Two Notions of Sub-behaviour for Session-based Client/Server Systems

Franco Barbanera\textsuperscript{1} and Ugo de’Liguoro\textsuperscript{2}

\textsuperscript{1}Dipartimento di Matematica e Informatica, Università di Catania
\textsuperscript{2}Dipartimento di Informatica, Università di Torino

PPDP’10 - July 27 2010, Hagenberg
Overview

Session

A session is a logic unit, collecting and structuring messages exchanged among a determined set of agents, sharing a private channel to prevent interference by third parties.

- **Session types** have been introduced to formalise *two-sided* sessions in type systems for the $\pi$-calculus.

We set up a behavioural semantic investigation of session types using the notion of **contract**.

- **Contracts** are a process algebraic formalism to describe the behaviour of services in a client/server scenario.
Session Types (Honda, Vasconcelos, Kubo)

Session types $=\,$ regular trees of ordinary types of (polyadic) $\pi$-calculus

If $\Gamma \vdash P$ is derivable and

$$\Gamma(x) = \mu X. \ ?(\text{Int}).&\langle \ell_0 : ![\text{Bool}]\text{end},
\ell_1 : \oplus \langle \ell_2 : \text{end},
\ell_3 : X \rangle \rangle$$

then channel $x$ is used in $P$ to carry the following “session”:

1. input an integer
2. on receiving the message $\ell_0$ send a boolean then stop
3. on receiving $\ell_1$ either issue $\ell_2$ then stop, or issue $\ell_3$ and start over the whole session
Session Types (Honda, Vasconcelos, Kubo)

The syntax:

\[ T ::= \textbf{Int} \mid \textbf{Bool} \mid \ldots \mid S \quad \text{ground/session type} \]

\[ S ::= \text{end} \quad \text{ended session} \]

\[ \quad \mid ?(T)S \quad \text{input of type } T, \text{ then } S \]

\[ \quad \mid ![T]S \quad \text{output of type } T, \text{ then } S \]

\[ \quad \mid \&\langle \ell_i : S_i \mid i \in I \rangle \quad \text{branching (} I \text{ finite)} \]

\[ \quad \mid \oplus\langle \ell_i : S_i \mid i \in I \rangle \quad \text{selection (} I \text{ finite)} \]

\[ \quad \mid X \quad \text{variable} \]

\[ \quad \mid \mu X. S \quad \text{recursion (} S \text{ not a variable)} \]

If \( T \) is restricted to ground types, these are first order session types; they are higher-order otherwise.
Session Types (Honda, Vasconcelos, Kubo)

The “duality” relation over session types:

\[
\begin{align*}
\text{end} & = \text{end} \\
?(T)S & = ![T]\overline{S} \\
\&\langle \ell_i : S_i \mid i \in I \rangle & = \oplus\langle \ell_i : \overline{S}_i \mid i \in I \rangle \\
\oplus\langle \ell_i : S_i \mid i \in I \rangle & = \&\langle \ell_i : \overline{S}_i \mid i \in I \rangle \\
\overline{X} & = X \\
\mu X. S & = \mu X. \overline{S}
\end{align*}
\]

The following rule is at the hearth of error freeness property within a typeable session:

\[
\Delta, x : S \vdash P \quad \Delta, x : \overline{S} \vdash Q \\
\hline
\Delta \vdash (\nu x)(P \mid Q)
\]
Subtyping Session Types (Gay-Hole)

Subtyping intuition

\( A <: B \) if and only if any channel that satisfies the stricter “protocol” \( A \) also satisfies the protocol \( B \)

The \( A <: B \) relation has been axiomatized by Gay and Hole.

They proved it *operationally* sound by showing that the *narrowing* rule:

\[
\frac{\Delta, x : B \vdash P \quad A <: B}{\Delta, x : A \vdash P}
\]

doesn’t break subject reduction.

Note that subsumption rule is just the dual of *subsumption* rule of the \( \lambda \)-calculus with subtyping.
Coinductive Axiomatization of FO-Subtyping

A coinductive reformulation: let $\Gamma = \{A_1 <: B_1, \ldots, A_k <: B_k\}$, then we derive judgements of the form $\Gamma \vdash A <: B$ by the rules:

- $\Gamma \vdash \mu X.A <: A\{\mu X.A/X\}$
- $\Gamma \vdash A\{\mu X.A/X\} <: \mu X.A$

- $\Gamma, \land_{i \in I} \langle \ell_i : A_i \rangle <: \land_{j \in J} \langle \ell_j : B_j \rangle \vdash A_i <: B_i \quad \forall i \in I \quad I \subseteq J$

- $\Gamma \vdash \land_{i \in I} \langle \ell_i : A_i \rangle$ $<$: $\land_{j \in J} \langle \ell_j : B_j \rangle$

- $\Gamma, \lor_{i \in I} \langle \ell_i : A_i \rangle <: \lor_{j \in J} \langle \ell_j : B_j \rangle \vdash A_j <: B_j \quad \forall j \in J \quad I \supseteq J$

- $\Gamma \vdash \lor_{i \in I} \langle \ell_i : A_i \rangle$ $<$: $\lor_{j \in J} \langle \ell_j : B_j \rangle$
Behavioural semantics of session types

Problem
Is there a semantic characterization of session subtyping?

Answer: behavioural semantics

- provide a formal definition of protocols as *behaviours*
- give a concept of *sub-behaviour*
- interpret session types as behaviours

We understand behaviours as a suitable kind of processes, for which we choose *contracts*
Contracts (Castagna, Laneve, Padovani)

- Contracts are abstract specifications of web-services (and of client queries).
- Central is the compliance relation among a client query and a server contract:
  \[ \rho \text{ complies with } \tau \quad (\rho \dashv \tau, \ \rho \text{ is a client for } \sigma) \]
  \[ \uparrow \]
  every request from \( \rho \) is satisfied by \( \sigma \)
- Compliance induces a subcontract relation:
  \( \sigma \) is a subcontract of \( \tau \) (\( \sigma \preceq \tau \)) \iff every client of \( \sigma \) is such of \( \tau \).
Contracts (Castagna, Laneve, Padovani)

Web contracts are parallel-free CCS terms (without $\tau$) generated by the grammar:

$$\sigma ::= 1 \mid \alpha.\sigma \mid \sigma + \sigma \mid \sigma \oplus \sigma \mid x \mid \text{rec } x.\sigma$$

where $\alpha \in \mathcal{N} \cup \overline{\mathcal{N}}$.

Semantics is defined by the LTS:

- $\alpha.\sigma \xrightarrow{\alpha} \sigma$
- $\sigma \xrightarrow{\alpha} \sigma' \Rightarrow \sigma + \rho \xrightarrow{\alpha} \sigma', \rho + \sigma \xrightarrow{\alpha} \sigma'$
- $\sigma \oplus \rho \xrightarrow{} \sigma, \sigma \oplus \rho \xrightarrow{} \tau$
- $\text{rec } x.\sigma \xrightarrow{} \sigma\{\text{rec } x.\sigma/x\}$
The contract of a ballot service might be:

\[ \text{rec } x. \text{Login.}(\text{Wrong}.x \oplus \text{Ok.}(\text{VoteA.}(V_{a1} + V_{a2}) + \text{VoteB.}(V_{b1} + V_{b2}))) \]
The contract of a ballot service might be:

```
rec x. Login.(Wrong.x ⊕ Ok.(VoteA.(Va1+Va2)+VoteB.(Vb1+Vb2)))
```

meaning:

- wait for a Login action
Example

The contract of a ballot service might be:

\[ \text{rec } x. \text{Login.} (\text{Wrong.} x \oplus \text{Ok.} (\text{VoteA.} (V_{a1} + V_{a2}) + \text{VoteB.} (V_{b1} + V_{b2}))) \]

meaning:

- wait for a Login action
- acknowledge the (in)correctness of login
The contract of a ballot service might be:

\[
\text{rec } x. \text{Login.} (\overline{\text{Wrong}}. x \oplus \overline{\text{Ok}}. (\text{VoteA.}(Va1 + Va2) + \text{VoteB.}(Vb1 + Vb2)))
\]

meaning:

- wait for a Login action
- acknowledge the (in)correctness of login
- in the negative restart
Example

The contract of a ballot service might be:

```
rec x. Login.(Wrong.x ⊕ Ok.(VoteA.(Va1+Va2)+VoteB.(Vb1+Vb2)))
```

meaning:

- wait for a Login action
- acknowledge the (in)correctness of login
- in the negative restart
- in the positive prompt for voting either A or B
The contract of a ballot service might be:

\[
\text{rec } x. \text{Login.} (\overline{\text{Wrong.} x} \oplus \overline{\text{Ok.} (\text{VoteA.} (\text{Va1 + Va2}) + \text{VoteB.} (\text{Vb1 + Vb2}))})
\]

meaning:

- wait for a Login action
- acknowledge the (in)correctness of login
- in the negative restart
- in the positive prompt for voting either A or B
- then offer the possibility for voting for a ticket
The contract of a ballot service might be:

\[ \text{rec } x. \text{Login.} (\overline{\text{Wrong}}. x \oplus \overline{\text{Ok}}. (\text{VoteA.}(\text{Va1} + \text{Va2}) + \text{VoteB.}(\text{Vb1} + \text{Vb2}))) \]

meaning:

- wait for a Login action
- acknowledge the (in)correctness of login
- in the negative restart
- in the positive prompt for voting either A or B
- then offer the possibility for voting for a ticket
Session Behaviours as Contracts interpreting Session Types

Consider the mapping from (first order) session types to contracts:

\[
\begin{align*}
[X] & = x \\
[end] & = 1 \\
[\mu X. A] & = \text{rec } x. [A] \\
[?\langle\gamma\rangle A] & = \gamma.[A] \\
[!\langle\gamma\rangle A] & = \overline{\gamma}.[A]
\end{align*}
\]

\[
\begin{align*}
[\&\langle\ell_i : B_i \mid i \in I\rangle] & = \sum_{i \in I} \ell_i.[B_i] \\
[\oplus\langle\ell_i : B_i \mid i \in I\rangle] & = \bigoplus_{i \in I} \ell_i.[B_i]
\end{align*}
\]

The image of the \([\cdot]\) map is a subset of the set of contracts.
Session Behaviours: the grammar

\[ S \ (\textit{Session Behaviours}) \text{ are the closed expressions among those defined by the grammar:} \]

\[
\begin{align*}
\sigma & ::= 1 \\
& \mid a_1.\sigma_1 + \cdots + a_n.\sigma_n \quad \text{external choice, } a_i \text{ distinct} \\
& \mid \bar{a}_1.\sigma_1 \oplus \cdots \oplus \bar{a}_n.\sigma_n \quad \text{internal choice, } \bar{a}_i \text{ distinct} \\
& \mid x \quad \text{variable} \\
& \mid \text{rec } x.\sigma \quad \text{recursion, } \sigma \text{ not a variable}
\end{align*}
\]

\textit{Contracts} describe the overall behaviour of a (client)server. 
\textit{Session Behaviors} describe the possible interactions of a process over a channel.
Compliance and Orthogonality

Extend the reduction relation to pairs of session-behaviours $\rho || \sigma$:

\[
\begin{align*}
\rho \overset{\alpha}{\longrightarrow} \rho' & \quad \sigma \overset{\alpha}{\longrightarrow} \sigma' \\
\rho || \sigma \quad \rightarrow & \quad \rho' || \sigma' \\
\rho \rightarrow \rho' & \\
\rho || \sigma \quad \rightarrow & \quad \rho' || \sigma
\end{align*}
\]

**Compliance:** the client $\rho$ complies with the server $\sigma$, $\rho \vdash \sigma$ if

\[
\forall \rho', \sigma', \quad \rho || \sigma \quad \overset{*}{\longrightarrow} \quad \rho' || \sigma' \quad \not\rightarrow \quad \Rightarrow \quad \rho' = 1
\]

i.e. any request of the client is eventually satisfied by the server.

**Orthogonality:**

\[
\rho \perp \sigma \quad \Leftrightarrow \quad \rho \vdash \sigma \quad \& \quad \sigma \vdash \rho
\]
Examples

\( \bar{a} \oplus \bar{b} \vdash a + b + c \) because:

\[
\bar{a} \oplus \bar{b} \parallel a + b + c \quad \rightarrow \quad \bar{a} \parallel a + b + c \quad \rightarrow \quad 1 \parallel 1
\]

\[
\bar{b} \parallel a + b + c \quad \rightarrow \quad 1 \parallel 1
\]

and also \( a + b + c \vdash \bar{a} \oplus \bar{b} \) hence \( \bar{a} \oplus \bar{b} \perp a + b + c \).

But \( \bar{a} \oplus \bar{b} \oplus \bar{c} \not\vdash a + b \) (and \( a + b \not\vdash \bar{a} \oplus \bar{b} \oplus \bar{c} \)) since:

\[
\bar{a} \oplus \bar{b} \oplus \bar{c} \parallel a + b \quad \rightarrow \quad \bar{c} \parallel a + b \not\rightarrow
\]

Note that \( \text{rec } x.a.x \vdash \text{rec } x.\bar{a}.x \) (without reaching \( 1 \parallel \cdots \)) since:

\[
\text{rec } x.a.x \parallel \text{rec } x.\bar{a}.x \quad \xrightarrow{2} \quad a.\text{rec } x.a.x \parallel \bar{a}.\text{rec } x.\bar{a}.x
\]

\[
\rightarrow \quad \text{rec } x.a.x \parallel \text{rec } x.\bar{a}.x \quad \rightarrow \quad \cdots
\]
For $\sigma, \rho \in S$, let

$$\text{Client}(\sigma) = \{\rho \in S \mid \rho \vdash \sigma\}, \quad \text{Server}(\rho) = \{\sigma \in S \mid \rho \vdash \sigma\}$$

Then define the relations:

1. $\sigma \preceq_s \sigma'$ if and only if $\text{Client}(\sigma) \subseteq \text{Client}(\sigma')$;
2. $\rho \preceq_c \rho'$ if and only if $\text{Server}(\rho) \subseteq \text{Server}(\rho')$.

In words: $\sigma \preceq_s \sigma'$ if the server $\sigma'$ has a larger set of clients than $\sigma$, and similarly for $\rho \preceq_c \rho'$.

Note. Our $\preceq_s$ is essentially the subcontract relation by Castagna et alii.
Duality in $S$

Let us extend the $\bar{\cdot}$ operation to all (also open) behaviours:

- $\bar{1} = 1$
- $\bar{a.\sigma} = \bar{a}.\bar{\sigma}$ and $\bar{a.\sigma} = a.\bar{\sigma}$
- $\bar{\sigma + \tau} = \bar{\sigma} \oplus \bar{\tau}$
- $\bar{\sigma \oplus \tau} = \bar{\sigma} + \bar{\tau}$
- $\bar{x} = x$
- $\bar{\text{rec } x.\sigma} = \text{rec } x.\bar{\sigma}$

If $\sigma \in S$ then $\bar{\sigma} \in S$, and $\bar{\bar{\sigma}} = \sigma$. Moreover:

$$\sigma = \llbracket A \rrbracket \text{ if and only if } \bar{\sigma} = \llbracket A \rrbracket$$
Duality in $S$

A relation exists between the syntactic operator $\bar{\cdot}$ and the server/client preorders:

**Proposition.** Let $\tau \in S$:

1. $\bar{\tau}$ is the minimum server among those of $\tau$:
   $$\forall \sigma \in \text{Server}(\tau). \quad \bar{\tau} \preceq_s \sigma$$

2. $\tau$ is the minimum client among those of $\tau$:
   $$\forall \rho \in \text{Client}(\tau). \quad \tau \preceq_c \rho$$

This does not hold outside of $S$:

- $\bar{a} \oplus \bar{a.b} \not\preceq a + a.b$
- the minimum of $\text{Client}(a + a.b)$ is actually $\bar{a}$
- $a + a.b \not\preceq \bar{a} \oplus \bar{a.b}$
- the minimum of $\text{Server}(a + a.b)$ is $\bar{a.b}$
- $\text{Server}(a\bar{b} + a\bar{c}) = \emptyset$
Let $A^\bot = \{ \sigma \in S \mid \exists \tau \in A. \sigma \perp \tau \}$ and $\sigma^\bot = \{ \sigma \}^\bot$:

$$\sigma \leq: \tau \iff A \tau \iff \sigma^\bot \subseteq \tau^\bot$$

**Theorem**

Behavioural subtyping is the intersection of both client and server-subbehaviour relations:

$$\leq: = \leq_c \cap \leq_s$$

It follows that or any $\sigma, \tau \in S$, $\overline{\sigma}$ is minimal in $\sigma^\bot$ w.r.t. $\leq:$. and

$$\sigma \leq: \tau \text{ if and only if } \overline{\tau} \leq: \overline{\sigma}$$

matching with the fact that $A <: B \iff \overline{B} <: \overline{A}$. 
**Higher-Order Behaviours** add input/output of behaviors to prefixes:

$$\sigma, \tau ::= \ldots | ?\sigma^p.\tau \mid !\sigma^p.\tau$$

where $p \in \{s, c\}$.

The higher-order LTS:

$$\begin{align*}
?\rho^p.\sigma & \xrightarrow{?\rho^p} \sigma \\
\sigma & \xrightarrow{?\rho_2^p} \sigma' \\
?\rho_1^p.\tau & \xrightarrow{!\rho_1^p} \tau' \\
\sigma \parallel \tau & \xrightarrow{\rho_1 \preceq_p \rho_2} \sigma' \parallel \tau' \\
!\rho^p.\sigma & \xrightarrow{!\rho^p} \sigma \\
\sigma & \xrightarrow{!\rho_1^p} \sigma' \\
!\rho_2^p.\tau & \xrightarrow{?\rho_2^p} \tau' \\
\sigma \parallel \tau & \xrightarrow{\rho_1 \preceq_p \rho_2} \sigma' \parallel \tau'
\end{align*}$$

*Note the use of $\preceq_s, \preceq_c$ in the LTS rules.*

The syntactical duality extends as:

$$\begin{align*}
?\sigma^p.\tau = !\sigma^p.\tau, \\
!\sigma^p.\tau = ?\sigma^p.\tau
\end{align*}$$
Interpreting Higher-Order Sessions

Higher-order session may send and receive session types:

\[ A, B, ::= \ldots | ?(A^p)B | ![A^p]B \quad \text{for } p = c, s \]

By considering higher-order behaviours we can extend the interpretation map to higher order session types straightforwardly:

\[
\llbracket ?(A^p)B \rrbracket = ?[A]^p[B], \quad \llbracket ![A^p]B \rrbracket = ![A]^p[B]
\]

Note. We have studied asymmetric session-types, with polarized channels to record either client or server role, elsewhere: see [Barbanera-Capecchi-de’Liguoro, Proc. of FSEN’09].
Subtyping Higher-Order Sessions

We decorate the sent/received session by a polarity:

\[ A, B, ::= \ldots | ?(A^p)B | ![A^p]B \quad \text{for} \quad p = c, s. \]

Then consider the (coinductive versions of) the Gay-Hole rules:

\[
\begin{align*}
\Gamma, ?(A^p)B <: ?(C^p)D &\vdash A <: C, B <: D \\
\hline
\Gamma &\vdash ?(A^p)B <: ?(C^p)D \\
\Gamma, ![A^p]B <: ![C^p]D &\vdash C <: A, B <: D \\
\hline
\Gamma &\vdash ![A^p]B <: ![C^p]D
\end{align*}
\]

**Fact** \( A <: B \) (according to Gay-Hole) if and only if \( \vdash A <: B \)
The Soundness Theorem

Main Theorem

Define:

1. $\models A <: B$ iff $\llbracket A \rrbracket \preceq \llbracket B \rrbracket$
2. $\models \Gamma$ iff $\models C <: D$ for all $C <: D \in \Gamma$
3. $\Gamma \models A <: B$ iff $\models \Gamma$ implies $\models A <: B$

then

$$\Gamma \vdash A <: B \Rightarrow \Gamma \models A <: B$$

Conjecture. Completeness holds, hence known decision algorithms for session subtyping are useful also for the subcontract relation in case of session behaviours.
Final Remarks

Results and conjectures:

- we have proposed an interpretation of session types into behaviours which is sound w.r.t. Gay-Hole subtyping
- we conjecture that the interpretation is actually complete
- when restricting to $S$, there is no theoretical loss w.r.t. the full set of contracts in the case of two-ended sessions

Further work:

- things are different when considering multiparty sessions and fairness concepts are involved
- the power of higher-order LTS in giving semantics to the typed $\pi$-calculus deserves further attention