

Session Types for Message-Passing Concurrency

Dan Frumin and Jorge A. Pérez

University of Groningen, The Netherlands

www.rug.nl/fse/fc - d.frumin | j.a.perez [[at]] rug.nl



UNIFYING
C•RECTNESS FOR
C•MMUNICATING
S•FTWARE

IPA Formal Methods - June 2022

(Part 2, v1.1)

Outline

Context

Intuitionistic Linear Logic for Session Types

Asynchronous Communication

Servers and the ! Modality

Classical Linear Logic and Session Types

Deadlock-Freedom and Priorities for CP

Analysis of MPSTs using APCP

Concluding Remarks

This Course

A bird's eye view on session types for message-passing concurrency, in two parts:

1. Session types as protocol abstractions (Jorge):
Motivation, key ideas, binary and multiparty session types.
2. **Session types as a discipline for communicating processes** (Dan):
The Curry-Howard correspondence between linear logic and session types (aka “propositions as sessions”).

Keywords

Concurrency Theory, Message-Passing, Programming Languages, Verification

Keywords

Concurrency Theory, Message-Passing, Programming Languages, Verification

- **Type systems**

Well-typed programs can't go wrong (Milner)

- **Session types** for communication correctness

What and **when** should be sent through a channel

- **Process calculi**

The π -calculus treats **processes** like the λ -calculus treats **functions**

Keywords

Concurrency Theory, Message-Passing, Programming Languages, Verification

- **Type systems**

Well-typed programs can't go wrong (Milner)

- **Session types** for communication correctness

What and **when** should be sent through a channel

- **Process calculi**

The π -calculus treats **processes** like the λ -calculus treats **functions**

- **Propositions as sessions**

Tight connection between **logic** and **type theory**.

Propositions As Types

Intuitionistic logic propositions \leftrightarrow types describing data
Natural deduction derivations \leftrightarrow λ -calculus terms
Proof normalization reductions \leftrightarrow β -reductions

aka Curry-Howard correspondence, formulae-as-types, proofs-as-programs...

Propositions As Sessions

Linear	logic propositions	\leftrightarrow	types describing behavior (sessions)
Sequence calculus	derivations	\leftrightarrow	π -calculus processes
Cut	reductions	\leftrightarrow	communication between processes

Propositions As Sessions

Linear logic propositions	\leftrightarrow	types describing behavior (sessions)
Sequence calculus derivations	\leftrightarrow	π -calculus processes
Cut reductions	\leftrightarrow	communication between processes

Plan:

- ▶ Linear Logic & sequent calculus;
- ▶ Session types from intuitionistic linear logic (π DILL, Caires & Pfenning 2010);
- ▶ Asynchronous communication (DeYoung et al, 2012)
- ▶ Session types from classical linear logic (CP, Wadler 2012) ...
- ▶ ... with priorities (APCP, van den Heuvel & Pérez, 2021)

Outline

Context

Intuitionistic Linear Logic for Session Types
Asynchronous Communication
Servers and the ! Modality

Classical Linear Logic and Session Types
Deadlock-Freedom and Priorities for CP
Analysis of MPSTs using APCP

Concluding Remarks

Linear Logic: Propositions

$A \wedge B$

$A \rightarrow B$

both A and B are true

if A is true, then B is true

Linear Logic: Propositions

$A \wedge B$

$A \rightarrow B$

both A and B are true
if A is true, then B is true

$A \otimes B$

I have both A and B

Linear Logic: Propositions

$A \wedge B$

both A and B are true

$A \rightarrow B$

if A is true, then B is true

$A \otimes B$

I have both A and B

$A \multimap B$

if you give me A , then I can produce B

Linear Logic: Linearity

$$A \wedge (A \rightarrow B) \rightarrow B$$

$$A \otimes (A \multimap B) \multimap B$$

Linear Logic: Linearity

$$A \wedge (A \rightarrow B) \rightarrow B$$

$$A \rightarrow A \wedge A$$

$$A \wedge B \rightarrow A$$

$$A \otimes (A \multimap B) \multimap B$$

$$A \not\multimap A \otimes A$$

$$A \otimes B \not\multimap A$$

Linear Logic: Linearity

$$A \wedge (A \rightarrow B) \rightarrow B$$

$$A \rightarrow A \wedge A$$

$$A \wedge B \rightarrow A$$

$$A \wedge (A \rightarrow B) \rightarrow A \wedge B$$

$$A \otimes (A \multimap B) \multimap B$$

$$A \not\multimap A \otimes A$$

$$A \otimes B \not\multimap A$$

$$A \otimes (A \multimap B) \not\multimap A \otimes B$$

Linear Logic is *substructural*, resource-aware.

Linear Logic: Sequent Calculus

Sequent

$$A_1, \dots, A_n \vdash B,$$

interpreted as $A_1 \otimes \dots \otimes A_n \multimap B$.

Linear Logic: Sequent Calculus

Sequent

$$A_1, \dots, A_n \vdash B,$$

interpreted as $A_1 \otimes \dots \otimes A_n \multimap B$.

$$A \vdash A \qquad \frac{\Delta_1 \vdash A \quad \Delta_2, A \vdash B}{\Delta_1, \Delta_2 \vdash B} \qquad \frac{\Delta_1, A, B, \Delta_2 \vdash C}{\Delta_1, B, A, \Delta_2 \vdash C}$$

Each connective is “explained” in sequent calculus with a left rule, a right rule, and the interactions with the cut rule.

Linear Logic: Sequent Calculus

$$\frac{\Delta_1 \vdash A \quad \Delta_2 \vdash B}{\Delta_1, \Delta_2 \vdash A \otimes B}$$

$$\frac{\Delta, A, B \vdash C}{\Delta, A \otimes B \vdash C}$$

Linear Logic: Sequent Calculus

$$\frac{\Delta_1 \vdash A \quad \Delta_2 \vdash B}{\Delta_1, \Delta_2 \vdash A \otimes B}$$

$$\frac{\Delta, A, B \vdash C}{\Delta, A \otimes B \vdash C}$$

$$\frac{\Delta, A \vdash B}{\Delta \vdash A \multimap B}$$

$$\frac{\Delta_1 \vdash A \quad B, \Delta_2 \vdash C}{\Delta_1, A \multimap B, \Delta_2 \vdash C}$$

Linear Logic: Sequent Calculus

$$\frac{\Delta_1 \vdash A \quad \Delta_2 \vdash B}{\Delta_1, \Delta_2 \vdash A \otimes B}$$

$$\frac{\Delta, A, B \vdash C}{\Delta, A \otimes B \vdash C}$$

$$\frac{\Delta, A \vdash B}{\Delta \vdash A \multimap B}$$

$$\frac{\Delta_1 \vdash A \quad B, \Delta_2 \vdash C}{\Delta_1, A \multimap B, \Delta_2 \vdash C}$$

$$\emptyset \vdash 1$$

$$\frac{\Delta \vdash C}{\Delta, 1 \vdash C}$$

Propositions as Session Types

$!U; S$	output value of type U , continue as S	$U \otimes S$
$?U; S$	input value of type U , continue as S	$U \multimap S$
end	terminate the session	1

Propositions as Session Types

$!U; S$	output value of type U , continue as S	$U \otimes S$
$?U; S$	input value of type U , continue as S	$U \multimap S$
end	terminate the session	1

$$x_1 : A_1, \dots, x_n : A_n \vdash P :: z : C$$

Propositions as Session Types

$!U; S$	output value of type U , continue as S	$U \otimes S$
$?U; S$	input value of type U , continue as S	$U \multimap S$
end	terminate the session	1

$x_1 : A_1, \dots, x_n : A_n \vdash P :: z : C$

Process P offers session C on channel z

Propositions as Session Types

$!U; S$	output value of type U , continue as S	$U \otimes S$
$?U; S$	input value of type U , continue as S	$U \multimap S$
end	terminate the session	1

$$x_1 : A_1, \dots, x_n : A_n \vdash P :: z : C$$

Process P offers session C on channel z

Assuming sessions A_1, \dots, A_n on channels x_1, \dots, x_n

Propositions as Session Types: $A \otimes B$

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle. (P \mid Q)) :: x : A \otimes B}$$

$$\frac{\Delta, y : A, x : B \vdash R :: z : C}{\Delta, x : A \otimes B \vdash x(y).R :: z : C}$$

Propositions as Session Types: $A \otimes B$

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle. (P \mid Q)) :: x : A \otimes B}$$

Create a new communication channel y

$$\frac{\Delta_1, x : B \vdash R :: z : C}{\Delta_1 \otimes B \vdash x(y).R :: z : C}$$

Propositions as Session Types: $A \otimes B$

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x} \langle y \rangle . (P \mid Q)) :: x : A \otimes B}$$

Create a new communication channel y

$$\frac{\Delta, x : B \vdash R :: z : C}{\Delta \otimes B \vdash x(\bar{y}) R}$$

Send y over z

Propositions as Session Types: $A \otimes B$

Execute P and Q in parallel

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x} \langle y \rangle . (P \mid Q)) :: x : A \otimes B}$$

Create a new communication channel y

$$\frac{\Delta, x : B \vdash R :: z : C}{\Delta \otimes B \vdash x(\bar{z}) R}$$


Send y over z

Propositions as Session Types: $A \otimes B$

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle. (P \mid Q)) :: x : A \otimes B}$$

$$\frac{\Delta, y : A, x : B \vdash R :: z : C}{\Delta, x : A \otimes B \vdash x(y).R :: z : C}$$

Input y on x



Propositions as Session Types: $A \multimap B$

$$\frac{\Delta, y : A \vdash P :: z : B}{\Delta \vdash z(y).P :: z : A \multimap B}$$

$$\frac{\Delta_1 \vdash P :: y : A \quad x : B, \Delta_2 \vdash Q :: z : C}{\Delta_1, x : A \multimap B, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle.(P \mid Q)) :: z : C}$$

Propositions as Session Types: 1 and forwarders

$$\emptyset \vdash \bar{x}\langle \rangle :: x : 1$$

$$\frac{\Delta \vdash Q :: z : C}{\Delta, x : 1 \vdash x().Q :: z : C}$$

$$x : A \vdash [y \leftarrow x] :: y : A$$

Propositions as Session Types: 1 and forwarders

Close the channel x


$$\emptyset \vdash \bar{x} \langle \rangle :: x : 1$$

$$\frac{\Delta \vdash Q :: z : C}{\Delta, x : 1 \vdash x().Q :: z : C}$$

$$x : A \vdash [y \leftarrow x] :: y : A$$

Propositions as Session Types: 1 and forwarders

Close the channel x

$$\emptyset \vdash \bar{x} \langle \rangle :: x : 1$$

$$\frac{\Delta \vdash Q :: z : C}{\Delta, x : 1 \vdash x().Q :: z : C}$$

Wait for the channel
 x to close


$$[y \leftarrow x] :: y : A$$

Propositions as Session Types: 1 and forwarders

$$\emptyset \vdash \bar{x}\langle \rangle :: x : 1$$

$$\frac{\Delta \vdash Q :: z : C}{\Delta, x : 1 \vdash x().Q :: z : C}$$

$$x : A \vdash [y \leftarrow x] :: y : A$$



Forward all messages
between x and y

Propositions as Session Types: branching

$$\frac{\Delta \vdash P :: x : A \quad \Delta \vdash Q :: x : A}{\Delta \vdash x \triangleright \{\text{inl} : P, \text{inr} : Q\} :: x : A \& B}$$

$$\frac{\Delta, x : A \vdash Q :: z : C}{\Delta, x : A \& B \vdash x \triangleleft \text{inl}; Q :: z : C}$$

$$\frac{\Delta, x : B \vdash Q :: z : C}{\Delta, x : A \& B \vdash x \triangleleft \text{inr}; Q :: z : C}$$

Propositions as Session Types: branching

Branch on x : proceed either as P or Q

$$\frac{\Delta \vdash P :: x : A \quad \Delta \vdash Q :: x : A}{\Delta \vdash x \triangleright \{ \text{inl} : P, \text{inr} : Q \} :: x : A \& B}$$

$$\frac{\Delta, x : A \vdash Q :: z : C}{\Delta, x : A \& B \vdash x \triangleleft \text{inl}; Q :: z : C}$$

$$\frac{\Delta, x : B \vdash Q :: z : C}{\Delta, x : A \& B \vdash x \triangleleft \text{inr}; Q :: z : C}$$

Propositions as Session Types: branching

Branch on x : proceed either as P or Q

$$\frac{\Delta \vdash P :: x : A \quad \Delta \vdash Q :: x : A}{\Delta \vdash x \triangleright \{ \text{inl} : P, \text{inr} : Q \} :: x : A \& B}$$

$$\frac{\Delta, x : A \vdash Q :: z : C}{\Delta, x : A \& B \vdash x \triangleleft \text{inl}; Q :: z : C}$$

$$\frac{\Delta, x : B \vdash Q :: z : C}{\Delta, x : A \& B \vdash x \triangleleft \text{inr}; Q :: z : C}$$

Select either left or right session continuation

Propositions as Session Types: branching

$$\frac{\Delta \vdash P_i :: x : A_i}{\Delta \vdash x \triangleright \{l_1 : P_1, \dots, l_n : P_n\} :: x : \&\{l_i : A_i\}_{1 \leq i \leq n}}$$

$$\frac{\Delta, x : A_i \vdash Q :: z : C}{\Delta, x : \&\{l_i : A_i\} \vdash x \triangleleft l_i; Q :: z : C}$$

Propositions as Session Types: Selection

$$\frac{\Delta \vdash P :: x : A}{\Delta \vdash x \triangleleft \text{inl}; P :: x : A \oplus B}$$

$$\frac{\Delta \vdash Q :: x : B}{\Delta \vdash x \triangleleft \text{inr}; Q :: x : A \oplus B}$$

$$\frac{\Delta, x : A \vdash P :: z : C \quad \Delta, x : B \vdash Q :: z : C}{\Delta, x : A \oplus B \vdash x \triangleright \{\text{inl} : P; \text{inr} : Q\} :: z : C}$$

Propositions as Session Types: Cut

$$\frac{\Delta_1 \vdash P :: x : A \quad \Delta_2, x : A \vdash Q :: z : C}{\Delta_1, \Delta_2 \vdash (\nu x)(P \mid Q) :: z : C}$$

Semantics of Session-typed Processes

$$\begin{aligned}(\nu x)(x \triangleright \{l_i : P_i\}_{i \in I} \mid x \triangleleft l_i; R) &\longrightarrow (\nu x)(P_i \mid R) \\(\nu x)(x \triangleleft l_i; R \mid x \triangleright \{l_i : P_i\}_{i \in I}) &\longrightarrow (\nu x)(R \mid P_i) \\(\nu x)((\nu y)\bar{x}\langle y \rangle.(P_1 \mid P_2) \mid x(z).Q) &\longrightarrow (\nu x)(P_2 \mid (\nu y)(P_1 \mid Q\{y/z\})) \\(\nu x)([x \leftarrow y] \mid P) &\longrightarrow P\{y/x\} \\P \longrightarrow P' &\implies (\nu x)(P \mid Q) \longrightarrow (\nu x)(P' \mid Q) \\&\dots\end{aligned}$$

Closed under structural congruence, noted \equiv .

Semantics of Session-typed Processes

$$\begin{aligned}(\nu x)(x \triangleright \{l_i : P_i\}_{i \in I} \mid x \triangleleft l_i; R) &\longrightarrow (\nu x)(P_i \mid R) \\(\nu x)(x \triangleleft l_i; R \mid x \triangleright \{l_i : P_i\}_{i \in I}) &\longrightarrow (\nu x)(R \mid P_i) \\(\nu x)((\nu y)\bar{x}\langle y \rangle.(P_1 \mid P_2) \mid x(z).Q) &\longrightarrow (\nu x)(P_2 \mid (\nu y)(P_1 \mid Q\{y/z\})) \\(\nu x)([x \leftarrow y] \mid P) &\longrightarrow P\{y/x\} \\P \longrightarrow P' &\implies (\nu x)(P \mid Q) \longrightarrow (\nu x)(P' \mid Q) \\&\dots\end{aligned}$$

Closed under structural congruence, noted \equiv .

All the reductions remove a cut, possibly introducing new cuts in the process.

Process Reductions from Cut Reductions

$$\frac{\frac{\Delta_1 \vdash P :: x : A \quad \Delta_1 \vdash Q :: x : A}{\Delta_1 \vdash x \triangleright \{\text{inl} : P, \text{inr} : Q\} :: x : A \& B} \quad \frac{\Delta_2, x : A \vdash R :: z : C}{\Delta_2, x : A \& B \vdash x \triangleleft \text{inl}; R :: z : C}}{\Delta_1, \Delta_2 \vdash (\nu x) (x \triangleright \{\text{inl} : P, \text{inr} : Q\} \mid x \triangleleft \text{inl}; R) :: z : C}$$

→

Process Reductions from Cut Reductions

$$\frac{\frac{\Delta_1 \vdash P :: x : A \quad \Delta_1 \vdash Q :: x : A}{\Delta_1 \vdash x \triangleright \{ \text{inl} : P, \text{inr} : Q \} :: x : A \& B} \quad \frac{\Delta_2, x : A \vdash R :: z : C}{\Delta_2, x : A \& B \vdash x \triangleleft \text{inl}; R :: z : C}}{\Delta_1, \Delta_2 \vdash (\nu x) (x \triangleright \{ \text{inl} : P, \text{inr} : Q \} \mid x \triangleleft \text{inl}; R) :: z : C}$$

→

$$\frac{\Delta_1 \vdash P :: x : A \quad \Delta_2, x : A \vdash R :: z : C}{}$$

Process Reductions from Cut Reductions

$$\frac{\frac{\Delta_1 \vdash P :: x : A \quad \Delta_1 \vdash Q :: x : A}{\Delta_1 \vdash x \triangleright \{\text{inl} : P, \text{inr} : Q\} :: x : A \& B} \quad \frac{\Delta_2, x : A \vdash R :: z : C}{\Delta_2, x : A \& B \vdash x \triangleleft \text{inl}; R :: z : C}}{\Delta_1, \Delta_2 \vdash (\nu x)(x \triangleright \{\text{inl} : P, \text{inr} : Q\} \mid x \triangleleft \text{inl}; R) :: z : C}$$

→

$$\frac{\Delta_1 \vdash P :: x : A \quad \Delta_2, x : A \vdash R :: z : C}{\Delta_1, \Delta_2 \vdash (\nu x)(P \mid R)}$$

Process Reductions from Cut Reductions

$$\frac{\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) \bar{x}(y).(P \mid Q) :: x : A \otimes B} \quad \frac{\Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_3, x : A \otimes B \vdash x(y).R :: z : C}}{\Delta_1, \Delta_2, \Delta_3 \vdash (\nu x) \left((\nu y) \bar{x}(y).(P \mid Q) \mid x(y).R \right) :: z : C}$$

→

Process Reductions from Cut Reductions

$$\frac{\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) \bar{x} \langle y \rangle . (P \mid Q) :: x : A \otimes B} \quad \frac{\Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_3, x : A \otimes B \vdash x(y) . R :: z : C}}{\Delta_1, \Delta_2, \Delta_3 \vdash (\nu x) \left((\nu y) \bar{x} \langle y \rangle . (P \mid Q) \mid x(y) . R \right) :: z : C}$$

→

Process Reductions from Cut Reductions

$$\frac{\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) \bar{x}(y).(P \mid Q) :: x : A \otimes B} \quad \frac{\Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_3, x : A \otimes B \vdash x(y).R :: z : C}}{\Delta_1, \Delta_2, \Delta_3 \vdash (\nu x) ((\nu y) \bar{x}(y).(P \mid Q) \mid x(y).R) :: z : C} \longrightarrow \frac{\Delta_1 \vdash P :: y : A \quad \Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_1, x : B, \Delta_3 \vdash (\nu y)(P \mid R) :: z : C}$$

Process Reductions from Cut Reductions

$$\frac{\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) \bar{x} \langle y \rangle . (P \mid Q) :: x : A \otimes B} \quad \frac{\Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_3, x : A \otimes B \vdash x(y) . R :: z : C}}{\Delta_1, \Delta_2, \Delta_3 \vdash (\nu x) \left((\nu y) \bar{x} \langle y \rangle . (P \mid Q) \mid x(y) . R \right) :: z : C}$$

$$\longrightarrow$$

$$\frac{\Delta_2 \vdash Q :: x : B \quad \frac{\Delta_1 \vdash P :: y : A \quad \Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_1, x : B, \Delta_3 \vdash (\nu y) (P \mid R) :: z : C}}{\Delta_1, \Delta_2, \Delta_3 \vdash (\nu x) \left(Q \mid (\nu y) (P \mid R) \right) :: z : C}$$

Properties of π DILL

Theorem (Subject reduction)

If $\Delta \vdash P :: z : C$ and $P \longrightarrow Q$ then $\Delta \vdash Q :: z : C$

Properties of π DILL

Theorem (Subject reduction)

If $\Delta \vdash P :: z : C$ and $P \longrightarrow Q$ then $\Delta \vdash Q :: z : C$

Theorem (Deadlock-freedom)

If $\Delta \vdash P :: z : C$ and $P \not\longrightarrow -$ then P is blocked on either z or a channel from Δ

Corollary

If $\emptyset \vdash P :: z : 1$ and $P \not\longrightarrow -$ then $P = \bar{z}\langle \rangle$.

Asynchronous Output

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle . (P \mid Q)) :: x : A \otimes B}$$

Asynchronous Output

Synchronize on
channel x , then
continue with Q

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle . (P \mid Q)) :: x : A \otimes B}$$

Asynchronous Output

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle . (P \mid Q)) :: x : A \otimes B}$$

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle \mid P \mid Q) :: x : A \otimes B}$$

Asynchronous Output

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle . (P \mid Q)) :: x : A \otimes B}$$

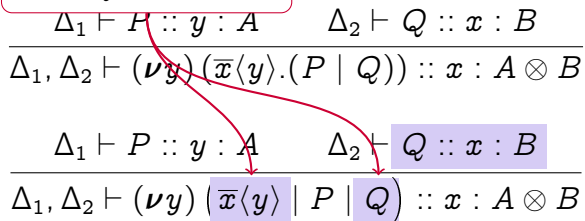
$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle \mid P \mid Q) :: x : A \otimes B}$$

For example: $(\nu y) \bar{x}\langle y \rangle . (P \mid z(k) . Q)$ vs $(\nu y) (\bar{x}\langle y \rangle \mid P \mid z(k) . Q)$

Asynchronous Output

Possible interference between $\bar{x}\langle y \rangle$ and actions on x from Q

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle . (P \mid Q)) :: x : A \otimes B}$$

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle \mid P \mid Q) :: x : A \otimes B}$$


Asynchronous Output

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle . (P \mid Q)) :: x : A \otimes B}$$

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\nu y) (\bar{x}\langle y \rangle \mid P \mid Q) :: x : A \otimes B}$$

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x' : B}{\Delta_1, \Delta_2 \vdash (\nu y)(\nu x') (\bar{x}\langle y, x' \rangle \mid P \mid Q) :: x : A \otimes B}$$

Asynchronous Output (& Matching Input)

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x' : B}{\Delta_1, \Delta_2 \vdash (\nu y)(\nu x')(\bar{x}\langle y, x' \rangle \mid P \mid Q) :: x : A \otimes B}$$

$$\frac{\Delta, y : A, x' : B \vdash R :: z : C}{\Delta, x : A \otimes B \vdash x(y, x').R :: z : C}$$

$$(\nu x)\left((\nu y)(\nu x')(\bar{x}\langle y, x' \rangle \mid P \mid Q) \mid x(y, x').R\right) \longrightarrow$$

$$(\nu y)\left(P \mid (\nu x')(Q \mid R)\right)$$

Asynchronous Output (& Matching Input)

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2 \vdash Q :: x' : B}{\Delta_1, \Delta_2 \vdash (\nu y)(\nu x') (\bar{x}\langle y, x' \rangle \mid P \mid Q) :: x : A \otimes B}$$

$$\frac{\Delta, y : A, x' : B \vdash R :: z : C}{\Delta, x : A \otimes B \vdash x(y, x').R :: z : C}$$

$$\frac{\Delta, y : A \vdash R :: x' : B}{\Delta \vdash x(y, x').P :: x : A \multimap B}$$

$$\frac{\Delta_1 \vdash P :: y : A \quad \Delta_2, x' : B \vdash Q :: z : C}{\Delta_1, x : A \multimap B, \Delta_2 \vdash (\nu y)(\nu x') (\bar{x}\langle y, x' \rangle \mid P \mid Q) :: z : C}$$

Unrestricted Resources

- ▶ So far we have talked only about *linear* resources

Unrestricted Resources

- ▶ So far we have talked only about *linear* resources
- ▶ A session type A describes a finite, linear session

Unrestricted Resources

- ▶ So far we have talked only about *linear* resources
- ▶ A session type A describes a finite, linear session
- ▶ Composition on disjoint sessions:

$$\frac{\Delta_1 \vdash P :: x : A \quad x : A, \Delta_2 \vdash Q :: z : C}{\Delta_1, \Delta_2 \vdash (\nu x)(P \mid Q) :: z : C}$$

Unrestricted Resources

- ▶ So far we have talked only about *linear* resources
- ▶ A session type A describes a finite, linear session
- ▶ Composition on disjoint sessions:

$$\frac{\Delta_1 \vdash P :: x : A \quad x : A, \Delta_2 \vdash Q :: z : C}{\Delta_1, \Delta_2 \vdash (\nu x)(P \mid Q) :: z : C}$$

Divide all the resources between P and Q

Unrestricted Resources

- ▶ So far we have talked only about *linear* resources
- ▶ A session type A describes a finite, linear session
- ▶ Composition on disjoint sessions:

$$\frac{\Delta_1 \vdash P :: x : A \quad x : A, \Delta_2 \vdash Q :: z : C}{\Delta_1, \Delta_2 \vdash (\nu x)(P \mid Q) :: z : C}$$

- ▶ Question: how to introduce *unrestricted*/non-linear sessions?

The ! Modality

- ▶ $!A$ - unlimited copies of the session A

The ! Modality

- ▶ $!A$ - unlimited copies of the session A
- ▶ $A_1, \dots, A_k; B_1, \dots, B_n \vdash C$ stands for $!A_1 \otimes \dots \otimes !A_k \otimes B_1 \otimes \dots \otimes B_n \multimap C$

The ! Modality

- ▶ $!A$ - unlimited copies of the session A
- ▶ $A_1, \dots, A_k; B_1, \dots, B_n \vdash C$ stands for $!A_1 \otimes \dots \otimes !A_k \otimes B_1 \otimes \dots \otimes B_n \multimap C$
- ▶ Unrestricted resources are non-linear and can be shared:

$$\frac{\Gamma; \Delta_1 \vdash P :: x : A \quad \Gamma; x : A, \Delta_2 \vdash Q :: z : C}{\Gamma; \Delta_1, \Delta_2 \vdash (\nu x)(P \mid Q) :: z : C}$$

$$\frac{x : A, y : A, \Gamma; \Delta \vdash P :: z : C}{x : A, \Gamma; \Delta \vdash P\{x/y\} :: z : C}$$

The ! Modality

- ▶ $!A$ - unlimited copies of the session A
- ▶ $A_1, \dots, A_k; B_1, \dots, B_n \vdash C$ stands for $!A_1 \otimes \dots \otimes !A_k \otimes B_1 \otimes \dots \otimes B_n \multimap C$
- ▶ Unrestricted resources are non-linear and can be shared:

$$\frac{\Gamma; \Delta_1 \vdash P :: x : A \quad \Gamma; x : A, \Delta_2 \vdash Q :: z : C}{\Gamma; \Delta_1, \Delta_2 \vdash (\nu x)(P \mid Q) :: z : C}$$

$$\frac{x : A, y : A, \Gamma; \Delta \vdash P :: z : C}{x : A, \Gamma; \Delta \vdash P\{x/y\} :: z : C}$$

The ! Modality

$$\frac{\Gamma; \emptyset \vdash P :: y : A}{\Gamma; \emptyset \vdash !u(y).P :: u : !A}$$

$$\frac{u : A, \Gamma; y : A, \Delta \vdash Q :: z : C}{u : A, \Gamma; \Delta \vdash (\nu y)(\bar{u}\langle y \rangle \mid Q) :: z : C}$$

$$\frac{\Gamma; x : !A, \Delta \vdash P :: z : C}{u : A, \Gamma; \Delta \vdash P\{u/x\} :: z : C}$$

The ! Modality

$$\frac{\Gamma; \emptyset \vdash P :: y : A}{\Gamma; \emptyset \vdash !u(y).P :: u : !A}$$

$$\frac{u : A, \Gamma; y : A, \Delta \vdash Q :: z : C}{u : A, \Gamma; \Delta \vdash (\nu y)(\bar{u}\langle y \rangle \mid Q) :: z : C}$$

$$\frac{\Gamma; x : !A, \Delta \vdash P :: z : C}{u : A, \Gamma; \Delta \vdash P\{u/x\} :: z : C}$$

$$(\nu u)(!u(y).P \mid (\nu y)(\bar{u}\langle y \rangle \mid Q)) \longrightarrow (\nu u)(!u(y).P \mid (\nu y)(P \mid Q))$$

Example: A Two-Buyer Protocol



Alice and Bob cooperate in buying a book from Seller:

1. Alice sends a book title to Seller, who sends a quote back.
2. Alice checks whether Bob can contribute in buying the book.
3. Alice uses the answer from Bob to interact with Seller, either:
 - a) completing the payment and arranging delivery details
 - b) canceling the transaction
4. In case 3(a) Alice contacts Bob to get his address, and forwards it to Seller.
- 4'. In case 3(b) Alice is in charge of gracefully concluding the conversation.

Example: A Two-Buyer Protocol

$$\text{sellerProto} = \text{BookId} \multimap N \otimes \& \left\{ \begin{array}{l} \text{buy} : \quad \text{PayId} \multimap \text{Address} \multimap 1 \\ \text{cancel} : 1 \end{array} \right\}$$

$$\text{bobProto} = N \multimap \oplus \{ \text{close} : 1; \text{share} : \text{Address} \otimes 1 \}$$

Example: A Two-Buyer Protocol

$$sellerProto = BookId \multimap N \otimes \& \left\{ \begin{array}{l} buy : \quad PayId \multimap Address \multimap 1 \\ cancel : 1 \end{array} \right\}$$

$$bobProto = N \multimap \oplus \{close : 1; \text{ share} : Address \otimes 1\}$$

$$\emptyset; \emptyset \vdash \text{Seller} :: u : !sellerProto$$

$$\emptyset; \emptyset \vdash \text{Bob} :: b : bobProto$$

$$p : PayId, book : BookId, s : sellerProto, b : bobProto \vdash \text{Alice} :: z : 1$$

$$p : PayId, book : BookId \vdash (\nu u)(\text{Seller} \mid (\nu b)(\text{Bob} \mid (\nu s)\bar{u}\langle s \rangle.\text{Alice})) :: z : 1$$

Example: A Two-Buyer Protocol

$$sellerProto = BookId \multimap N \otimes \& \left\{ \begin{array}{l} buy : \quad PayId \multimap Address \multimap 1 \\ cancel : 1 \end{array} \right\}$$

$$bobProto = N \multimap \oplus \{close : 1; \text{ share} : Address \otimes 1\}$$

$$\bar{s}\langle book \rangle . s(\text{price}) . \bar{b}\langle price \rangle . b \triangleright \left\{ \begin{array}{l} close : s \triangleleft cancel; b().s() \\ share : b(addr).s \triangleleft buy; \bar{s}\langle p \rangle . \bar{s}\langle addr \rangle . b().s() \end{array} \right\}$$

Outline

Context

Intuitionistic Linear Logic for Session Types

Asynchronous Communication

Servers and the ! Modality

Classical Linear Logic and Session Types

Deadlock-Freedom and Priorities for CP

Analysis of MPSTs using APCP

Concluding Remarks

π DILL Processes

Theorem (Deadlock-freedom)

If $\Delta \vdash P :: z : C$ and $P \not\rightarrow -$ then P is blocked on either z or a channel from Δ

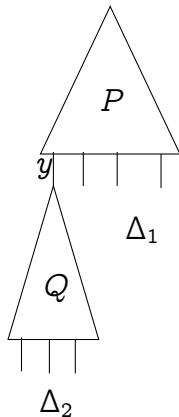
π DILL Processes

Theorem (Deadlock-freedom)

If $\Delta \vdash P :: z : C$ and $P \not\rightarrow -$ then P is blocked on either z or a channel from Δ

$$\frac{\Delta_2 \vdash Q :: y : A \quad \Delta_1, y : A \vdash P :: z : C}{\Delta_1, \Delta_2 \vdash (\nu x)(P \mid Q) :: z : C}$$

π DILL processes are organized in a tree-like topology



Classical Linear Logic

ILL vs CLL:

$1, \otimes, \multimap, \oplus, \&$

$A_1, \dots, A_n \vdash A$

$x_1 : A_1, \dots, x_n : A_n \vdash P :: z : A$

$1, \perp, \otimes, \wp, \oplus, \&$

$\vdash B_1, \dots, B_k$

$P \vdash x_1 : B_1, \dots, x_k : B_k$

implicit duality of connectives

explicit notion of duality used for composition

Duality

ILL: \otimes is “somewhat” dual to \multimap , CLL brings this duality closer to the one between \oplus and $\&$.

Duality

ILL: \otimes is “somewhat” dual to \multimap , CLL brings this duality closer to the one between \oplus and $\&$.

$$(A \oplus B)^\perp = A^\perp \& B^\perp$$

$$(A \& B)^\perp = A^\perp \oplus B^\perp$$

$$(A \otimes B)^\perp = A^\perp \wp B^\perp$$

$$(A \wp B)^\perp = A^\perp \otimes B^\perp$$

$$1^\perp = \perp$$

$$\perp^\perp = 1$$

Duality

ILL: \otimes is “somewhat” dual to \multimap , CLL brings this duality closer to the one between \oplus and $\&$.

$$(A \oplus B)^\perp = A^\perp \& B^\perp$$

$$(A \& B)^\perp = A^\perp \oplus B^\perp$$

$$(A \otimes B)^\perp = A^\perp \wp B^\perp$$

$$(A \wp B)^\perp = A^\perp \otimes B^\perp$$

$$1^\perp = \perp$$

$$\perp^\perp = 1$$

$$A = A^{\perp\perp}, A \multimap B = A^\perp \wp B$$

Composition Through Duality

$$\frac{P \vdash \Delta_1, x : A \quad Q \vdash \Delta_2, y : A^\perp}{(\nu xy)(P \mid Q) \vdash \Delta_1, \Delta_2}$$

Composition Through Duality

$$\frac{P \vdash \Delta_1, x : A \quad Q \vdash \Delta_2, y : A^\perp}{(\nu xy)(P \mid Q) \vdash \Delta_1, \Delta_2}$$

Create a single channel with endpoints x and y

Composition Through Duality

$$\frac{P \vdash \Delta_1, x : A \quad Q \vdash \Delta_2, y : A^\perp}{(\nu xy)(P \mid Q) \vdash \Delta_1, \Delta_2}$$

$$\frac{P \vdash \Delta_1 \quad Q \vdash \Delta_2}{P \mid Q \vdash \Delta_1, \Delta_2}$$

$$\frac{P \vdash \Delta, x : A, y : A^\perp}{(\nu xy)P \vdash \Delta_1, \Delta_2}$$

$$[x \leftarrow y] \vdash y : A, x : A^\perp$$

Interpretation of the Connectives

$$\bar{x}\langle y, x' \rangle \vdash x : A \otimes B, y : A^\perp, x' : B^\perp$$

$$\frac{P \vdash \Delta, y : A, x' : B}{x(y, x').P \vdash \Delta, x : A \wp B}$$

Interpretation of the Connectives

$$\bar{x}\langle y, x' \rangle \vdash x : A \otimes B, y : A^\perp, x' : B^\perp$$

$$\frac{P \vdash \Delta, y : A, x' : B}{x(y, x').P \vdash \Delta, x : A \wp B}$$

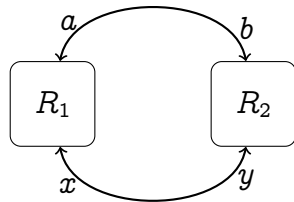
$$\frac{j \in I}{\bar{x}\langle x' \rangle \triangleleft 1_j \vdash x : \oplus\{1_i : A_i\}, x' : A_j^\perp}$$

$$\frac{\forall i \in I. P_i \vdash \Delta, x' : A_i}{x(x') \triangleright \{1_i : P_i\}_{i \in I} \vdash \Delta, \&\{1_i : A_i\}_{i \in I}}$$

Example Cyclic Process

$$R_1 = a(v, a').x(w, x').R'_1$$

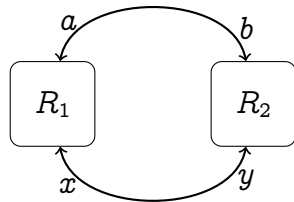
$$R_2 = (\nu y' y_1)(\nu b' b_1)(\bar{y}\langle w, y'\rangle \mid \bar{b}\langle v, b'\rangle \mid R'_2)$$



Example Cyclic Process

$$R_1 = a(v, a').x(w, x').R'_1$$

$$R_2 = (\nu y' y_1)(\nu b' b_1)(\bar{y}\langle w, y'\rangle \mid \bar{b}\langle v, b'\rangle \mid R'_2)$$



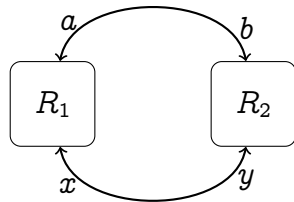
$$\frac{R'_1 \vdash v : A, a' : C, w : B, x' : D}{R_1 \vdash a : A \wp C, x : B \wp D}$$

$$\frac{R'_2 \vdash y_1 : D^\perp, b_1 : C^\perp}{R_2 \vdash b : A^\perp \otimes C^\perp, y : B^\perp \otimes D^\perp, w : A, v : B}$$

Example Cyclic Process

$$R_1 = a(v, a').x(w, x').R_1'$$

$$R_2 = (\nu y' y_1)(\nu b' b_1)(\bar{y}\langle w, y'\rangle \mid \bar{b}\langle v, b'\rangle \mid R_2')$$



$$\frac{R_1' \vdash v : A, a' : C, w : B, x' : D}{R_1 \vdash a : A \wp C, x : B \wp D}$$

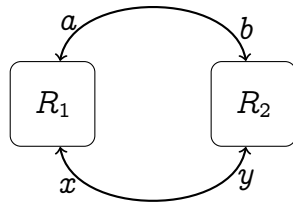
$$\frac{R_2' \vdash y_1 : D^\perp, b_1 : C^\perp}{R_2 \vdash b : A^\perp \otimes C^\perp, y : B^\perp \otimes D^\perp, w : A, v : B}$$

$$(\nu ab)(\nu xy)(R_1 \mid R_2) \longrightarrow$$

Example Cyclic Process

$$R_1 = a(v, a').x(w, x').R'_1$$

$$R_2 = (\nu y' y_1)(\nu b' b_1)(\bar{y}\langle w, y'\rangle \mid \bar{b}\langle v, b'\rangle \mid R'_2)$$



$$\frac{R'_1 \vdash v : A, a' : C, w : B, x' : D}{R_1 \vdash a : A \wp C, x : B \wp D}$$

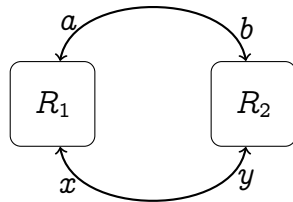
$$\frac{R'_2 \vdash y_1 : D^\perp, b_1 : C^\perp}{R_2 \vdash b : A^\perp \otimes C^\perp, y : B^\perp \otimes D^\perp, w : A, v : B}$$

$$(\nu ab)(\nu xy)(R_1 \mid R_2) \longrightarrow (\nu xy)(\nu b' b_1)(x(w, x').R'_1\{b'/a'\} \mid (\nu y' y_1)(\bar{y}\langle w, y_1\rangle \mid R'_2))$$

Example Cyclic Process

$$R_1 = a(v, a') \ x(w, x').R'_1$$

$$R_2 = (\nu y' y_1)(\nu b' b_1) (\bar{y}\langle w, y' \rangle \mid \bar{b}\langle v, b' \rangle \mid R'_2)$$



$$\frac{R'_1 \vdash v : A, a' : C, w : B, x' : D}{R_1 \vdash a : A \wp C, x : B \wp D}$$

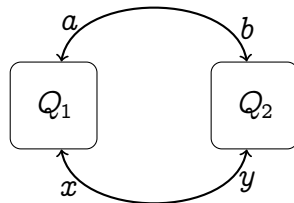
$$\frac{R'_2 \vdash y_1 : D^\perp, b_1 : C^\perp}{R_2 \vdash b : A^\perp \otimes C^\perp, y : B^\perp \otimes D^\perp, w : A, v : B}$$

$$\begin{aligned} (\nu ab)(\nu xy)(R_1 \mid R_2) &\longrightarrow (\nu xy)(\nu b' b_1) (x(w, x').R'_1\{b'/a'\} \mid (\nu y' y_1) (\bar{y}\langle w, y_1 \rangle \mid R'_2)) \\ &\longrightarrow (\nu b' b_1)(\nu y' y_1) (R'_1\{b'/a'\}\{y'/x'\} \mid R'_2) \end{aligned}$$

Example Deadlock Process

$$Q_1 = a(v, a').(\nu x'x_1)(\bar{x}\langle w, x'\rangle \mid Q'_1)$$

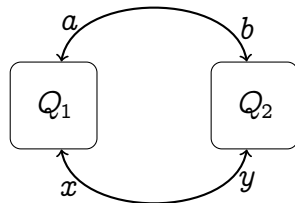
$$Q_2 = y(w, y').(\nu b'b_1)(\bar{b}\langle v, b'\rangle \mid Q'_2)$$



Example Deadlock Process

$$Q_1 = a(v, a').(\nu x'x_1)(\bar{x}\langle w, x'\rangle \mid Q'_1)$$

$$Q_2 = y(w, y').(\nu b'b_1)(\bar{b}\langle v, b'\rangle \mid Q'_2)$$



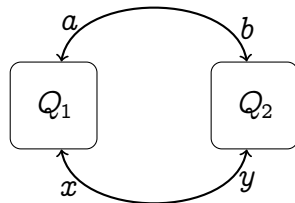
$$\frac{Q'_1 \vdash v : A, a' : C, x_1 : D^\perp}{Q_1 \vdash a : A \wp C, x : B^\perp \otimes D^\perp}$$

$$\frac{Q'_2 \vdash w : B, y' : D, b_1 : C^\perp}{Q_2 \vdash b : A^\perp \otimes C^\perp, y : B \wp D, v : A}$$

Example Deadlock Process

$$Q_1 = a(v, a').(\nu x'x_1)(\bar{x}\langle w, x'\rangle \mid Q'_1)$$

$$Q_2 = y(w, y').(\nu b'b_1)(\bar{b}\langle v, b'\rangle \mid Q'_2)$$



$$\frac{Q'_1 \vdash v : A, a' : C, x_1 : D^\perp}{Q_1 \vdash a : A \wp C, x : B^\perp \otimes D^\perp}$$

$$\frac{Q'_2 \vdash w : B, y' : D, b_1 : C^\perp}{Q_2 \vdash b : A^\perp \otimes C^\perp, y : B \wp D, v : A}$$

$$(\nu ab)(\nu xy)(Q_1 \mid Q_2) \not\rightarrow$$

Priorities

$$A, B ::= A \otimes^o B \mid A \wp^o B \mid \dots \qquad o \in \mathbb{N} \cup \{\infty\}$$

Priorities

$$A, B ::= A \otimes^{\circ} B \mid A \wp^{\circ} B \mid \dots \qquad \circ \in \mathbb{N} \cup \{\infty\}$$

$$(A \otimes^{\circ} B)^{\perp} = A^{\perp} \wp^{\circ} B^{\perp} \dots$$

Priorities

$$A, B ::= A \otimes^{\circ} B \mid A \wp^{\circ} B \mid \dots \quad \circ \in \mathbb{N} \cup \{\infty\}$$

$$(A \otimes^{\circ} B)^{\perp} = A^{\perp} \wp^{\circ} B^{\perp} \dots$$

$$\frac{P \vdash \Delta, y : A, x' : B \quad \circ < pr(\Delta)}{x(y, x').P \vdash \Delta, x : A \wp^{\circ} B}$$

Example with Priorities

$$\frac{\frac{Q'_1 \vdash v : A, a' : C, x_1 : D^\perp}{(\nu x' x_1)(\bar{x}\langle w, x' \rangle \mid R') \vdash v : A, a' : C, x : B^\perp \otimes^{o_2} D^\perp} \quad o_1 < o_2}{a(v, a').(\nu x' x_1)(\bar{x}\langle w, x' \rangle \mid Q'_1) \vdash a : A \wp^{o_1} C, x : B^\perp \otimes^{o_2} D^\perp}$$

Example with Priorities

$$\frac{
 \frac{
 Q'_1 \vdash v : A, a' : C, x_1 : D^\perp
 }{
 (\nu x' x_1)(\bar{x}\langle w, x' \rangle \mid R') \vdash v : A, a' : C, x : B^\perp \otimes^{o_2} D^\perp \quad o_1 < o_2
 }
 }{
 a(v, a').(\nu x' x_1)(\bar{x}\langle w, x' \rangle \mid Q'_1) \vdash a : A \wp^{o_1} C, x : B^\perp \otimes^{o_2} D^\perp
 }$$

$$\frac{
 \frac{
 Q'_2 \vdash w : B, y' : D, b_1 : C^\perp
 }{
 (\nu b' b_1)(\bar{b}\langle v, b' \rangle \mid Q') \vdash w : B, y' : D, b : A^\perp \otimes^{o_1} C^\perp \quad o_2 < o_1
 }
 }{
 y(w, y').(\nu b' b_1)(\bar{b}\langle v, b' \rangle \mid Q'_2) \vdash b : A^\perp \otimes^{o_1} C^\perp, y : B \wp^{o_2} D, v : A
 }$$

Properties of APCP

Theorem (Deadlock-freedom)

If $P \vdash \emptyset$ and $P \not\rightarrow -$ then P does not contain actions or prefixes.

Application: Analysis of MPSTs

Even though APCP features **binary** session types, it can serve as a language for implementing and analyzing **multiparty** protocols.

Application: Analysis of MPSTs

Consider the following global type:

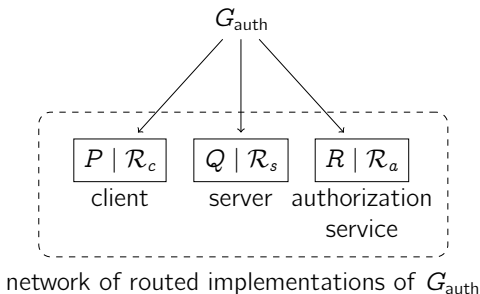
$$G_{\text{auth}} = \mu X . s \rightarrow c \left\{ \begin{array}{l} \text{login} . c \rightarrow a : \text{passwd} . a \rightarrow s : \text{auth} . X, \\ \text{quit} . c \rightarrow a : \text{quit} . \text{end} \end{array} \right\}$$

Application: Analysis of MPSTs

Consider the following global type:

$$G_{\text{auth}} = \mu X . s \rightarrow c \left\{ \begin{array}{l} \text{login} . c \rightarrow a : \text{passwd} . a \rightarrow s : \text{auth} . X, \\ \text{quit} . c \rightarrow a : \text{quit} . \text{end} \end{array} \right\}$$

We have devised a **router-based analysis**:

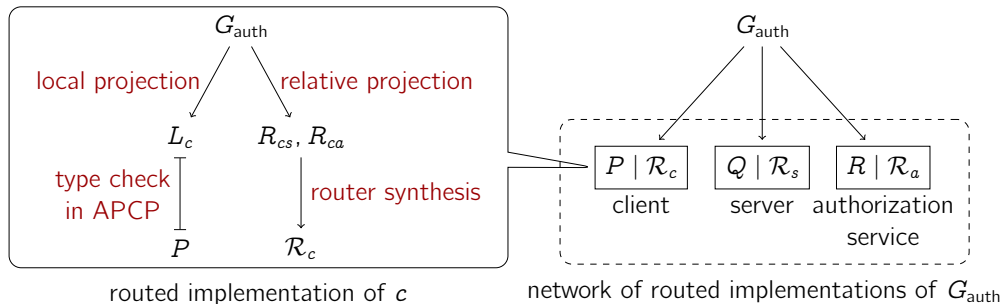


Application: Analysis of MPSTs

Consider the following global type:

$$G_{\text{auth}} = \mu X . s \rightarrow c \left\{ \begin{array}{l} \text{login} . c \rightarrow a : \text{passwd} . a \rightarrow s : \text{auth} . X, \\ \text{quit} . c \rightarrow a : \text{quit} . \text{end} \end{array} \right\}$$

We have devised a **router-based analysis**:



Outline

Context

Intuitionistic Linear Logic for Session Types

Asynchronous Communication

Servers and the ! Modality

Classical Linear Logic and Session Types

Deadlock-Freedom and Priorities for CP

Analysis of MPSTs using APCP

Concluding Remarks

Summary

We first overviewed key notions underlying **binary session types** and **multiparty session types** (without committing to a process model)

We then discussed concurrent interpretations of various flavors of linear logic that

- Clarify the logical foundations of binary session types, in the spirit of the **Curry-Howard isomorphism**
- Identified a number of session-types systems based on logical foundations, in which processes enjoy fidelity, safety, and progress
- Identifies a class of π -calculus processes which enjoy fidelity, safety, and progress

Further Topics

Research on session types has long addressed topics not mentioned here, including:

- Different **liveness properties** (progress, deadlock-freedom, and lock-freedom)
- Synchronous / asynchronous **communication disciplines**
- Connections between session types and **automata theory**
- **Security properties** (secure information flow, access control)
- Session types into **object-oriented, functional, and imperative** and languages
- **Behavioral equivalences** as informed by session types
- Session types and models of **exceptions, reversibility, run-time monitoring** and **adaptation**