cdot \rrbracket$ from Sect. 3, which removes annotations from types in MCP. We extend it to environments Γ in MCP by adding the straightforward rules:

$$\llbracket \cdot \rrbracket = \cdot \qquad \llbracket \Gamma, x^p : A \rrbracket = \llbracket \Gamma \rrbracket, \llbracket A \rrbracket$$

Let us write $\vdash \Gamma$ in MCP whenever $P \vdash \Gamma$ in MCP for some P. Then, we can prove the following result.

Theorem 6 (Derivable Judgements in MCP and CLL) $\vdash \Gamma$ *in MCP if and only if* $\vdash [\![\Gamma]\!]$ *in CLL.*

Proof Observe that removing proof terms in MCP yields a pure logic that differs from CLL only for two aspects. Firstly, rule MCut in MCP is different from rule Cut in CLL. Secondly, propositions in MCP are annotated with roles and the channels they type. However, we know that rule MCut is admissible in MCP (Theorem 5), just like rule Cut is admissible in CLL [15]. Therefore, when evaluating the expressivity of the two systems wrt the derivability of judgements, we can limit our comparison to the cut-free fragments of MCP and CLL without loss of generality. The only difference between these two fragments is that MCP propositions are annotated with channels and roles. But such annotations are used only by rule MCut and can be freely chosen in all other rules. As a consequence, cut-free MCP is completely equivalent to the system with the same rules but without annotations; CLL is that system.   □

Our proof of Theorem 6 relies on the fact that rule MCut in MCP and rule Cut in CLL do not add any new provable judgements to their respective systems. We now move from judgements to proofs. What is the difference between the sets of proofs that can be respectively constructed in MCP and CLL? Since coherence coincides with duality (Proposition 4) in the case of sessions with two participants, it is straightforward to show that an application of Cut in CLL corresponds to an application of MCut in MCP. Formally, let us write $\mathcal{P} \vdash_2 \Gamma$ for a judgement derived in MCP with proof $\mathcal{P}$ by using only "binary" applications of MCut, i.e., where the number of processes composed at each application of MCut is exactly two. We write $[\![\mathcal{P}]\!]$ for the encoding of $\mathcal{P}$ in MCP into a proof in CLL, defined as an homomorphism where the only relevant change is that each application of a binary MCut is replaced with an equivalent application of rule Cut, as follows (we omit the proof terms):

$$\left[\!\!\left[\; \frac{\mathcal{Q} \vdash \Gamma_1, x^{p_1} : A_1 \quad \mathcal{R} \vdash \Gamma_2, x^{p_2} : A_2 \quad \vDash p_1 : A_1, p_2 : A_2}{\vdash \Gamma_1, \Gamma_2} \; \text{MCut} \;\right]\!\!\right]$$

$$=$$

$$\frac{[\![\mathcal{Q}]\!] \vdash [\![\Gamma_1, x^{p_1} : A_1]\!] \quad [\![\mathcal{R}]\!] \vdash [\![\Gamma_2, x^{p_2} : A_2]\!]}{\vdash [\![\Gamma_1, \Gamma_2]\!]} \; \text{Cut}$$

We can now prove:

Theorem 7 *(Binary MCP and CLL)* $\mathcal{P} \vdash_2 \Gamma$ *in MCP if and only if* $[\![\mathcal{P}]\!] \vdash [\![\Gamma]\!]$ *in CLL.*

Proof By induction on the construction of $[\![\mathcal{P}]\!]$. The interesting case is that of MCut. The thesis follows from Proposition 4.   □

Theorem 7 cannot be generalised to all of MCP, since CLL cannot compose more than two proofs at the same time as done in our rule MCut. Hence, MCut must be somehow simulated using a different proof structure. An interesting line of work in this direction is the notion of "medium processes" studied in [4]. Given some processes that have compatible local types for a multiparty session, as composed in our rule MCut, a medium process corresponds to a proof in (intuitionistic) linear logic that can be composed with such processes using the standard Cut rule. This medium process is synthesised by the original global type used to type the processes and acts as a middleware: all communications over the session are centralised on the medium, which distributes messages to the processes by following the original global

type. This approach adds a layer of indirection (processes do not communicate directly, but through the medium) that is not present in the original theory of multiparty session types, and is also not necessary in MCP. However, it points to an interesting relationship between global types and the class of proofs in linear logic that correspond to medium processes (P. Wadler, personal communication, 2015).

8 Related work and discussion

Curry–Howard correspondences for session types. The works closest to ours are the Curry–Howard correspondences between binary session types and linear logic [6,28]. We extended this line of work considerably by introducing multiparty sessions, which required generalising the notion of type compatibility in linear logic to address multiple types (coherence). Coherence reconstructs the standard relationship between the global and local views found in multiparty session types. We then used coherence to develop a new proof theory that conservatively extends linear logic to capture multiparty interactions (all derivable judgements in linear logic are derivable also in our framework, and vice versa). Furthermore, our work provides, for the first time, a notion of session fidelity in the context of a Curry–Howard correspondence between linear logic and session types (Sect. 5, Theorem 4). In this work we have not treated polymorphism and existential/universal quantification, which we believe can be naturally added to MCP following the lines presented in [5,28] for binary sessions.

Our work inverts the interpretation of $\otimes$ as output and $\otimes$ as input given in [3]. This makes our process terms in line with previous developments of multiparty session types, where communications go from one sender to many receivers [12]. Using the standard interpretation would yield a join mechanism where multiple senders synchronise with a single receiver. Formally, in the standard interpretation of $\otimes$ as output and $\otimes$ as input, the rules for $\otimes$ and $\otimes$ in MCP and $\otimes\otimes$ in coherence would be as follows:

$$\frac{P \vdash \Gamma, y^p : A \quad Q \vdash \Delta, x^p : B}{\overline{x}^{pq}(y); (P \mid Q) \vdash \Gamma, \Delta, x^p : A \otimes^q B} \otimes \qquad \frac{P \vdash \Gamma, y^p : A, x^p : B}{x^{p\tilde{q}}(y); P \vdash \Gamma, x^p : A \otimes^{\tilde{q}} B} \otimes$$

$$\frac{G \vDash \Theta, \ p : B, \ \{q_i : D_i\}_i \quad G' \vDash \ p : A, \ \{q_i : C_i\}_i}{\tilde{q} \rightarrow p : \langle G' \rangle; G \ \vDash \ \Theta, \ p : A \otimes^{\tilde{q}} B, \ \{q_i : C_i \otimes^p D_i\}_i} \otimes\otimes$$

Note that there would be no need to re-prove our results, since the proof theory would not change.

The standard cut rule in CLL forces the graph of connections among processes to be a tree [1], a known sufficient condition for deadlock-freedom in session types [7]. A multi-cut rule is proposed in [1] to allow two processes to share multiple channels. This enables reasoning on networks with cyclic inter-connections, but breaks the deadlock-freedom property guaranteed by linear logic, since duality is no longer a sufficient condition when multiple resources are involved (also noted in [28]). For the first time, MCP processes can have cyclic inter-connections (e.g., our example in Sect. 2), but they are still guaranteed to be deadlock-free. The key twist is to use coherence as a principle to check that the inter-connections are safely resolved by communications. This suggests that coherence may be useful also in other settings related to linear logic, for reasoning about the sharing of resources among multiple entities (in our case, sessions). We leave this investigation as interesting future work.

Multiparty session types (MPSTs) Our work concisely unifies many of the ideas found in separate developments of multiparty session types. Our global types with multicasting are inspired from [12], to which we added nested and replicated types; both additions arise naturally from our proof theory. Our nesting of global types can be seen as a logical reconstruction of (a simplification of) those originally presented in [13], while repetitions in global types reconstruct the concept presented in [10].

Our proof system for coherence is inspired by the notion of well-formedness found in MPSTs [12,17], in the context of synchronous communications [2]. Since coherence is a proof system, projection and extraction are derived from proof equivalences, rather than being defined separately as in [17,19]. A benefit is that our projection and extraction are guaranteed to be correct by construction, whereas in previous works they have to be proven correct separately wrt the auxiliary notion of well-formedness.

In [12], MPSTs are combined with an ordering on session names to guarantee deadlock-freedom. Our deadlock-freedom result, instead, is based on the structure of our proofs. In some cases, our technique is more precise; for example, consider the deadlock-free system:

$$?a^p(x_a); \ ?b^r(x_b); \ \overline{x_a}^{\,pq}(w_1); \ x_b^{\,rs}(w_2); \ (\,\overline{x_a}^{\,pt}(w_3); \ P_1 \mid P_2\,)$$
$$!a^q(x_a); \ x_a^{\,qp}(w_1); \ (P_3 \mid P_4) \quad !a^t(x_a); \ x_a^{\,tp}(w_3); \ (P_5 \mid P_6) \quad !b^s(x_b); \ \overline{x_b}^{\,sr}(w_2); \ P_7$$

If we compose these processes in parallel, restricting sessions a and b accordingly, we obtain a typable MCP process. Instead, the system in [12] rejects it, since the actions performed by the first process create a cycle between the names x_a and x_b. In [23], the approach in [12] is refined to type processes such as the one above by ordering the I/O actions of each session.

We conjecture that MCP can be used to naturally extend the work in [9], where linear logic is used to type choreography programs, obtaining a Curry–Howard correspondence for the calculus of compositional choreographies typed with multiparty session types [22].

Coherence Coherence can be generalised, e.g., in Fig. 2: (1) rule !? could allow for more than one client; (2) similarly, rule 1⊥ could be relaxed to allow for more than one ⊥ type; (3) rule ⊗⅋ could allow the involved participants to play different roles in the nested session they create, as in [13] (adding such roles as an extra annotation to each type respectively). We leave these extensions as interesting future work. Point (2) influences greatly the complexity of the cut admissibility proof for MCP (Theorem 5), because it would imply that the cut reduction of a terminated session could lead to having more than one process in the reductum (all the processes typed with ⊥), whereas now we have only one. This means that we would have to type a parallel composition of processes without restriction, requiring to extend our framework in the fashion of the logic presented in [9]. While extending the proof theory of MCP would be easy, (extending coherence to allow for missing participants to be added later, as in [22]), it would also cause an explosion in the number of cases to consider in the proof [9]. As future work, we will investigate how our rule MCut and the notion of coherence can affect the mapping from the functional language GV [21,28].

In [11], a proof system similar to the multiplicative-additive fragment without channel passing of our coherence is embedded in the calculus of constructions. Differently from our approach, no correspondence between global types and proofs is provided; hence, extraction does not follow automatically from the theory (and is not presented).

Appendix: Proof derivations for β-reductions

In this section, we present the full proof derivations for the process reductions in MCP given in Fig. 8. While our structural equivalences are straightforward (they are similar to those in [27], extended to parallel contexts), our process reductions are quite new because they introduce multiparty synchronisations.

To improve the readability of our proof transformations, we adopt a short-hand notation for MCP proof trees, similar to that in [6]. Calligraphic letters $\mathcal{D}$, $\mathcal{E}$, ... denote proof derivations in either MCP or coherence (proofs, for short). We use "=" to define proofs, ":" to state that a proof ends with a given judgement, and $(\mathcal{D}_i)_i$ to denote a list of proofs $\mathcal{D}_1, \ldots, \mathcal{D}_n$. We also write a proof name, e.g., $\mathcal{D}$, above a judgement to denote that $\mathcal{D}$ is the proof used to derive that judgement. We use the notation $\mathsf{MCut}(\mathcal{D}_1, \ldots, \mathcal{D}_n)(\mathcal{G})$ to denote a proof that ends with an application of rule MCut, where the $\mathcal{D}_i$ are the left-hand premises and $\mathcal{G}$ the coherence proof of such application respectively. In each case below, we adopt the convention that $\mathcal{L}$ denotes the left-hand side proof of a β-reduction in Fig. 8 (our starting point, or hypothesis) and $\mathcal{R}$ the right-hand side of the reduction (our goal).

Case $\beta_{\otimes \mathbin{\bindnasrepma}}$. Given

$$\mathcal{L} = \mathsf{MCut}\,((\mathcal{Q}_i)_i, \mathcal{P}, (\mathcal{D}_j)_j)\,(\mathcal{G}) \qquad P \vdash \{\Gamma_j\}_j, \Gamma, \{\Delta_i\}_i, \{\Gamma_i\}_i$$

where $P = (\nu x : p \mathrel{->} \tilde{q} : \langle G' \rangle; G)\Big(\prod_i x^{q_i\,p}(y); (P_i \mid Q_i) \mid \overline{x}^{p\tilde{q}}(y); R \mid \prod_j P_j\Big)$ and such that

$$\mathcal{G} \;=\; \dfrac{\overset{\mathcal{G}''}{G \vDash \{r_j : E_j\}_j,\, p : B,\, \{q_i : D_i\}_i} \quad \overset{\mathcal{G}'}{G' \vDash p : A,\, \{q_i : C_i\}_i}}{p \mathrel{->} \tilde{q} : \langle G' \rangle; G \; \vDash \; \{r_j : E_j\}_j,\; p : A \mathbin{\bindnasrepma}^{\tilde{q}} B,\; \{q_i : C_i \otimes^p D_i\}_i} \; {\otimes \mathbin{\bindnasrepma}}$$

$$\mathcal{Q}_i \;=\; \dfrac{\overset{\mathcal{Q}_i'}{P_i \vdash \Gamma_i,\, y^{q_i} : C_i} \quad \overset{\mathcal{Q}_i''}{Q_i \vdash \Delta_i,\, x^{q_i} : D_i}}{x^{q_i\,p}(y); (P_i \mid Q_i) \vdash \Gamma_i, \Delta_i,\, x^{q_i} : C_i \otimes^p D_i} \; {\otimes}$$

$$\mathcal{P} \;=\; \dfrac{\overset{\mathcal{P}'}{R \vdash \Gamma,\, y^p : A,\, x^p : B}}{\overline{x}^{p\tilde{q}}(y); R \vdash \Gamma,\, x^p : A \mathbin{\bindnasrepma}^{\tilde{q}} B} \; {\mathbin{\bindnasrepma}}$$
$$\mathcal{D}_j \;:\; P_j \vdash \Gamma_j,\, x^{r_j} : E_j$$

we can construct

$$\mathcal{R}' = \mathsf{MCut}\,((\mathcal{Q}_i'')_i, \mathcal{P}', (\mathcal{D}_j)_j)\,(\mathcal{G}'')$$
$$: \; (\nu x : G)\,\Big(\prod_i Q_i \mid R \mid \prod_j P_j\big)\Big) \vdash \{\Gamma_j\}_j, \Gamma, \{\Delta_i\}_i,\, y^p : A$$

and, finally:

$$\mathcal{R} = \mathsf{MCut}\,((\mathcal{Q}_i')_i, \mathcal{R}')\,(\mathcal{G}')$$
$$: \; (\nu y : G')\Big(\prod_i P_i \mid (\nu x : G)\,(\prod_i Q_i \mid R \mid \prod_j P_j)\Big) \vdash \{\Gamma_j\}_j, \Gamma, \{\Delta_i\}_i, \{\Gamma_i\}_i$$

Case $\beta_{\oplus\&1}$. Without loss of generality we only consider this reduction. The case for $\beta_{\oplus\&2}$ is similar. Given

$$\mathcal{L} = \mathsf{MCut}\,(\mathcal{D}, (\mathcal{E}_i)_i, (\mathcal{F}_j)_j)\,(\mathcal{G})$$
$$: (\nu x\colon p \to \tilde{q}\colon \&(G_1, G_2))\,\big(x^{p\tilde{q}}.\mathsf{inl};\,P \mid \textstyle\prod_i x^{q_i\,p}.\mathsf{case}(Q_i, R_i) \mid \prod_j P_j\big) \vdash \Gamma, \{\Delta_i\}_i, \{\Gamma_j\}_j$$

where

$$\mathcal{G} \;=\; \cfrac{\cfrac{\mathcal{G}_1}{G_1 \vDash \{r_j\colon E_j\}_j,\, p\colon A,\, \{q_i\colon C_i\}_i} \quad \cfrac{\mathcal{G}_2}{G_2 \vDash \{r_j\colon E_j\}_j,\, p\colon B,\, \{q_i\colon D_i\}_i}}{p \to \tilde{q}\colon \&(G_1, G_2) \vDash \{r_j\colon E_j\}_j,\, p\colon A \oplus^{\tilde{q}} B,\, \{q_i\colon C_i \&^p D_i\}_i}\ \oplus\&$$

$$\mathcal{D} \;=\; \cfrac{\cfrac{\mathcal{D}'}{P \vdash \Gamma,\, x^p\colon A}}{x^{p\tilde{q}}.\mathsf{inl};\,P \vdash \Gamma,\, x^p\colon A \oplus^{\tilde{q}} B}\ \oplus 1$$

$$\mathcal{E}_i \;=\; \cfrac{\cfrac{\mathcal{E}'_i}{Q_i \vdash \Delta_i,\, x^{q_i}\colon C_i} \quad \cfrac{\mathcal{E}''_i}{R_i \vdash \Delta_i,\, x^{q_i}\colon D_i}}{x^{q_i\,p}.\mathsf{case}(Q_i, R_i) \vdash \Delta_i,\, x^{q_i}\colon C_i \&^p D_i}\ \&$$

$$\mathcal{F}_j \;:\; P_j \vdash \Gamma_j,\, x^{q_j}\colon E_j$$

we can construct:

$$\mathcal{R} = \mathsf{MCut}\,(\mathcal{D}', (\mathcal{E}'_i)_i, (\mathcal{F}_j)_j)\,(\mathcal{G}_1)$$
$$: (\nu x\colon G_1)\,\Big(P \mid \textstyle\prod_i Q_i \mid \prod_j P_j\Big) \vdash \Gamma, \{\Delta_i\}_i, \{\Gamma_j\}_j$$

Case $\beta_{1\perp}$. Given

$$\mathcal{L} = \mathsf{MCut}\,(\mathcal{P}, (\mathcal{Q}_i)_i)\,(\mathcal{G})$$
$$: (\nu x\colon \mathsf{end}^{p\tilde{q}})\,\big(\mathsf{wait}\,x^p;\,P \mid \textstyle\prod_i \mathsf{close}\,x^{q_i}\big) \vdash \Gamma$$

where

$$\mathcal{G} \;=\; \cfrac{}{\mathsf{end}^{p\tilde{q}} \vDash p\colon \perp,\, \{q_i\colon 1\}_i}\ 1\perp$$

$$\mathcal{Q}_i \;=\; \cfrac{}{\mathsf{close}\,x^{q_i} \vdash x^{q_i}\colon 1}\ 1$$

$$\mathcal{P} \;=\; \cfrac{\cfrac{\mathcal{P}'}{P \vdash \Gamma}}{\mathsf{wait}\,x^p;\,P \vdash \Gamma,\, x^p\colon \perp}\ \perp$$

we can construct:

$$\mathcal{R} = \mathcal{P}'$$
$$: P \vdash \Gamma$$

Case $\beta_{!?}$. Given

$$\mathcal{L} = \mathsf{MCut}\,(\mathcal{P}, (\mathcal{Q}_i)_i)\,(\mathcal{G})$$
$$: (\nu x\colon{?}p \to {!}\tilde{q}\colon \langle G\rangle)\,\Big({?}x^p(y);\,P \mid \textstyle\prod_i {!}x^{q_i}(y);\,Q_i\Big) \vdash \Gamma, \{\Delta_i\}_i$$

where

$$\mathcal{G} \quad = \quad \cfrac{\cfrac{\mathcal{G}'}{G \vDash p : A, \{q_i : B_i\}_i}}{?p \rightarrow !\tilde{q} : \langle G \rangle \vDash p : ?A, \{q_i : !B_i\}_i} \; !?$$

$$\mathcal{Q}_i \quad = \quad \cfrac{\cfrac{\mathcal{Q}'_i}{Q_i \vdash ?\Delta_i, y^{q_i} : A_i}}{!x^{q_i}(y); Q_i \vdash ?\Delta_i, x^{q_i} : !A_i} \; !$$

$$\mathcal{P} \quad = \quad \cfrac{\cfrac{\mathcal{P}'}{P \vdash \Gamma, y^p : B}}{?x^p(y); P \vdash \Gamma, x^p : ?B} \; ?$$

we can construct:

$$\mathcal{R} = \mathsf{MCut}\,(\mathcal{P}', (\mathcal{Q}'_i)_i)\,(\mathcal{G}')$$
$$: \; (\boldsymbol{\nu} y : G) \left(P \; \mid \; \textstyle\prod_i Q_i \right) \vdash \Gamma, \{\Delta_i\}_i$$

Case $\beta_{!W}$. Given

$$\mathcal{L} = \mathsf{MCut}\,((\mathcal{Q}_i)_i, \mathcal{P})\,(\mathcal{G})$$
$$: \; (\boldsymbol{\nu} x : ?p \rightarrow !\tilde{q} : \langle G \rangle) \left(\textstyle\prod_i !x^{q_i}(y); Q_i \; \mid \; P \right) \vdash \Gamma, \{\Delta_i\}_i$$

where

$$\mathcal{G} \quad = \quad \cfrac{\cfrac{\mathcal{G}'}{G \vDash p : A, \{q_i : B_i\}_i}}{?p \rightarrow !\tilde{q} : \langle G \rangle \vDash p : ?A, \{q_i : !B_i\}_i} \; !?$$

$$\mathcal{Q}_i \quad = \quad \cfrac{\cfrac{\mathcal{Q}'_i}{A_i \vdash ?\Delta_i, y^{q_i} : A_i}}{!x^{q_i}(y); Q_i \vdash ?\Delta_i, x^{q_i} : !A_i} \; !$$

$$\mathcal{P} \quad = \quad \cfrac{\cfrac{\mathcal{P}'}{P \vdash \Gamma}}{P \vdash \Gamma, x^p : ?B} \; \mathsf{Weaken}$$

we can construct:

$$\mathcal{R}' = \mathcal{P}'$$
$$: \; P \vdash \Gamma$$

Finally, since all types in Δ_i are prefixed with ?, we can iteratively applying Weaken to $\mathcal{R}'$ for $\{\Delta_i\}_i$ and obtain:

$$\mathcal{R} : P \vdash \Gamma, \{\Delta_i\}_i$$

Case $\beta_{!C}$. Given

$$\mathcal{L} = \mathsf{MCut}\,((\mathcal{Q}_i)_i, \mathcal{P})\,(\mathcal{G})$$
$$: \; (\boldsymbol{\nu} x : ?p \rightarrow !\tilde{q} : \langle G \rangle) \left(\textstyle\prod_i !x^{q_i}(w); Q_i \; \mid \; P[x/y][x/z] \right) \vdash \Gamma, \{\Delta_i\}_i$$

where

$$\mathcal{G} = \dfrac{\dfrac{\mathcal{G}'}{G \vDash p : A, \{q_i : B_i\}_i}}{?p \mathbin{-\!\!>} !\tilde{q} : \langle G\rangle \vDash p : ?A, \{q_i : {!B_i}\}_i} \ {!?}$$

$$\mathcal{Q}_i = \dfrac{Q_i \vdash ?\Delta_i, w^{q_i} : B_i}{!x^{q_i}(w) ; Q_i \vdash ?\Delta_i, x^{q_i} : {!B_i}} \ {!}$$

$$\mathcal{P} = \dfrac{\dfrac{\mathcal{F}'}{P \vdash \Gamma, y^p : ?A, z^p : ?A}}{P[x/y][x/z] \vdash \Gamma, x^p : ?A} \ \text{Contract}$$

we can construct:

$$\mathcal{R}' = \mathsf{MCut}\,((\mathcal{Q}_i)_i, \mathcal{P}')\,(\mathcal{G})$$
$$: (\boldsymbol{\nu} z : ?p \mathbin{-\!\!>} !\tilde{q} : \langle G\rangle)\left(\textstyle\prod_i !z^{q_i}(w) ; Q_i \ \mid\ P\right) \vdash \Gamma, \{\Delta_i\}_i, y^p : ?A$$

Then, we can iteratively apply $\mathsf{Contract}$ to $\mathcal{R}'$ and, finally, rule MCut to obtain:

$$\mathcal{R} : (\boldsymbol{\nu} y : ?p \mathbin{-\!\!>} !\tilde{q} : \langle G\rangle)\left(\textstyle\prod_i !y^{q_i}(w) ; Q_i \ \mid\ (\boldsymbol{\nu} z : ?p \mathbin{-\!\!>} !\tilde{q} : \langle G\rangle)\left(\textstyle\prod_i !z^{q_i}(w) ; Q_i \mid P\right)\right) \vdash \Gamma, \{\Delta_i\}_i$$

Case β_+. Given

$$\mathcal{L} = \mathsf{MCut}\,(\mathcal{P}, (\mathcal{Q}_i)_i)\,(\mathcal{G})$$
$$: (\boldsymbol{\nu} x : G)\left((P_1 + P_2) \ \mid\ \textstyle\prod_i Q_i\right) \vdash \Delta, \{\Gamma_i\}_i$$

where

$$\mathcal{P} = \dfrac{\overset{\mathcal{P}_1}{P_1 \vdash \Gamma, x^p : A} \quad \overset{\mathcal{P}_2}{P_2 \vdash \Gamma, x^p : A}}{P_1 + P_2 \vdash \Delta, x^p : A} \ {+}$$

we can construct, for $i = 1$ in Fig. 8:

$$\mathcal{R} = \mathsf{MCut}\,(\mathcal{P}_1, (\mathcal{Q}_i)_i)\,(\mathcal{G})$$
$$: (\boldsymbol{\nu} x : G)\left(P_1 \ \mid\ \textstyle\prod_i Q_i\right) \vdash \Delta, \{\Gamma_i\}_i$$

Otherwise, for $i = 2$ in Fig. 8, we construct:

$$\mathcal{R} = \mathsf{MCut}\,(\mathcal{P}_2, (\mathcal{Q}_i)_i)\,(\mathcal{G})$$
$$: (\boldsymbol{\nu} x : G)\left(P_2 \ \mid\ \textstyle\prod_i Q_i\right) \vdash \Delta, \{\Gamma_i\}_i$$

References

1. Abramsky, S., Gay, S.J., Nagarajan, R.: Interaction categories and the foundations of typed concurrent programming. In: NATO ASI DPD, pp. 35–113 (1996)
2. Bejleri, A., Yoshida, N.: Synchronous multiparty session types. Electr. Notes Theor. Comput. Sci. **241**, 3–33 (2009)
3. Bellin, G., Scott, P.J.: On the pi-calculus and linear logic. Theor. Comput. Sci. **135**(1), 11–65 (1994)
4. Caires, L., Pérez, J.A.: A typeful characterization of multiparty structured conversations based on binary sessions. CoRR, abs/1407.4242 (2014)
5. Caires, L., Pérez, J.A., Pfenning, F., Toninho, B.: Behavioral polymorphism and parametricity in session-based communication. In: ESOP, pp. 330–349 (2013)
6. Caires, L., Pfenning, F.: Session types as intuitionistic linear propositions. In: CONCUR, pp. 222–236 (2010)

7. Carbone, M., Debois, S.: A graphical approach to progress for structured communication in web services. In: Proceedings of ICE'10 (2010)
8. Carbone, M., Montesi, F.: Deadlock-freedom-by-design: multiparty asynchronous global programming. In: POPL, pp. 263–274 (2013)
9. Carbone, M., Montesi, F., Schürmann, C.: Choreographies, logically. In: CONCUR, pp. 47–62 (2014)
10. Castagna, G., Dezani-Ciancaglini, M., Padovani, L.: On global types and multi-party session. LMCS, **8**(1), 1–45 (2012)
11. Ciobanu, Gabriel, Horne, Ross: Behavioural analysis of sessions using the calculus of structures. In: Proceedings of the 10th International Andrei Ershov Informatics Conference, Perspectives of System Informatics (PSI 2015), volume to appear of *LNCS*. Springer (2016)
12. Coppo, M., Dezani-Ciancaglini, M., Yoshida, N., Padovani, L.: Global progress for dynamically interleaved multiparty sessions. MSCS **760**, 1–65 (2015)
13. Demangeon, R., Honda, K.: Nested protocols in session types. In: CONCUR, pp. 272–286 (2012)
14. Deniélou, P.-M., Yoshida, N.: Multiparty compatibility in communicating automata: characterisation and synthesis of global session types. ICALP **2**, 174–186 (2013)
15. Girard, J.-Y.: Linear logic. Theor. Comput. Sci. **50**, 1–102 (1987)
16. Honda, K., Vasconcelos, V., Kubo, M.: Language primitives and type disciplines for structured communication-based programming. In: ESOP, pp. 22–138 (1998)
17. Honda, K., Yoshida, N., Carbone, M.: Multiparty asynchronous session types. In: Proceedings of POPL, vol. 43(1), pp. 273–284. ACM (2008)
18. Kouzapas, D., Yoshida, N.: Globally governed session semantics. LMCS **10** (2015)
19. Lange, J., Tuosto, E.: Synthesising choreographies from local session types. In: CONCUR, pp. 225–239 (2012)
20. Lange, J., Tuosto, E., Yoshida, N.: From communicating machines to graphical choreographies. In: POPL 2015, pp. 221–232. ACM (2015)
21. Lindley, S., Garrett M.J.: A semantics for propositions as sessions. In: ESOP, pp. 560–584 (2015)
22. Montesi, F., Yoshida, N.: Compositional choreographies. In: CONCUR, pp. 425–439 (2013)
23. Padovani, L., Vasconcelos, V.T., Vieira, H.T.: Typing liveness in multiparty communicating systems. In: COORDINATION, pp. 147–162 (2014)
24. Sangiorgi, D., Walker, D.: The π-calculus: A Theory of Mobile Processes. Cambridge University Press, Cambridge (2001)
25. Scribble Project Home Page. http://www.scribble.org
26. Vasconcelos, V.T.: Fundamentals of session types. Inf. Comput. **217**, 52–70 (2012)
27. Wadler, P.: Propositions as sessions. In: ICFP, pp. 273–286 (2012)
28. Wadler, P.: Propositions as sessions. J. Funct. Prog. **24**(2–3), 384–418 (2014)