

# Fair Subtyping for Multi-Party Session Types

Luca Padovani

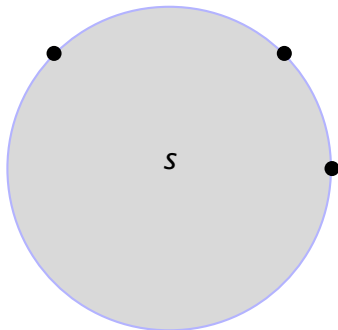
Dipartimento di Informatica, Università di Torino

COORDINATION'11

# Sessions and session types

$s[p]$

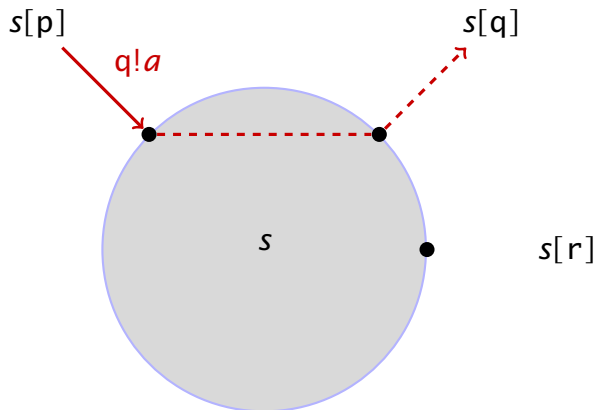
$s[q]$



$s[r]$

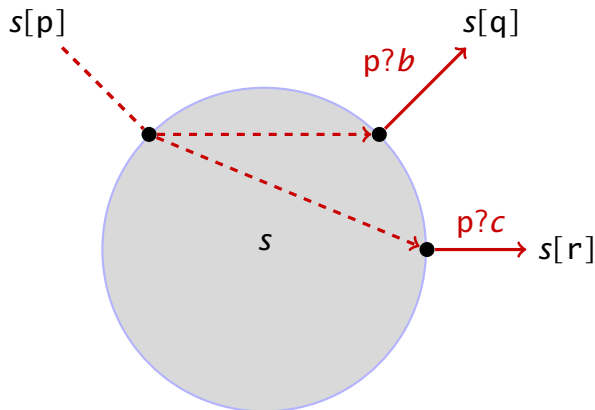
- $s[p] : T = q!a.T \oplus q!b.r!c.end$
- $s[q] : S = p?a.S + p?b.end$
- $s[r] : p?c.end$

# Sessions and session types



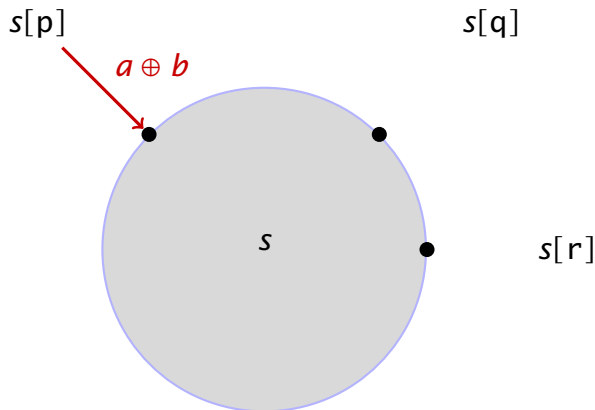
- $s[p] : T = q!a.T \oplus q!b.r!c.end$
- $s[q] : S = p?a.S + p?b.end$
- $s[r] : p?c.end$

# Sessions and session types



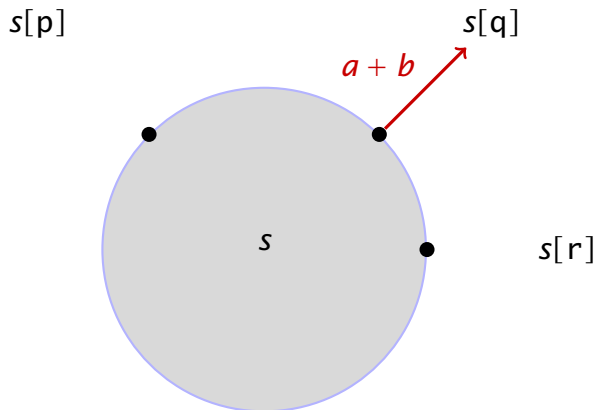
- $s[p] : T = q!a.T \oplus q!b.r!c.end$
- $s[q] : S = p?a.S + p?b.end$
- $s[r] : p?c.end$

# Sessions and session types



- $s[p] : T = q!a.T \oplus q!b.r!c.end$
- $s[q] : S = p?a.S + p?b.end$
- $s[r] : p?c.end$

# Sessions and session types



- $s[p] : T = q!a.T \oplus q!b.r!c.end$
- $s[q] : S = p?a.S + p?b.end$
- $s[r] : p?c.end$

# Session correctness = safety + liveness

## Safety

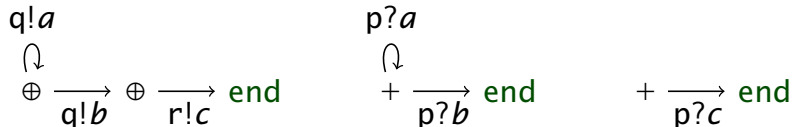
- no message of unexpected type is ever sent

## Liveness

- every non-terminated participant eventually makes progress

# Example: multi-party session

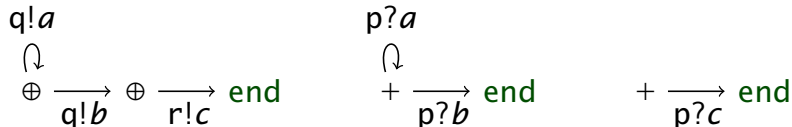
- $s[p] : T = q!a.T \oplus q!b.r!c.end$
- $s[q] : S = p?a.S + p?b.end$
- $s[r] : p?c.end$



Is this session correct?

# Example: multi-party session

- $s[p] : T = q!a.T \oplus q!b.r!c.end$
- $s[q] : S = p?a.S + p?b.end$
- $s[r] : p?c.end$



Is this session correct? **Yes, under a fairness assumption**

# Subtyping for session types

- Gay, Hole, **Subtyping for session types in the pi calculus**, 2005

$\text{end} \leq_{\text{GH}} \text{end}$

$$\frac{T_i \leq_{\text{GH}} S_i \quad (i \in I)}{\sum_{i \in I} ?a_i.T_i \leq_{\text{GH}} \sum_{i \in I \cup J} ?a_i.S_i}$$

$$\frac{T_i \leq_{\text{GH}} S_i \quad (i \in I)}{\bigoplus_{i \in I \cup J} !a_i.T_i \leq_{\text{GH}} \bigoplus_{i \in I} !a_i.S_i}$$

$T \leq_{\text{GH}} S$  means...

- it is safe to use a channel of type  $T$  where a channel of type  $S$  is expected, or...
- it is safe to use a process that behaves as  $S$  where a process that behaves as  $T$  is expected

# Subtyping for session types

- Gay, Hole, **Subtyping for session types in the pi calculus**, 2005

$\text{end} \leq_{\text{GH}} \text{end}$

$$\frac{T_i \leq_{\text{GH}} S_i \quad (i \in I)}{\sum_{i \in I} p?a_i.T_i \leq_{\text{GH}} \sum_{i \in I \cup J} p?a_i.S_i}$$

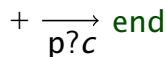
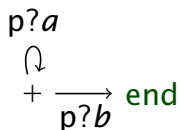
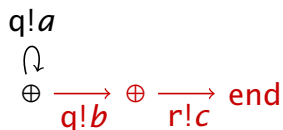
$$\frac{T_i \leq_{\text{GH}} S_i \quad (i \in I)}{\bigoplus_{i \in I \cup J} p!a_i.T_i \leq_{\text{GH}} \bigoplus_{i \in I} p!a_i.S_i}$$

$T \leq_{\text{GH}} S$  means...

- it is safe to use a channel of type  $T$  where a channel of type  $S$  is expected, or...
- it is safe to use a process that behaves as  $S$  where a process that behaves as  $T$  is expected

# Example: multi-party session (and subtyping)

- $p : T = q!a.T \oplus q!b.r!c.end$
- $q : S = p?a.S + p?b.end$
- $r : p?c.end$



# Example: multi-party session (and subtyping)

- $p : T = q!a.T$
- $q : S = p?a.S + p?b.end$
- $r : p?c.end$

$$q!a$$
$$\Downarrow$$
$$\oplus$$
$$p?a$$
$$\Downarrow$$
$$+ \xrightarrow{p?b} end$$
$$+ \xrightarrow{p?c} end$$

Is this session correct?

# Dyadic vs multi-party sessions

In the dyadic setting. . .

- $\leq_{GH}$  preserves both safety and liveness

$p!a.T \not\leq_{GH} \text{end}$

(a process owning an endpoint is required to use it)

In the multi-party setting. . .

- $\leq_{GH}$  preserves safety
- $\leq_{GH}$  does not (necessarily) preserve liveness

# How to fix subtyping

## Definition (**correct** session)

- $T_1 \mid \dots \mid T_n$  **correct** if  
 $T_1 \mid \dots \mid T_n \Rightarrow S_1 \mid \dots \mid S_n$  implies  
 $S_1 \mid \dots \mid S_n \Rightarrow \text{end} \mid \dots \mid \text{end}$

## Definition (fair subtyping)

- $\llbracket T \rrbracket = \{M \mid (T \mid M) \text{ is } \mathbf{correct}\}$
- $T \leq S$  iff  $\llbracket T \rrbracket \subseteq \llbracket S \rrbracket$

# How to fix subtyping

## Definition (**correct** session)

- $T_1 \mid \dots \mid T_n$  **correct** if  
 $T_1 \mid \dots \mid T_n \Rightarrow S_1 \mid \dots \mid S_n$  implies  
 $S_1 \mid \dots \mid S_n \Rightarrow \text{end} \mid \dots \mid \text{end}$

## Definition (fair subtyping)

- $\llbracket T \rrbracket = \{M \mid (T \mid M) \text{ is } \mathbf{correct}\}$
- $T \leq S$  iff  $\llbracket T \rrbracket \subseteq \llbracket S \rrbracket$

# How to fix subtyping

## Definition (**correct** session)

- $T_1 \mid \dots \mid T_n$  **correct** if  
 $T_1 \mid \dots \mid T_n \Rightarrow S_1 \mid \dots \mid S_n$  implies  
 $S_1 \mid \dots \mid S_n \Rightarrow \text{end} \mid \dots \mid \text{end}$

## Definition (fair subtyping)

- $\llbracket T \rrbracket = \{M \mid (T \mid M) \text{ is } \text{correct}\}$
- $T \leq S$  iff  $\llbracket T \rrbracket \subseteq \llbracket S \rrbracket$

# How to fix subtyping

## Definition (**correct** session)

- $T_1 \mid \dots \mid T_n$  **correct** if  
 $T_1 \mid \dots \mid T_n \Rightarrow S_1 \mid \dots \mid S_n$  implies  
 $S_1 \mid \dots \mid S_n \Rightarrow \text{end} \mid \dots \mid \text{end}$

## Definition (fair subtyping)

- $\llbracket T \rrbracket = \{M \mid (T \mid M) \text{ is } \text{correct}\}$
- $T \leq S$  iff  $\llbracket T \rrbracket \subseteq \llbracket S \rrbracket$

# Dilemma

$\leq_{GH}$  versus  $\leq$

- $\leq_{GH}$  is intuitive but unsound
- $\leq$  is sound but obscure

# $\leq_{\text{GH}}$ and $\leq$ are incomparable

$$\begin{aligned} T &= p!a.T \\ S &= q?b.S \end{aligned}$$

$$\begin{aligned} T &\leq S \\ S &\leq T \end{aligned}$$

$$\begin{aligned} \llbracket T \rrbracket &= \emptyset \\ \llbracket S \rrbracket &= \emptyset \end{aligned}$$

$$\begin{aligned} T &\not\leq_{\text{GH}} S \\ S &\not\leq_{\text{GH}} T \end{aligned}$$

# $\leq_{\text{GH}}$ and $\leq$ are incomparable

$$\begin{array}{llll} T = p!a.T & T \leq S & \llbracket T \rrbracket = \emptyset & T \not\leq_{\text{GH}} S \\ S = q?b.S & S \leq T & \llbracket S \rrbracket = \emptyset & S \not\leq_{\text{GH}} T \end{array}$$

not viable

$$\llbracket \text{fail} \rrbracket = \llbracket T \rrbracket = \llbracket S \rrbracket = \dots = \emptyset$$

$$\llbracket \dots \rrbracket \neq \emptyset$$

viable

$$\leq \subseteq \leq_{\text{GH}}$$

# A normal form for session types

$T$  is in **normal form** if either

- $T = \text{fail}$ , or
- $\text{end} \in \text{trees}(S)$  for every  $S \in \text{trees}(T)$

## Proposition

*For every  $T$  there exists  $S \preceq T$  in nf*

## Theorem

*Let  $T, S \neq \text{fail}$  be in nf. Then  $T \preceq S$  implies  $T \preceq_{\text{GH}} S$*

# A normal form for session types

$T$  is in **normal form** if either

- $T = \text{fail}$ , or
- $\text{end} \in \text{trees}(S)$  for every  $S \in \text{trees}(T)$

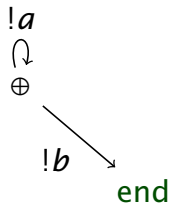
## Proposition

*For every  $T$  there exists  $S \preceq T$  in nf*

## Theorem

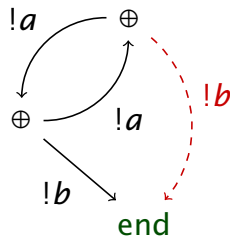
*Let  $T, S \neq \text{fail}$  be in nf. Then  $T \preceq S$  implies  $T \preceq_{\text{GH}} S$*

# Experiment 1



$$T = !a.T \oplus !b.end$$

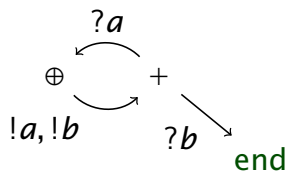
$\stackrel{?}{\leq}$



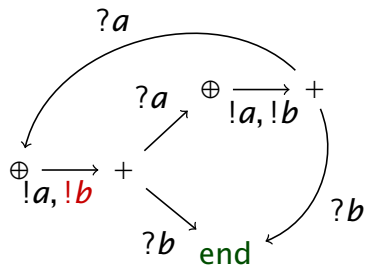
$$S = !a.!a.S \oplus !b.end$$

Is there a context  $M$  that discriminates between  $T$  and  $S$ ?

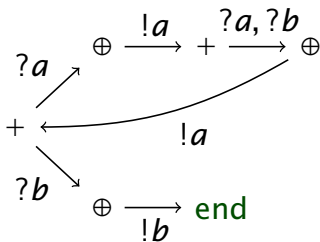
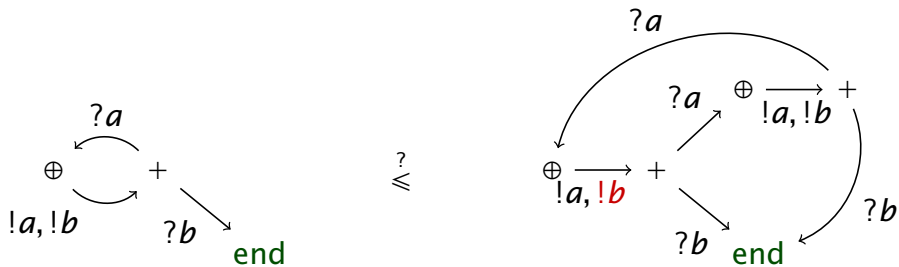
# Experiment 2



$\cong$



# Experiment 2



# Semantic subtyping comes to rescue

$$\begin{array}{lcl} T \leq S & \stackrel{\text{def}}{\iff} & \llbracket T \rrbracket \subseteq \llbracket S \rrbracket \\ T \leq S & \iff & \llbracket T \rrbracket \setminus \llbracket S \rrbracket = \emptyset \\ T \text{ not viable} & \stackrel{\text{def}}{\iff} & \llbracket T \rrbracket = \emptyset \end{array}$$

## Idea

- 1 Compute  $T - S$  such that  $\llbracket T - S \rrbracket = \llbracket T \rrbracket \setminus \llbracket S \rrbracket$
- 2 Reduce  $T \leq S$  to checking  $T - S$  not viable

# Semantic subtyping comes to rescue

$$T \leq S \stackrel{\text{def}}{\iff} \llbracket T \rrbracket \subseteq \llbracket S \rrbracket$$

$$T \leq S \iff \llbracket T \rrbracket \setminus \llbracket S \rrbracket = \emptyset$$

$$T \text{ not viable} \stackrel{\text{def}}{\iff} \llbracket T \rrbracket = \emptyset$$

## Idea

- 1 Compute  $T - S$  such that  $\llbracket T - S \rrbracket = \llbracket T \rrbracket \setminus \llbracket S \rrbracket$
- 2 Reduce  $T \leq S$  to checking  $T - S$  not viable

# Semantic subtyping comes to rescue

$$\begin{array}{lll} T \leq S & \stackrel{\text{def}}{\iff} & \llbracket T \rrbracket \subseteq \llbracket S \rrbracket \\ T \leq S & \iff & \llbracket T \rrbracket \setminus \llbracket S \rrbracket = \emptyset \\ T \text{ not viable} & \stackrel{\text{def}}{\iff} & \llbracket T \rrbracket = \emptyset \end{array}$$

## Idea

- 1 Compute  $T - S$  such that  $\llbracket T - S \rrbracket = \llbracket T \rrbracket \setminus \llbracket S \rrbracket$
- 2 Reduce  $T \leq S$  to checking  $T - S$  not viable

# Semantic subtyping comes to rescue

$$\begin{array}{lll} T \leq S & \stackrel{\text{def}}{\iff} & \llbracket T \rrbracket \subseteq \llbracket S \rrbracket \\ T \leq S & \iff & \llbracket T \rrbracket \setminus \llbracket S \rrbracket = \emptyset \\ T \text{ not viable} & \stackrel{\text{def}}{\iff} & \llbracket T \rrbracket = \emptyset \end{array}$$

## Idea

- 1 Compute  $T - S$  such that  $\llbracket T - S \rrbracket = \llbracket T \rrbracket \setminus \llbracket S \rrbracket$
- 2 Reduce  $T \leq S$  to checking  $T - S$  not viable

# Behavioral difference $\llbracket T - S \rrbracket = \llbracket T \rrbracket \setminus \llbracket S \rrbracket$

## Intuitively

- Along every path shared by both  $T$  and  $S$ ...
- ...turn **end** to **fail**

## Formally

$$\text{end} - \text{end} = \text{fail}$$

$$\sum_{i \in I} p?a_i.T_i - \sum_{i \in I \cup J} p?a_i.S_i = \sum_{i \in I} p?a_i.(T_i - S_i)$$

$$\bigoplus_{i \in I \cup J} p!a_i.T_i - \bigoplus_{i \in I} p!a_i.S_i = \bigoplus_{i \in I} p!a_i.(T_i - S_i) \oplus \bigoplus_{j \in J} p!a_j.T_j$$

## Proposition

$$\llbracket T - S \rrbracket \neq \emptyset \iff \llbracket T \rrbracket \setminus \llbracket S \rrbracket \neq \emptyset$$

# Fair subtyping, at last

$\text{fail} \leq_A T$        $\text{end} \leq_A \text{end}$

$$\frac{T_i \leq_A S_i \quad (i \in I)}{\sum_{i \in I} p?a_i.T_i \leq_A \sum_{i \in I \cup J} p?a_i.S_i}$$

$$\frac{T_i \leq_A S_i \quad (i \in I) \quad \text{nf}(T - S) = \text{fail}}{T = \bigoplus_{i \in I \cup J} p!a_i.T_i \leq_A \bigoplus_{i \in I} p!a_i.S_i = S}$$

## Theorem

$T \leq S$  iff  $\text{nf}(T) \leq_A \text{nf}(S)$

# Fair subtyping, at last

$\text{fail} \leq_A T$      $\text{end} \leq_A \text{end}$

$$\frac{T_i \leq_A S_i \quad (i \in I)}{\sum_{i \in I} p?a_i.T_i \leq_A \sum_{i \in I \cup J} p?a_i.S_i}$$

$$\frac{T_i \leq_A S_i \quad (i \in I) \quad \text{nf}(T - S) = \text{fail}}{T = \bigoplus_{i \in I \cup J} p!a_i.T_i \leq_A \bigoplus_{i \in I} p!a_i.S_i = S}$$

## Theorem

$T \leq S$  iff  $\text{nf}(T) \leq_A \text{nf}(S)$

# Fair subtyping, at last

**fail**  $\leq_A T$       **end**  $\leq_A$  **end**

$$\frac{T_i \leq_A S_i \quad (i \in I)}{\sum_{i \in I} p?a_i.T_i \leq_A \sum_{i \in I \cup J} p?a_i.S_i}$$

$$\frac{T_i \leq_A S_i \quad (i \in I) \quad \text{nf}(T - S) = \mathbf{fail}}{T = \bigoplus_{i \in I \cup J} p!a_i.T_i \leq_A \bigoplus_{i \in I} p!a_i.S_i = S}$$

## Theorem

$T \leq S$  iff  $\text{nf}(T) \leq_A \text{nf}(S)$

# Fair subtyping, at last

**fail**  $\leq_A T$       **end**  $\leq_A$  **end**

$$\frac{T_i \leq_A S_i \quad (i \in I)}{\sum_{i \in I} p?a_i.T_i \leq_A \sum_{i \in I \cup J} p?a_i.S_i}$$

$$\frac{T_i \leq_A S_i \quad (i \in I) \quad \text{nf}(T - S) = \mathbf{fail}}{T = \bigoplus_{i \in I \cup J} p!a_i.T_i \leq_A \bigoplus_{i \in I} p!a_i.S_i = S}$$

## Theorem

$T \leq S$  iff  $\text{nf}(T) \leq_A \text{nf}(S)$

# (Fair) subtyping = (fair) testing preorder

- $P$  passes test  $T$
- $P \sqsubseteq Q$  iff  $P$  passes test  $T$  implies  $Q$  passes test  $T$

## “Unfair” testing

- De Nicola, Hennessy, **Testing equivalences for processes**, 1983
- ...

## Fair testing

- Cleaveland, Natarajan, **Divergence and fair testing**, 1995
- Rensink, Vogler, **Fair testing**, 2007

# Fair testing vs fair subtyping

## Fair testing

- Cleaveland, Natarajan, **Divergence and fair testing**, 1995
- Rensink, Vogler, **Fair testing**, 2007
- denotational (= obscure) characterization
- no complete deduction system
- exponential

## Fair subtyping

- + operational (= hopefully less obscure) characterization
- + complete deduction system
- + polynomial

# More on fair subtyping

- Padovani, **Fair Subtyping for Multi-Party Session Types**, COORDINATION 2011
- 
- + formal definitions and proofs
  - + algorithms (viability, normal form, subtyping)

# Work in progress: fair type checking

$$T = !a.T \oplus !b.\text{end}$$

$$P = u!a.P$$

$$\frac{\frac{u : T \vdash P}{u : !a.T \vdash u!a.P} \text{ (T-Output)} \quad T \leq !a.T}{u : T \vdash P} \text{ (T-Narrow)}$$

**thank you**