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 CORRECTNESS FOR COMMUNICATING SOFTWARE

IPA Formal Methods - June 2022
(Part 2, v1.1)

Outline

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 (πDILL, 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

Linear Logic: Propositions

$A \wedge B$	both A and B are true
$A \to B$	if A is true, then B is true

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

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

$$A \wedge (A \to B) \to B \qquad\qquad A \otimes (A \multimap B) \multimap B$$

Linear Logic: Linearity

$$A \wedge (A \rightarrow B) \rightarrow B \qquad A \otimes (A \multimap B) \multimap B$$

$$A \rightarrow A \wedge A \qquad A \not\multimap A \otimes A$$

$$A \wedge B \rightarrow A \qquad A \otimes B \not\multimap A$$

Linear Logic: Linearity

$$A \wedge (A \to B) \to B \qquad A \otimes (A \multimap B) \multimap B$$

$$A \to A \wedge A \qquad A \not\multimap A \otimes A$$

$$A \wedge B \to A \qquad A \otimes B \not\multimap A$$

$$A \wedge (A \to B) \to A \wedge B \qquad A \otimes (A \multimap B) \not\multimap A \otimes B$$

Linear Logic is *substructural*, resource-aware.

Linear Logic: Sequent Calculus

Sequent

$$A_1, \ldots, A_n \vdash B,$$

interpreted as $A_1 \otimes \ldots \otimes A_n \vdash B$.

Linear Logic: Sequent Calculus

Sequent

$$A_1, \ldots, A_n \vdash B,$$

interpreted as $A_1 \otimes \ldots \otimes A_n \vdash B$.

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

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

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

Linear Logic: Sequent Calculus

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

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

Linear Logic: Sequent Calculus

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

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

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

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	$U \multimap S$
$\mathtt{end}$	**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	$U \multimap S$
$\mathtt{end}$	**terminate** the session	1

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

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	$U \multimap S$
$\mathtt{end}$	**terminate** the session	1

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

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	$U \multimap S$
$\mathtt{end}$	**terminate** the session	1

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

Process P offers session C on channel z

Assuming sessions $A_1, \ldots, A_n$ on channels $x_1, \ldots, x_n$

Propositions as Session Types: $A \otimes B$

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

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

Propositions as Session Types: $A \otimes B$

$$\frac{\Delta_1 \vdash P :: y : A \qquad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash {\color{red}(\boldsymbol{\nu} y)}\, (\overline{x}\langle y \rangle.(P \mid Q)) :: x : A \otimes B}$$

Create a new communication channel y

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

Propositions as Session Types: $A \otimes B$

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

$, x : B \vdash R :: z : C$

$\otimes B \vdash x($

Create a new communication channel y

Send y over z

Propositions as Session Types: $A \otimes B$

Execute P and Q in parallel

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

$$\frac{\ldots, x : B \vdash R :: z : C}{\ldots \otimes B \vdash x(\ldots$$

Create a new communication channel y

Send y over z

Propositions as Session Types: $A \otimes B$

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

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

Input y on x

Propositions as Session Types: $A \multimap B$

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

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

Propositions as Session Types: 1 and forwarders

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

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

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

Propositions as Session Types: 1 and forwarders

Close the channel x

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

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

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

Propositions as Session Types: 1 and forwarders

Close the channel x

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

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

Wait for the channel x to close

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

Propositions as Session Types: 1 and forwarders

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

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

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

Forward all messages between x and y

Propositions as Session Types: branching

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

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

Propositions as Session Types: branching

Branch on x: proceed either as P or Q

$$\frac{\Delta \vdash P :: x : A \qquad \Delta \vdash Q :: x : A}{\Delta \vdash x \triangleright \{\mathtt{inl} : P, \mathtt{inr} : Q\} :: x : A \mathbin{\&} B}$$

$$\frac{\Delta, x : A \vdash Q :: z : C}{\Delta, x : A \mathbin{\&} B \vdash x \triangleleft \mathtt{inl}; Q :: z : C} \qquad \frac{\Delta, x : B \vdash Q :: z : C}{\Delta, x : A \mathbin{\&} B \vdash x \triangleleft \mathtt{inr}; Q :: z : C}$$

Propositions as Session Types: branching

Branch on x: proceed either as P or Q

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

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

Select either left or right session continuation

Propositions as Session Types: branching

$$\frac{\Delta \vdash P_i :: x : A_i}{\Delta \vdash x \triangleright \{\mathtt{l}_1 : P_1, \ldots, \mathtt{l}_n : P_n\} :: x : \&\{\mathtt{l}_i : A_i\}_{1 \leq i \leq n}}$$

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

Propositions as Session Types: Selection

$$\frac{\Delta \vdash P :: x : A}{\Delta \vdash x \triangleleft \texttt{inl}; P :: x : A \oplus B} \qquad \frac{\Delta \vdash Q :: x : B}{\Delta \vdash x \triangleleft \texttt{inr}; Q :: x : A \oplus B}$$

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

Propositions as Session Types: Cut

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

Semantics of Session-typed Processes

$$
\begin{array}{rcl}
(\boldsymbol{\nu} x)\big(x \triangleright \{\mathtt{l_i} : P_i\}_{i \in I} \mid x \triangleleft \mathtt{l_i}; R\big) & \longrightarrow & (\boldsymbol{\nu} x)\big(P_i \mid R\big) \\
(\boldsymbol{\nu} x)\big(x \triangleleft \mathtt{l_i}; R \mid x \triangleright \{\mathtt{l_i} : P_i\}_{i \in I}\big) & \longrightarrow & (\boldsymbol{\nu} x)\big(R \mid P_i\big) \\
(\boldsymbol{\nu} x)\big((\boldsymbol{\nu} y)\overline{x}\langle y\rangle.(P_1 \mid P_2) \mid x(z).Q\big) & \longrightarrow & (\boldsymbol{\nu} x)\big(P_2 \mid (\boldsymbol{\nu} y)(P_1 \mid Q\{^y/_z\})\big) \\
(\boldsymbol{\nu} x)([x \leftarrow y] \mid P) & \longrightarrow & P\{^y/_x\} \\
P \longrightarrow P' & \Longrightarrow & (\boldsymbol{\nu} x)(P \mid Q) \longrightarrow (\boldsymbol{\nu} x)(P' \mid Q) \\
 & \cdots &
\end{array}
$$

Closed under structural congruence, noted $\equiv$.

Semantics of Session-typed Processes

$$
\begin{array}{rcl}
(\boldsymbol{\nu} x)\big(x \triangleright \{\mathtt{l_i} : P_i\}_{i \in I} \mid x \triangleleft \mathtt{l_i}; R\big) & \longrightarrow & (\boldsymbol{\nu} x)\big(P_i \mid R\big) \\
(\boldsymbol{\nu} x)\big(x \triangleleft \mathtt{l_i}; R \mid x \triangleright \{\mathtt{l_i} : P_i\}_{i \in I}\big) & \longrightarrow & (\boldsymbol{\nu} x)\big(R \mid P_i\big) \\
(\boldsymbol{\nu} x)\big((\boldsymbol{\nu} y)\overline{x}\langle y \rangle.(P_1 \mid P_2) \mid x(z).Q\big) & \longrightarrow & (\boldsymbol{\nu} x)\big(P_2 \mid (\boldsymbol{\nu} y)(P_1 \mid Q\{y/z\})\big) \\
(\boldsymbol{\nu} x)([x \leftarrow y] \mid P) & \longrightarrow & P\{y/x\} \\
P \longrightarrow P' & \Longrightarrow & (\boldsymbol{\nu} x)(P \mid Q) \longrightarrow (\boldsymbol{\nu} x)(P' \mid Q) \\
& \dots &
\end{array}
$$

Closed under structural congruence, noted $\equiv$.

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

Process Reductions from Cut Reductions

$$
\dfrac{\dfrac{\Delta_1 \vdash P :: x : A \qquad \Delta_1 \vdash Q :: x : A}{\Delta_1 \vdash x \triangleright \{\mathtt{inl} : P, \mathtt{inr} : Q\} :: x : A \mathbin{\&} B} \qquad \dfrac{\Delta_2, x : A \vdash R :: z : C}{\Delta_2, x : A \mathbin{\&} B \vdash x \triangleleft \mathtt{inl}; R :: z : C}}{\Delta_1, \Delta_2 \vdash (\boldsymbol{\nu} x)\Big(x \triangleright \{\mathtt{inl} : P, \mathtt{inr} : Q\} \mid x \triangleleft \mathtt{inl}; R\Big) :: z : C}
$$

$$\longrightarrow$$

Process Reductions from Cut Reductions

$$\dfrac{\dfrac{\boxed{\Delta_1 \vdash P :: x : A} \qquad \Delta_1 \vdash Q :: x : A}{\Delta_1 \vdash x \triangleright \{\texttt{inl} : P, \texttt{inr} : Q\} :: x : A \,\&\, B} \qquad \dfrac{\boxed{\Delta_2, x : A \vdash R :: z : C}}{\Delta_2, x : A \,\&\, B \vdash x \triangleleft \texttt{inl}; R :: z : C}}{\Delta_1, \Delta_2 \vdash (\boldsymbol{\nu} x)\Big(x \triangleright \{\texttt{inl} : P, \texttt{inr} : Q\} \mid x \triangleleft \texttt{inl}; R\Big) :: z : C}$$

$$\longrightarrow$$

$$\dfrac{\boxed{\Delta_1 \vdash P :: x : A} \qquad \boxed{\Delta_2, x : A \vdash R :: z : C}}{}$$

Process Reductions from Cut Reductions

$$
\dfrac{\dfrac{\colorbox{pink}{$\Delta_1 \vdash P :: x : A$} \qquad \Delta_1 \vdash Q :: x : A}{\Delta_1 \vdash x \triangleright \{\mathtt{inl} : P, \mathtt{inr} : Q\} :: x : A \,\&\, B} \qquad \dfrac{\colorbox{lavender}{$\Delta_2, x : A \vdash R :: z : C$}}{\Delta_2, x : A \,\&\, B \vdash x \triangleleft \mathtt{inl}; R :: z : C}}{\Delta_1, \Delta_2 \vdash (\boldsymbol{\nu} x)\Big(x \triangleright \{\mathtt{inl} : P, \mathtt{inr} : Q\} \mid x \triangleleft \mathtt{inl}; R\Big) :: z : C}
$$

$$\longrightarrow$$

$$
\dfrac{\colorbox{pink}{$\Delta_1 \vdash P :: x : A$} \qquad \colorbox{lavender}{$\Delta_2, x : A \vdash R :: z : C$}}{\Delta_1, \Delta_2 \vdash (\boldsymbol{\nu} x)(P \mid R)}
$$

Process Reductions from Cut Reductions

$$\dfrac{\dfrac{\Delta_1 \vdash P :: y : A \qquad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\boldsymbol{\nu} y)\overline{x}\langle y\rangle.(P \mid Q) :: x : A \otimes B} \qquad \dfrac{\Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_3, x : A \otimes B \vdash x(y).R :: z : C}}{\Delta_1, \Delta_2, \Delta_3 \vdash (\boldsymbol{\nu} x)\Big((\boldsymbol{\nu} y)\overline{x}\langle y\rangle.(P \mid Q) \mid x(y).R\Big) :: z : C}$$

$$\longrightarrow$$

Process Reductions from Cut Reductions

$$
\dfrac{\dfrac{\boxed{\Delta_1 \vdash P :: y : A} \qquad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\boldsymbol{\nu} y)\overline{x}\langle y\rangle.(P \mid Q) :: x : A \otimes B} \qquad \dfrac{\boxed{\Delta_3, y : A, x : B \vdash R :: z : C}}{\Delta_3, x : A \otimes B \vdash x(y).R :: z : C}}{\Delta_1, \Delta_2, \Delta_3 \vdash (\boldsymbol{\nu} x)\Big((\boldsymbol{\nu} y)\overline{x}\langle y\rangle.(P \mid Q) \mid x(y).R\Big) :: z : C}
$$

$$\longrightarrow$$

Process Reductions from Cut Reductions

$$\dfrac{\dfrac{\Delta_1 \vdash P :: y : A \qquad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\boldsymbol{\nu} y)\overline{x}\langle y\rangle.(P \mid Q) :: x : A \otimes B} \qquad \dfrac{\Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_3, x : A \otimes B \vdash x(y).R :: z : C}}{\Delta_1, \Delta_2, \Delta_3 \vdash (\boldsymbol{\nu} x)\Big((\boldsymbol{\nu} y)\overline{x}\langle y\rangle.(P \mid Q) \mid x(y).R\Big) :: z : C}$$

$$\longrightarrow$$

$$\dfrac{\Delta_1 \vdash P :: y : A \qquad \Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_1, x : B, \Delta_3 \vdash (\boldsymbol{\nu} y)(P \mid R) :: z : C}$$

Process Reductions from Cut Reductions

$$\dfrac{\dfrac{\Delta_1 \vdash P :: y : A \qquad \boxed{\Delta_2 \vdash Q :: x : B}}{\Delta_1, \Delta_2 \vdash (\boldsymbol{\nu} y)\overline{x}\langle y\rangle.(P \mid Q) :: x : A \otimes B} \qquad \dfrac{\Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_3, x : A \otimes B \vdash x(y).R :: z : C}}{\Delta_1, \Delta_2, \Delta_3 \vdash (\boldsymbol{\nu} x)\Big((\boldsymbol{\nu} y)\overline{x}\langle y\rangle.(P \mid Q) \mid x(y).R\Big) :: z : C}$$

$$\longrightarrow$$

$$\dfrac{\boxed{\Delta_2 \vdash Q :: x : B} \qquad \dfrac{\Delta_1 \vdash P :: y : A \qquad \Delta_3, y : A, x : B \vdash R :: z : C}{\Delta_1, x : B, \Delta_3 \vdash (\boldsymbol{\nu} y)(P \mid R) :: z : C}}{\Delta_1, \Delta_2, \Delta_3 \vdash (\boldsymbol{\nu} x)\Big(Q \mid (\boldsymbol{\nu} y)(P \mid R)\Big) :: z : C}$$

Properties of πDILL

Theorem (Subject reduction)

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

Properties of πDILL

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* $P = \overline{z}\langle\rangle$.

Asynchronous Output

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

Asynchronous Output

Synchronize on channel x, then continue with Q

$$\frac{\Delta_1 \vdash P :: y : A \qquad \Delta_2 \vdash Q :: x : B}{\Delta_1, \Delta_2 \vdash (\boldsymbol{\nu} y)\left(\overline{x}\langle y\rangle.(P \mid Q)\right) :: x : A \otimes B}$$

Asynchronous Output

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

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

Asynchronous Output

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

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

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

Asynchronous Output

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

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

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

Asynchronous Output

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

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

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

Asynchronous Output (& Matching Input)

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

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

$$(\boldsymbol{\nu} x)\Big((\boldsymbol{\nu} y)(\boldsymbol{\nu} x')\,(\overline{x}\langle y, x'\rangle \mid P \mid Q) \mid x(y, x').R\Big) \longrightarrow$$

$$(\boldsymbol{\nu} y)\Big(P \mid (\boldsymbol{\nu} x')(Q \mid R)\Big)$$

Asynchronous Output (& Matching Input)

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

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

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

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:

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

Unrestricted Resources

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

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

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:

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

- 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
- $A_1, \ldots, A_k; B_1, \ldots, B_n \vdash C$ stands for $!A_1 \otimes \ldots \otimes !A_k \otimes B_1 \otimes \ldots \otimes B_n \multimap C$

The ! Modality

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

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

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

The ! Modality

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

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

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

The ! Modality

$$\frac{\Gamma; \emptyset \vdash P :: y : A}{\Gamma; \emptyset \vdash !u(y).P :: u : !A}
\qquad
\frac{u : A, \Gamma; y : A, \Delta \vdash Q :: z : C}{u : A, \Gamma; \Delta \vdash (\boldsymbol{\nu} y)(\overline{u}\langle y \rangle \mid Q) :: z : C}$$

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

The ! Modality

$$\frac{\Gamma; \emptyset \vdash P :: y : A}{\Gamma; \emptyset \vdash !u(y).P :: u : !A} \qquad \frac{u : A, \Gamma; y : A, \Delta \vdash Q :: z : C}{u : A, \Gamma; \Delta \vdash (\boldsymbol{\nu} y)(\overline{u}\langle y \rangle \mid Q) :: z : C}$$

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

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

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

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

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

Example: A Two-Buyer Protocol

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

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

$$\emptyset; \emptyset \vdash \mathsf{Seller} :: u : !sellerProto \qquad\qquad \emptyset; \emptyset \vdash \mathsf{Bob} :: b : bobProto$$

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

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

Example: A Two-Buyer Protocol

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

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

$$\overline{s}\langle \mathit{book}\rangle.s(\mathit{price}).\overline{b}\langle \mathit{price}\rangle.b \triangleright \left\{\begin{array}{ll} \texttt{close}: & s \triangleleft \texttt{cancel}; b().s() \\ \texttt{share}: & b(\mathit{addr}).s \triangleleft \texttt{buy}; \overline{s}\langle p\rangle.\overline{s}\langle \mathit{addr}\rangle.b().s() \end{array}\right\}$$

Outline

πDILL Processes

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

πDILL Processes

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

TODO: processes have a tree-like structure

Classical Linear Logic

ILL vs CLL:

$1,\ \otimes,\ \multimap,\ \oplus,\ \&$	$1,\ \bot,\ \otimes,\ \parr,\ \oplus,\ \&$
$A_1, \ldots, A_n \vdash A$	$\vdash B_1, \ldots, B_k$
$x_1 : A_1, \ldots, x_n : A_n \vdash P :: z : A$	$P \vdash x_1 : B_1, \ldots, x_k : B_k$
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 $\&$.

$$
\begin{aligned}
(A \oplus B)^{\perp} &= A^{\perp} \mathbin{\&} B^{\perp} \\
(A \mathbin{\&} B)^{\perp} &= A^{\perp} \oplus B^{\perp} \\
(A \otimes B)^{\perp} &= A^{\perp} \mathbin{⅋} B^{\perp} \\
(A \mathbin{⅋} B)^{\perp} &= A^{\perp} \otimes B^{\perp} \\
1^{\perp} &= \perp \\
\perp^{\perp} &= 1
\end{aligned}
$$

Duality

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

$$\begin{array}{rcl}
(A \oplus B)^\perp & = & A^\perp \mathbin{\&} B^\perp \\
(A \mathbin{\&} B)^\perp & = & A^\perp \oplus B^\perp \\
(A \otimes B)^\perp & = & A^\perp \mathbin{⅋} B^\perp \\
(A \mathbin{⅋} B)^\perp & = & A^\perp \otimes B^\perp \\
1^\perp & = & \perp \\
\perp^\perp & = & 1
\end{array}$$

$A = A^{\perp\perp}, A \multimap B = A^\perp \mathbin{⅋} B$

Composition Through Duality

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

Composition Through Duality

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

Create a single channel with endpoints x and y

Composition Through Duality

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

$$\frac{P \vdash \Delta_1 \qquad Q \vdash \Delta_2}{P \mid Q \vdash \Delta_1, \Delta_2} \qquad \frac{P \vdash \Delta, x : A, y : A^\perp}{(\boldsymbol{\nu} xy)P \vdash \Delta_1, \Delta_2} \qquad [x \leftarrow y] \vdash y : A, x : A^\perp$$

Interpretation of the Connectives

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

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

Interpretation of the Connectives

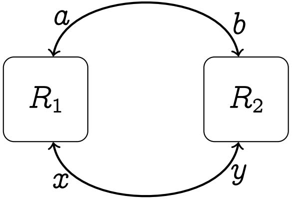
$$\overline{x}\langle y, x' \rangle \vdash x : A \otimes B, y : A^{\perp}, x' : B^{\perp} \qquad \frac{P \vdash \Delta, y : A, x' : B}{x(y, x').P \vdash \Delta, x : A \mathbin{⅋} B}$$

$$\frac{j \in I}{\overline{x}\langle x' \rangle \triangleleft \mathtt{l}_j \vdash x : \oplus\{\mathtt{l}_i : A_i\}, x' : A_j^{\perp}} \qquad \frac{\forall i \in I. \qquad P_i \vdash \Delta, x' : A_i}{x(x') \triangleright \{\mathtt{l}_i : P_i\}_{i \in I} \vdash \Delta, \&\{\mathtt{l}_i : A_i\}_{i \in I}}$$

Example Cyclic Process

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

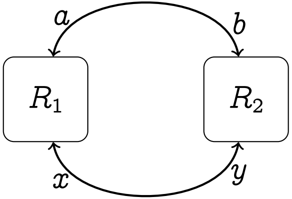
$$R_2 = (\boldsymbol{\nu} y' y_1)(\boldsymbol{\nu} b' b_1)\Big(\overline{y}\langle w, y'\rangle \mid \overline{b}\langle v, b'\rangle \mid R_2'\Big)$$



Example Cyclic Process

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

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



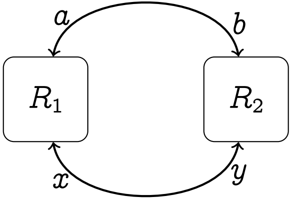
$$\frac{R_1' \vdash v : A, a' : C, w : B, x' : D}{R_1 \vdash a : A \parr C, x : B \parr D}$$

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

Example Cyclic Process

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

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



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

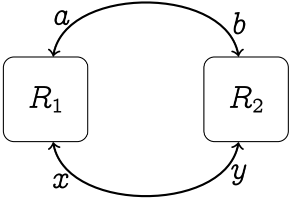
$$\frac{R_2' \vdash y_1 : D^{\perp}, b_1 : C^{\perp}}{R_2 \vdash b : A^{\perp} \otimes C^{\perp}, y : B^{\perp} \otimes D^{\perp}, w : A, v : B}$$

$$(\boldsymbol{\nu} ab)(\boldsymbol{\nu} xy)\Big(R_1 \mid R_2\Big) \longrightarrow$$

Example Cyclic Process

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

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



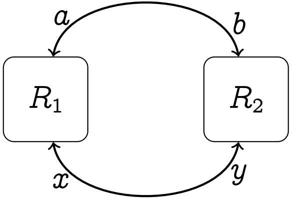
$$\frac{R_1' \vdash v : A, a' : C, w : B, x' : D}{R_1 \vdash a : A \mathbin{⅋} C, x : B \mathbin{⅋} D} \qquad \frac{R_2' \vdash y_1 : D^\perp, b_1 : C^\perp}{R_2 \vdash b : A^\perp \otimes C^\perp, y : B^\perp \otimes D^\perp, w : A, v : B}$$

$$(\boldsymbol{\nu} ab)(\boldsymbol{\nu} xy)\big(R_1 \mid R_2\big) \longrightarrow (\boldsymbol{\nu} xy)(\boldsymbol{\nu} b' b_1)\big(x(w, x').R_1'\{b'/a'\} \mid (\boldsymbol{\nu} y' y_1)(\overline{y}\langle w, y_1\rangle \mid R_2')\big)$$

Example Cyclic Process

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

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



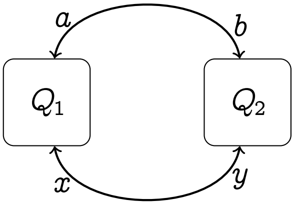
$$\frac{R_1' \vdash v : A, a' : C, w : B, x' : D}{R_1 \vdash a : A \parr C, x : B \parr D}$$

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

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

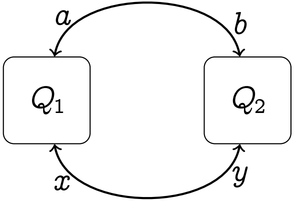
Example Deadlock Process

$$Q_1 = a(v, a').(\boldsymbol{\nu} x' x_1)(\overline{x}\langle w, x'\rangle \mid Q_1')$$
$$Q_2 = y(w, y').(\boldsymbol{\nu} b' b_1)\left(\overline{b}\langle v, b'\rangle \mid Q_2'\right)$$



Example Deadlock Process

$$Q_1 = a(v, a').(\boldsymbol{\nu} x' x_1)(\overline{x}\langle w, x'\rangle \mid Q_1')$$
$$Q_2 = y(w, y').(\boldsymbol{\nu} b' b_1)\left(\overline{b}\langle v, b'\rangle \mid Q_2'\right)$$

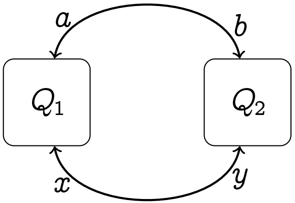


$$\frac{Q_1' \vdash v : A, a' : C, x_1 : D^\perp}{Q_1 \vdash a : A \parr C, x : B^\perp \otimes D^\perp} \qquad \frac{Q_2' \vdash w : B, y' : D, b_1 : C^\perp}{Q_2 \vdash b : A^\perp \otimes C^\perp, y : B \parr D, v : A}$$

Example Deadlock Process

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

$$Q_2 = y(w, y').(\boldsymbol{\nu} b' b_1)\big(\overline{b}\langle v, b'\rangle \mid Q_2'\big)$$



$$\frac{Q_1' \vdash v : A, a' : C, x_1 : D^\perp}{Q_1 \vdash a : A \parr C, x : B^\perp \otimes D^\perp} \qquad \frac{Q_2' \vdash w : B, y' : D, b_1 : C^\perp}{Q_2 \vdash b : A^\perp \otimes C^\perp, y : B \parr D, v : A}$$

$$(\boldsymbol{\nu} ab)(\boldsymbol{\nu} xy)\big(Q_1 \mid Q_2\big) \not\longrightarrow$$

Priorities

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

Priorities

$$A, B ::= A \otimes^{o} B \mid A \mathbin{⅋}^{o} B \mid \ldots \qquad\qquad o \in \mathbb{N} \cup \{\infty\}$$

$$(A \otimes^{o} B)^{\perp} = A^{\perp} \mathbin{⅋}^{o} B^{\perp} \ldots$$

Priorities

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

$$(A \otimes^{o} B)^{\perp} = A^{\perp} \parr^{o} B^{\perp} \ldots$$

$$\frac{P \vdash \Delta, y : A, x' : B \qquad o < pr(\Delta)}{x(y, x').P \vdash \Delta, x : A \parr^{o} B}$$

Example with Priorities

$$\dfrac{\dfrac{Q_1' \vdash v : A, a' : C, x_1 : D^{\perp}}{(\boldsymbol{\nu} x' x_1)(\overline{x}\langle w, x'\rangle \mid R') \vdash v : A, a' : C, x : B^{\perp} \otimes^{o_2} D^{\perp}} \quad o_1 < o_2}{a(v, a').(\boldsymbol{\nu} x' x_1)(\overline{x}\langle w, x'\rangle \mid Q_1') \vdash a : A \parr^{o_1} C, x : B^{\perp} \otimes^{o_2} D^{\perp}}$$

Example with Priorities

$$\dfrac{\dfrac{Q_1' \vdash v : A, a' : C, x_1 : D^\perp}{(\boldsymbol{\nu} x' x_1)(\overline{x}\langle w, x' \rangle \mid R') \vdash v : A, a' : C, x : B^\perp \otimes^{o_2} D^\perp} \quad o_1 < o_2}{a(v, a').(\boldsymbol{\nu} x' x_1)(\overline{x}\langle w, x' \rangle \mid Q_1') \vdash a : A \mathbin{⅋}^{o_1} C, x : B^\perp \otimes^{o_2} D^\perp}$$

$$\dfrac{\dfrac{Q_2' \vdash w : B, y' : D, b_1 : C^\perp}{(\boldsymbol{\nu} b' b_1)\Big(\overline{b}\langle v, b' \rangle \mid Q'\Big) \vdash w : B, y' : D, b : A^\perp \otimes^{o_1} C^\perp} \quad o_2 < o_1}{y(w, y').(\boldsymbol{\nu} b' b_1)\Big(\overline{b}\langle v, b' \rangle \mid Q_2'\Big) \vdash b : A^\perp \otimes^{o_1} C^\perp, y : B \mathbin{⅋}^{o_2} D, v : A}$$

Properties of APCP

Theorem (Deadlock-freedom)

If $P \vdash \emptyset$ *and* $P \not\longrightarrow -$ *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:

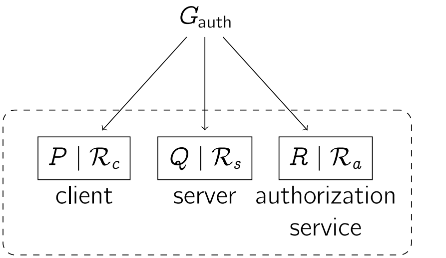
$$G_{\mathsf{auth}} = \mu X \,.\, s \to c \left\{ \begin{array}{l} \mathsf{login} \,.\, c \to a : \mathsf{passwd} \,.\, a \to s : \mathsf{auth} \,.\, X \,, \\ \mathsf{quit} \,.\, c \to a : \mathsf{quit} \,.\, \mathsf{end} \end{array} \right\}$$

Application: Analysis of MPSTs

Consider the following global type:

$$G_{\text{auth}} = \mu X \,.\, s \to c \begin{Bmatrix} \text{login} \,.\, c \to a : \text{passwd} \,.\, a \to s : \text{auth} \,.\, X\,, \\ \text{quit} \,.\, c \to a : \text{quit} \,.\, \text{end} \end{Bmatrix}$$

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



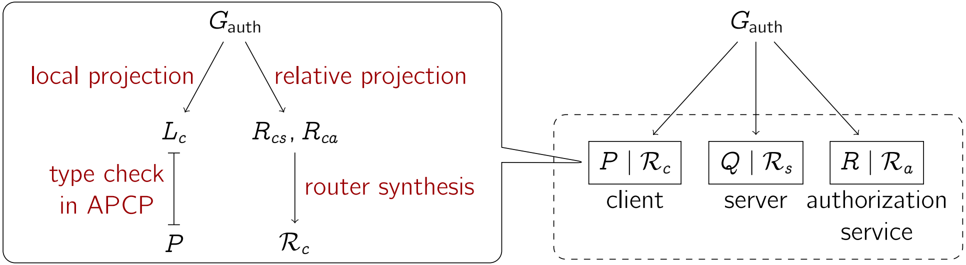
network of routed implementations of G_{auth}

Application: Analysis of MPSTs

Consider the following global type:

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

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



routed implementation of c

network of routed implementations of G_{auth}

Outline

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**