Propositions in Linear Multirole Logic as Multiparty Session Types

Hongwei Xi and Hanwen Wu
Boston University
hwxi,hwwu@cs.bu.edu

***Abstract*—We identify multirole logic as a new form of logic and formalize linear multirole logic (LMRL) as a natural generalization of classical linear logic (CLL). Among various meta-properties established for LMRL, we obtain multi-cut elimination stating that every cut between *three (or more)* sequents can be eliminated as a generalization of a cut between only two sequents, thus extending the celebrated result of cut-elimination by Gentzen. We also present a variant of π-calculus for multi-party sessions that demonstrates a tight correspondence between process communication in this variant and multi-cut elimination in LMRL, thus extending some recent results by Caires and Pfenning (2010) and Wadler (2012), among others, along a similar line of work.**

***Index Terms*—multirole logic, linear logic, multiparty, session types, cut-elimination, π-calculus**

I. Introduction[1]

Foundational research on concurrency has long been pursued. Along one line, a tight correspondence between linear logic [1] and π-calculus [2] [3] was established [4] [5]. Along another line, the notion of session types [6] [7] [8] was introduced to ensure that communication protocols are properly followed between parties in a session. Recently, these two lines of work are unified [9] [10]. In particular, the *propositions-as-session-types* correspondence is established for *binary/dyadic* sessions, where cut-elimination on the logic side matches communication on the process-calculus side. In an attempt to account for *multiparty* sessions, Carbone et al. generalized the notion of *duality* in logic into *coherence* [11] [12], establishing a correspondence between multi-sequent cut and multiparty communication.

In this paper, we are to present an alternative approach to establishing a correspondence between multi-sequent cut and multiparty communication. Let us first take a look at the identity axiom in classical linear logic (CLL):

$$\frac{}{A \vdash A}\,\textbf{id}$$

which can also be given as follows in a one-sided formulation:

$$\frac{}{\vdash A, A^{\perp}}\,\textbf{id}$$

where $A^{\perp}$ is the dual (linear negation) of A. The *signature* of an occurence of A is defined as the parity of the number of times A has been negated [13]. Thus, the one-sided identity axiom in CLL can also be written as

$$\frac{}{\vdash [A]_0, [A]_1}\,\textbf{id}$$

where $[A]_0, [A]_1$ are interpretations of A based on signature 0 and 1, respectively.

Assumed, but not explicitly stated, is that there are only two interpretations available and one is the dual of the other. *What if we forego such an assumption and instead, allow more than two interpretations for each formula?* This move directly leads us to the discovery of **multirole logic**. For illustration, let us see as follows the identity axiom[2] in linear multirole logic (LMRL):

$$\frac{R_1 \uplus \cdots \uplus R_n = \overline{\varnothing}}{\vdash [A]_{R_1}, \ldots, [A]_{R_n}}\,\textbf{id}$$

A *role* is represented as a natural number and the concept of role somewhat corresponds to the notion of signature. An interpretation of A is not based on singleton role as one might expect; it is instead based on a set R of roles. The important side condition for the identity axiom in LMRL is that all of the involved role sets are disjoint and their union form the underlying full set of roles (that may be infinite). Note that we use $\uplus$ for disjoint union and $\overline{\varnothing}$ for the underlying full set of roles.

We can readily see that the identity axiom in LMRL degrades into the identity axiom in CLL when $\overline{\varnothing} = \{0, 1\}$ and both R_1 and R_2 are singleton sets.

The cut-rule can be generalized similarly. In CLL, it is of the following form:

$$\frac{\vdash \Gamma, [A]_0 \quad \vdash \Delta, [A]_1}{\vdash \Gamma, \Delta}\,\textbf{2-cut}$$

In LMRL, one may naturally expect that the rule for multi-cut should be of the following form:

$$\frac{R_1 \uplus \cdots \uplus R_n = \overline{\varnothing} \quad \vdash \Gamma_1, [A]_{R_1} \quad \cdots \quad \vdash \Gamma_n, [A]_{R_n}}{\vdash \Gamma_1, \ldots, \Gamma_n}$$

However, it should be soon clear that the side condition is *incorrect*. And the correct rule for multi-cut should be as follows:

$$\frac{\overline{R}_1 \uplus \cdots \uplus \overline{R}_n = \overline{\varnothing} \quad \vdash \Gamma_1, [A]_{R_1} \quad \cdots \quad \vdash \Gamma_n, [A]_{R_n}}{\vdash \Gamma_1, \ldots, \Gamma_n}\,n\textbf{-cut}$$

[1]We assume some prior knowledge about session types, π-calculus, and linear logic.

[2]Strictly speaking, the identity axiom requires that A be an atomic formula.

where $\overline{R}$ is the complement of R w.r.t. $\overline{\varnothing}$. The intuition on the side condition will be given when we present the cut-elimination proof for the n-**cut** rule.

Again, we can easily see that the n-**cut** rule of LMRL degrades naturally into the 2-**cut** rule of CLL when $\overline{\varnothing} = \{0, 1\}$ and R_1, R_2 are singleton sets. However, the correct and incorrect side conditions cannot be distinguished in CLL in that both $R_1 \uplus R_2 = \{0, 1\}$ and $\overline{R}_1 \uplus \overline{R}_2 = \{0, 1\}$ are true. Only when $\overline{\varnothing}$ contains more than two roles, the two side conditions become distinct, thus offering a deeper understanding of duality.

Following Girard [13], we say that the axiom and the cut in CLL are dual to each other. From the side conditions of **id** and n-**cut**, we learn that this is also the case in LMRL. And we can think of the dual (linear negation) of $[A]_R$ as $[A]_{\overline{R}}$ where R and $\overline{R}$ form a partition of $\overline{\varnothing}$, and $[A]_{\overline{R}}$ is the "negated" interpretation of $[A]_R$. In general, if $\overline{R}_1 \uplus \cdots \uplus \overline{R}_n = \overline{\varnothing}$ holds, we say that $[A]_{R_1}, \ldots, [A]_{R_n}$ are *coherent* in the sense of [11] [12]. Therefore coherence can be seen in multirole logic as a natural generalization of duality. Dual interpretations of a formula can be cut away in CLL. As a natural generalization, coherent interpretations of a formula can be cut away in LMRL.

There is yet more surprise. Instead of cutting away multiple interpretations of a formula completely, the following rule in LMRL tells us what an *incomplete* cut is like:

$$\frac{\overline{R}_1 \cap \overline{R}_2 = \varnothing \quad \vdash \Gamma_1, [A]_{R_1} \quad \vdash \Gamma_2, [A]_{R_2}}{\vdash \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2}} \; \textbf{2-cut-spill}$$

As one can see, the rule 2-**cut-spill** has a relaxed side condition in that the involved role sets are still disjoint but their union is no longer required to form the underlying full set. It will be clear that n-**cut** can be readily established by applying 2-**cut-spill** repeatedly.

Contributions: In CLL, with only two roles available, a lot of subtle details are hidden in plain sight. In LMRL, with multiple interpretations (based on sets of roles), these subtleties are suddenly made clear. The primary contribution of the paper lies in the identification of multirole logic as a new form of logic and the presented formalization of linear multirole logic (LMRL). We consider the formulation and proof of various meta-properties on LMRL a large part of this contribution. In particular, we formulate a cut-rule for multiple sequents in LMRL and prove its admissibility, naturally extending the celebrated result of cut-elimination by Gentzen [14] [15].

Primarily for the purpose of comparing LMRL with intuitionistic linear logic and classical linear logic, we also present a variant of π-calculus for multiparty sessions that demonstrates a tight correspondence between process communication in this variant and multi-cut elimination in LMRL, thus extending some recent results on encoding session types as propositions in linear logic [9] [10]. The focus of the paper is first and foremost on LMRL, and the presentation of πLMRL can be seen as an exercise to facilitate the understanding of LMRL.

Organization: The rest of the papers is organized as follows. We formulate LMRL in Section II, establishing various meta-properties. We primarily focus on conjunctive LMRL (LMRL$_\wedge$) while briefly mentioning disjunctive LMRL (LMRL$_\vee$) as the obvious dual of LMRL$_\wedge$. We then present in Section III a process-calculus πLMRL, which can be seen as a session-typed variant of π-calculus. The session types in πLMRL are just the formulas in LMRL, and the reduction semantics of πLMRL is directly based on cut-elimination in LMRL. In each section, we may present some clarifying comments at the end on subtle issues and closely related works specific to that section. We also point out in Section IV certain implementations of session types based on LMRL, making it possible for anyone interested to actually gain a bit of taste of the theory developed here. Lastly, we mention some related works and conclude.

II. Linear Multirole Logic

We have already presented a bit of intuition on generalizing CLL into LMRL in Section I. In this section, we are to elaborate on the intuition and give a formal presentation of LMRL.

Roles: Given a natural number N, we use R_N for the set consisting of all of the natural numbers less than N, and R_ω for the set of natural numbers. In addition, we use $\mathcal{R}$ for either R_N (for some $n \geq 2$) or R_ω, and may refer to each number in $\mathcal{R}$ as a role. Note that multirole logic is parameterized over a chosen underlying set $\mathcal{R}$ of roles, and we may use $\overline{\varnothing}$ to refer to this set $\mathcal{R}$. Given a subset R of some $\mathcal{R}$, we use $\overline{R}$ for the complement of R in $\mathcal{R}$ (assuming that this particular $\mathcal{R}$ can be readily inferred from the context). Also, we use $R_1 \uplus R_2$ for the union of two disjoint sets R_1 and R_2.

Interpreted Formulas: Given a formula A and a set R of roles, we write $[A]_R$ for an i-formula, which is an interpretation of A based on R. We may also write $[A]_r$ for an interpretation of A under a singleton set $\{r\}$, namely $[A]_{\{r\}}$. Correspondingly, logical connectives are decorated with roles as well. For instance, the interpretation of $\wedge_r$ based on R is $\wedge$-like if $r \in R$, and it is $\vee$-like otherwise. We are to formally introduce these connectives later.

It is crucial to realize that interpretations should be based on *sets* of roles rather than just individual roles due to the need of complement roles.

Conjunctive and Disjunctive Interpretation: We have developed two flavors of LMRL, a conjunctive one, and a disjunctive one. Intuitively speaking, a conjunctive multirole logic is one that for each $r \in \overline{\varnothing}$, there is a logical connective (for instance) $\wedge_r$ such that $\wedge_r$ is given a $\wedge$-like interpretation by the side playing role r and a $\vee$-like interpretation otherwise. If we think of the universal quantifier $\forall$ as an infinite form of conjunction, then what is said about $\wedge$ can be readily applied to $\forall$ as well. In fact, additive, multiplicative, and exponential connectives in linear logic [1] can all be treated in a similar manner. Evidently, a disjunctive multirole logic can be formulated *dually* by giving $\wedge_r$ a $\vee$-like interpretation if the side plays the role r and a $\wedge$-like interpretation otherwise.

In this paper, we focus on formalizing conjunctive multirole logic since the disjunctive flavor is its exact dual. Thus we omit detailed discussions about disjunctive multirole logic. While we also developed *classical* multirole logic, we solely focus on *linear* multirole logic (LMRL) in this paper, which is based on classical linear logic. We may refer to LMRL$_\wedge$ as simply LMRL if there is no risk of ambiguity.

Sequents: A sequent Γ in multirole logic is a bag (multiset) of i-formulas. Note that two identical i-formulas are allowed to appear in one sequent. We use $\cdot$ for an empty sequent and $(\Gamma, [A]_R)$ for any sequent that can be formed by inserting $[A]_R$ into Γ (at any position). The parentheses in $(\Gamma, [A]_R)$ may be dropped if there is no confusion. We use $\vdash \Gamma$ for a judgment meaning that Γ is derivable and may write $(\Gamma_1; \cdots; \Gamma_n) \Rightarrow \Gamma_c$ for an inference rule of the following form:

$$\frac{\vdash \Gamma_1 \quad \cdots \quad \vdash \Gamma_n}{\vdash \Gamma_c}$$

where $\vdash \Gamma_1, \ldots, \vdash \Gamma_n$ are the premisses of the rule and $\vdash \Gamma_c$ the conclusion.

Cut: As can be readily expected, the cut-rule (for two sequents) in (either conjunctive or disjunctive) LMRL is of the following form:

$$\frac{\vdash \Gamma_1, [A]_R \quad \vdash \Gamma_2, [A]_{\overline{R}}}{\vdash \Gamma_1, \Gamma_2}\ \textbf{2-cut}$$

The cut-rule can be interpreted as communication between two parties in distributed programming [4] [16] [5] [9] [10]. For communication between multiple parties, it is natural to seek a generalization of the cut-rule that involve more than two sequents. In conjunctive LMRL, the admissibility of the following n-**cut**$_\wedge$ can be established for each $n \geq 1$:

$$\frac{\overline{R}_1 \uplus \cdots \uplus \overline{R}_n = \overline{\varnothing} \quad \vdash \Gamma_1, [A]_{R_1} \quad \cdots \quad \vdash \Gamma_n, [A]_{R_n}}{\vdash \Gamma_1, \ldots, \Gamma_n}\ n\textbf{-cut}_\wedge$$

In disjunctive LMRL, the admissibility of the following cut-rule (n-**cut**$_\vee$) can be established for each $n \geq 1$:

$$\frac{R_1 \uplus \cdots \uplus R_n = \overline{\varnothing} \quad \vdash \Gamma_1, [A]_{R_1} \quad \cdots \quad \vdash \Gamma_n, [A]_{R_n}}{\vdash \Gamma_1, \ldots, \Gamma_n}\ n\textbf{-cut}_\vee$$

We will give explanation later on the case where $n = 1$.

A. Syntax

We use t for (first-order) terms in LMRL, which are standard. For each $r \in \mathcal{R}$, there exist logical connectives $\otimes_r$, $\&_r$, $!_r$, and $\forall_r$. The formulas in LMRL are defined as follows:

$$\text{Formulas } A, B \quad ::= \quad a \mid A \otimes_r B \mid A \&_r B \mid !_r(A) \mid \forall_r(\lambda x.A)$$

where a ranges over primitive formulas. Parentheses may be omitted if the context is clear.

The interpretation of $\otimes_r$ under roles R is $\otimes$-like as in CLL when $r \in R$, and is $⅋$-like as in CLL when $r \notin R$. Similarly, $\&_r$ under R is $\&$-like when $r \in R$, and $\oplus$-like otherwise. $!_r$ under R is !-like when $r \in R$, and ?-like otherwise. $\forall_r$ under R is $\forall$-like when $r \in R$, and $\exists$-like otherwise.

$$\frac{R_1 \uplus \cdots \uplus R_n = \overline{\varnothing}}{\vdash [a]_{R_1}, \ldots, [a]_{R_n}}\ \textbf{id}_\wedge$$

$$\frac{r \notin R \quad \vdash \Gamma, [A]_R, [B]_R}{\vdash \Gamma, [A \otimes_r B]_R}\ \otimes\textbf{-neg}$$

$$\frac{r \in R \quad \vdash \Gamma_1, [A]_R \quad \vdash \Gamma_2, [B]_R}{\vdash \Gamma_1, \Gamma_2, [A \otimes_r B]_R}\ \otimes\textbf{-pos}$$

$$\frac{r \notin R \quad \vdash \Gamma, [A]_R}{\vdash \Gamma, [A \&_r B]_R}\ \&\textbf{-neg-l} \qquad \frac{r \notin R \quad \vdash \Gamma, [B]_R}{\vdash \Gamma, [A \&_r B]_R}\ \&\textbf{-neg-r}$$

$$\frac{r \in R \quad \vdash \Gamma, [A]_R \quad \vdash \Gamma, [B]_R}{\vdash \Gamma, [A \&_r B]_R}\ \&\textbf{-pos}$$

$$\frac{r \in R \quad \vdash ?\Gamma, [A]_R}{\vdash ?\Gamma, [!_r A]_R}\ \textbf{!-pos} \qquad \frac{r \notin R \quad \vdash \Gamma}{\vdash \Gamma, [!_r A]_R}\ \textbf{!-neg-weaken}$$

$$\frac{r \notin R \quad \vdash \Gamma, [A]_R}{\vdash \Gamma, [!_r A]_R}\ \textbf{!-neg-derelict}$$

$$\frac{r \notin R \quad \vdash \Gamma, [!_r A]_R, [!_r A]_R}{\vdash \Gamma, [!_r A]_R}\ \textbf{!-neg-contract}$$

$$\frac{r \notin R \quad \vdash \Gamma, [A\{^t/_x\}]_R}{\vdash \Gamma, [\forall_r(\lambda x.A)]_R}\ \forall\textbf{-neg}$$

$$\frac{r \in R \quad x \notin \Gamma \quad \vdash \Gamma, [A]_R}{\vdash \Gamma, [\forall_r(\lambda x.A)]_R}\ \forall\textbf{-pos}$$

Fig. 1: LMRL$_\wedge$ Inference Rules

Recall that an i-formula in LMRL is of the form $[A]_R$, and a sequent Γ is a bag (multiset) of i-formulas. We may write $[?A]_R$ to mean $[!_r A]_R$ for some $r \notin R$, and $?\Gamma$ to mean that each i-formula in Γ is of the form $[?A]_R$. Given a sequent Γ and an i-formula $[A]_R$, we use $(\Gamma, [A]_R)$ for a sequent obtained from inserting $[A]_R$ into Γ (at any position). Given two sequents Γ_1 and Γ_2, we use (Γ_1, Γ_2) for a sequent obtained from merging Γ_1 with Γ_2 (in any order). The parentheses in $(\Gamma, [A]_R)$ and (Γ_1, Γ_2) may be dropped if there is no confusion.

B. LMRL$_\wedge$: Conjunctive LMRL

The inference rules for LMRL$_\wedge$ are given in Figure 1. Positive rules correspond to positive interpretations of connectives, which are given when $r \in R$ holds. Negative rules correspond to negative interpretations, which are given when $r \notin R$ holds. There are one positive rule and one negative rule for each $\otimes_r$ and one positive rule and two negative rules for each $\&_r$, and one positive rule and one negative rule for each $\forall_r$.

The principal i-formula of a rule is defined as the formula in the conclusion that has the connective of interest, e.g. the i-formula $[A \otimes_r B]_R$ in $\otimes$-**neg**. The principal i-formulas for the other rules (excluding the rule **id**$_\wedge$) should be clear as well. For the rule **id**$_\wedge$, each $[a]_{R_i}$ is referred to as a principal i-formula.

We use $|A|$ for the size of A, which is the number of connectives contained in A. We use $\mathcal{D}$ for a derivation tree and $ht(\mathcal{D})$ for the height of the derivation tree. Also, we use $\mathcal{D} :: \Gamma$ for a derivation of Γ.

Proposition 1 (Substitution). *Given a sequent Γ, a variable x and a term t, we use $\Gamma\{^t/_x\}$ for the sequent obtained from replacing each i-formula $[A]_R$ in Γ with $[A\{^t/_x\}]_R$. Assume $\mathcal{D}_1 :: \Gamma$. Then we can construct a derivation $\mathcal{D}_2$ of the sequent $\Gamma\{^t/_x\}$.*

Proof. By structural induction on $\mathcal{D}_1$. □

We may use $\mathcal{D}_1\{^t/_x\}$ for the $\mathcal{D}_2$ constructed in Proposition 1.

Lemma 1 (1-cut). *The following rule is admissible in LMRL$_\wedge$:*

$$(\Gamma, [A]_\varnothing) \Rightarrow \Gamma$$

Proof. By structural induction on the derivation of $\mathcal{D} :: (\Gamma, [A]_\varnothing)$. □

Note that Lemma 1 simply states the admissibility of the n-**cut**$_\wedge$ for $n = 1$. So it actually makes sense to have a cut involving only *one* sequent! Intuitively speaking, an empty interpretation of a formula can be eliminated.

Lemma 2 (η-expansion). *The following rule is admissible in LMRL$_\wedge$:*

$$\cdot \Rightarrow [A]_{R_1}, \dots, [A]_{R_n}$$

where $R_1 \uplus \cdots \uplus R_n = \overline{\varnothing}$ and $\cdot$ is the empty sequent.

Proof. By structural induction on A. □

The next lemma is the most crucial one in this paper. It shows that, with multiple roles in the logic, we have not only a *complete* cut as we know in CLL but also a cut with *spills* of the form $[A]_{R_1 \cap R_2}$! This is only possible in multirole logic since it allows $R_1 \cap R_2$ to be non-empty. On the contrary, in CLL, $R_1 \cap R_2$ is always empty thus the rule will degrade into traditional 2-**cut** by Lemma 1.

Lemma 3 (2-cut with spill). *Assume that $\overline{R}_1$ and $\overline{R}_2$ are disjoint. Then the following rule is admissible in LMRL$_\wedge$:*

$$(\Gamma_1, [A]_{R_1}; \Gamma_2, [A]_{R_2}) \Rightarrow \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2}$$

Given an i-formula $[A]_R$, let us use $\{[A]_R\}$ for either a singleton sequent consisting of only $[A]_R$ if A is not of the form $!_r B$ for $r \notin R$, or some repeated occurrences of $[A]_R$ if A is of the form $!_r B$ for $r \notin R$.

Due to the explicit presence of the three structural rules **!-neg-weaken**, **!-neg-derelict**, and **!-neg-contract** in LMRL$_\wedge$, we need to prove a strengthened version of Lemma 3 stating that the following rule is admissible in LMRL$_\wedge$:

$$(\Gamma_1, \{[A]_{R_1}\}; \Gamma_2, \{[A]_{R_2}\}) \Rightarrow \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2}$$

Note that the proof strategy we use is essentially adopted from the one in a proof of cut-elimination for classical linear logic (CLL) [17]. Assume $\mathcal{D}_1 :: (\Gamma_1, \{[A]_{R_1}\})$ and $\mathcal{D}_2 :: (\Gamma_2, \{[A]_{R_2}\})$. We proceed by induction on $|A|$ and $ht(\mathcal{D}_1) + ht(\mathcal{D}_2)$, lexicographically ordered. The rest of the proof is given in Appendix C, which should be readily accessible for someone familiar with a standard proof of cut-elimination (for classical logic).

Lemma 4 (2-cut). *The following rule is admissible in LMRL$_\wedge$:*

$$(\Gamma_1, [A]_R; \Gamma_2, [A]_{\overline{R}}) \Rightarrow \Gamma_1, \Gamma_2$$

Proof. Lemma 4 is just a special case of Lemma 5 where $n = 2$. □

Lemma 5 (n-cut). *Assume that $R_1, R_2, \dots, R_n$ are subsets of $\overline{\varnothing}$ for some $n \geq 1$ such that:*

$$\overline{R}_1 \uplus \overline{R}_2 \uplus \cdots \uplus \overline{R}_n = \overline{\varnothing}$$

Then the following rule is admissible in LMRL$_\wedge$:

$$(\Gamma_1, [A]_{R_1}; \Gamma_2, [A]_{R_2}; \cdots ; \Gamma_n, [A]_{R_n}) \Rightarrow \Gamma_1, \Gamma_2, \dots, \Gamma_n$$

One can prove the lemma directly by the standard technique similar to the proof for Lemma 3. But here, we present an alternative proof using Lemma 3.

Proof. Assume $\mathcal{D}_i :: (\Gamma_i, [A]_{R_i})$ for $1 \leq i \leq n$. The proof proceeds by induction on n. The base case (where $n = 1$) is simply covered by Lemma 1. For $n \geq 2$, we can apply Lemma 3 to $\mathcal{D}_1$ and $\mathcal{D}_2$ to obtain a derivation $\mathcal{D}_{12} :: (\Gamma_1, \Gamma_2, [A]_{R_{12}})$ where $R_{12} = R_1 \cap R_2$. Clearly, $\overline{R}_{12} = \overline{R}_1 \uplus \overline{R}_2$ holds, and we can invoke induction hypothesis on $\mathcal{D}_{12}$ and the remaining derivations $\mathcal{D}_i$ for $3 \leq i \leq n$ to obtain a derivation of $\Gamma_1, \dots, \Gamma_n$. □

In Section I, we pointed out an incorrect guess for the side condition of n-**cut** that states $R_1 \uplus \cdots \uplus R_n = \overline{\varnothing}$. With such a side condition, one can readily see that the inductive proof of Lemma 5 no longer works. This can be seen from another angle. Suppose we are to prove Lemma 5 directly (instead of making use of Lemma 3). Let us take the principal cut of $\otimes_r$ as an example. We clearly need one of the last applied rules to be $\otimes$-**neg** (with one premise) and all of the others to be $\otimes$-**pos** (with two premises) to allow induction to proceed. Again we can see that this is only possible if r belongs to all but one R's, namely $\overline{R}_1 \uplus \cdots \uplus \overline{R}_n = \overline{\varnothing}$ and $r \notin R_k$ for some $1 \leq k \leq n$.

Lemma 6 (Splitting). *The following rule is admissible in LMRL$_\wedge$:*

$$(\Gamma, [A]_{R_1 \uplus R_2}) \Rightarrow \Gamma, [A]_{R_1}, [A]_{R_2}$$

Proof. Assume $\mathcal{D}_1 :: (\Gamma, [A]_{R_1 \uplus R_2})$. By Lemma 2, we have a derivation:

$$\mathcal{D}_2 :: ([A]_{\overline{R}_1 \cap \overline{R}_2}, [A]_{R_1}, [A]_{R_2})$$

By applying Lemma 4 to $\mathcal{D}_1$ and $\mathcal{D}_2$, we obtain a derivation of $(\Gamma, [A]_{R_1}, [A]_{R_2})$. □

C. LMRL$_\vee$ as the Dual of LMRL$_\wedge$: Disjunctive LMRL

For a bit of completeness, we briefly mention disjunctive LMRL (LMRL$_\vee$), which is the exact *dual* of LMRL$_\wedge$. The inference rules for LMRL$_\vee$ are listed in Figure 9 in Appendix A, which can simply be obtained by replacing each set (of roles) in Figure 1 with its complement. The various lemmas established for LMRL$_\wedge$ have their counterparts in LMRL$_\vee$, which are omitted for brevity.

D. Clarifying Discussions

CLL as a Special Case of LMRL: When there are only two roles, namely $\overline{\varnothing} = \{0, 1\}$, LMRL degrades into CLL.

The logical connectives $\otimes_0$ and $\otimes_1$ in LMRL$_\wedge$ correspond to $\otimes$ and $\parr$ in CLL, respectively. $\&_0$ and $\&_1$ correspond to $\&$ and $\oplus$, respectively. The quantifiers $\forall_0$ and $\forall_1$ correspond to $\forall$ and $\exists$, respectively. And the modality operators $!_0$ and $!_1$ correspond to ! and ?, respectively.

Let $[\![A]\!]$ be the translation of A such that each connective/quantifier/modality operator in LMRL is translated to a corresponding one in CLL based on the above rules. Then each sequent Γ in LMRL can be translated into a sequent $\Delta_1 \vdash \Delta_0$ in CLL as follows such that Γ is derivable in LMRL if and only if $\Delta_1 \vdash \Delta_0$ is derivable in CLL:

- If $[A]_{\overline{\varnothing}}$ is in Γ, then **1** goes into Δ_0. Note that **1** is the unit of $\otimes$ in CLL.
- If $[A]_{\varnothing}$ is in Γ, then **1** goes into Δ_1
- If $[A]_{\{0\}}$ is in Γ, then $[\![A]\!]$ goes into Δ_0
- If $[A]_{\{1\}}$ is in Γ, then $[\![A]\!]$ goes into Δ_1

Therefore, CLL is just a special case of LMRL$_\wedge$[3]. As mentioned in Section I, the availability of multiple roles and interpretations based on sets of roles reveals hidden subtleties and gives rise to rules such as 2-**cut-spill**. Such a correspondence clearly demonstrated that LMRL is a genuine generalization of classical linear logic.

Duality (Linear Negation) and Coherence: In CLL, linear negation is introduced as follows:

- Each atomic formula is given in two forms, positive A, and negative $A^\perp$.
- Linear negation for composed formulas is defined by *de Morgan* laws.

As we already mentioned in Section I, we forego the assumption that formulas have only two forms and generalize that to allow multiple interpretations. Thus, duality in the sense of CLL is gone, and instead, we use the relationship of roles to describe the relationship of different interpretations.

For instance, *dual/negation* (in the sense of LMRL) of $[A]_R$ is just the "negated" interpretation $[A]_{\overline{R}}$ where $\overline{R}$ is the complement of R. And *coherence* (in the sense of [11] [12]) is expressed as $\overline{R}_1 \uplus \cdots \uplus \overline{R}_n = \overline{\varnothing}$ for formulas $[A]_{R_1}, \ldots, [A]_{R_n}$. For example, if we write $\otimes$ for $\otimes_r$ for some $r \in R$, and write $\parr$ for $\otimes_r$ for some $r \notin R$, then we can say that $[A \parr B]_R$ is dual to $[A \otimes B]_{\overline{R}}$. Or we can say that $[A \parr B]_{R_1}$ is coherent with $[A \otimes B]_{R_2}$, $[A \otimes B]_{R_3}$ when $\overline{R}_1 \uplus \overline{R}_2 \uplus \overline{R}_3 = \overline{\varnothing}$.

Such a definition of coherence is a natural result of cut-elimination proof and it explains the reason that the coherent conditions in LMRL is the almost the same as those found in [11] [12]. The one single exception is the axiom. Because we allow more than two interpretations for a formula, principal i-formulas in the conclusion of **id**$_\wedge$ are no longer dual to each other, and we can not call them coherent either since the side condition is $R_1 \uplus \cdots \uplus R_n = \overline{\varnothing}$. If we say coherent interpretations involved in a cut means communication among *matched* parties, then axiom means a *different* kind of communication among "matched" parties (in a different sense) since they are not "coherent" in the same way. Indeed, this is in line with the existing work where axiom is interpreted as *forwarding* instead of regular communication [10] [9] [11] [12].

Furthermore, if we really want to introduce linear negation as a primitive connective, like

$$\frac{\vdash \Gamma, [A]_R}{\vdash \Gamma, [A^\perp]_{\overline{R}}}\ \textbf{not}$$

We then essentially assumes there are only two interpretations of a formula, and it will *invalidate* the admissibility of any n-**cut** where $n \neq 2$.

Since LMRL is multirole by design, we see LMRL as a form of negation-less logic: The negation of each i-formula $[A]_R$ is simply the i-formula $[A]_{\overline{R}}$ but there is no negation for the formula A *per se*. In other words, the notion of negation *cannot be internalized* within LMRL if one wants to preserve multi-cut.

Machine Checked Proof: We also have machine-checked proofs of all meta-properties of conjunctive/disjunctive linear/non-linear multirole logics available on-line at http://multirolelogic.org. These proofs are encoded in ATS[4], a functional programming language based on the Applied Type System framework [18] that supports dependent types (of DML-style), linear types, and programming with theorem-proving. The proofs are type-checked and the generated constraints are solved with the use of Z3 [19].

III. LMRL as a Process Calculus

As the very first inspiration for LMRL stems from studies on multiparty session types, it only seems fit if we present a typed variant (πLMRL) of π-calculus [2] [3] in which the types are directly based on the formulas in LMRL. We are to closely follow some recent work by Frank Pfenning [9] and Philip Wadler [10] in our presentation of πLMRL. In particular, we shall mostly adopt the notational convention used by the latter and thus refer the reader to the original paper for a detailed explanation. We see the encoding of cut-elimination of LMRL in πLMRL mostly as an exercise to facilitate the understanding of LMRL (but LMRL can certain have meaning that is beyond πLMRL or even concurrent programming).

[3]Indeed, the non-linear version of this correspondence is also true, that is, the classical logic is a special case of classical multirole logic where there are only two roles.

[4]Available at http://www.ats-lang.org

It should be noted that there exists a fundamental difference between πLMRL and π-calculus: The point-to-point communication in the latter is replaced with a form of broadcasting in the former. We are to mention at the end a simple extension of πLMRL that can support point-to-point communication directly.

A. *Session Types*

The session types (or types for short) are just formulas in LMRL extended with $\mathbf{1}_r$ (the unit for $\otimes_r$) for each role r:

$$\text{Session Types } A, B \quad ::= \quad a \mid \mathbf{1}_r \mid A \otimes_r B \mid A \&_r B \mid !_r A$$

where a is for primitive types. We drop quantified formulas for brevity. The meaning of various forms of session types is to be given later.

B. *Process Terms*

The syntax for the terms representing processes in πLMRL are given in Figure 2. We use x and y for names (of channels). Given x and R, the term x_R refers to an endpoint of the name x such that the endpoint is supposed to be held by a party playing the roles contained in R. We use P, Q for a process and $\vec{P}$ for a composition of processes of the form $(P_1 \mid \cdots \mid P_n)$ where $n \geq 1$.

P, Q	$::=$	$id_A(x_{R_1}, \ldots, x_{R_n})$	link
		$\nu x{:}A.\vec{P}$	parallel composition
		$x_R()_r.0$	$(r \in R)$ empty input
		$x_R[\,]_r.P$	$(r \notin R)$ empty output
		$x_R(y)_r.(P \mid Q)$	$(r \in R)$ input
		$x_R[y]_r.P$	$(r \notin R)$ output
		$x_R(case)_r.(P, Q)$	$(r \in R)$ offer choices
		$x_R[inl]_r.P$	$(r \notin R)$ choose left
		$x_R[inr]_r.P$	$(r \notin R)$ choose right
		$!x_R(y)_r.P$	$(r \in R)$ server accept
		$?x_R[y]_r.P$	$(r \notin R)$ client request

Fig. 2: The Syntax of πLMRL Processes

In $\nu x{:}A.\vec{P}$, the name x is bound in $\vec{P}$; in $x_R(y)_r.(P \mid Q)$, the name y is bound in P (but not in Q); in $x_R[y]_r.P$, $!x_R(y)_r.P$, and $?x_R[y]_r.P$, the name y is bound in P. Given a process P, we write $fn(P)$ for the set of free names in P. Note that the presence of annotations like A, R, and r in process terms is solely for supporting a form of Church typing. Such annotations may be *omitted* if there is no risk of confusion. In particular, they are not needed in the formulation of reduction semantics for πLMRL.

Interpretation: The meaning of various terms in Figure 2 is largely standard. $id_A(x_{R_1}, \ldots, x_{R_n})$ is a forwarding link among multiple endpoints following session type A; each message received on one endpoint will be broadcast to all other linked endpoints. $\nu x{:}A.(P_1 \mid \cdots \mid P_n)$, which is a combination of name restriction and parallel composition (after a cut), means to compose n processes $P_1, \ldots, P_n$ under channel x of session type A such that each P_k is holding an endpoint x_{R_k} of channel x, where $1 \leq k \leq n$ and $\overline{R}_1 \uplus \cdots \uplus \overline{R}_n = \varnothing$. Note that we mostly follow Wadler's notation [10]. Basically, we use $(\cdot)$ for an action to receive/offer/wait/accept and $[\cdot]$ for an action to send/choose/close/request.

$[A]_R, [B]_R$	$::=$	$[a]_R$	primitive
		$[\mathbf{1}_r]_R$	$r \in R$, wait ($\mathbf{1}$-like)
			$r \notin R$, close ($\perp$-like)
		$[A \otimes_r B]_R$	$r \in R$, input ($\otimes$-like)
			$r \notin R$, output ($\parr$-like)
		$[A \&_r B]_R$	$r \in R$, offer ($\&$-like)
			$r \notin R$, choose ($\oplus$-like)
		$[!_r A]_R$	$r \in R$, server accept ($!$-like)
			$r \notin R$, client request ($?$-like)

Fig. 3: The i-types of πLMRL

C. *Statics of πLMRL*

Given a session type A and a role set R, we can form an i-type $[A]_R$ as shown in Figure 3. If one thinks that A is a *global type*, then $[A]_R$ is a *local type* for the endpoint of a channel that is supposed to be held by a party playing the roles in R. We can see that generalizing the number of interpretations of a formula A from two (as in CLL) to many (as in LMRL) corresponds to generalizing local interpretations of a global session type from being of two-party to being of multiparty. The meaning for various forms of session types is to become clear when the typing rules for πLMRL are presented in Figure 4.

Interpretation: Let us write $[A \otimes B]_R$ ($[A \parr B]_R$) to mean $[A \otimes_r B]_R$ for some $r \in R$ ($r \notin R$). $A \otimes B$ is interpreted as *input A then behave as B* as is stipulated by the proof of cut-elimination. Dually, $A \parr B$ is interpreted as *output A then behave as B* in πLMRL. Let us write $[A \& B]_R$ ($[A \oplus B]_R$) to mean $[A \&_r B]_R$ for some $r \in R$ ($r \notin R$); $A \& B$ intuitively means offering choice of A or B and $A \oplus B$ means selecting from A or B. Let us write $[!A]_R$ ($[?A]_R$) to mean $[!_r A]_R$ for some $r \in A$ ($r \notin A$); $!A$ means the ability to repeatedly spawn processes of the type A while $?A$ means to request such a process to be spawned.

On Output v.s. Input: In the literature, $A \otimes B$ is often intuitively interpreted as *output A then behave as B* [9] [10] [12] based on the management of typing context, except [11]. In πLMRL, we interpret $\otimes$ as input.

In the cut-elimination step for $\otimes$ rules (please see Figure 10 in Appendix B, $\beta_\otimes$, for an example), there is one negative rule ($\parr$) v.s. multiple positive rules ($\otimes$). This is the *only possible senario* for a principal cut of $\otimes/\parr$ to be pushed up, and anything else would have broken the proof. The observation is confirmed by the coherence rules for $\otimes$/$\parr$ presented in [11] [12] as well. If we interpret $A \otimes B$ as input and $A \parr B$ as output, then this correspond to a *broadcast* from one endpoint to all other endpoints. If one inverts the input/output interpretation, then it will result in a kind of joining operation that requires all sending endpoints to synchronize with that one receiver [12], which may be a possibility but is not currently in line with the semantics of multiparty session types (which is traditionally based on broadcasting/multicasting).

Furthermore, by interpreting $\otimes$ as input and $\parr$ as output, it becomes natural to support multicasting/broadcasting as is also pointed out in [11]. The subsequent work in [12] reverses the stance taken in [11], interpreting $\otimes$ as output and removing the direct support for multicasting. In all the examples from [12], multicasting is simulated through multiple point-to-point communication due to the fact that their Axiom represents a link with only *two* ends. In that case, one must choose how to interpret $\otimes$, since there is only one $\otimes$ and one $\parr$ in a (principal) cut. This design does not seem to make full use of the coherence rule for $\otimes\parr$, which can involve one $\parr$-formula and *many* $\otimes$-formulas.

We choose to interpret $\otimes$ as output based on the cut-elimination proof of LMRL. As a consequence, we naturally support broadcasting and multicasting in πLMRL.

Judgements: In πLMRL, we use Γ for an environment associating distinct names with i-types. Let Γ be the following environment:

$$x_1 : [A_1]_{R_1}, \ldots, x_n : [A_n]_{R_n}$$

Then the domain $\mathbf{dom}(\Gamma)$ of Γ equals $\{x_1, \ldots, x_n\}$. Given another environment Γ', we write Γ, Γ' for the union of Γ and Γ' whenever $\mathbf{dom}(\Gamma) \cap \mathbf{dom}(\Gamma') = \varnothing$. We refer to $x : [A]_R$ as an association in an environment.

A typing judgment in πLMRL is of the form

$$P \vdash x_1 : [A_1]_{R_1}, \ldots, x_n : [A_n]_{R_n}$$

meaning that process P communicates along each endpoint x_k of role R_k in a channel specified by the session type A_k for $k = 1, \ldots, n$. Erasing process P and the channel names x_k from the judgment yields a judgment in LMRL (extended with $\mathbf{1}_r$ for each role r).

D. Dynamics of πLMRL

We present a reduction semantic for πLMRL based on Lemma 5, which states the admissibility of the following rule in LMRL:

$$(\Gamma_1, [A]_{R_1}; \cdots ; \Gamma_n, [A]_{R_n}) \Rightarrow \Gamma_1, \ldots, \Gamma_n$$

where $\overline{R_1} \uplus \cdots \uplus \overline{R_n} = \overline{\varnothing}$. Ideally, we would formulate such a reduction semantics by directly following the proof of Lemma 5, which essentially implies following the proof of Lemma 3. Unfortunately, it is not yet clear to us how the latter can be encoded in a process calculus[5]. Instead, we are to proceed by following a direct proof of Lemma 5, which is largely parallel to a standard proof of cut-elimination for classical linear logic (e.g., the one used by [10]).

[5]Even though, our implementations based on LMRL fully supports Lemma 3.

Structural Equivalence: We use $\equiv$ for a structural equivalence relation on processes and assume it to contain the following equivalence rules **perm** and **assoc**:

$$\nu x.\vec{P_1} \equiv \nu x.\vec{P_2} \qquad \textbf{perm}$$
$$\nu x.(\nu y.(P \mid \vec{P_1}) \mid \vec{P_2}) \equiv \nu y.(\nu x.(P \mid \vec{P_2}) \mid \vec{P_1}) \qquad \textbf{assoc}$$

For **perm**, $\vec{P_2}$ is assumed to be a permutation of $\vec{P_1}$. For **assoc**, both $x \notin fn(P_1)$ and $y \notin fn(P_2)$ are assumed to hold.

Reduction: We use $\longrightarrow$ for a single-step reduction relation on processes. We first introduce η-expansions, and then proper rules for reducing principal cuts and lastly introduce another set of rules for turning non-principal cuts into principal ones based on commuting conversions.

η-expansions: In LMRL, the **id** (axiom) rule is restricted to primitive formulas, and that can be generalized to composed formulas with Lemma 2. In πLMRL, a forwarding link is allowed to follow compound session types. Thus we need η-expansions for reductions to go through, as explained in [12]. We omit any further explanation and refer readers to the original paper. We only present rules in Figure 5. Without loss of generality, let us assume $r \in R_1$, and we present only 3-party η-expansions.

Principal Cuts: Let us assume a typing derivation $\mathcal{D}$ of the following form (where the last applied rule is n-**cut** for $n \geq 2$):

$$\frac{\overline{R_1} \uplus \cdots \uplus \overline{R_n} = \overline{\varnothing} \quad \mathcal{D}_i :: P_i \vdash \Gamma_i, x_{R_i} : [A]_{R_i} \quad 1 \leq i \leq n}{\nu x{:}A.(P_1 \mid \cdots \mid P_n) \vdash \Gamma_1, \ldots, \Gamma_n}$$

The cut is a *principal* one if each association $x : [A]_{R_i}$ is introduced by the last applied rule in $\mathcal{D}_i$ for $1 \leq i \leq n$. Principal cut reductions in LMRL is β-reductions in πLMRL. Without loss of generality, rules for β-reduction (of three parties without derivation trees) is shown in Figure 7. Please refer to Figure 10 (with derivation trees) in the appendix for a better illustrate of such correspondence.

We explains a bit more on β_{id}. As an illustration, see Figure 6. Let full set be $\{0, 1, 2\}$. Let $R_i = \{i\}$ for $i = 0, 1, 2$. Then three axiom links before reduction can be typed as follows for some proposition A.

$$id_A(x, x_0) \vdash x : [A]_{\overline{R_0}}, x_0 : [A]_{R_0}$$
$$id_A(x, x_1) \vdash x : [A]_{\overline{R_1}}, x_1 : [A]_{R_1}$$
$$id_A(x, x_2) \vdash x : [A]_{\overline{R_2}}, x_2 : [A]_{R_2}$$

The cut reduction is performed on x, and the result is

$$\nu x{:}A.(id_A(x, x_0) | id_A(x, x_1) | id_A(x, x_2)) \longrightarrow id_A(x_0, x_1, x_2)$$

Types for endpoints x_0, x_1 and x_2 suggests that they are *to be used/held* by users of *dual* types. Such inversion is also noted in [10]. Here we can think of each user creating an axiom link of two dual endpoints, reserving one for self-usage, and giving the other end to others as if everyone is

$$\frac{R_1 \uplus \cdots \uplus R_n = \overline{\varnothing}}{id_A(x_{R_1}, \ldots, x_{R_n}) \vdash x_{R_1} : [A]_{R_1}, \ldots, x_{R_n} : [A]_{R_n}}\ \textbf{id}$$

$$\frac{\overline{R}_1 \uplus \cdots \uplus \overline{R}_n = \overline{\varnothing} \quad P_1 \vdash \Gamma_1, x_{R_1} : [A]_{R_1} \quad \cdots \quad P_n \vdash \Gamma_n, x_{R_n} : [A]_{R_n}}{\nu x{:}A.(P_1 \,|\, \cdots \,|\, P_n) \vdash \Gamma_1, \ldots, \Gamma_n}\ n\textbf{-cut}$$

$$\frac{r \notin R \quad P \vdash \Gamma, y_R : [A]_R, x_R : [B]_R}{x_R[y]_r.P \vdash \Gamma, x_R : [A \otimes_r B]_R}\ \otimes\textbf{-neg} \qquad \frac{r \in R \quad P \vdash \Gamma_1, y_R : [A]_R \quad Q \vdash \Gamma_2, x_R : [B]_R}{x_R(y)_r.(P \,|\, Q) \vdash \Gamma_1, \Gamma_2, x_R : [A \otimes_r B]_R}\ \otimes\textbf{-pos}$$

$$\frac{r \notin R \quad P \vdash \Gamma, x_R : [A]_R}{x_R[inl]_r.P \vdash \Gamma, x_R : [A \&_r B]_R}\ \&\textbf{-neg-l} \qquad \frac{r \notin R \quad P \vdash \Gamma, x_R : [B]_R}{x_R[inr]_r.P \vdash \Gamma, x_R : [A \&_r B]_R}\ \&\textbf{-neg-r}$$

$$\frac{r \in R \quad P \vdash \Gamma, x_R : [A]_R \quad Q \vdash \Gamma, x_R : [B]_R}{x_R(case)_r.(P, Q) \vdash \Gamma, x_R : [A \&_r B]_R}\ \&\textbf{-pos}$$

$$\frac{r \notin R \quad P \vdash \Gamma, y_R : [A]_R}{?x_R[y]_r.P \vdash \Gamma, x_R : [!_r A]_R}\ \textbf{!-neg-derelict} \qquad \frac{r \in R \quad P \vdash ?\Gamma, y_R : [A]_R}{!x_R(y)_r.P \vdash ?\Gamma, x_R : [!_r A]_R}\ \textbf{!-pos} \qquad \frac{r \notin R \quad P \vdash \Gamma}{P \vdash \Gamma, x_R : [!_r A]_R}\ \textbf{!-neg-weaken}$$

$$\frac{r \notin R \quad P \vdash \Gamma, y_R : [!_r A]_R, z_R : [!_r A]_R}{P\{{}^{x_R}/_{y_R, z_R}\} \vdash \Gamma, x_R : [!_r A]_R}\ \textbf{!-neg-contract} \qquad \frac{r \notin R \quad P \vdash \Gamma}{x_R[\,]_r.P \vdash \Gamma, x_R : [\mathbf{1}_r]_R}\ \textbf{1-neg} \qquad \frac{r \in R}{x_R(\,)_r.0 \vdash x_R : [\mathbf{1}_r]_R}\ \textbf{1-pos}$$

Fig. 4: Typing Rules for πLMRL

$$\begin{aligned}
id_{A \otimes_r B}(x_{R_1}, x_{R_2}, x_{R_3}) &\xrightarrow{\eta} x_{R_3}[y_{R_3}]_r.x_{R_2}[y_{R_2}]_r.x_{R_1}(y_{R_1})_r.(id_A(y_{R_1}, y_{R_2}, y_{R_3}) \mid id_B(x_{R_1}, x_{R_2}, x_{R_3}))\\
id_{A \&_r B}(x_{R_1}, x_{R_2}, x_{R_3}) &\xrightarrow{\eta} x_{R_1}(case)_r.(x_{R_3}[inl]_r.x_{R_2}[inl]_r.id_A(x_{R_1}, x_{R_2}, x_{R_3}), x_{R_3}[inr]_r.x_{R_2}[inr]_r.id_B(x_{R_1}, x_{R_2}, x_{R_3}))\\
id_{\mathbf{1}_r}(x_{R_1}, x_{R_2}, x_{R_3}) &\xrightarrow{\eta} x_{R_3}[\,]_r.x_{R_2}[\,]_r.x_{R_1}()_r.0\\
id_{!_r A}(x_{R_1}, x_{R_2}, x_{R_3}) &\xrightarrow{\eta} !x_{R_1}(y_{R_1})_r.?x_{R_3}[y_{R_3}]_r.?x_{R_2}[y_{R_2}]_r.id_A(y_{R_1}, y_{R_2}, y_{R_3})
\end{aligned}$$

Fig. 5: η-expansion

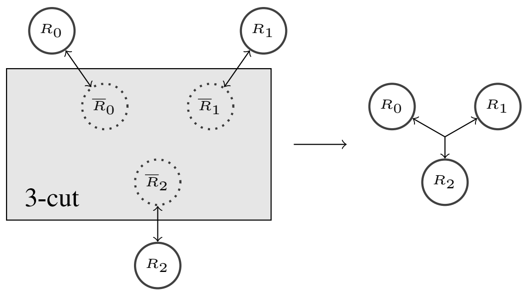

Fig. 6: Example 3-cut upon Axioms

Assuming $R_0 \uplus R_1 \uplus R_2 = \overline{\varnothing}$. i-formulas are represented just by their corresponding roles in circles. The $\leftrightarrow$ are links representing axioms. $\longrightarrow$ represents cut reductions. This figure shows that a 3-cut combines three 2-party sessions into a single 3-party session.

communicating with his/her dual. Data that flows into one endpoint of the axiom link will flow out from the other end. And within the three x's in the cut, data flows out of one endpoint, will be broadcasted to other endpoints. The reduction simplifies these three sessions down to one, by performing substitution and linking the three endpoints directly using the axiom. After reduction, users can *use* x_0, x_1 and x_2 to communicate directly. Note that users are not shown in the figure. Only endpoints are shown.

Non-Principal Cuts: The rules for handling non-principal cuts are based on commuting conversions, each of which pushes a cut inside a communication prefix. It can be readily checked that a cut must be a principal one if none of the rules in Figure 8 applies. As πLMRL is not the focus of this work, please see [10] for details on commuting conversions.

E. Subject Reduction and Deadlock-Freeness

Let us define $P \Longrightarrow Q$ as $P \equiv P' \longrightarrow Q' \equiv Q$ for some P' and Q'. Let $\Longrightarrow^*$ be the transitive closure of $\Longrightarrow$.

Theorem 7 (Subject Reduction)**.** *Assume that $P \vdash \Gamma$ is derivable. If $P \Longrightarrow P'$, then $P' \vdash \Gamma$ is also derivable.*

Proof. It simply follows the way in which $\longrightarrow$ is defined. □

Theorem 8 (Deadlock-Freeness)**.** *Assume that $P \vdash \Gamma$ is derivable. If P is a cut, then $P \Longrightarrow P'$ holds for some P'.*

Proof. (Sketch) It follows from a simple case analysis on P. For details, one may see the proof of Theorem 2 [20]. Note that we use cut-elimination in a rather liberal sense: It means the elimination of a particular cut (and P' may be allowed to contain cuts). The point is to guarantee progress being made rather than the eventuality of reaching some terminal state. □

$$
\begin{aligned}
\nu x{:}A.(id_A(x_1,x) \,|\, id_A(x_2,x) \,|\, id_A(x_3,x)) &\longrightarrow id_A(x_1,x_2,x_3) && (\beta_{id})\\
\nu x{:}A \otimes_r B.(x[y].P \,|\, x(y_2).(P_2 \,|\, Q_2) \,|\, x(y_3).(P_3 \,|\, Q_3)) &\longrightarrow \nu y{:}A.(\nu x{:}B.(P \,|\, Q_2 \,|\, Q_3) \,|\, P_2\{y/y_2\} \,|\, P_3\{y/y_3\}) && (\beta_{\otimes})\\
\nu x{:}A \&_r B.(x[inl].P \,|\, x(case).(P_2,Q_2) \,|\, x(case).(P_3 \,|\, Q_3)) &\longrightarrow \nu x{:}A.(P \,|\, P_2 \,|\, P_3) && (\beta_{\& l})\\
\nu x{:}A \&_r B.(x[inr].P \,|\, x(case).(P_2,Q_2) \,|\, x(case).(P_3 \,|\, Q_3)) &\longrightarrow \nu x{:}B.(P \,|\, Q_2 \,|\, Q_3) && (\beta_{\& r})\\
\nu x{:}\mathbf{1}_r.(x[].P \,|\, x().0 \,|\, x().0) &\longrightarrow P && (\beta_{\mathbf{1}})\\
\nu x{:}!_r A.(?x[y].P \,|\, !x(y_2).P_2 \,|\, !x(y_3).P_3) &\longrightarrow \nu y{:}A.(P \,|\, P_2\{y/y_2\} \,|\, P_3\{y/y_3\}) && (\beta_{!})\\
\nu x{:}!_r A.(P \,|\, !x(y_2).P_2 \,|\, !x(y_3).P_3) &\longrightarrow P && (\beta_{!W})\\
\nu x{:}!_r A.(P\{x/m,n\} \,|\, !x(y_2).P_2 \,|\, !x(y_3).P_3) &\longrightarrow \\
&\nu n{:}!_r A.(\nu m{:}!_r A.(P \,|\, !m(y_2).P_2 \,|\, !m(y_3).P_3) \,|\, !n(y_2).P_2 \,|\, !n(y_3).P_3) && (\beta_{!C})
\end{aligned}
$$

Fig. 7: β-reduction of πLMRL

Without loss of generality, we present here a 3-party principal cut reduction without annotations. Assume names are of role set R_1, R_2, and R_3 in the order of parallel composition. Assume $\overline{R}_1 \uplus \overline{R}_2 \uplus \overline{R}_3 = \overline{\varnothing}$ and $r \notin R_1$.

$$
\begin{aligned}
\nu x_0.(x[y].P_1 \mid \vec{P}) &\longrightarrow x[y].\nu x_0.(P_1 \mid \vec{P})\\
\nu x_0.(x(y).(P_1 \mid P_2) \mid \vec{P}) &\longrightarrow x(y).(\nu x_0.(P_1 \mid \vec{P}) \mid P_2)\\
&\qquad\qquad \text{if } x_0 \in fn(P_1)\\
\nu x_0.(x(y).(P_1 \mid P_2) \mid \vec{P}) &\longrightarrow x(y).(P_1 \mid \nu x_0.(P_2 \mid \vec{P}))\\
&\qquad\qquad \text{if } x_0 \in fn(P_2)\\
\nu x_0.(x[inl].P_1 \mid \vec{P}) &\longrightarrow x[inl].\nu x_0.(P_1 \mid \vec{P})\\
\nu x_0.(x[inr].P_1 \mid \vec{P}) &\longrightarrow x[inr].\nu x_0.(P_1 \mid \vec{P})\\
\nu x_0.(x(case).(P_1, P_2) \mid \vec{P}) &\longrightarrow x(case).\nu x_0.(P_1 \mid \vec{P}, P_2 \mid \vec{P})\\
\nu x_0.(!x[y].P_1 \mid \vec{P}) &\longrightarrow !x[y].\nu x_0.(P_1 \mid \vec{P})\\
\nu x_0.(?x(y).P_1 \mid \vec{P}) &\longrightarrow ?x(y).\nu x_0.(P_1 \mid \vec{P})\\
\nu x_0.(x[].P_1 \mid \vec{P}) &\longrightarrow x[].\nu x_0.(P_1 \mid \vec{P})
\end{aligned}
$$

Fig. 8: Commuting Conversions without Proof Terms

F. Clarifying Discussions

On Starting Off a Proof: In LMRL (also in CLL), there are two ways to start off a proof, **1** and **id**, and they are different from the perspective of πLMRL. If we start from **1**, we derive a single process and its type. If we start from the axiom **id**, we derive a *system* [21] of matching processes and their types, and corresponding roles satisfy similar side-conditions of **id**, which is $R_1 \uplus \cdots \uplus R_n = \overline{\varnothing}$. Cutting coherent ($\overline{R}_1 \uplus \cdots \uplus \overline{R}_n = \overline{\varnothing}$) processes derived by **1** will combine them into a system, and the system will reduces to inactive process 0 whose type is **1**. Cutting several systems derived by **id** is cutting coherent endpoints from each system, and reduce these systems to one or several simpler systems by explicit *name substitution*, as demoed in Figure 6. It is crucial to note that the resulting process from the cut of systems is inherently an *arbiter* [21] process that precisely described the internal communication of name-passing for forming *new* (sub-)sessions. [12] noted that reducing global types results in arbiter processes as well.

On the Duality of Axiom and Cut: It is well known that axiom and cut play dual role [13]. It is also reflected in πLMRL. Following [9], we interpret the identity axiom as forwarding links, where a message flows into one endpoint will flow out of all of the other endpoints, and n-**cut** as regular communication, where a message flows out of (sent from) one endpoint will flow into (be received by) all of the other endpoints. We see the identity axiom as communication within a system [21], and cut as communication among several systems (of the same global session type).

On Unary Connectives, Point-2-Point Communication, and LMRL as a Framework for Coordination: In CLL, formulas have only two interpretations. Thus the only possible unary connective is the negation. In LMRL, i-formulas have multiple interpretations. Thus we can introduce new unary connectives.

For instance, let us introduce a unary connective $msg_{r,s,T}(A)$ for sending a message of linear type T from endpoint of role r to endpoint of role s, and then continue as session type A. Its interpretation by a party playing roles in R is as follows. Note that if cut-elimination is the main concern, the type T of the payload is not important and we can omit it for simpler presentation.

- $r \in R, s \notin R$, send the message onto the channel (which means this endpoint is inputting a message from its owner) and then proceed as A.
- $r \notin R, s \in R$, receive the message from the channel (which means this endpoint is outputting a message to its owner) and then proceed as A.
- $r, s \in R$, skip the operation and then proceed as A since it is not relevant.
- $r, s \notin R$, skip the operation and then proceed as A since it is a self-loop.

We can introduce corresponding primitives into πLMRL and use msg (without payload type) to type them as follows.

$$
\frac{r \in R \land s \notin R \quad P \vdash \Gamma, x_R : [A]_R}{x_R(recv)_{r,s}.P \vdash x_R : [msg_{r,s}(A)]_R}
$$

$$\frac{r \notin R \wedge s \in R \quad P \vdash \Gamma, x_R : [A]_R}{x_R[send]_{r,s}.P \vdash x_R : [msg_{r,s}(A)]_R}$$
$$\frac{r \in R \wedge s \in R \quad P \vdash \Gamma, x_R : [A]_R}{x_R(skip)_{r,s}.P \vdash x_R : [msg_{r,s}(A)]_R}$$
$$\frac{r \notin R \wedge s \notin R \quad P \vdash \Gamma, x_R : [A]_R}{x_R[skip]_{r,s}.P \vdash x_R : [msg_{r,s}(A)]_R}$$

One can easily verify that adding the cut-elimination step for unary connective msg is straightforward, and the corresponding reduction step implements point-2-point value-passing communication.

For transmitting values, previous works either piggy-back on empty name sending ($\mathbf{1}$ and $\bot$) [9], or has to orient an axiom link so that it behaves like input/output pairs [12]. While in πLMRL, LMRL offers a *framework* to coherently interpret logical connectives such as msg at all local endpoints so that we don't need to piggy-back on $\mathbf{1}/\bot$ or modify the interpretation of the axiom. And the key is the ability to have multiple interpretations for a formula, as a generalization of only two as in CLL.

It is rather obvious to introduce multicast as well. For instance, $msg_{x,\vec{y},T}(A)$ will serve the purpose well, where $\vec{y}$ is a set of target roles different from the source role x. Its interpretation is similar to that of point-2-point messaging, and we left that to the reader.

We see LMRL *as a framework to safely coordinate among different parties*, offering possibilities for programmers to come up with new unary connectives and interpretations.

G. Examples

In this section, following conventions, we present some examples based on internet commerce. For a cleaner presentation, we might use the same name across different processes to make name substitution inexplicit. These examples (including Appendix D) are demonstrating the same 2-Buyer protocol but with different tastes. In all the examples, we omit annotations on the calculus since the number of participants is fixed and the redexes are intuitively clear. We may also omit the role r in $\mathbf{1}_r$ since whoever initiate the session termination is not important. We use _ to represent some unused name.

Example 1. We give an example of 2-Buyer protocol, using unary connective msg we just introduced. There are three participants, Seller, Buyer1 and Buyer2, identified as role 0, 1 and 2 respectively. The full set is therefore $\{0, 1, 2\}$. Let $\mathbf{N}$, $\mathbf{Q}$, $\mathbf{C}$, $\mathbf{A}$ and $\mathbf{D}$ be types of payload. Their interactions are as follows. Buyer1 sends to Seller a book title of type $\mathbf{N}$; Seller replies to Buyer1 and Buyer2 a quote price of type $\mathbf{Q}$; Buyer1 sends to Buyer2 the amount he/she can contribute, of type $\mathbf{C}$; Dependents on how much he/she still needs to pay, Buyer2 makes a choice; Buyer2 either choose to buy and sends to Seller his/her address of type $\mathbf{A}$, and Seller replies a projected delivery date of type $\mathbf{D}$; or Buyer2 choose to cancel and terminates the session.

Thus we have the session type as

$$\begin{aligned} G \overset{\text{def}}{=}\ & msg_{1,0,\mathbf{N}}(\mathbf{1}) \otimes_1 msg_{0,\{1,2\},\mathbf{Q}}(\mathbf{1}) \otimes_0 msg_{1,2,\mathbf{C}}(\mathbf{1}) \otimes_1 \\ & ((msg_{2,0,\mathbf{A}}(\mathbf{1}) \otimes_2 msg_{0,2,\mathbf{D}}(\mathbf{1}) \otimes_0 \mathbf{1}) \&_2 \mathbf{1}) \end{aligned}$$

and their processes are (assuming Buyer2 chooses to buy)

$$\begin{aligned} S(x) \overset{\text{def}}{=}\ & x(n).(n(recv).n().0 | x[q].q[send].q[]. \\ & x(c).(c(skip).c().0 | x(case).(\\ & \quad x(a).(a(recv).a().0 | x[d].d[send].d[].x().0), \\ & \quad x().0))) \\ B_1(x) \overset{\text{def}}{=}\ & x[n].n[send].n[].x(q).(q(recv).q().0 | \\ & x[c].c[send].c[].x(case).(\\ & \quad x(a).(a(skip).a().0 | x(d).(d(skip).d().0 | x().0)), \\ & \quad x().0)) \\ B_2(x) \overset{\text{def}}{=}\ & x(n).(n(skip).n().0 | x(q).(q(recv).q().0 | \\ & x(c).(c(recv).c().0 | x[inl].x[a].a[send].a[]. \\ & x(d).(d(recv).d().0 | x[].\_.0)))) \end{aligned}$$

We can then use 3-**cut** to compose these three processes, and derive

$$\frac{S(x) \vdash \Gamma_1, x : [G]_{\overline{0}} \quad B_1(x) \vdash \Gamma_2, x : [G]_{\overline{1}} \quad B_2(x) \vdash \Gamma_3, x : [G]_{\overline{2}}, \_ : [\mathbf{1}]_{\overline{\varnothing}}}{\nu x{:}G.(S(x)|B_1(x)|B_2(x)) \vdash \Gamma_0, \Gamma_1, \Gamma_2, \_ : [\mathbf{1}]_{\overline{\varnothing}}}$$

and the reduction below shows our intended interactions among three participants. In every step, we may rearrange cut-formulas via **perm** for cleaner presentation.

$$\begin{aligned} & \nu x{:}G.(S(x)|B_1(x)|B_2(x)) \\ \longrightarrow^* \ & \nu n{:}msg_{1,0,\mathbf{N}}.(n(recv).n().0 | n(skip).n().0 | n[send].n[]. \\ & \nu q{:}msg_{0,\{1,2\},\mathbf{Q}}.(q(recv).q().0 | q(recv).q().0 | q[send].q[]. \\ & \nu c{:}msg_{1,2,\mathbf{C}}.(c(recv).c().0 | c(skip).c().0 | c[send].c[]. \\ & \nu a{:}msg_{2,0,\mathbf{A}}.(a(recv).a().0 | a(skip).a().0 | a[send].a[]. \\ & \nu d{:}msg_{0,2,\mathbf{D}}.(d(recv).d().0 | d(skip).d().0 | d[send].d[]. \\ & \nu x{:}\mathbf{1}.(x().0 | x().0 | x[].\_.0)))))) \\ \longrightarrow^* \ & 0 \end{aligned}$$

Note that this style is essentially creating many small sub-sessions for transmitting values, which is in line with [11], [12]. This way, the calculus has a tighter correspondens with the logic, at the cost of diverging from traditional multiparty session types. In Appendix D, we will show other styles that 1) does not create many sub-sessions by avoiding $\otimes$'s which makes it closer to traditional multiparty session types 2) piggy-back on $\mathbf{1}$'s as a comparision 3) start the proof using axioms to show how the systems can be reduced to an arbiter process.

IV. Implementations

We have done in ATS several implementations of session types that are primarily based on πLMRL and used them in teaching. In one implementation, process communication is done via shared-memory and a process is implemented as a pthread. In another implementation, a process is mapped to a process in Erlang/Elixir. In yet another implementation, a process is mapped to either the web browser or some web worker running behind. It is simply beyond the scope of the paper to give any form of detailed description on these implementations. For anyone who is interested in trying out them, please first locate the google-group of the name *ats-lang-users*[6] and then search for information related to session types.

V. Related Work and Conclusion

While the development of LMRL is the major contribution of the paper, we find it rather difficult to *meaningfully* compare LMRL with other logics at this stage (of development of LMRL). Instead, we only mention work in the area of session-types.

The work in [11] and [12] is already covered in Section II-D and Section III-F. In what follows, we primarily comment on some related work from other sources.

The very notion of session types was introduced by Honda [6] and further extended by [7], [22]. There have since been extensive theoretical studies on session types in the literature (e.g., [9], [10], [23]–[27]). Multiparty session types, as a generalization of binary/dyadic session types, were introduced by Honda and others [8], together with the notion of global types, local types, projection and coherence.

Introduced by Milner and others [2], [3], π-calculus allows channel names to be communicated along the channels themselves, making it possible to describe concurrent computations with changing network configuration. Connections between π-calculus and linear logic have been actively studied from early on [4], [5], and it is demonstrated in some recent work [9], [10], [20], [28] that a tight proofs-as-processes correspondences exists for dyadic sessions. In contrast to the work in [10], [20], [28], which is based on classical linear logic (CLL), the work in [9] is based on dual intuitionistic linear logic (DILL). Clearly, πLMRL is also closely related to CLL.

In [11], Carbone et al. introduced MCP, a variant of CLL that admits MCut, a generalized cut-rule for composing multiple proofs. MCut requires coherent proofs (obtained through a separate proof system) as a side condition. Their follow-up work [12] introduced a variant of MCP, and a translation from MCP to CP [10] via GCP (an intermediate calculus) that interprets a coherence proof as an arbiter process that mediates communications in a multiparty session. Despite similarities, our work is fundamentally different from theirs in that their work focuses on translating MCP to CP with process-calculus in mind, whereas ours primarily aims at generalizing CLL into a new form of logic. As a result, LMRL is a full generalization of CLL, including a generalized version of the identity axiom that attests to the duality between the identity axiom and the cut. LMRL also admits Lemma 1, Lemma 3, Lemma 6, and unary connectives beyond negation, which are only made possible by multiple interpretations of a formula.

There have also been studies on multirole parties [29], [30], where such parties play multiple roles by holding channels belonging to multiple sessions. We see no direct relation between their multirole party and our multirole endpoint as formulated in this paper.

There is also very recent work on encoding multiparty session types based on binary session types [21], which relies on an arbiter process to mediate communications between multiple parties while preserving global ordering information. This form of mediating (formulated based on automata theory) corresponds a cut to multiple systems in πLMRL as discussed in previous sections.

In traditional multiparty session types [8], [31]–[34], well-formed orderings on session names are usually employed to guarantee deadlock-freeness. Their offer/choose primitives are not global and are limited by mergeability [33]. In LMRL, offer/choice are global synchronising points. Thus LMRL can describe protocols that traditional MPSTs cannot. For instance, a 3-party protocol $(A \otimes_1 C) \&_0 (B \otimes_2 C)$ where endpoint $\overline{0}$ makes a choice that either endpoint $\overline{1}$ sends A then proceed as C or endpoint $\overline{2}$ sends B then proceed as C.

Conclusion: We conclude that linear multirole logic is a genius generalization of classical linear logic. It is achieved by generalizing the number of interpretations of a formula from two to many, which corresponds to generalizing the local interpretations of a global session type from two to many. As a result, we obtain a cut rule that combines more than two sequents. The admissibility of such n-**cut** generalizes the celebreted result on cut-elimination by Gentzen. While multirole logic stems from studies on multiparty session types, it should certainly not be restricted to such studies. Just as the notion of linearity (as in linear logic) that has greatly enriched the study of logics and programming languages, we hope that the notion of multirole (as in multirole logic) can exert a significant impact in this regard as well.

[6] https://groups.google.com/forum/#!forum/ats-lang-users

References

[1] J.-Y. Girard, "Linear logic," vol. 50, no. 1, pp. 1–101, Jan. 1987.
[2] R. Milner, J. Parrow, and D. Walker, "A Calculus of Mobile Processes, I," *Inf. Comput.*, 1992.
[3] ——, "A Calculus of Mobile Processes, II," *Inf. Comput.*, 1992.
[4] S. Abramsky, "Proofs as Processes." *Theor. Comput. Sci.*, 1994.
[5] G. Bellin and P. J. Scott, "On the pi-Calculus and Linear Logic." *Theor. Comput. Sci.*, 1994.
[6] K. Honda, "Types for Dyadic Interaction." *CONCUR*, 1993.
[7] K. Honda, V. T. Vasconcelos, and M. Kubo, "Language primitives and type discipline for structured communication-based programming," in *Programming Languages and Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, Mar. 1998, pp. 122–138.
[8] K. Honda, N. Yoshida, and M. Carbone, "Multiparty asynchronous session types." *POPL*, pp. 273–284, 2008.
[9] L. Caires and F. Pfenning, "Session Types as Intuitionistic Linear Propositions." in *CONCUR*, 2010.
[10] P. Wadler, "Propositions as sessions." *ICFP*, pp. 273–286, 2012.
[11] M. Carbone, F. Montesi, C. Schürmann, and N. Yoshida, "Multiparty Session Types as Coherence Proofs." *CONCUR*, pp. 412–426, 2015.
[12] M. Carbone, S. Lindley, F. Montesi, and C. Schürmann, "Coherence Generalises Duality: a logical explanation of multiparty session types," in *CONCUR*, 2016, pp. 33:1–33:14.

[13] J.-Y. Girard, P. Taylor, and Y. Lafont, *Proofs and types*. Ecole Normale Superieure: Cambridge University Press, Apr. 1989.
[14] G. Gentzen, "Untersuchungen über das logische Schließen. I," *Mathematische Zeitschrift*, vol. 39, no. 1, pp. 176–210, 1935.
[15] ——, "Untersuchungen über das logische Schließen. II," *Mathematische Zeitschrift*, vol. 39, no. 1, pp. 405–431, 1935.
[16] S. Abramsky, "Computational Interpretations of Linear Logic." *Theor. Comput. Sci.*, 1993.
[17] A. S. Troelstra, "Lectures on Linear Logic," 1991.
[18] H. Xi, "Applied Type System," in *Types for Proofs and Programs*. Berlin, Heidelberg: Springer Berlin Heidelberg, Apr. 2003, pp. 394–408.
[19] L. de Moura and N. Bjørner, "Z3: An Efficient SMT Solver," in *Programming Languages and Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 337–340.
[20] P. Wadler, "Propositions as sessions," *Journal of Functional Programming*, vol. 24, no. 2-3, pp. 384–418, 2014.
[21] L. Caires and J. A. Pérez, "Multiparty session types within a canonical binary theory, and beyond," *International Conference on Formal Techniques for ...*, 2016.
[22] K. Takeuchi, K. Honda, and M. Kubo, "An Interaction-based Language and its Typing System." *PARLE*, 1994.
[23] G. Castagna, M. Dezani-Ciancaglini, E. Giachino, and L. Padovani, "Foundations of session types." *PPDP*, pp. 219–230, 2009.
[24] S. J. Gay and V. T. Vasconcelos, "Linear type theory for asynchronous session types," *Journal of Functional ...*, 2010.
[25] B. Toninho, L. Caires, and F. Pfenning, "Dependent session types via intuitionistic linear type theory." *PPDP*, 2011.
[26] V. T. Vasconcelos, "Fundamentals of session types." *Inf. Comput.*, 2012.
[27] S. Lindley and J. G. Morris, "A Semantics for Propositions as Sessions." *ESOP*, vol. 9032, no. Chapter 23, pp. 560–584, 2015.
[28] L. Caires, F. Pfenning, and B. Toninho, "Linear logic propositions as session types," *Mathematical Structures in ...*, 2016.
[29] N. Yoshida and P.-m. Deniélou, "Dynamic multirole session types." *POPL*, pp. 435–446, 2011.
[30] R. Neykova and N. Yoshida, "Multiparty Session Actors." *COORDINATION*, pp. 131–146, 2014.
[31] A. Bejleri and N. Yoshida, "Synchronous Multiparty Session Types," *Electronic Notes in Theoretical Computer Science*, vol. 241, pp. 3–33, Jul. 2009.
[32] L. Bettini, M. Coppo, L. D'Antoni, M. De Luca, M. Dezani-Ciancaglini, and N. Yoshida, "Global Progress in Dynamically Interleaved Multiparty Sessions." *CONCUR*, vol. 5201, no. Chapter 33, pp. 418–433, 2008.
[33] N. Yoshida, P.-m. Deniélou, A. Bejleri, and R. Hu, "Parameterised Multiparty Session Types." *FoSSaCS*, vol. 6014, no. Chapter 10, pp. 128–145, 2010.
[34] A. Scalas and N. Yoshida, "Lightweight Session Programming in Scala." *ECOOP*, 2016.
[35] H. Xi, Z. Ren, H. Wu, and W. Blair, "Session Types in a Linearly Typed Multi-Threaded Lambda-Calculus," *arXiv.org*, 2016.

Appendix A
Rules of Disjunctive LMRL

See Figure 9.

Appendix B
Detailed Principal Cut Reductions

See Figure 10.

Appendix C
Proof of Lemma 3

Proof.

Case (A is primitive)**.** Then it is a simple routine to verify that the sequent $\vdash \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2}$ follows from an application of the rule $\mathbf{id}_\wedge$.

Case (A is of the form $A_1 \otimes_r A_2$)**.** We have three possibilities

- $r \in R_1$ and $r \notin R_2$
- $r \notin R_1$ and $r \in R_2$

$$\frac{\overline{R}_1 \uplus \cdots \uplus \overline{R}_n = \varnothing}{\vdash [a]_{R_1}, \ldots, [a]_{R_n}}\ \mathbf{id}_\vee$$

$$\frac{r \in R \quad \vdash \Gamma, [A]_R, [B]_R}{\vdash \Gamma, [A \otimes_r B]_R}\ \mathbf{⅋\text{-pos}}$$

$$\frac{r \notin R \quad \vdash \Gamma_1, [A]_R \quad \vdash \Gamma_2, [B]_R}{\vdash \Gamma_1, \Gamma_2, [A \otimes_r B]_R}\ \mathbf{⅋\text{-neg}}$$

$$\frac{r \in R \quad \vdash \Gamma, [A]_R}{\vdash \Gamma, [A \&_r B]_R}\ \mathbf{\oplus\text{-pos-l}} \qquad \frac{r \in R \quad \vdash \Gamma, [B]_R}{\vdash \Gamma, [A \&_r B]_R}\ \mathbf{\oplus\text{-pos-r}}$$

$$\frac{r \notin R \quad \vdash \Gamma, [A]_R \quad \vdash \Gamma, [B]_R}{\vdash \Gamma, [A \&_r B]_R}\ \mathbf{\oplus\text{-neg}}$$

$$\frac{r \notin R \quad \vdash ?\Gamma, [A]_R}{\vdash ?\Gamma, [!_r A]_R}\ ?\mathbf{\text{-neg}}$$

$$\frac{r \in R \quad \vdash \Gamma, [A]_R}{\vdash \Gamma, [!_r A]_R}\ ?\mathbf{\text{-pos-derelict}}$$

$$\frac{r \in R \quad \vdash \Gamma}{\vdash \Gamma, [!_r A]_R}\ ?\mathbf{\text{-pos-weaken}}$$

$$\frac{r \in R \quad \vdash \Gamma, [!_r A]_R, [!_r A]_R}{\vdash \Gamma, [!_r A]_R}\ ?\mathbf{\text{-pos-contract}}$$

$$\frac{r \in R \quad \vdash \Gamma, [A\{{}^t/_x\}]_R}{\vdash \Gamma, [\forall_r(\lambda x.A)]_R}\ \exists\mathbf{\text{-pos}}$$

$$\frac{r \notin R \quad x \notin \Gamma \quad \vdash \Gamma, [A]_R}{\vdash \Gamma, [\forall_r(\lambda x.A)]_R}\ \exists\mathbf{\text{-neg}}$$

Fig. 9: $\text{LMRL}_\vee$ Inference Rules

- $r \in R_1$ and $r \in R_2$

Subcase ($r \in R_1$ and $r \notin R_2$)**.** Then $\mathcal{D}_1$ is of the following form,

$$\frac{\mathcal{D}_{11} :: (\Gamma_{11}, [A_1]_{R_1}) \quad \mathcal{D}_{12} :: (\Gamma_{12}, [A_2]_{R_1})}{\vdash \Gamma_1, [A]_{R_1}}\ \otimes\mathbf{\text{-pos}}$$

and $\mathcal{D}_2$ is of the following form,

$$\frac{\mathcal{D}_{21} :: (\Gamma_2, [A_1]_{R_2}, [A_2]_{R_2})}{\vdash \Gamma_2, [A]_{R_2}}\ \otimes\mathbf{\text{-neg}}$$

Then, we have

$$\begin{array}{rr}
\mathcal{D}'_{11} :: (\Gamma_{11}, \Gamma_2, [A_1]_{R_1 \cap R_2}, [A_2]_{R_2}) & \text{by I.H. on } \mathcal{D}_{11}, \mathcal{D}_{21} \\
\mathcal{D}'_{12} :: (\Gamma, [A_1]_{R_1 \cap R_2}, [A_2]_{R_1 \cap R_2}) & \text{by I.H. on } \mathcal{D}_{12}, \mathcal{D}'_{11} \\
\vdash \Gamma, [A]_{R_1 \cap R_2} & \text{by rule } \otimes\mathbf{\text{-neg}}
\end{array}$$

Subcase ($r \notin R_1$ and $r \in R_2$)**.** This case is analogous to the previous one.

Subcase ($r \in R_1$ and $r \in R_2$)**.** Then $\mathcal{D}_k$ is of the following form for each of the cases $k = 1$ and $k = 2$,

$$\frac{\mathcal{D}_{k1} :: (\Gamma_{k1}, [A_1]_{R_k}) \quad \mathcal{D}_{k2} :: (\Gamma_{k2}, [A_2]_{R_k})}{\vdash \Gamma_k, [A]_{R_k}}\ \otimes\mathbf{\text{-pos}}$$

Then, we have

$$\begin{array}{ll} \mathcal{D}'_1 :: (\Gamma_{11}, \Gamma_{21}, [A_1]_{R_1 \cap R_2}) & \text{by I.H. on } \mathcal{D}_{11}, \mathcal{D}_{21} \\ \mathcal{D}'_2 :: (\Gamma_{12}, \Gamma_{22}, [A_2]_{R_1 \cap R_2}) & \text{by I.H. on } \mathcal{D}_{12}, \mathcal{D}'_{22} \\ \vdash \Gamma, [A]_{R_1 \cap R_2} & \text{by rule } \otimes\textbf{-pos} \end{array}$$

Case (A is of the form $A_1 \&_r A_2$)**.** We have the same three possibilities as above.

Subcase ($r \in R_1$ and $r \notin R_2$)**.** Then $\mathcal{D}_1$ is of the following form,

$$\frac{\mathcal{D}_{11} :: (\Gamma_1, [A_1]_{R_1}) \quad \mathcal{D}_{12} :: (\Gamma_1, [A_2]_{R_1})}{\vdash \Gamma_1, [A]_{R_1}} \ \&\textbf{-pos}$$

and $\mathcal{D}_2$ is of the following form for $k = 1, 2$,

$$\frac{\mathcal{D}_{2k} :: (\Gamma_2, [A_k]_{R_2})}{\vdash \Gamma_2, [A]_{R_2}} \ \&\textbf{-neg-l(r)}$$

where the last rule in $\mathcal{D}_2$ is either $\&$**-neg-l** or $\&$**-neg-l** depending on k. Then,

$$\begin{array}{ll} \mathcal{D}'_k :: (\Gamma_1, \Gamma_2, [A_k]_{R_1 \cap R_2}) & \text{by I.H. on } \mathcal{D}_{1k}, \mathcal{D}_{2k} \\ \vdash \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2} & \text{by either } \&\textbf{-neg-l} \text{ or } \&\textbf{-neg-r} \end{array}$$

Subcase ($r \notin R_1$ and $r \in R_2$)**.** This case is analogous to the previous one.

Subcase ($r \in R_1$ and $r \in R_2$)**.** Then $\mathcal{D}_k$ is of the following form for each of the cases $k = 1$ and $k = 2$,

$$\frac{\mathcal{D}_{k1} :: (\Gamma_{k1}, [A_1]_{R_k}) \quad \mathcal{D}_{k2} :: (\Gamma_{k2}, [A_2]_{R_k})}{\vdash \Gamma_k, [A]_{R_k}} \ \&\textbf{-pos}$$

Then we have,

$$\begin{array}{ll} \mathcal{D}'_1 :: (\Gamma_{11}, \Gamma_{21}, [A_1]_{R_1 \cap R_2}) & \text{by I.H. on } \mathcal{D}_{11}, \mathcal{D}_{21} \\ \mathcal{D}'_2 :: (\Gamma_{12}, \Gamma_{22}, [A_2]_{R_1 \cap R_2}) & \text{by I.H. on } \mathcal{D}_{12}, \mathcal{D}_{22} \\ \vdash \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2} & \text{by } \&\textbf{-pos} \text{ on } \mathcal{D}'_1, \mathcal{D}'_2 \end{array}$$

Case (A is of the form $!_r A'$)**.** This is the most involved case. We have the same three possibilities as above.

Subcase ($r \in R_1$ and $r \notin R_2$). Then $\mathcal{D}_1$ is of the following form,

$$\frac{\mathcal{D}_{11} :: (?\Gamma_1, [A']_{R_1})}{\vdash ?\Gamma_1, [A]_{R_1}} \ \textbf{!-pos}$$

And there are three possibilities of $\mathcal{D}_2$ depending on which negetive rule is the last rule.

- $\mathcal{D}_2$ is of the following form,

$$\frac{\mathcal{D}_{21} :: (\Gamma_2, \{[A]_{R_2}\})}{\vdash \Gamma_2, \{[A]_{R_2}\}} \ \textbf{!-neg-weaken}$$

  and we have $\Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2}$ by I.H. on $\mathcal{D}_1$ and $\mathcal{D}_{21}$.
- $\mathcal{D}_2$ is of the following form,

$$\frac{\mathcal{D}_{21} :: (\Gamma_2, \{[A]_{R_2}\}, [A']_R)}{\vdash \Gamma_2, \{[A]_{R_2}\}} \ \textbf{!-neg-derelict}$$

  and then

$$\begin{array}{ll} \mathcal{D}_{121} :: (\Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2}, [A']_R) & \text{by I.H. on } \mathcal{D}_1, \mathcal{D}_{21} \\ \mathcal{D}'_{121} :: (\Gamma_1, \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2}) & \text{by I.H. on } \mathcal{D}_{11}, \mathcal{D}_{121} \\ \vdash \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2} & \text{by } \textbf{!-neg-contract} \text{ repeatedly} \end{array}$$

- $\mathcal{D}_2$ is of the form,

$$\frac{\mathcal{D}_{21} :: (\Gamma_2, \{[A]_{R_2}\}, [A]_{R_2})}{\vdash \Gamma_2, \{[A]_{R_2}\}} \ \textbf{!-neg-contract}$$

  We then have $\vdash \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2}$ by I.H. on $\mathcal{D}_1, \mathcal{D}_2$

Subcase ($r \notin R_1$ and $r \in R_2$)**.** This subcase is an analogous to the previous one.

Subcase ($r \in R_1$ and $r \in R_2$)**.** Then $\mathcal{D}_k$ is of the following form for each of the cases $k = 1$ and $k = 2$,

$$\frac{\mathcal{D}_{k1} :: (?\Gamma_k, [A']_{R_k})}{\vdash ?\Gamma_k, [A]_{R_k}} \ \textbf{!-pos}$$

Then, we have

$$\begin{array}{ll} D'_{12} :: (?\Gamma_1, ?\Gamma_2, [A']_{R_1 \cap R_2}) & \text{by I.H. on } \mathcal{D}_{11}, \mathcal{D}_{21} \\ \vdash ?\Gamma_1, ?\Gamma_2, [A]_{R_1 \cap R_2} & \text{by } \textbf{!-pos} \text{ on } \mathcal{D}'_{12} \end{array}$$

Case (A is of the form $\forall(\lambda x.A')$)**.** We have the same three possibilities as above.

Subcase ($r \in R_1$ and $r \notin R_2$)**.** Then $\mathcal{D}_1$ is of the following form,

$$\frac{\mathcal{D}_{11} :: (\Gamma_1, [A']_{R_1})}{\vdash \Gamma, [A]_{R_1}} \ \forall\textbf{-pos}$$

where x is not free in Γ_1, and $\mathcal{D}_2$ is of the following form,

$$\frac{\mathcal{D}_{21} :: (\Gamma_2, [A'\{^t/_x\}]_{R_2})}{\vdash \Gamma_, [A]_{R_2}} \ \forall\textbf{-neg}$$

Let $\mathcal{D}'_{11}$ be $\mathcal{D}_{11}\{^t/_x\}$, which is a derivation of $(\Gamma_1, [A'\{^t/_x\}]_{R_1})$.

$$\begin{array}{ll} \mathcal{D}_{121} :: (\Gamma_1, \Gamma_2, [A'\{^t/_x\}]_{R_1 \cap R_2}) & \text{by I.H. on } \mathcal{D}'_{11}, \mathcal{D}_{21} \\ \vdash \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2} & \text{by } \forall\textbf{-neg} \text{ on } \mathcal{D}_{121} \end{array}$$

Subcase ($r \notin R_1$ and $r \in R_2$)**.** This case is analogous to the previous one.

Subcase ($r \in R_1$ and $r \in R_2$)**.** Then $\mathcal{D}_k$ is of the following form for each of the cases $k = 1$ and $k = 2$,

$$\frac{\mathcal{D}_{k1} :: (\Gamma_k, [A]_{R_k}, [A']_{R_k})}{\vdash \Gamma_k, [A]_{R_k}} \ \forall\textbf{-pos}$$

where x is not free in Γ_k.

$$\begin{array}{ll} \mathcal{D}'_{12} :: (\Gamma_1, \Gamma_2, [A']_{R_1 \cap R_2}) & \text{by I.H. on } \mathcal{D}_{11}, \mathcal{D}_{21} \\ \vdash \Gamma_1, \Gamma_2, [A]_{R_1 \cap R_2} & \text{by } \forall\textbf{-pos} \text{ on } \mathcal{D}'_{12} \end{array}$$

All of the cases are covered where the cut-formula is the principal formula of both $\mathcal{D}_1$ and $\mathcal{D}_2$. For brevity, we omit the cases where the cut-formula is not principal in either $\mathcal{D}_1$ or $\mathcal{D}_2$, which can be trivially handled [17].

□

Appendix D
More Examples

In this section, we give examples of the same 2-Buyer protocol but with different flavors. Example 1 in the main text uses $\otimes$ in combination with msg, which is essentially creating many small sessions of type msg for transmitting values. Although this style provides a tighter correspondence with the logic, it is less similar to traditional multiparty session types. Therefore we give Example 2 that solely uses msg that closely resemble real world uses of traditional multiparty session types. We also provide Example 3 that piggy-backs on $\mathbf{1}$ as has been done in other works, e.g. [9]. Using $\mathbf{1}$ means the actual value-passing part is basiclly left unspecified, or only specified informally. Whereas in LMRL, by using unary connective msg, we are able to specify it. Lastly, we present Example 4 that shows a cut among systems [21], which reduces into an arbiter/medium process [21].

Example 2. Based on the same protocol, we give an native point-to-point example by using unary connective msg only. To avoid deep nesting, we write $msg_{i,j,T} :: A$ for $msg_{i,j,T}(A)$ with :: being right associative. We write $msg_{i,\{j_1,\dots,j_n\},T}$ for multicasting from i to $j_1,\dots,j_n$. Again, we omit the r from $\mathbf{1}_r$ assuming we have fixed some valid $r \in \{0,1,2\}$.

Let global session type be

$$\begin{aligned} G \overset{\text{def}}{=}\ & msg_{1,0,\mathbf{N}} :: msg_{0,\{1,2\},\mathbf{Q}} :: msg_{1,2,\mathbf{C}} \\ & :: ((msg_{2,0,\mathbf{A}} :: msg_{0,2,\mathbf{D}} :: \mathbf{1}) \&_2 \mathbf{1}) \end{aligned}$$

Let processes be

$$\begin{aligned} S(x) &\overset{\text{def}}{=} x(recv).x[send].x(skip). \\ &\quad x(case).(x(recv).x[send].x().0), x().0) \\ B_1(x) &\overset{\text{def}}{=} x[send].x(recv).x[send]. \\ &\quad x(case).(x(skip).x(skip).x().0, x().0) \\ B_2(x) &\overset{\text{def}}{=} x(skip).x(recv).x(recv). \\ &\quad x[inl].x[send].x(recv).x[].\_.0 \end{aligned}$$

We can then use 3-**cut** to compose these three processes, and derive

$$\frac{S(x) \vdash \Gamma_1, x:[G]_{\bar{0}} \qquad B_1(x) \vdash \Gamma_2, x:[G]_{\bar{1}} \qquad B_2(x) \vdash \Gamma_3, x:[G]_{\bar{2}}, \_:[\mathbf{1}]_{\bar{\varnothing}}}{\nu x{:}G.(S(x) \,|\, B_1(x) \,|\, B_2(x)) \vdash \Gamma_1, \Gamma_2, \Gamma_3, \_:[\mathbf{1}]_{\bar{\varnothing}}}$$

and the reduction below shows our intended interactions among three participants. In every step, we use **perm** to rearrange cut formulas, and we only show the parts correspond to the cut formulas.

$$\begin{aligned} & \nu x.(S(x) \,|\, B_1(x) \,|\, B_2(x)) \\ & \longrightarrow \nu x.(x(recv)\dots \,|\, x[send]\dots \,|\, x(skip)\dots) \\ & \longrightarrow \nu x.(x[send]\dots \,|\, x(recv)\dots \,|\, x(recv)\dots) \\ & \longrightarrow \nu x.(x(skip)\dots \,|\, x[send]\dots \,|\, x(recv)\dots) \\ & \longrightarrow \nu x.(x(case)\dots \,|\, x(case)\dots \,|\, x[inl]\dots) \\ & \longrightarrow \nu x.(x(recv)\dots \,|\, x(skip)\dots \,|\, x[send]\dots) \\ & \longrightarrow \nu x.(x[send]\dots \,|\, x(skip)\dots \,|\, x(recv)\dots) \\ & \longrightarrow \nu x.(x().0 \,|\, x().0 \,|\, x[].\_.0) \\ & \longrightarrow 0 \end{aligned}$$

Although capturing the same 2-Buyer protocol, this presentation is more direct, simpler and closer to real world implementations. In Example 1, channel name x is used by processes S, B_1 and B_2 to broadcast other names (e.g. n, q, etc,.) of session types like msg, and then those names are used to form new sub-sessions to exchange data. However, in Example 2, we reuse the same channel name x for all the communications, eliminating the need for additional sub-sessions. It is made possible because msg is a unary connective in LMRL.

As an extension, one may allow the payload to be a name, resulting in high-order sessions in the form of msg (as compared to high-order sessions in the form of $\otimes$, which is already supported in LMRL.). [11] mentioned that "coherence can be generalized … rule $\otimes ⅋$ coulde allow the involved participants to play different roles in the nested session they create." In the sense of LMRL, this is saying that logical rules can relate formulas of different interpretations, as a generalization from logical rules that relate formulas of the same signature as in CLL [13]. We have not explored such extensions. But there is a possibility that $msg_{i,j,T}(A)$ where T is a linear typed value, including not only primitive values but also session, could provide a logical account for the aforementioned extensions. Indeed, it may provide a logical account for a deadlock-freeness proof in [35]. We leave this as a future work.

Example 3. This time, instead of using unary connective msg, we piggy-back on $\mathbf{1}$ as in [9]. Compared to [9], [11], [12], [20], unary connective msg offers formal specifications for sending values in the session types.

This time, $\mathbf{N}$, $\mathbf{Q}$, $\mathbf{C}$, $\mathbf{A}$, $\mathbf{D}$ are all aliases of $\mathbf{1}$. Let

$$G \overset{\text{def}}{=} \mathbf{N} \otimes_1 \mathbf{Q} \otimes_0 \mathbf{C} \otimes_1 ((\mathbf{A} \otimes_2 \mathbf{D} \otimes_0 \mathbf{1}) \&_2 \mathbf{1})$$

Let processes be

$$\begin{aligned} S(x) &\overset{\text{def}}{=} x(n).(\text{get-name}_n | x[q].\text{put-quote}_q.x(c).(c().0| \\ &\quad x(case).(x(a).(\text{get-addr}_a | x[d].\text{put-date}_d.x().0), \\ &\quad x().0))) \\ B_1(x) &\overset{\text{def}}{=} x[n].\text{put-name}_n.x(q).(\text{get-quote}_q | x[c].\text{put-contrib}_c. \\ &\quad x(case).(x(a).(a().0 | x(d).(d().0 | x().0)), x().0)) \\ B_2(x) &\overset{\text{def}}{=} x(n).(n().0 | x(q).(\text{get-quote}_q | x(c).(\text{get-contrib}_c | \\ &\quad x[inl].x[a].\text{put-addr}_a.x(d).(\text{get-date}_d | x[].\_.0)))) \end{aligned}$$

where get-name_n is an alias/shortcuts for $n().0$. It receives a book title on channel n. put-quote_q is an alias/shortcuts for $q[]$. It sends a book price on channel q. Other shortcuts are defined similarly.

We can then use 3-**cut** to compose these three processes, and derive

$$\frac{S(x) \vdash \Gamma_0, x : [G]_{\overline{0}} \qquad B_1(x) \vdash \Gamma_1, x : [G]_{\overline{1}} \qquad B_2(x) \vdash \Gamma_2, x : [G]_{\overline{2}}}{\nu x{:}G.(S(x)|B_1(x)|B_2(x)) \vdash \Gamma_0, \Gamma_1, \Gamma_2}$$

and the reduction below shows our intended interactions among three participants.

$$\begin{aligned}
&\nu x{:}G.(S(x)|B_1(x)|B_2(x)) \\
&\longrightarrow^* \nu n{:}\mathbf{1}.(\text{get-name}_n|n().0|\text{put-name}_n. \\
&\quad \nu q{:}\mathbf{1}.(\text{get-quote}_q|\text{get-quote}_q|\text{put-quote}_q. \\
&\quad\quad \nu c{:}\mathbf{1}.(\text{get-contrib}_c|c().0|\text{put-contrib}_c. \\
&\quad\quad\quad \nu a{:}\mathbf{1}.(\text{get-addr}_a|a().0|\text{put-addr}_a. \\
&\quad\quad\quad\quad \nu d{:}\mathbf{1}.(\text{get-date}_d|d().0|\text{put-date}_d. \\
&\quad\quad\quad\quad\quad \nu x{:}\mathbf{1}.(x[].0|x[].0|x().\_.0))))) \\
&\longrightarrow^* 0
\end{aligned}$$

Note that, in [10], [12], instead of piggy-back on **1**, they use atomic propositions for **N**, **Q**, etc,. In that case, all those shortcuts such as get-name$_n$ will be axiom links since the only way to derive atomic propositions is through axioms. However, these works left such details mostly unexplained, and simply assumes that we have processes of such atomic types without using axioms. We believe this is one reason why [9] starts a proof using **1** instead, since it avoids deriving a process that involves not only the endpoint of interest, but also the other endpoint that comes with the axiom link. We will see in Example 4 what will actually happen when we start a proof using the axiom.

Example 4. We give again a 2-Buyer protocol. But this time we start all proofs using the axiom.

This time, **N**, **Q**, **C**, **A**, **D** are some atomic propositions. Let the global session type be

$$G \overset{\text{def}}{=} \mathbf{N} \otimes_1 \mathbf{Q} \otimes_0 \mathbf{C} \otimes_1 ((\mathbf{A} \otimes_2 \mathbf{D} \otimes_0 \mathbf{1}) \&_2 \mathbf{1})$$

and processes be

$$\begin{aligned}
S'(x) \overset{\text{def}}{=}\ & x_0[n_0].x(n).(id_{\mathbf{N}}(n, n_0)| \\
& \quad x[q].x_0(q_0).(id_{\mathbf{Q}}(q, q_0)| \\
& \quad\quad x_0[c_0].x(c).(id_{\mathbf{C}}(c, c_0)| \\
& \quad\quad\quad x(case).(\\
& \quad\quad\quad\quad x_0[inl].x_0[a_0].x(a).(id_{\mathbf{A}}(a, a_0)| \\
& \quad\quad\quad\quad\quad x[d].x_0(d_0).(id_{\mathbf{D}}(d, d_0)|id_{\mathbf{1}}(x, x_0))), \\
& \quad\quad\quad\quad x_0[inr].id_{\mathbf{1}}(x, x_0)))))
\end{aligned}$$

$$\begin{aligned}
B_1'(x) \overset{\text{def}}{=}\ & x[n].x_1(n_1).(id_{\mathbf{N}}(n, n_1)| \\
& \quad x_1[q_1].x(q).(id_{\mathbf{Q}}(q, q_1)| \\
& \quad\quad x[c].x_1(c_1).(id_{\mathbf{C}}(c, c_1)| \\
& \quad\quad\quad x(case).(\\
& \quad\quad\quad\quad x_1[inl].x_1[a_1].x(a).(id_{\mathbf{A}}(a, a_1)| \\
& \quad\quad\quad\quad\quad x_1[d_1].x(d).(id_{\mathbf{D}}(d, d_1)|id_{\mathbf{1}}(x, x_1))), \\
& \quad\quad\quad\quad x_1[inr].id_{\mathbf{1}}(x, x_1)))))
\end{aligned}$$

$$\begin{aligned}
B_2'(x) \overset{\text{def}}{=}\ & x_2[n_2].x(n).(id_{\mathbf{N}}(n, n_2)| \\
& \quad x_2[q_2].x(q).(id_{\mathbf{Q}}(q, q_2)| \\
& \quad\quad x_2[c_2].x(c).(id_{\mathbf{C}}(c, c_2)| \\
& \quad\quad\quad x_2(case).(\\
& \quad\quad\quad\quad x[inl].x[a].x_2(a_2).(id_{\mathbf{A}}(a, a_2)| \\
& \quad\quad\quad\quad\quad x_2[d_2].x(d).(id_{\mathbf{D}}(d, d_2)|id_{\mathbf{1}}(x, x_2))), \\
& \quad\quad\quad\quad x[inr].id_{\mathbf{1}}(x, x_2)))))
\end{aligned}$$

We can derive

$$\frac{S'(x) \vdash x : [G]_{\overline{0}}, x_0 : [G]_0 \qquad B_1'(x) \vdash x : [G]_{\overline{1}}, x_1 : [G]_1 \qquad B_2'(x) \vdash x : [G]_{\overline{2}}, x_2 : [G]_2}{\nu x{:}G.(S'(x)|B_1'(x)|B_2'(x)) \vdash x_0 : [G]_0, x_1 : [G]_1, x_2 : [G]_2}$$

And the reduction is as follows

$$\begin{aligned}
&\nu x{:}G.(S'(x)|B_1'(x)|B_2'(x)) \longrightarrow \\
&\quad x_0[n_0].x_2[n_2].x_1(n_1).(id_{\mathbf{N}}(n_0, n_1, n_2)| \\
&\quad x_1[q_1].x_2[q_2].x_0(q_0).(id_{\mathbf{Q}}(q_0, q_1, q_2)| \\
&\quad x_0[c_0].x_2[c_2].x_1(c_1).(id_{\mathbf{C}}(c_0, c_1, c_2)| \\
&\quad x_2(case).(\\
&\quad\quad x_0[inl].x_1[inl].x_0[a_0].x_1[a_1].x_2(a_2).(id_{\mathbf{A}}(a_0, a_1, a_2)| \\
&\quad\quad\quad x_1[d_1].x_2[d_2].x_0(d_0).(id_{\mathbf{D}}(d_0, d_1, d_2)|id_{\mathbf{1}}(x_0, x_1, x_2))), \\
&\quad\quad x_0[inr].x_1[inr].id_{\mathbf{1}}(x_0, x_1, x_2)))))
\end{aligned}$$

The resulting process can be viewed as a medium process [21]. It cannot reduce any further since it awaits external communications from users. The seller/buyers processes will *hold* or *use* endpoints to interact with others. Compare this with Example 3 where we start the proof using **1** and x's *are* the users. Note, $S'(x)$ is actually an η-expansion of

$$id_G(x, x_0) \vdash x : [G]_{\overline{0}}, x_0 : [G]_0$$

Indeed, by composing this process with users from Example 3, we get a system [21] that reduces to 0.

$$\begin{aligned}
&\nu x_0{:}G.(S(x_0)\,|\,\nu x_1{:}G.(B_1(x_1)\,|\,\nu x_2{:}G.(B_2(x_2)\,| \\
&\qquad\qquad \nu x{:}G.(S'(x)\,|\,B_1'(x)\,|\,B_2'(x))))) \longrightarrow 0
\end{aligned}$$

Note that those axiom links at the innermost level are basically left unexplained. It is up to the programmer to interpret them and implement the desired protocol. As a comparision, using msg can formally specify what to do at each endpoint, thus offer better control in the type system.

$$\dfrac{\dfrac{}{id_A(x_1,x)\vdash x_1,x}\textbf{id}\quad\dfrac{}{id_A(x_2,x)\vdash x_2,x}\textbf{id}\quad\dfrac{}{id_A(x_3,x)\vdash x_3,x}\textbf{id}}{\nu x{:}A.(id_A(x_1,x)\,|\,id_A(x_2,x)\,|\,id_A(x_3,x))\vdash x_1,x_2,x_3}\textbf{3-cut}\quad\longrightarrow\quad\dfrac{}{id_A(x_1,x_2,x_3)\vdash x_1,x_2,x_3}\textbf{id}\qquad(\beta_{id})$$

$$\dfrac{\dfrac{r\notin R_1\quad P\vdash y,x}{x[y].P\vdash x}\otimes\textbf{-neg}\quad\dfrac{r\in R_2\quad P_2\vdash y_2\quad Q_2\vdash x}{x(y_2).(P_2|Q_2)\vdash x}\otimes\textbf{-pos}\quad\dfrac{r\in R_3\quad P_3\vdash y_3\quad Q_3\vdash x}{x(y_3).(P_3|Q_3)\vdash x}\otimes\textbf{-pos}}{\nu x{:}A\otimes_r B.(x[y].P\,|\,x(y_2).(P_2\,|\,Q_2)\,|\,x(y_3).(P_3\,|\,Q_3))\vdash\cdot}\textbf{3-cut}$$

$$\longrightarrow\quad\dfrac{\dfrac{P\vdash y,x\quad Q_2\vdash x\quad Q_3\vdash x}{\nu x{:}B.(P\,|\,Q_2\,|\,Q_3)\vdash y}\textbf{3-cut}\qquad P_2\{y/y_2\}\vdash y\quad P_3\{y/y_3\}\vdash y}{\nu y{:}A.(\nu x{:}B.(P\,|\,Q_2\,|\,Q_3)\,|\,P_2\{y/y_2\}\,|\,P_3\{y/y_3\})\vdash\cdot}\textbf{3-cut}\qquad(\beta_\otimes)$$

$$\dfrac{\dfrac{r\notin R_1\quad P\vdash x}{x[inl].P\vdash x}\&\textbf{-neg-l}\quad\dfrac{r\in R_2\quad P_2\vdash x\quad Q_2\vdash x}{x.case(P_2,Q_2)\vdash x}\&\textbf{-pos}\quad\dfrac{r\in R_3\quad P_3\vdash x\quad Q_3\vdash x}{x.case(P_3,Q_3)\vdash x}\&\textbf{-pos}}{\nu x{:}A\&_r B.(x[inl].P\,|\,x.case(P_2,Q_2)\,|\,x.case(P_3,Q_3))\vdash\cdot}\textbf{3-cut}$$

$$\longrightarrow\quad\dfrac{P\vdash x\quad P_2\vdash x\quad P_3\vdash x}{\nu x{:}A.(P|P_2|P_3)\vdash\cdot}\textbf{3-cut}\qquad(\beta_{\&l})$$

$$\dfrac{\dfrac{r\notin R_1\quad P\vdash x}{x[inr].P\vdash x}\&\textbf{-neg-r}\quad\dfrac{r\in R_2\quad P_2\vdash x\quad Q_2\vdash x}{x.case(P_2,Q_2)\vdash x}\&\textbf{-pos}\quad\dfrac{r\in R_3\quad P_3\vdash x\quad Q_3\vdash x}{x.case(P_3,Q_3)\vdash x}\&\textbf{-pos}}{\nu x{:}A\&_r B.(x[inr].P\,|\,x.case(P_2,Q_2)\,|\,x.case(P_3,Q_3))\vdash\cdot}\textbf{3-cut}$$

$$\longrightarrow\quad\dfrac{P\vdash x\quad Q_2\vdash x\quad Q_3\vdash x}{\nu x{:}B.(P|Q_2|Q_3)\vdash\cdot}\textbf{3-cut}\qquad(\beta_{\&r})$$

$$\dfrac{\dfrac{r\notin R_1\quad P\vdash\cdot}{x[\,].P\vdash x}\textbf{1-neg}\quad\dfrac{r\in R_2}{x(\,).0\vdash x}\textbf{1-pos}\quad\dfrac{r\in R_3}{x(\,).0\vdash x}\textbf{1-pos}}{\nu x{:}\mathbf{1}.(x[].P\,|\,x().0\,|\,x().0)\vdash\cdot}\textbf{3-cut}\quad\longrightarrow\quad P\vdash\cdot\qquad(\beta_{\mathbf{1}})$$

$$\dfrac{\dfrac{r\notin R_1\quad P\vdash y}{?x(y).P\vdash x}\textbf{!-neg-derelict}\quad\dfrac{r\in R_2\quad P_2\vdash y_2}{!x(y_2).P_2\vdash x}\textbf{!-pos}\quad\dfrac{r\in R_3\quad P_3\vdash y_3}{!x(y_3).P_3\vdash x}\textbf{!-pos}}{\nu x{:}!_r A.(?x[y].P\,|\,!x(y_2).P_2\,|\,!x(y_3).P_3)\vdash\cdot}\textbf{3-cut}$$

$$\longrightarrow\quad\dfrac{P\vdash y\quad P_2\{y/y_2\}\vdash y\quad P_3\{y/y_3\}\vdash y}{\nu y{:}A.(P\,|\,P_2\{y/y_2\}\,|\,P_3\{y/y_3\})\vdash\cdot}\textbf{3-cut}\qquad(\beta_!)$$

$$\dfrac{\dfrac{r\notin R_1\quad P\vdash\cdot}{P\vdash x}\textbf{!-neg-weaken}\quad\dfrac{r\in R_2\quad P_2\vdash y_2}{!x(y_2).P_2\vdash x}\textbf{!-pos}\quad\dfrac{r\in R_3\quad P_3\vdash y_3}{!x(y_3).P_3\vdash x}\textbf{!-pos}}{\nu x{:}!_r A.(P\,|\,!x(y_2).P_2\,|\,!x(y_3).P_3)\vdash\cdot}\textbf{3-cut}\quad\longrightarrow\quad\dfrac{\underline{P\vdash\cdot}}{P\vdash\cdot}\textbf{!-neg-weaken}\qquad(\beta_{!W})$$

$$\dfrac{\dfrac{r\notin R_1\quad P\vdash m,n}{P\{x/m,n\}\vdash x}\textbf{!-neg-contract}\quad\dfrac{r\in R_2\quad P_2\vdash y_2}{!x(y_2).P_2\vdash x}\textbf{!-pos}\quad\dfrac{r\in R_3\quad P_3\vdash y_3}{!x(y_3).P_3\vdash x}\textbf{!-pos}}{\nu x{:}!_r A.(P\{x/m,n\}\,|\,!x(y_2).P_2\,|\,!x(y_3).P_3)\vdash\cdot}\textbf{3-cut}$$

$$\longrightarrow\quad\dfrac{\underline{\dfrac{\dfrac{P\vdash m,n\quad\dfrac{r\in R_2\quad P_2'\vdash y_2}{!m(y_2).P_2'\vdash m}\textbf{!-pos}\quad\dfrac{r\in R_3\quad P_3'\vdash y_3}{!m(y_3).P_3'\vdash m}\textbf{!-pos}}{\nu m{:}!_r A.(P\,|\,!m(y_2).P_2'\,|\,!m(y_3).P_3')\vdash n}\textbf{3-cut}\quad\dfrac{r\in R_2\quad P_2''\vdash y_2}{!n(y_2).P_2''\vdash n}\textbf{!-pos}\quad\dfrac{r\in R_3\quad P_3''\vdash y_3}{!n(y_3).P_3''\vdash n}\textbf{!-pos}}{\nu n{:}!_r A.(\nu m{:}!_r A.(P\,|\,!m(y_2).P_2'\,|\,!m(y_3).P_3')\,|\,!n(y_2).P_2''\,|\,!n(y_3).P_3'')\vdash\cdot}\textbf{3-cut}}}{\nu n{:}!_r A.(\nu m{:}!_r A.(P\,|\,!m(y_2).P_2\,|\,!m(y_3).P_3)\,|\,!n(y_2).P_2\,|\,!n(y_3).P_3)\vdash\cdot}\textbf{!-neg-contract}\qquad(\beta_{!C})$$

Fig. 10: Principal Cut Reduction for 3-Party Sessions

Without loss of generality, we present here a 3-party principal cut reduction without proof terms. Assume endpoints are of role R_1, R_2, and R_3 respectively, in the same order as they occur in the parallel composition. $\overline{R}_1 \uplus \overline{R}_2 \uplus \overline{R}_3 = \overline{\varnothing}$ and $r \notin R_1$. Assume P's correspond to type A, and Q's correspond to type B. Double rule means multiple applications of that rule.