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

Franco Barbanera
Dept. of Mathematics and Computer Science
Università di Catania
v.le A. Doria 6, I-95125 Catania, Italy
barba@dmi.unict.it

Ugo de'Liguoro
Dept. of Computer Science
Università di Torino
c.so Svizzera 185, I-10149 Torino, Italy
deliguoro@di.unito.it

## Abstract

We propose a refinement and a simplification of the behavioural semantics of session types, based on the concepts of compliance and sub-behaviour from the theory of web contracts. We introduce two relations, representing the idea of sub-behaviour from the point of view of the client and the server, respectively, and characterize the sub-behaviour relation (from the literature) as the intersection of the other two. We show that a proper subclass of behaviours, called "session behaviors", and the sub-behaviour relations model session types and subtyping, clarifying the otherwise problematic extension of session type subtyping with concepts from the theory of contracts.

***Categories and Subject Descriptors*** F.1.2 [*Modes of Computation*]: Parallelism and concurrency; F.3.1 [*Specifying and Verifying and Reasoning about Programs*]: Logics of programs; F.3.3 [*Studies of Program Constructs*]: Type structure

***General Terms*** Theory, Verification

***Keywords*** Client/server interaction, Session Types, Behavioural Semantics, Sub-behaviour, Semantics of Subtyping, Coinduction.

## 1. Introduction

A great deal of work is presently devoted to the formalization of interaction through the network, widening the research area on protocols and investigating its basic concepts. Among them a central one is that of 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.
Restricting attention to dyadic sessions, namely among two partners, it is natural to consider dual views of the same session according to the role of the participants in each communication action. A formalization of this idea can be found in Honda's et alii *session type* systems introduced in [12]. Session types represent the usage of each session channel by a regular tree of types (which is itself considered as a type), abstractly representing all sequences of actions of which the typed channel is "subject" in the $\pi$-calculus jargon. In the theory of session types each type has a dual, describing the same interaction from the point of view of the process holding the opposite end of the channel; the exact correspondence of dual typings of the same channel ensures error freeness (but not deadlock freeness: see [9]).
A similar situation shows up in the theory of *contracts* [4, 7, 13]. A contract is some kind of abstract interface offered by a server to its possible clients, describing the server overall behaviour in an interaction. To formally check whether a client will comply with the server, a dual contract can be used, describing this time the client requests. This leads to an asymmetric view of the client/server interaction, because of a bias to the client side: it is only the client that is expected to complete in any interaction with the server, and not vice versa.
Session types and contracts are quite different: the former represent the usage of single channels, disregarding their arrangement into the system and how their actions, possibly badly, interleave. The latter describe, on the contrary, the behaviour of the server, or client, as a whole, and so could be considered as the "type" of the process itself. In spite of this, and of the difficulties emerging from a first attempt of comparison in [14], contracts have been the basis for some proposals of behavioural semantics for session types in a series of papers: see [5, 8, 17].

In this paper we propose a refinement and a simplification of behavioural semantics of session types, based on the concepts of compliance and sub-behaviour, that we analyze into two symmetric relations. The idea of behavioural semantics of types consists of an interpretation of types into terms of some process algebra such that relevant properties of types are modeled by properties of a transition system over the process terms. By considering a sub-language of CCS without $\tau$, behaviours (more precisely the behavioural part of a contract as defined in [14]) are terms of the language defined by the grammar:

$$\sigma ::= \mathbf{1} \mid \alpha.\sigma \mid \sigma + \sigma \mid \sigma \oplus \sigma \mid x \mid \mathsf{rec}\, x.\sigma \qquad (1)$$

where $\alpha$ represents the CCS action prefixing. However this language is too large for the sake of modeling session types, because it interferes badly with the essential notion of duality, so that we look for a suitable subset of behaviours which we call *session behaviours*. The central issue is that of getting a simple syntactical description of dual behaviours just in terms of the interpretation of dual type constructs, that is reflected by dual actions: input and output actions are represented by names $a$ and co-names $\overline{a}$; branching types, expressing the ability of reacting to some set of messages, are interpreted by external choices, that is by sums $a_1.\sigma_1 + \cdots + a_k.\sigma_k$, where the $a_i$ are pairwise distinct and cannot be co-names (a first restriction w.r.t. the grammar (1)), and the $\sigma_i$ interpret the corresponding continuations; selection types, expressing the possibility that one out of a finite set of requests will be the actual one, are interpreted by internal choices $\overline{a}_1.\sigma_1 \oplus \cdots \oplus \overline{a}_k.\sigma_k$ where again the $\overline{a}_i$ are pairwise distinct and have to be co-names (a second and symmetric restriction), that can evolve internally to $\overline{a}_i.\sigma_i$ for some $i$. The correspondence of dual types is modeled by interaction in a CCS style: a pair of behaviours $\rho, \sigma$ interacts resulting in the continuations $\rho', \sigma'$, written $\rho \| \sigma \longrightarrow \rho' \| \sigma'$ if $\rho$ and $\sigma$ have complementary capabilities $\alpha$ and $\overline{\alpha}$, possibly after some internal actions. To see an example, consider the following variation of an example in [14] (omitting the trailing $\mathbf{1}$):

$$\begin{aligned}\sigma_1 = \mathtt{Login}.(&\overline{\mathtt{Wrong}} \oplus \\ &\overline{\mathtt{Ok}}.(\mathtt{VoteA}.(\mathtt{Va1} + \mathtt{Va2}) + \\ &\quad \mathtt{VoteB}.(\mathtt{Vb1} + \mathtt{Vb2})))\end{aligned}$$

It is a session behaviour that represents a Ballot Service. The service can receive a login and, if correct, signals to the client, by the message Ok, the enabling to vote either for A or for B. After that, the server enables also the voting for one of two possible vice-candidates. The login can be found incorrect, and in such a case the communication Wrong is issued to the client. The following is a recursive version of $\sigma_1$, which represents a service that allows the client to retry the login action in case of failure:

$$\begin{aligned}\sigma_2 = \mathsf{rec}\, x.\mathtt{Login}.(&\overline{\mathtt{Wrong}}.x \oplus \\ &\overline{\mathtt{Ok}}.(\mathtt{VoteA}.(\mathtt{Va1} + \mathtt{Va2}) + \\ &\quad \mathtt{VoteB}.(\mathtt{Vb1} + \mathtt{Vb2})))\end{aligned}$$

We avoid divergence, namely infinite internal actions, by admitting guarded recursion only (which is the case of recursive session types as well and is our last restriction to the grammar (1)). Hence any session behaviour reduces in a finite number of steps into either a $+$ or a $\oplus$-sum (more precisely into a finite set of $\oplus$ summands, which by definition are prefixed by co-names). Then, by writing $\Longrightarrow$ as reflexive and transitive closure of $\longrightarrow$, we say that a *client* $\rho$ is *compliant* w.r.t. a *server* $\sigma$ if whenever $\rho \| \sigma \Longrightarrow \rho' \| \sigma'$ and the latter is an irreducible pair, then $\rho' = \mathbf{1}$, the completed behaviour; this is written as $\rho \dashv \sigma$. For example we have: $a + b \dashv \overline{a} \oplus \overline{b}$ because of either $a + b \| \overline{a} \oplus \overline{b} \longrightarrow a + b \| \overline{a} \longrightarrow \mathbf{1} \| \mathbf{1}$ or $a + b \| \overline{a} \oplus \overline{b} \longrightarrow a + b \| \overline{b} \longrightarrow \mathbf{1} \| \mathbf{1}$. Vice versa $a + b \not\dashv \overline{a} \oplus \overline{c}$ since $a + b \| \overline{a} \oplus \overline{c} \longrightarrow a + b \| \overline{c} \not\longrightarrow$. Note that nothing is requested about $\sigma'$ when $\rho \dashv \sigma$ and $\rho \| \sigma \Longrightarrow \rho' \| \sigma' \not\longrightarrow$, that is it might be $\sigma' \neq \mathbf{1}$: this expresses the idea that the client is entitled to abandon the interaction at any time, while the server is expected to react properly to all client requests.
Consider the behaviour:

$$\rho_1 = \overline{\mathtt{Login}}.(\mathtt{Wrong} + \mathtt{Ok}.(\overline{\mathtt{VoteA}} \oplus \overline{\mathtt{VoteB}}))$$

Then $\rho_1 \dashv \sigma_1$ even if the client $\rho_1$ is not prepared to vote for vice-candidates; also $\rho_1 \dashv \sigma_2$ because a client willing to login once will comply with a server admitting repeated login actions after failure. We have remarked that divergence is excluded by admitting only guarded recursive behaviours, namely of the form $\mathsf{rec}\, x.\sigma$ where $\sigma$ is not a variable (in particular $\sigma \neq x$). As usual, such a behaviour internally reduces to its unfolding: $\mathsf{rec}\, x.\sigma \longrightarrow \sigma\{\mathsf{rec}\, x.\sigma / x\}$ which is the substitution of $x$ by $\mathsf{rec}\, x.\sigma$ in $\sigma$ by avoiding variable clashes. We conclude that e.g. $\mathsf{rec}\, x.\ a.x \dashv \mathsf{rec}\, x.\ \overline{a}.x$, although the reduction out of $\mathsf{rec}\, x.\ a.x \| \mathsf{rec}\, x.\ \overline{a}.x$ is infinite.
The concept of compliance induces two relations: $\sigma \preceq_s \tau$ if all compliant clients of $\sigma$ are such w.r.t. $\tau$, and symmetrically $\sigma \preceq_c \tau$ if all server that satisfy $\sigma$, do the same w.r.t. $\tau$. These are instances of the substitutivity criterion: if $\sigma \preceq_s \tau$ then any server whose behaviour is described by $\sigma$ can be safely replaced by a server satisfying the behavioural description $\tau$; a similar property holds for clients when $\sigma \preceq_c \tau$. To have a glimpse of these relations observe that $a \preceq_s a.b$ and $a.b \preceq_c a$; on the other hand $a \preceq_p a + b$ and $\overline{a} \oplus \overline{b} \preceq_p \overline{a}$ for both $p = c, s$. Applying the above definitions to the previous examples we find that $\sigma_1 \preceq_s \sigma_2$ since intuitively all the clients that comply with a server allowing for just one login, will do with the more liberal ones that allow for more login attempts, as it is described by $\sigma_2$. As far as the $\preceq_c$ relation is concerned, let

$$\rho_2 = \overline{\mathtt{Login}}.(\mathtt{Wrong} + \mathtt{Ok}).$$

Then $\rho_1 \preceq_c \rho_2$ since $\rho_2$ represents the behaviour of voters wishing to return a blank ballot, i.e. wishing to partecipate to the elections but without voting for any candidate; then clients described by $\rho_2$ will comply with ballot servers with which also the actually voting clients described by $\rho_1$ will do.
These relations are evidently close to the idea of subtyping. In typed $\lambda$-calculus and in functional or OO programming languages, $A <: B$ ($A$ is a subtype of $B$) means that any value of type $A$ can masquerade as a value of type $B$, as formalized by the usual subsumption rule. In the case of $\pi$-calculus and related systems, types are not assigned to values but to channel names, so that subtyping is applied in the opposite direction: if $A <: B$ then any channel typeable by $B$ can mascherade as a channel with type $A$ (this is the narrowing rule: see e.g. [20], chap. 7). Following the suggestions in the literature, we define first the concept of orthogonality: $\sigma \perp \tau$ if and only if $\sigma \dashv \tau$ and $\tau \dashv \sigma$; then $\sigma$ is a sub-behaviour of $\tau$ if all the $\rho$ orthogonal w.r.t. $\sigma$ are such w.r.t. $\tau$: we write $\sigma \preceq: \tau$ and call this relation *behavioural subtyping*.
Consider the voter:

$$\rho_3 = \overline{\mathtt{Login}}.(\mathtt{Wrong} + \mathtt{Ok}.(\overline{\mathtt{VoteA}}.\overline{\mathtt{Va2}} \oplus \overline{\mathtt{VoteB}}.\overline{\mathtt{Vb1}}))$$

Then it is easy to see that $\sigma_1 \perp \rho_3$; but $\sigma_1 \not\perp \rho_1$ because $\sigma_1 \not\dashv \rho_1$. As a matter of fact the $\preceq:$ behaves the same as $\preceq_p$ for $p = c, s$ w.r.t. $+$ and $\oplus$, namely covariantly in the number of $+$ summands and contravariantly in the number of the $\oplus$-summands (and covariantly w.r.t. the respective continuations), that is session behaviors and behavioural subtyping mirror branching and selection session types w.r.t. (syntactic) subtyping, respectively. To these properties the sub-behaviour relations $\preceq_p$ add dual forms of subtyping in depth that does not hold in Gay and Hole theory.
Given the above notions, some results can be obtained. First we can extend the definition of the $\bar{\cdot}$ operation, originally concerning names and co-names only, to any behavior: $\overline{\sigma}$ is obtained from $\sigma$ by replacing each prefix $\alpha$ by $\overline{\alpha}$ (recall that $\bar{\cdot}$ is an involutive operator), and by exchanging $+$ with $\oplus$ (and ? with ! in case of higher order behaviors: see below). Then we discover that $\overline{\sigma}$ is the minimal server of $\sigma$, namely it is minimal w.r.t. $\preceq_s$ among all the server of $\sigma$; this also holds w.r.t. $\preceq_c$. Second we prove that $\preceq: \; = \; \preceq_s \cap \preceq_c$ as expected, so that by using this fact we have that $\overline{\sigma}$ is the minimal orthogonal behaviour of $\sigma$ w.r.t. $\preceq:$ . This a payoff of our definition of session behaviours: similar properties have been studied e.g. in [14] for the larger set of "contract behaviours", leading to intricate characterizations of the "dual" of a behaviour.
Next we consider session types, whose treatment we split into two parts, first-order and higher-order session types, for readability. Since the behaviours have been tailored to session types, the interpretation mapping is straightforward: ground types are just names, which are interpreted into the respective co-names when used as the types of outputs. The branching type $\&\langle \cdots \rangle$ is interpreted by a $+$-sum, and the selection type $\oplus\langle \cdots \rangle$ by a $\oplus$-sum. Now we prove that a system which is equivalent to Gay and Hole's subtyping [11] is sound w.r.t. the interpretation of subtyping into behavioural subtyping. More: by adding two axioms: $\mathsf{end} \leq_s A$ or $A \leq_c \mathsf{end}$ (where end is the type of the completed session) we obtain a sound (and we conjecture complete) axiomatization of $\preceq_s$ and $\preceq_c$ respectively.

A *higher-order session* includes the sending of the receiving of other sessions. This makes sense in the case of process algebras (and programming languages) having the ability to exchange channel names as first class values, a property called mobility. In [1] we studied how the idea of asymmetric sessions, namely sessions with

a bias toward the client side, could be accommodated in the framework of $\pi$-calculus with session types, and we came out with a type relation $\preccurlyeq$ which we dubbed "prefix". Roughly $A \preccurlyeq B$ if channels typed by $B$ allows for longer patterns of actions than those typed by $A$. Surprisingly we realized that this is not a consistent extension of subtyping, since there it holds the principle that $A <: B$ implies $\overline{B} <: \overline{A}$ (where $\overline{A}$ is the syntactical dual type of $A$): since $\overline{\mathsf{end}} = \mathsf{end}$, by adding the axiom $\mathsf{end} <: A$ all types collapse.
The behavioural modeling of higher-order sessions is more problematic, and has been addressed in [8] by means of a calculus generalizing CCS with value passing. By applying the same idea to our first-order behaviors, we obtain higher-order behaviors $?\sigma_1^p\sigma_2$ and $!\sigma_1^p\sigma_2$, representing the receiving and the sending of a session $\sigma_1$ with continuation $\sigma_2$ respectively. The decoration $p = c, s$ that we add as a superscript of $\sigma_1$ is to record the role either as a client or as a server that the actor of the transmitted session is supposed to play. This is important when clients and servers have different rights and duties: a session can be started by $P$ and $Q$, and after a while it can be delegated, say by $P$, to some third party $R$; if $P$ was behaving either as a server or as a client w.r.t. $Q$, it is necessary that $R$ will play the same role w.r.t. $Q$. The point of polarities is to make this apparent in the abstract behavioural description of the various interacting parties. For example the Ballot Service could be modularized in an Authentication Service and a Vote Bookkeeper. The authentication service checks the right of the client to vote, and then passes to him a session in which it can express, as a client, its vote to the Vote Bookkeeper. The behaviour of the Authentication Service is described by the following higher-order session behaviour:

$$\mathtt{Login}.(\overline{\mathtt{Wrong}} \oplus \overline{\mathtt{Ok}}.!(\overline{\mathtt{VoteA}}.(\overline{\mathtt{Va1}} \oplus \overline{\mathtt{Va2}}) \oplus \overline{\mathtt{VoteB}}.(\overline{\mathtt{Vb1}} \oplus \overline{\mathtt{Vb2}}))^c)$$

In view of the above discussion about the sub-behaviour relations, the interaction with $?\sigma_1\sigma_2$ should be triggered by any $!\rho_1^p\rho_2$ such that $\rho_1 \preceq_p \sigma_1$ and not just when $\rho_1 = \sigma_1$.
The definition of the LTS rises some technical problems. First the relations $\preceq_p$ are involved in the definition of the LTS of higher-order behaviours; but they are defined in terms of the compliance relation $\dashv$ which in turn depends on the LTS. This is not novel problem, as it appears also in [5, 17], where it is observed that a suitable stratification of the definition of the LTS actually solves the problem. There is however a second and deeper difficulty, which is connected with the logical complexity of the relations $\dashv$ and $\preceq_p$ that are not immediately seen to be decidable, and not even $\Sigma^0_1$. It is for this reason that we insist in viewing behaviours as a mathematical model of session types, whose subtyping theory is axiomatizable and indeed it is decidable. We eventually prove that the basic properties of compliance and of the sub-behaviour relations hold for the full system of higher-order session behaviours, and in particular that they model both subtyping of session types and two natural extensions of this theory.

The paper is structured as follows: in Section 2 we introduce first-order behaviours, the LTS and the relation of compliance, the central concepts of client/server sub-behaviours. In Section 3 we study orthogonality and the related concept of behavioural subtyping, establishing that (first-order) session types with subtyping are modeled by behaviours and behavioural subtyping. We then introduce in Section 4 higher-order behaviours. We characterize coinductively the relations $\preceq_p$ extended to higher-order behaviours, and prove that behavioural subtyping is the intersection of client/server sub-behaviour relations. We conclude the section by showing that the full session type system with subtyping is modeled by higher-oder behaviours. In Section 5 we relate our work to the literature by crediting the basic notions we study, but also stressing where we depart from the original definitions. Then we conclude.

## 2. Session Behaviours and Sub-Behaviour relations

A Session Behaviour is an abstraction of what is performed by a process on its end of a channel, through which the process interacts with the system it is embedded in.

DEFINITION 2.1 (Session Behaviours). *Let $\mathcal{N}$ be some countable set of symbols and $\overline{\mathcal{N}} = \{\overline{a} \mid a \in \mathcal{N}\}$, with $\mathcal{N} \cap \overline{\mathcal{N}} = \emptyset$. The set $\mathcal{B}$ of* raw behaviour expressions *is defined by the grammar:*

$$\begin{array}{rcl} \sigma & ::= & \mathbf{1} \\ & & \mid a_1.\sigma_1 + \cdots + a_n.\sigma_n \\ & & \mid \overline{a}_1.\sigma_1 \oplus \cdots \oplus \overline{a}_n.\sigma_n \\ & & \mid x \\ & & \mid \mathsf{rec}\, x.\sigma \end{array}$$

*where $n$ is any positive integer, $a_i \in \mathcal{N}$ (hence $\overline{a}_i \in \overline{\mathcal{N}}$) for all $i$, $x$ is a session behaviour variable out of a denumerable set and it is bound in $\mathsf{rec}\, x.\, \sigma$.*
*The set $\mathcal{S}$ of* session behaviours *is the subset of closed raw behaviour expressions such that in $a_1.\sigma_1 + \cdots + a_n.\sigma_n$ and in $\overline{a}_1.\sigma_1 \oplus \cdots \oplus \overline{a}_n.\sigma_n$ the $a_i$ and the $\overline{a}_i$ are, respectively, pairwise distinct, and in $\mathsf{rec}\, x.\sigma$ the expression $\sigma$ is not a variable.*

We abbreviate $a_1.\sigma_1 + \cdots + a_n.\sigma_n$ by $\sum_{i=1}^n a_i.\sigma_i$, and $\overline{a}_1.\sigma_1 \oplus \cdots \oplus \overline{a}_n.\sigma_n$ by $\bigoplus_{i=1}^n \overline{a}_i.\sigma_i$. Moreover, $\sum_{i=1}^1 a_i.\sigma_i$ and $\bigoplus_{i=1}^1 \overline{a}_i.\sigma_i$ are abbreviated by $a_1.\sigma_1$ and $\overline{a}_i.\sigma_1$, respectively. We also use the notations $\sum_{i\in I} a_i.\sigma_i$ and $\bigoplus_{i\in I} \overline{a}_i.\sigma_i$ for finite and not empty $I$. The trailing $\mathbf{1}$ is normally omitted: we write e.g. $a+b$ for $a.\mathbf{1}+b.\mathbf{1}$. Note that recursion in $\mathcal{S}$ is guarded (and hence contractive).

The semantics of session behaviours is given in terms of the following LTS:

DEFINITION 2.2. *Let* $\mathbf{Act} = \mathcal{N} \cup \overline{\mathcal{N}}$ *and* $\oplus, \mathsf{rec} \notin \mathbf{Act}$*; then define the LTS:*

$$a_1.\sigma_1 + \cdots + a_n.\sigma_n \xrightarrow{a_i} \sigma_i \qquad\qquad \overline{a}.\sigma \xrightarrow{\overline{a}} \sigma$$

$$\overline{a}_1.\sigma_1 \oplus \cdots \oplus \overline{a}_n.\sigma_n \xrightarrow{\oplus} \overline{a}_i.\sigma_i \qquad \mathsf{rec}\, x.\sigma \xrightarrow{\mathsf{rec}} \sigma\{\mathsf{rec}\, x.\sigma/x\}$$

*where $n > 1$ in $\overline{a}_1.\sigma_1 \oplus \cdots \oplus \overline{a}_n.\sigma_n$*

We abbreviate $\longrightarrow = \xrightarrow{\oplus} \cup \xrightarrow{\mathsf{rec}}$. Note that neither $\oplus$ nor $\mathsf{rec}$ are names, and do not have any corresponding co-name, so that they are unobservable, and used just as a metanotation in the statements of some lemmas; indeed we adopt the standard $\longrightarrow$ (from CCS without $\tau$) in the subsequent definition of the parallel operator for testing. As usual, we write $\Longrightarrow = \longrightarrow^*$, $\overset{\alpha}{\Longrightarrow} = \longrightarrow^* \xrightarrow{\alpha} \longrightarrow^*$ for $\alpha \in \mathbf{Act}$, $\sigma \overset{s}{\Longrightarrow} \sigma'$ if $s = \alpha_1 \cdots \alpha_n$ and $\sigma \overset{\alpha_1}{\Longrightarrow} \cdots \overset{\alpha_n}{\Longrightarrow} \sigma'$. Also we write $\sigma \longrightarrow$ and $\sigma \xrightarrow{\alpha}$ if there exists $\sigma'$ s.t. $\sigma \longrightarrow \sigma'$ and $\sigma \xrightarrow{\alpha} \sigma'$ respectively, and $\sigma \not\longrightarrow$ when $\neg(\sigma \longrightarrow)$.

For any $\sigma \in \mathcal{S}$ and $R \subseteq \mathcal{S}$, define $\sigma \Downarrow R$ if and only if $R = \{\sigma' \in \mathcal{S} \mid \sigma \Longrightarrow \sigma' \not\longrightarrow\}$.

The syntax of our session behaviours prevents the possibility of infinite $\longrightarrow$ reductions, as formally shown in Lemma 2.3 below. A raw behavioural expression like $\mathsf{rec}\, x.x$ is not, in fact, a session behaviour, since our sintax does not allow the rec operator to be applied on variables. Also an expression like $\mathsf{rec}\, x.(y \oplus z)$ is not a proper session behaviours (nor a raw session behaviour either), since we allow the $+$ and $\oplus$ operators only on *prefixed* behaviours. Such syntactical restrictions do not prevent an abstract representation, in our process algebra, of branching and selection types. At the same time they rule out diverging expression like $\mathsf{rec}\, x.x$ and $\mathsf{rec}\, x.\mathsf{rec}\, y.(x \oplus x)$ which would be meaningless in our session viewpoint of behaviours.

LEMMA 2.3. *For any $\sigma \in \mathcal{S}$, there exists no infinite $\longrightarrow$ reduction out of it. Moreover, for all $\sigma \in \mathcal{S}$ there exists a unique finite and non empty $R \subseteq \mathcal{S}$ such that $\sigma \Downarrow R$, which takes one of the following forms:*

$$\{\mathbf{1}\},\ \{\sum_{i=1}^{n} a_i.\sigma_i\},\ \{\overline{a}_1.\sigma_1, \ldots, \overline{a}_n.\sigma_n\}\ (n > 0).$$

*Proof.* By cases of $\sigma \in \mathcal{S}$: in particular, if $\sigma = \mathsf{rec}\, x.\sigma'$ we know that $\sigma' \neq x$ and that, in case $\sigma'$ be a $\oplus$-term, its components are prefixed. So the longest $\longrightarrow$ reduction sequence out of $\sigma$ necessarily consists in a number of $\xrightarrow{\mathsf{rec}}$ steps (which is equal to the number of occurrences of rec prefixing $\sigma$), followed by at most one $\xrightarrow{\oplus}$ step. $\square$

We write $\sigma \Downarrow \mathbf{1}$, $\sigma \Downarrow \sum_{i=1}^{n} a_i.\sigma_i$ and $\sigma \Downarrow \bigoplus_{i=1}^{n} \overline{a}_i.\sigma_i$ whenever $\sigma \Downarrow R$ with $R$ of one of the three forms above; equivalently we could have set $\sigma \Downarrow \rho$ where $\rho$ is the unique session behaviour such that $\sigma \xrightarrow{\mathsf{rec}}{}^{*} \rho$.

According to standard CCS, the interaction over a channel, between a client and a server is modeled by extending the LTS semantics to pairs of behaviours put in parallel, written $\rho \| \sigma$. Following [13, 16], we introduce a notion of *compliance* between a *client* and a *server* behaviour (roles that are defined in terms of the relation itself).

DEFINITION 2.4 (Communication and Compliance). *Let $\rho\|\sigma$ denote a pair of session behaviors, then over the set of such pairs define the LTS:*

$$\frac{\rho \xrightarrow{\alpha} \rho' \quad \sigma \xrightarrow{\overline{\alpha}} \sigma'}{\rho\|\sigma \longrightarrow \rho'\|\sigma'} \qquad \frac{\rho \longrightarrow \rho'}{\rho\|\sigma \longrightarrow \rho'\|\sigma} \qquad \frac{\sigma \longrightarrow \sigma'}{\rho\|\sigma \longrightarrow \rho\|\sigma'}$$

*where $\alpha \in$ **Act** and $\overline{\alpha}$ is the usual involution.*

*We say that the* client $\rho$ *is* compliant *with the* server $\sigma$, *written $\rho \dashv \sigma$, if for all $\rho', \sigma'$, $\rho\|\sigma \longrightarrow^{*} \rho'\|\sigma' \not\longrightarrow$ implies $\rho' = \mathbf{1}$.*

The informal meaning of $\rho \dashv \sigma$ is that in any reduction out of $\rho\|\sigma$ all of the actions $\alpha$ on the $\rho$'s side are eventually matched by the correspondent co-actions $\overline{\alpha}$ on the $\sigma$'s side. Simple examples of compliance are: $\overline{a}\oplus\overline{b} \dashv a+b$ and also $\overline{a}\oplus\overline{b} \dashv a+b+c$; on the other hand $a+b \not\dashv \overline{a}\oplus\overline{c}$ because of the reduction $a+b\|\overline{a}\oplus\overline{c} \longrightarrow a+b\|\overline{c}$. The $\dashv$ relation is not symmetric: indeed $\mathbf{1} \dashv \sigma$ for any $\sigma$, but e.g. $a \not\dashv \mathbf{1}$. Observe that in [13, 16] $a \oplus b \not\dashv \overline{a} \oplus \overline{b}$ while $a + b \dashv \overline{a} + \overline{b}$. On the other hand, by our syntactical restrictions neither $a \oplus b$ nor $\overline{a} + \overline{b}$ are well formed session behaviours.

REMARK 2.5. Our notion of compliance has been inspired by [13]. W.r.t. the original definition, the present one differs under some relevant respects: first, no expression in $\mathcal{S}$ may diverge, i.e. there is no infinite $\longrightarrow$ reduction out of any $\sigma \in \mathcal{S}$.
Second, if we extended the definition of $\dashv$ to expressions in $\mathcal{B}$, in order to take into account the unguarded recursive expression $\Omega = \mathsf{rec}\, x.\, x$, we would get $\Omega \dashv \sigma$ for all $\sigma$, which is not the case in [13]. This is not a weakness of our approach: $\rho \dashv \sigma$ does not mean that $\rho$ will eventually terminate, but just that any request by $\rho$ will be eventually satisfied by $\sigma$. Therefore we can look at the $\rho \dashv \sigma$ relation as some kind of fairness property with a bias toward $\rho$.
Moreover, and more importantly, we accept the idea that $\rho$'s "requests" might be satisfied infinitely many times, which happens in a non trivial way in cases like $\mathsf{rec}\, x.a.x \dashv \mathsf{rec}\, x.\overline{a}.x$. We would then vacuously have $\Omega \dashv \sigma$ since $\Omega$ has no "request" at all. As a matter of fact, it seems that the present concept of compliance is related to fair testing [19] (but note that we are about two sided communications, whereas the relevant aspects of fairness appear with multiple interactions) and induces a sub-behaviour relation similar to the sub-session concept in [17].
We eventually observe that the definition of $\dashv$ in 2.4 above is literally the same as in Definition 3 in [16]. In fact, by admitting unguarded recursion, one immediately gets $\sigma \dashv \Omega$ for any $\sigma$, which is unacceptable.

LEMMA 2.6. *Let $\rho, \sigma \in \mathcal{S}$:*

1. *$\rho\|\sigma \Longrightarrow \rho'\|\sigma'$ iff for some $s \in \mathbf{Act}^*$ and $\rho', \sigma' \in \mathcal{S}$, $\rho \overset{s}{\Longrightarrow} \rho'$ and $\sigma \overset{\overline{s}}{\Longrightarrow} \sigma'$;*
2. *$\rho\|\sigma \not\longrightarrow$ iff either $\rho = \mathbf{1}$ or $\sigma = \mathbf{1}$ or*
$$\rho \not\longrightarrow\ \&\ \ \sigma \not\longrightarrow\ \&\ \ \neg\exists\alpha \in \mathbf{Act}.\ (\rho \xrightarrow{\alpha}\ \&\ \ \sigma \xrightarrow{\overline{\alpha}})\ ;$$
3. *if $\rho \dashv \sigma$ and $\rho\|\sigma \Longrightarrow \rho'\|\sigma'$ then $\rho' \dashv \sigma'$.*

*Proof.* Immediate consequences of Def. 2.4. $\square$

It is possible to get a coinductive characterization of the compliance relation.

DEFINITION 2.7 (Coinductive Compliance). *Define the operator $\mathcal{F} : \mathcal{S}^2 \to \mathcal{S}^2$ such that for all $\mathcal{R}$, $(\rho, \sigma) \in \mathcal{F}(\mathcal{R})$ if and only if one of the following conditions holds:*

1. *$\rho \Downarrow \mathbf{1}$*
2. *$\rho \Downarrow \sum_{i\in I} a_i.\rho_i$ & $\exists J \subseteq I, \{\sigma_j\}_{j\in J}.[\ J \neq \emptyset$ & $\sigma \Downarrow \bigoplus_{j\in J} \overline{a}_j.\sigma_j$ & $\forall j \in J.\ \rho_j \mathcal{R} \sigma_j]$*
3. *$\rho \Downarrow \bigoplus_{i\in I} \overline{a}_i.\rho_i$ & $I \neq \emptyset$ &
$\exists J \supseteq I, \{\sigma_j\}_{j\in J}.\ [\sigma \Downarrow \sum_{j\in J} a_j.\sigma_j$ & $\forall i \in I.\ \rho_i \mathcal{R} \sigma_i]$.*

*A relation $\mathcal{R} \subseteq \mathcal{S}^2$ is a* coinductive compliance *if and only if $\mathcal{R} \subseteq \mathcal{F}(\mathcal{R})$.*

It is immediate to check that $\mathcal{F}$ is monotonic. Set $\mathcal{F}^0 = \mathcal{S}^2$, and $\mathcal{F}^{k+1} = \mathcal{F}(\mathcal{F}^k)$: by Tarski theorem $\bigcap_k \mathcal{F}^k$ is the greatest fixed point of $\mathcal{F}$, which is a coinductive compliance. We write $|s|$ for the length of $s \in \mathbf{Act}^*$.

LEMMA 2.8. *For all $k$ and $\rho, \sigma \in \mathcal{S}$, if $(\rho, \sigma) \in \mathcal{F}^k$ then for all $s \in \mathbf{Act}^*$ with $|s| < k$ and such that $\rho \overset{s}{\Longrightarrow} \rho'$ and $\sigma \overset{\overline{s}}{\Longrightarrow} \sigma'$ for some $\rho', \sigma' \in \mathcal{S}$, either $\rho' \Downarrow \mathbf{1}$, or $\rho'\|\sigma' \longrightarrow$.*

*Proof.* By induction over $k$. The case $k = 0$ is trivial, since there exists no $s$ with $|s| < 0$.
Suppose that $(\rho, \sigma) \in \mathcal{F}^{k+1} = \mathcal{F}(\mathcal{F}^k)$, and that $\rho \overset{s}{\Longrightarrow} \rho'$ and $\sigma \overset{\overline{s}}{\Longrightarrow} \sigma'$ for some $s$ with $|s| < k + 1$. If $\rho' \Downarrow \mathbf{1}$ then there is nothing to prove. Otherwise suppose that $|s| = 0$: then either $\rho \Downarrow \sum_{i\in I} a_i.\rho_i$ and $\sigma \Downarrow \bigoplus_{j\in J} \overline{a}_j.\sigma_j$ for some $J \subseteq I$, or $\rho \Downarrow \bigoplus_{i\in I} \overline{a}_i.\rho_i$ and $\sigma \Downarrow \sum_{j\in J} a_j.\sigma_j$ for some $J \supseteq I$. In both cases there exists $\alpha$ such that $\rho \Longrightarrow \rho' \overset{\alpha}{\Longrightarrow}$ and $\sigma \Longrightarrow \sigma' \overset{\overline{\alpha}}{\Longrightarrow}$, which implies $\rho'\|\sigma' \longrightarrow$.
If instead $|s| > 0$, we have that for some $\alpha, s', \rho'', \sigma''$, $s = \alpha s'$, $\rho \overset{\alpha}{\Longrightarrow} \rho'' \overset{s'}{\Longrightarrow} \rho'$, and that $(\rho'', \sigma'') \in \mathcal{F}^k$. Then, observing that $|s'| < k$, the thesis follows by the induction hypothesis. $\square$

PROPOSITION 2.9. *The relation $\dashv$ is the largest coinductive compliance.*

*Proof.* First we prove that $\dashv$ is a coinductive compliance, namely that $\dashv \subseteq \mathcal{F}(\dashv)$. Let $\rho \dashv \sigma$ and $\rho \Downarrow \rho'$, $\sigma \Downarrow \sigma'$. By Lemma 2.3 there are three possible cases according to the shape of $\rho'$:

$\rho' = \mathbf{1}$: then $(\rho, \sigma) \in \mathcal{F}(\dashv)$ by condition (1) of Def. 2.7.

$\rho' = \sum_{i\in I} a_i.\rho_i$: since $\rho' \neq \mathbf{1}$, $\rho' \not\longrightarrow$, and $\rho' \dashv \sigma'$ by (3) of Lemma 2.6, it is the case that $\sigma' \xrightarrow{\overline{a}_j} \sigma_j$ for some $j \in I$, and some $\sigma_j$, such that $\rho_j \dashv \sigma_j$; hence clause (2) of Def. 2.7 is satisfied by taking $J = \{j\}$.

$\rho' = \bigoplus_{i\in I} \overline{a}_i.\rho_i$: similarly it must be the case that $\rho' \dashv \sigma'$; now for any $i \in I$ we have $\rho' \xrightarrow{\overline{a}_i} \rho_i$, so that by (2) of Lemma 2.6, for all $i \in I$ there is $\sigma_i$ s.t. $\sigma' \xrightarrow{a_i} \sigma_i$ and $\rho_i \dashv \sigma_i$. By the definition of $\mathcal{S}$ this is only possible if $\sigma'$ is a sum of the shape $\sum_{j\in J} a_j.\sigma_j$ where $J \supseteq I$. Consequently also (3) of Def. 2.7 is satisfied.

Vice versa suppose that $\rho \not\dashv \sigma$, that is for certain $\rho', \sigma'$ it is the case that $\rho\|\sigma \Longrightarrow \rho'\|\sigma' \not\longrightarrow$ but $\rho' \neq \mathbf{1}$. By (2) of Lemma 2.6 there exists $s \in \mathbf{Act}^*$ such that $\rho \overset{s}{\Longrightarrow} \rho'$ and $\sigma \overset{\overline{s}}{\Longrightarrow} \sigma'$; on the other hand, since $\rho'\|\sigma' \not\longrightarrow$ and $\rho' \neq \mathbf{1}$ there exists no $\alpha \in \mathbf{Act}$ such that both $\rho' \xrightarrow{\alpha}$ and $\sigma' \xrightarrow{\overline{\alpha}}$. Therefore, by Lemma 2.8, $(\rho, \sigma) \notin \mathcal{F}^k$, for all $k > |s|$. □

### 2.1 Client/Server Sub-behaviours

The notion of compliance, being an asymmetric one, naturally induces two notions of sub-behaviour, for client and server side respectively.

DEFINITION 2.10 (Client/Server Sub Behaviours). *For $\sigma, \rho \in \mathcal{S}$, let*
$\mathsf{Client}(\sigma) = \{\rho \in \mathcal{S} \mid \rho \dashv \sigma\}$ *and* $\mathsf{Server}(\rho) = \{\sigma \in \mathcal{S} \mid \rho \dashv \sigma\}$. *Then define the relations:*

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

*We also set $\sigma \simeq_s \sigma'$ if both $\sigma \preceq_s \sigma'$ and $\sigma' \preceq_s \sigma$, and define $\sigma \simeq_c \sigma'$ similarly.*

A syntactical concept of *duality* on $\mathcal{S}$ can be obtained by interchanging $a$ with $\overline{a}$ and $+$ with $\oplus$; the *dual* of $\sigma$ is denoted by $\overline{\sigma}$, where $\overline{\mathbf{1}} = \mathbf{1}$, $\overline{x} = x$ and $\overline{\mathsf{rec}\, x.\sigma} = \mathsf{rec}\, x.\overline{\sigma}$. As expected, $\overline{\overline{\sigma}} = \sigma$ for all $\sigma$.

It is easy to check that the relation $\mathcal{R} = \{(\sigma, \overline{\sigma}) \mid \sigma \in \mathcal{S}\}$ is a coinductive compliance; hence $\sigma \dashv \overline{\sigma}$ for any $\sigma$, by Proposition 2.9. By involutivity of $\bar{\cdot}$ we also have $\mathcal{R} = \{(\overline{\sigma}, \sigma) \mid \sigma \in \mathcal{S}\}$, so that also $\overline{\sigma} \dashv \sigma$. It follows that $\overline{\sigma} \in \mathsf{Client}(\sigma) \cap \mathsf{Server}(\sigma)$.
It is false, instead, that $\rho \dashv \sigma$ implies $\overline{\rho} \dashv \overline{\sigma}$: in fact $a+b+c \dashv \overline{a} \oplus \overline{b}$, but $\overline{a} \oplus \overline{b} \oplus \overline{c} \not\dashv a + b$.

We prove below that $\overline{\sigma}$ is the minimal behaviour in $\mathsf{Client}(\sigma)$ w.r.t. $\preceq_c$ and the minimal behaviour in $\mathsf{Server}(\sigma)$ w.r.t. $\preceq_s$.

LEMMA 2.11. *For any $\sigma, \sigma' \in \mathcal{S}$ and $s \in \mathbf{Act}^*$, if $\sigma \overset{s}{\Longrightarrow} \sigma'$ then $\overline{\sigma} \overset{\overline{s}}{\Longrightarrow} \overline{\sigma'}$; similarly if $\sigma \Downarrow \tau$ then $\overline{\sigma} \Downarrow \overline{\tau}$*

*Proof.* Straightforward. □

LEMMA 2.12. *For all $\rho, \sigma, \tau \in \mathcal{S}$, if $\rho \dashv \tau$ and $\overline{\tau} \dashv \sigma$ then $\rho \dashv \sigma$.*

*Proof.* By Prop. 2.9 it suffices to prove that $(\rho, \sigma) \in \mathcal{F}^k$ for all $k$. More precisely we prove that for all $k$ and for all $\rho', \tau', \sigma'$, if $(\rho', \tau') \in \mathcal{F}^k$ and $(\overline{\tau'}, \sigma') \in \mathcal{F}^k$ then $(\rho', \sigma') \in \mathcal{F}^k$.
Let $k > 0$ and $\neg(\rho \Downarrow \mathbf{1})$, since otherwise the thesis is trivial. There are two cases: if $\rho \Downarrow \sum_{i\in I} a_i.\rho_i$ then, by $\rho \dashv \tau$, $(\rho, \tau) \in \mathcal{F}^k$, so that there exists a non empty $J \subseteq I$ such that $\tau \Downarrow \bigoplus_{j\in J} \overline{a}_j.\tau_j$ and $(\rho_j, \tau_j) \in \mathcal{F}^{k-1}$ for all $j \in J$. By Lemma 2.11 $\overline{\tau} \Downarrow \sum_{j\in J} a_j.\overline{\tau}_j$ so that, because of $\overline{\tau} \dashv \sigma$, $(\overline{\tau}, \sigma) \in \mathcal{F}^k$ and there exists a non empty $H \subseteq J$ such that $\sigma \Downarrow \bigoplus_{h\in H} \overline{a}_h.\sigma_h$ and $(\overline{\tau}_h, \sigma_h) \in \mathcal{F}^{k-1}$ for all $h \in H$. Since $H \subseteq J \subseteq I$, by the induction hypothesis we have that $(\rho_h, \sigma_h) \in \mathcal{F}^{k-1}$ for all $h \in H$, so that $(\rho, \sigma) \in \mathcal{F}^k$ by definition of $\mathcal{F}$.
The case $\rho \Downarrow \bigoplus_{i\in I} \overline{a}_i.\rho_i$ is similar. □

PROPOSITION 2.13. *Let $\tau \in \mathcal{S}$. Then*

1. *$\overline{\tau}$ is the minimum client of $\tau$, i.e. $\forall \rho \in \mathsf{Client}(\tau).\ \overline{\tau} \preceq_c \rho$.*
2. *$\overline{\tau}$ is the minimum server of $\tau$, i.e. $\forall \sigma \in \mathsf{Server}(\tau).\ \overline{\tau} \preceq_s \sigma$.*

*Proof.* We observed that $\overline{\tau} \in \mathsf{Client}(\tau) \cup \mathsf{Server}(\tau)$, hence it remains to show the minimality property w.r.t. $\preceq_c$ and $\preceq_s$ respectively. (1) Let $\rho \in \mathsf{Client}(\tau)$. In order to establish $\overline{\tau} \preceq_c \rho$, let $\sigma \in \mathsf{Server}(\overline{\tau})$. Then we have $\rho \dashv \tau$ and $\overline{\tau} \dashv \sigma$. By Lemma 2.12 we know that $\rho \dashv \sigma$, i.e. $\sigma \in \mathsf{Server}(\rho)$. Hence $\overline{\tau} \preceq_c \rho$. The proof of (2) is similar, using Lemma 2.12 by observing that the $\bar{\cdot}$ operation is involutive. □

REMARK 2.14. The strong properties stated in Proposition 2.13 above are due to the lack of *implicit* nondeterminism in session behaviours. The nondeterminism in our system is in fact only *explicit*, i.e. exclusively due to the $\oplus$ operator.
A light form of implicit nondeterminism could be introduced by relaxing the constraint imposing prefixes to be pairwise distinct: in fact, if we let $a + a.b$ be a correct session behaviour, $(a + a.b)\|\overline{a}$ could be reduced both to $\mathbf{1}$ and $b$. By introducing such implicit nondeterminism, it is easy to check that $\overline{a} \oplus \overline{a}.\overline{b} \not\dashv a + a.b$ and that the minimum of $\mathsf{Client}(a + a.b)$ is actually $\overline{a}$. On the other hand, $a + a.b \not\dashv \overline{a} \oplus \overline{a}.\overline{b}$ and the minimum of $\mathsf{Server}(a + a.b)$ is $\overline{a}.\overline{b}$. We could also have no minimum at all: it is immediate to check that, by allowing such an implicit nondeterminism, for the term $a.\overline{b} + a.\overline{c}$ we would have $\mathsf{Server}(a.\overline{b} + a.\overline{c}) = \emptyset$.
Relaxing the constraint that $a_1, \ldots, a_n \in \mathcal{N}$ in $a_1.\sigma_1 + \cdots + a_n$ and $\overline{a}_1, \ldots, \overline{a}_n \in \overline{\mathcal{N}}$ in $\overline{a}_1.\sigma_1 \oplus \cdots \oplus \overline{a}_n.\sigma_n$, would, instead, introduce even more implicit nondeterminins, taking us completely outside of the "session" context.

## 3. Orthogonality and Subtyping

We define a concept of sub-behaviour in terms of the greatest symmetric relation included in $\dashv$, which we call *orthogonality*. It comes out that the sub-behavior relation is the intersection of $\preceq_c$ and $\preceq_s$, and that it is a model of Gay-Hole subtyping theory of session types introduced in [11].

DEFINITION 3.1 (Orthogonality). *We say that $\rho$ is* orthogonal *to $\sigma$, written $\rho \perp \sigma$, if $\rho \dashv \sigma$ and $\sigma \dashv \rho$.*

A coinductive characterization of $\perp$ can be obtained by defining a binary relation $\mathcal{R} \subseteq \mathcal{S}^2$ as a *coinductive orthogonality* if it is a symmetric coinductive compliance.

LEMMA 3.2. *$\perp$ is the largest coinductive orthogonality.*

*Proof.* Immediate by observing that, by Prop. 2.9, $\perp\ = \dashv \cap \dashv^{-1}$ implies that $\rho \perp \sigma$ iff $(\rho, \sigma), (\sigma, \rho) \in \mathcal{F}^k$ for all $k$. □

For $A \subseteq \mathcal{S}$, define $A^\perp = \{\sigma \in \mathcal{S} \mid \exists \tau \in A.\ \sigma \perp \tau\}$; below by $\sigma^\perp$ it is meant $\{\sigma\}^\perp$.

LEMMA 3.3.

1. *$\mathbf{1}^\perp = \{\sigma \in \mathcal{S} \mid \sigma \Downarrow \mathbf{1}\}$;*
2. *$(\mathsf{rec}\, x.\sigma)^\perp = (\sigma\{\mathsf{rec}\, x.\sigma/x\})^\perp$;*
3. *$(\sum_{i\in I} a_i.\sigma_i)^\perp = \{\tau \in \mathcal{S} \mid \exists J \subseteq I, \{\tau_j\}_{j\in J}.\ \tau \Downarrow \bigoplus_{j\in J} \overline{a}_j.\tau_j\ \&\ \forall j \in J.\ \tau_j \in \sigma_j^\perp\}$;*
4. *$(\bigoplus_{i\in I} \overline{a}_i.\sigma_i)^\perp = \{\tau \in \mathcal{S} \mid \exists J \supseteq I, \{\tau_j\}_{j\in J}.\ \tau \Downarrow \sum_{j\in J} a_j.\tau_j\ \&\ \forall i \in I.\ \tau_i \in \sigma_i^\perp\}$.*

*Proof.* (1): $\sigma \perp \mathbf{1}$ iff $\sigma \dashv \mathbf{1}$ (since $\mathbf{1} \dashv \sigma$ for any $\sigma$) which is equivalent to $\sigma \Downarrow \mathbf{1}$ by Prop. 2.9.
By Lemma 2.3, $\mathsf{rec}\, x.\sigma \Downarrow \tau$ iff $\sigma\{\mathsf{rec}\, x.\sigma/x\} \Downarrow \tau$; hence (2) holds by Lemma 3.2 and Def. 2.7.
If $\sum_{i\in I} a_i.\sigma_i \dashv \tau$ then by Prop. 2.9 $\tau \Downarrow \bigoplus_{j\in J} \overline{a}_j.\tau_j$ for some non empty $J \subseteq I$, where $\sigma_j \dashv \tau_j$ for all $j \in J$. Vice versa if $\tau \dashv \sum_{i\in I} a_i.\sigma_i$ then $\tau \Downarrow \bigoplus_{k\in K} \overline{a}_k.\tau'_k$: by the unicity of the $\tau'$ s.t. $\tau \Downarrow \tau'$ (Lemma 2.3) we conclude that $\bigoplus_{j\in J} \overline{a}_j.\tau_j = \bigoplus_{k\in K} \overline{a}_k.\tau'_k$, which establishes (3).
The proof of (4) is analogous. □

Next we define a concept of sub-behavior which is the semantic counterpart of the notion of subtyping. This concept is the restriction to $\mathcal{S}$ of the subsieve relation in [5]. We name it *semantic subtyping* because it is based on a set theoretic interpretation of the subtyping relation, and because, as we prove below, it models Gay and Hole subtype relation among session types.

DEFINITION 3.4 (Semantic Subtyping).
*For $\sigma, \tau \in \mathcal{S}$, $\sigma \preceq: \tau$ if and only if $\sigma^{\perp} \subseteq \tau^{\perp}$.*

As anticipated at the beginning of this section, there is a precise correspondence between the client/server sub-behaviour relations and the semantic subtyping, namely:

THEOREM 3.5. $\preceq: \; = \; \preceq_c \cap \preceq_s$.

The proof of this theorem is deferred to Section 4, where it is established for the more general case of higher-order session behaviours.

COROLLARY 3.6. *For any $\sigma \in \mathcal{S}$, $\overline{\sigma}$ is minimal in $\sigma^{\perp}$ w.r.t. $\preceq:$ . Therefore, for any $\sigma, \tau \in \mathcal{S}$,*

$$\sigma \preceq: \tau \quad \textit{if and only if} \quad \overline{\tau} \preceq: \overline{\sigma}.$$

*Proof.* Concerning the minimality, if $\tau \in \sigma^{\perp}$ then $\tau \in \mathsf{Client}(\sigma) \cap \mathsf{Server}(\sigma)$; by Prop. 2.13 both $\overline{\sigma} \preceq_c \tau$ and $\overline{\sigma} \preceq_s \tau$, so that $\overline{\sigma} \preceq: \tau$ by Theorem 3.5.
For the only if implication, if $\sigma \preceq: \tau$ then $\overline{\sigma} \in \sigma^{\perp} \subseteq \tau^{\perp}$, which implies $\overline{\tau} \preceq: \overline{\sigma}$, as $\overline{\tau}$ is minimal in $\tau^{\perp}$. The inverse implication follows by the involutivity of $\overline{\cdot}$. □

To provide the correlation between semantic subtyping and the subtype relation on session types, let us first recall the definition of session type and give a formal interpretation of session types into session behaviours.
By considering only the session types (the live channel types) and disregarding sorts in the terminology used in [12], a *first order session type* $A$ is an expression of the language defined by the grammar:

$$\begin{array}{rl} A, B ::= & \mathsf{end} \mid ?(\gamma)A \mid ![\gamma]A \mid \&\langle \ell_i : B_i \mid i \in I\rangle \mid \\ & \oplus\langle \ell_i : B_i \mid i \in I\rangle \mid X \mid \mu X.\ A \end{array}$$

where $\gamma = \mathbf{Int}, \mathbf{Bool}, \ldots$ is some ground type and the recursion $\mu X.\ A$ is guarded. Given that, it is straightforward to interpret session types into $\mathcal{S}$:

$$\begin{array}{rclcrcl} [\![X]\!] & = & x & & & & \\ [\![\mathsf{end}]\!] & = & \mathbf{1} & & [\![\mu X.\ A]\!] & = & \mathsf{rec}\, x.\, [\![A]\!] \\ [\![?(\gamma)A]\!] & = & \gamma.[\![A]\!] & & [\![![\gamma]A]\!] & = & \overline{\gamma}.[\![A]\!] \end{array}$$

$$[\![\&\langle \ell_i : B_i \mid i \in I\rangle]\!] \;=\; \textstyle\sum_{i\in I} \ell_i.[\![B_i]\!]$$

$$[\![\oplus\langle \ell_i : B_i \mid i \in I\rangle]\!] \;=\; \textstyle\bigoplus_{i\in I} \overline{\ell}_i.[\![B_i]\!]$$

given a canonic choice of the variables $x$ associated to $X$ (e.g. just the identity) and that the ground types $\gamma$ and the labels $\ell$ are in $\mathcal{N}$.
Over session types it is defined a notion of duality: $\overline{A}$ is obtained from $A$ by exchanging $?$ with $!$ and $\&$ with $\oplus$. Then we immediately have:

PROPOSITION 3.7. *For any first-order session type $A$ and $\sigma \in \mathcal{S}$:*

$$\sigma = [\![A]\!] \quad \textit{if and only if} \quad \overline{\sigma} = [\![\overline{A}]\!].$$

*Proof.* By a straightforward induction over $A$. □

In [11] the subtyping system is proved sound by a subject reduction theorem. An equivalent formulation of the system is given in Figure 1. It is inspired by the analogous system in [2], for subtyping recursive types of simply typed $\lambda$-calculus.

The judgments $\Gamma \vdash A <: B$ are formed by a finite set $\Gamma$ of type inequalities $C <: D$, and $\Gamma, C <: D$ abbreviates $\Gamma \cup \{C <: D\}$, assuming that $C <: D \notin \Gamma$.

DEFINITION 3.8 (Session Subtyping).
*Let $\Gamma = \{A_i <: B_i \mid i \leq k\}$ be a finite (possibly empty) set of statements $A_i <: B_i$ where $A_i, B_i$ are session types; then we say that $A$ is a subtype of $B$, written $A <: B$, if $\emptyset \vdash A <: B$ is derivable using the axioms and rules in Figure 1.*

The system in Figure 1 essentially embodies the coinductive definition in [11]. We omit the proof that the two systems are equivalent; rather we show by an example how the apparently circular rule (T-SUB-&) can be used.
Given $A = \mu X.\&\langle \ell : X\rangle$ and $B = \mu X.\&\langle \ell : X, \ell' : C\rangle$, set $\Gamma = \{\&\langle \ell : A\rangle <: \&\langle \ell : B, \ell' : C\rangle\}$.
By (T-SUB-HYP) we have $\Gamma \vdash \&\langle \ell : A\rangle <: \&\langle \ell : B, \ell' : C\rangle$, while $\Gamma \vdash A <: \&\langle \ell : A\rangle$ by (T-SUB-UNFOLD), and
$\Gamma \vdash \&\langle \ell : B, \ell' : C\rangle <: B$ by (T-SUB-FOLD).
Hence, by (T-SUB-TRANS), we derive $\Gamma \vdash A <: B$, which is an instance of the premise of rule (T-SUB-&). Therefore we have $\vdash \&\langle \ell : A\rangle <: \&\langle \ell : B, \ell' : C\rangle$, from which we obtain $\vdash A <: B$ using (T-SUB-UNFOLD), (T-SUB-FOLD) and (T-SUB-TRANS).
The usage of rule (T-SUB-$\oplus$) is similar.

The main result of this section is that if $A <: B$ (or more precisely if $\vdash A <: B$ is derivable) then $A \preceq: B$. As usual we prove a more general statement: first define $\models A <: B$ iff $[\![A]\!] \preceq: [\![B]\!]$; $\models \Gamma$ iff $\models C <: D$ for all $C <: D \in \Gamma$; and $\Gamma \models A <: B$ iff $\models \Gamma$ implies $\models A <: B$. Then we show that $\Gamma \vdash A <: B$ implies $\Gamma \models A <: B$ using stratified versions of the sub-behaviour relations and of the definitions just given.

DEFINITION 3.9. *For all $k \in \mathbb{N}$, let $\preceq:_k \subseteq \mathcal{S}^2$ be inductively defined by $\preceq:_0 = \mathcal{S}^2$ and by $\sigma \preceq:_{k+1} \tau$ iff:*

1. *$\sigma \Downarrow \mathbf{1}$ implies $\tau \Downarrow \mathbf{1}$*
2. *$\sigma \Downarrow \sum_{i\in I} a_i.\sigma_i$ implies $\tau \Downarrow \sum_{j\in J} a_j.\tau_j$ for some $J \supseteq I$ and $\sigma_i \preceq:_k \tau_i$ for all $i \in I$, and*
3. *$\sigma \Downarrow \bigoplus_{i\in I} \overline{a}_i.\sigma_i$ implies $\tau \Downarrow \bigoplus_{j\in J} \overline{a}_j.\tau_j$ for some $\emptyset \neq J \subseteq I$ and $\sigma_j \preceq:_k \tau_j$ for all $j \in J$.*

LEMMA 3.10. $\preceq: \; = \bigcap_{k\in\mathbb{N}} \preceq:_k$.

*Proof.* Similar to that of Prop. 2.9 using Lemma 3.3. □

We define:

1. $\models_k A <: B$ iff $[\![A]\!] \preceq:_k [\![B]\!]$;
2. $\models_k \Gamma$ iff $\models_k C <: D$ for all $C <: D \in \Gamma$;
3. $\Gamma \models_k A <: B$ iff $\models_k \Gamma$ implies $\models_k A <: B$.

THEOREM 3.11 (Soundness w.r.t. Subtyping).
*If $\Gamma \vdash A <: B$ then $\Gamma \models A <: B$.*

*Proof.* By Lemma 3.10 we know that $\Gamma \models A <: B$ iff $\Gamma \models_k A <: B$ for all $k \in \mathbb{N}$. The proof is via a principal induction

$$\Gamma \vdash A <: A \ (\text{T-SUB-ID}) \qquad \frac{\Gamma \vdash A <: B \quad \Gamma \vdash B <: C}{\Gamma \vdash A <: C} \ (\text{T-SUB-TRANS}) \qquad \Gamma, A <: B \vdash A <: B \ (\text{T-SUB-HYP})$$

$$\Gamma \vdash \mu X.A <: A\{\mu X.A/X\} \ (\text{T-SUB-UNFOLD}) \qquad \Gamma \vdash A\{\mu X.A/X\} <: \mu X.A \ (\text{T-SUB-FOLD})$$

$$\frac{\Gamma, \&_{i\in I}\langle \ell_i : A_i\rangle <: \&_{j\in J}\langle \ell_j : B_j\rangle \vdash A_i <: B_i \quad \forall i \in I \quad I \subseteq J}{\Gamma \vdash \&_{i\in I}\langle \ell_i : A_i\rangle <: \&_{j\in J}\langle \ell_j : B_j\rangle} \ (\text{T-SUB-}\&)$$

$$\frac{\Gamma, \oplus_{i\in I}\langle \ell_i : A_i\rangle <: \oplus_{j\in J}\langle \ell_j : B_j\rangle \vdash A_j <: B_j \quad \forall j \in J \quad I \supseteq J}{\Gamma \vdash \oplus_{i\in I}\langle \ell_i : A_i\rangle <: \oplus_{j\in J}\langle \ell_j : B_j\rangle} \ (\text{T-SUB-}\oplus)$$

**Figure 1.** Coinductive subtyping of first order session types

over the derivation of $\Gamma \vdash A <: B$, and a subordinate induction over $k$.
All cases are trivial but for rules T-SUB-& and T-SUB-$\oplus$. Let us consider the first case (the second one is similar).
$\Gamma \models_0 \&_{i\in I}\langle \ell_i : A_i\rangle <: \&_{j\in J}\langle \ell_j : B_j\rangle$ is trivially true, since $[\![\&_{i\in I}\langle \ell_i : A_i\rangle]\!] \preceq:_0 [\![\&_{j\in J}\langle \ell_j : B_j\rangle]\!]$ holds always.
To establish $\Gamma \models_{k+1} \&_{i\in I}\langle \ell_i : A_i\rangle <: \&_{j\in J}\langle \ell_j : B_j\rangle$, assume $\models_{k+1} \Gamma$.
Since $\preceq:_k \supseteq \preceq:_{k+1}$, we have that $\models_k \Gamma$; by the secondary induction hypothesis: $\Gamma \models_k \&_{i\in I}\langle \ell_i : A_i\rangle <: \&_{j\in J}\langle \ell_j : B_j\rangle$, so that it must be the case that $\models_k \&_{i\in I}\langle \ell_i : A_i\rangle <: \&_{j\in J}\langle \ell_j : B_j\rangle$.
It follows that $\models_k \Gamma, \&_{i\in I}\langle \ell_i : A_i\rangle <: \&_{j\in J}\langle \ell_j : B_j\rangle$; hence by the primary induction hypothesis:
$\Gamma, \&_{i\in I}\langle \ell_i : A_i\rangle <: \&_{j\in J}\langle \ell_j : B_j\rangle \models A_i <: B_i$ for all $i \in I$.
We conclude that $\models_k A_i <: B_i$ for all $i \in I$.
Let $[\![\&_{i\in I}\langle \ell_i : A_i\rangle]\!] = \sum_{i\in I} \ell_i.\sigma_i$, where $\sigma_i = [\![A_i]\!]$, and similarly $[\![\&_{j\in J}\langle \ell_j : B_j\rangle]\!] = \sum_{j\in J} \ell_j.\tau_j$ with $\tau_j = [\![B_j]\!]$. Then trivially $\sum_{i\in I} \ell_i.\sigma_i \Downarrow \sum_{i\in I} \ell_i.\sigma_i$ and $\sum_{j\in J} \ell_j.\tau_j \Downarrow \sum_{j\in J} \ell_j.\tau_j$.
On the other hand we know that $\sigma_i \preceq:_k \tau_i$ for all $i \in I$, since $\models_k A_i <: B_i$. Hence $\sum_{i\in I} \ell_i.\sigma_i \preceq:_{k+1} \sum_{j\in J} \ell_j.\tau_j$, that is $\models_{k+1} \&_{i\in I}\langle \ell_i : A_i\rangle <: \&_{j\in J}\langle \ell_j : B_j\rangle$ as desired. $\square$

A relevant consequence of this theorem and of (1) of Lemma 3.3 is that it is inconsistent with the behavioural interpretation of session types to postulate $\mathsf{end} <: A$ for any $A$ (a similar but weaker result was stated in [1]). However this can be done by introducing two new relations among types, denoted by $\leq_c$ and $\leq_s$ below, that can be naturally interpreted by $\preceq_c$ and $\preceq_s$.

DEFINITION 3.12 ($\leq_s$ and $\leq_c$. First-order).

1. *$\leq_c$ is the relation on session types obtained from the axioms and rules of Figure 1 by replacing $<:$ by $\leq_c$ and by adding the axiom*

$$\Gamma \vdash A \leq_c \mathsf{end} \quad (\text{T-AX-C})$$

2. *$\leq_c$ is the relation on session types defined as $\leq_c$ but using the following axiom instead of* (T-AX-C)*:*

$$\Gamma \vdash \mathsf{end} \leq_s A \quad (\text{T-AX-S})$$

The above defined relations can be proved sound w.r.t. the sub-behaviour relations. Let the relations $\Gamma \models A \leq_c B$ and $\Gamma \models A \leq_s B$ be defined similarly to $\Gamma \models A <: B$, with $\preceq_c$ and $\preceq_s$ in place of $\preceq:$ respectively.

THEOREM 3.13 (Soundness w.r.t. Sub-behaviours).

1. *If* $\Gamma \vdash A \leq_c B$ *then* $\Gamma \models A \leq_c B$.
2. *If* $\Gamma \vdash A \leq_s B$ *then* $\Gamma \models A \leq_s B$.

*Proof.* Observe that $[\![\mathsf{end}]\!] = \mathbf{1}$ implies $[\![\mathsf{end}]\!] \preceq_s [\![A]\!]$ and $[\![A]\!] \preceq_c [\![\mathsf{end}]\!]$ for any $A$. The rest of the proof follows the same lines of the proof of Theorem 3.11. $\square$

REMARK 3.14. In [1], the relation '$\preccurlyeq$' on Session Types was introduced in order to formalize the notion of "enabling *longer* sequences of interactions". Its definition essentially corresponds to that of $\leq_s$ above, but replacing the side conditions of (T-SUB-&) and (T-SUB-$\oplus$) by the condition $I = J$. Now we can put '$\preccurlyeq$' in a broader and clearer context since it is possible to prove that

$$\leq_s \;=\; <: \circ \preccurlyeq \;=\; \preccurlyeq \circ <: \text{ and } \leq_c \;=\; <: \circ \succcurlyeq \;=\; \succcurlyeq \circ <:$$

We claim that similar decomposition properties apply to the higher-order session types as well.
It is natural to use our sub-session relations in type systems for processes that interact through session typed channels and such that the two partners of an interaction over a channel c know the role they shall be playing over their respective ends of c, either as a client or a server. The interactions can be asymmetric and the relations $\leq_s$ and $\leq_c$ can be used in two distinct subsumption rules, as shown in the following example rules, resembling the rules of the well known process calculus with session types introduced in [12]. In such rules, polarities (denoted by the superscripts 's' and 'c') on the (live) channel $\kappa$ are used not only to match the two ends of the channel, but also to distinguish the role played by the process $P$ over its end of $\kappa$.

$$\frac{\Gamma \vdash P \triangleright \Delta \cdot \kappa^s : S \quad S' \leq_s S}{\Gamma \vdash P \triangleright \Delta \cdot \kappa^s : S'} \qquad \frac{\Gamma \vdash P \triangleright \Delta \cdot \kappa^c : S \quad S' \leq_c S}{\Gamma \vdash P \triangleright \Delta \cdot \kappa^c : S'}$$

## 4. Higher-order Session Behaviours

A session is higher order if the private channel of the session can either send or receive channel names. This is reflected in the session type theory by extending the grammar of first order types as follows:

$$A, B ::= \ldots \mid ?(A)B \mid ![A]B,$$

where $A$ can be any session type. The modeling of (full) session types with behaviours requires the introduction of *higher order session behaviours* which are essentially terms of CCS with value passing, but for the fact that the set of values coincides with the set of behaviours itself.

DEFINITION 4.1 (Higher-Order Session Behaviour). *A* higher order behaviour *is defined by extending the grammar in Definition 2.1:*

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

*where $p \in \{s, c\}$. We call $\mathcal{HS}$ the resulting set.*

$$?\rho^p.\sigma \xrightarrow{?\rho^p} \sigma \qquad\qquad !\rho^p.\sigma \xrightarrow{!\rho^p} \sigma$$

$$\frac{\sigma \xrightarrow{?\rho_2^p} \sigma' \quad \tau \xrightarrow{!\rho_1^p} \tau' \quad \rho_1 \preceq_p \rho_2}{\sigma \| \tau \longrightarrow \sigma' \| \tau'}$$

$$\frac{\sigma \xrightarrow{!\rho_1^p} \sigma' \quad \tau \xrightarrow{?\rho_2^p} \tau' \quad \rho_1 \preceq_p \rho_2}{\sigma \| \tau \longrightarrow \sigma' \| \tau'}$$

**Figure 2.** The higher order labelled transition system

This definition follows similar constructions in [5, 17]. A technical difference is the use of polarities; they have been introduced in [11] and used in [21] to keep track of the pairing of the two ends of private channels of sessions; we use polarities also to distinguish among the actions by a server ($p = s$) or by a client ($p = c$).

DEFINITION 4.2. *The syntactic dual $\overline{\sigma}$ of $\sigma \in \mathcal{HS}$ is defined as for behaviours in $\mathcal{S}$ by adding the clauses:*

$$\overline{?\sigma^p.\tau} = !\sigma^p.\overline{\tau}, \quad \overline{!\sigma^p.\tau} = ?\sigma^p.\overline{\tau}.$$

The modeling of full session types, namely including higher order sessions, involves higher order labeled transition systems, as it is the case in [8, 17]. An LTS is *higher order* if the set of states is somehow included into that of actions; more, one can either define (coinductively) relations over the LTS that take into account also the actions (see e.g. [18]), or a coinductive relation can be used in the side conditions of transition rules, as in [5, 17].

DEFINITION 4.3. *Let* **HoAct** $=$ **Act** $\cup \{?\sigma^p, !\sigma^p \mid \sigma \in \mathcal{HS}, p \in \{s, c\}\}$*. Then we extend the definition of the LTS modeling both the actions and the interaction, as well the definitions of $\dashv$, $\preceq_s$ and $\preceq_c$ to $\mathcal{HS}$ by adding the rules in Figure 2.*

The intuitive meaning of the new communication rules in Figure 2 is that $?\sigma^s$ is the action representing the reception of a channel on which the receiver has to behave as a *server*. It is then consistent with the idea of compliance to allow such an action to synchronize with $!\tau^s$, in case $\tau \preceq_s \sigma$, since then any client that is compliant with the protocol $\tau$, will do the same with $\sigma$.
Vice versa, and for the same reason, when the input action is $?\sigma^c$ the receiver is expected to behave on the received channel as a *client*: then this input action will safely synchronize with any $!\tau^c$ such that $\tau \preceq_c \sigma$.

REMARK 4.4. The higher order LTS rises two technical difficulties. The first one, as noted in [17], is that there is a circularity in the definitions of $\rho \dashv \sigma$ and of $\preceq_s, \preceq_c$, where the latter relations must be redefined in terms of higher order computations (a similar problem of circularity has been coped with also in [5]). This can be remedied by observing that the syntactical complexity of $\sigma$ is less than the complexity of $?\sigma^p.\tau$ and of $!\sigma^p.\tau$, and by stratifying the behaviour syntax accordingly, obtaining a hierarchy of relations $\preceq_s^i, \preceq_c^i$. This is left implicit for readability; however the coinductive characterizations of $\dashv$ and of the other relations studied so far requires a more careful analysis, and unfortunately a rather complex one.
Second, we haven't provided a complete axiomatization of the relations $\preceq_c$ and $\preceq_s$, as their definition involves a universal quantification over behaviours and reductions. Therefore, to the state of our knowledge, the higher order LTS should not be considered as a formal system, rather as an abstract mathematical model.

Lemma 2.3 still holds for higher order behaviours and LTS: in particular if $\sigma \Longrightarrow ?\sigma_1^p.\sigma_2$ or $\sigma \Longrightarrow !\sigma_1^p.\sigma_2$, these are unique, so that we can write $\sigma \Downarrow ?\sigma_1^p.\sigma_2$ and $\sigma \Downarrow !\sigma_1^p.\sigma_2$.
Also Lemma 2.6 holds for the new LTS, with **Act** replaced by **HoAct**.

For any $\mathcal{R} \subseteq \mathcal{HS}^2$ define:

$$\mathcal{G}_c(\mathcal{R}) = \{(\sigma, \tau) \in \mathcal{HS}^2 \mid \forall \rho. \sigma \mathcal{R} \rho \Rightarrow \tau \mathcal{R} \rho\}$$
$$\mathcal{G}_s(\mathcal{R}) = \{(\sigma, \tau) \in \mathcal{HS}^2 \mid \forall \rho. \rho \mathcal{R} \sigma \Rightarrow \rho \mathcal{R} \tau\}$$

and write $\sigma \mathcal{G}_p(\mathcal{R}) \tau$ to abbreviate $(\sigma, \tau) \in \mathcal{G}_p(\mathcal{R})$, for $p \in \{c, s\}$. By their definition $\preceq_c = \mathcal{G}_c(\dashv)$, and $\preceq_s = \mathcal{G}_s(\dashv)$. Now let $\mathcal{F}' : \mathcal{HS}^2 \to \mathcal{HS}^2$ be defined as the least extension of $\mathcal{F}$ in Def. 2.7 such that $(\rho, \sigma) \in \mathcal{F}'(\mathcal{R})$ if, for $p \in \{c, s\}$:

1. $\rho \Downarrow ?\rho_1^p.\rho_2$ implies $\sigma \Downarrow !\sigma_1^p.\sigma_2$ & $\sigma_1 \mathcal{G}_p(\mathcal{R}) \rho_1$ & $\rho_2 \mathcal{R} \sigma_2$
2. $\rho \Downarrow !\rho_1^p.\rho_2$ implies $\sigma \Downarrow ?\sigma_1^p.\sigma_2$ & $\rho_1 \mathcal{G}_p(\mathcal{R}) \sigma_1$ & $\rho_2 \mathcal{R} \sigma_2$

Lemma 2.8 still holds (by considering two more cases), with $\mathcal{F}'$ in place of $\mathcal{F}$, and we establish:

LEMMA 4.5. *$\mathcal{F}'$ is monotonic and $\dashv = \bigcap_k \mathcal{F}'^k$ namely the greatest fixed point of $\mathcal{F}'$.*

*Proof.* The $\supseteq$ inclusion is established as in Prop. 2.9 by using (the extended version of) Lemma 2.8. For the $\subseteq$ inclusion it suffices to prove that $\dashv \subseteq \mathcal{F}'(\dashv)$. The cases of $\rho \Downarrow \rho'$ with $\rho' = \mathbf{1}, \sum_{i \in I} a_i.\rho_i, \bigoplus_{i \in I} \overline{a}_i.\rho_i$ are the same as in the proof of Prop. 2.9. Suppose that $\rho' = ?\rho_1^p.\rho_2$ and that $\sigma \Downarrow \sigma'$. Then $\rho \| \sigma \Longrightarrow \rho' \| \sigma' \not\longrightarrow$ since $\rho' \neq \mathbf{1}$ and $\rho' \dashv \sigma'$ by hypothesis. It follows that $\sigma' = ?\sigma_1^p.\sigma_2$ for certain $\sigma_1, \sigma_2$ s.t. $\sigma_1 \preceq_p \rho_1$ and $\rho_2 \dashv \sigma_2$, and the thesis follows since $\preceq_p = \mathcal{G}_p(\dashv)$. The case $\rho' = !\rho_1^p.\rho_2$ is analogous. □

Again, it is easily seen that the relation $\{(\sigma, \overline{\sigma}) \mid \sigma \in \mathcal{HS}\}$ is a prefixed point of $\mathcal{F}'$, so that by the last lemma we have that both $\sigma \dashv \overline{\sigma}$ and $\overline{\sigma} \dashv \sigma$.
Now it is possible to provide, also for the respective extensions to higher order behaviours, a coinductive characterization of the the sub-behaviour relations. We stress that such a characterization is essentially more complex than in the first order case, because the sub-behaviour relations depend on the compliance relation; in turn compliance depends on the sub-behaviour relations, which are involved in the definition of the LTS. This forces a mutually coinductive construction, which is the contents of the following definition.

DEFINITION 4.6 (Coinductive Client/Server Relation Pair). *A pair $(\mathcal{R}_c, \mathcal{R}_s)$ of binary relations over $\mathcal{HS}$ is a* coinductive client/server *relation pair if $\{\sigma \mid \sigma \mathcal{R}_c \mathbf{1}\} = \{\tau \mid \mathbf{1} \mathcal{R}_s \tau\}$ and if $(\sigma, \tau) \in \mathcal{R}_p$ and $p, q \in \{s, c\}$, then:*

1. *$\sigma \Downarrow \sum_{i \in I} a_i.\sigma_i$ implies*
   *$\exists J \supseteq I.\ \tau \Downarrow \sum_{j \in J} a_j.\tau_j$ & $\forall j \in J.\ \sigma_j \mathcal{R}_p \tau_j$*
2. *$\sigma \Downarrow \bigoplus_{i \in I} \overline{a}_i.\sigma_i$ implies*
   *$\exists J \subseteq I.\ \tau \Downarrow \bigoplus_{j \in J} \overline{a}_j.\tau_j$ & $\forall j \in J.\ \sigma_j \mathcal{R}_p \tau_j$*
3. *$\sigma \Downarrow ?\sigma_1^q.\sigma_2$ implies $\tau \Downarrow ?\tau_1^q.\tau_2$ & $\sigma_1 \mathcal{R}_q \tau_1$ & $\sigma_2 \mathcal{R}_p \tau_2$*
4. *$\sigma \Downarrow !\sigma_1^q.\sigma_2$ implies $\tau \Downarrow !\tau_1^q.\tau_2$ & $\tau_1 \mathcal{R}_q \sigma_1$ & $\sigma_2 \mathcal{R}_p \tau_2$*

LEMMA 4.7. *$(\preceq_c, \preceq_s)$ is the largest client/server relation pair w.r.t. componentwise inclusion.*

*Proof.* (Sketch) If $(\mathcal{R}_c, \mathcal{R}_s)$ is a client/server relation pair then $\mathcal{R}_c \subseteq \preceq_c$ and $\mathcal{R}_s \subseteq_s \preceq_s$ is shown in the standard way, namely by considering the operator $\mathcal{H} : \mathcal{HS}^2 \times \mathcal{HS}^2 \to \mathcal{HS}^2 \times \mathcal{HS}^2$ determined by Def. 4.6 and showing that if $\mathcal{R}_c \not\subseteq \preceq_c$ and $\mathcal{R}_s \not\subseteq \preceq_s$ then $(\mathcal{R}_c, \mathcal{R}_s) \not\subseteq \mathcal{H}^k$ for some $k \in \mathbb{N}$ contradicting that hypothesis.

$$\frac{\Gamma,\ ?(A^p)B \leq ?(C^p)D \vdash A \leq C, B \leq D}{\Gamma \vdash ?(A^p)B \leq ?(C^p)D}\ \textsc{T-Sub-In} \qquad \frac{\Gamma,\ ![A^p]B \leq ![C^p]D \vdash C \leq A, B \leq D}{\Gamma \vdash ![A^p]B \leq ![C^p]D}\ \textsc{T-Sub-Out}$$

**Figure 3.** Rules for polarized higher order session types

To show that $(\preceq_c, \preceq_s)$ is a client/server relation pair we proceed by cases. We consider here only the cases of higher order input/output:

$\sigma \Downarrow ?\sigma_1^q.\sigma_2$: if $\sigma \preceq_c \tau$ then for all $\rho$, the fact that $\sigma \dashv \rho$ implies $\tau \dashv \rho$; we know that $\sigma \dashv \overline{\sigma}$ where $\overline{\sigma} = !\sigma_1^q.\overline{\sigma}_2$, so that $\tau \dashv \overline{\sigma}$. By Lemma 4.5 it follows that $\tau \Downarrow ?\tau_1^q.\tau_2$ for some $\tau_1, \tau_2$ such that $\sigma_1 \preceq_q \tau_1$ and $\tau_2 \dashv \overline{\sigma}_2$. On the other hand $\tau_2 \dashv \overline{\sigma}_2$ implies that, for all $\rho'$, if $\sigma \dashv \rho'$ then $\tau \dashv \rho'$ by (the extension to higher-order of) Lemma 2.12, hence $\tau_2 \preceq_c \sigma_2$ as desired.

$\sigma \Downarrow !\sigma_1^q.\sigma_2$: we reason as before, obtaining that $\tau \Downarrow !\tau_1^q.\tau_2$ where $\tau_1 \preceq_q \sigma_1$ and $\tau_2 \dashv \overline{\sigma}_2$. From $\tau_2 \dashv \overline{\sigma}_2$ we conclude that $\tau_2 \preceq_c \sigma_2$ as above; note that $\tau_1 \preceq_q \sigma_1$ has been inverted w.r.t. the previous case.

In both cases if $\sigma \preceq_s \tau$ then $\rho \dashv \sigma$ implies $\rho \dashv \tau$, and we proceed similarly. □

LEMMA 4.8. *With respect to the higher order LTS, all points of Lemma 3.3 remain valid, and moreover, for $p \in \{c, s\}$:*

1. $(?\sigma_1^p.\sigma_2)^{\perp} = \{\tau \in \mathcal{HS} \mid \exists \tau_1, \tau_2.\, \tau \Downarrow !\tau_1^p.\tau_2\ \&\ \tau_1 \preceq_p \sigma_1\ \&\ \tau_2 \in \sigma_2^{\perp}\}$;
2. $(!\sigma_1^p.\sigma_2)^{\perp} = \{\tau \in \mathcal{HS} \mid \exists \tau_1, \tau_2.\, \tau \Downarrow ?\tau_1^p.\tau_2\ \&\ \sigma_1 \preceq_p \tau_1\ \&\ \tau_2 \in \sigma_2^{\perp}\}$.

*Proof.* Analogous to that of Lemma 3.3, using Lemma 4.7. □

We are eventually in place to prove the main result for higher-order session behaviours.

THEOREM 4.9. $\preceq: \ = \ \preceq_c \cap \preceq_s$.

*Proof.* Assume that $\sigma \preceq_c \tau$ and $\sigma \preceq_s \tau$. If $\rho \in \sigma^{\perp}$ then $\sigma \dashv \rho$ and $\rho \dashv \sigma$, that is $\rho \in \mathsf{Server}(\sigma) \cap \mathsf{Client}(\sigma)$. By assumption $\mathsf{Server}(\sigma) \cap \mathsf{Client}(\sigma) \subseteq \mathsf{Server}(\tau) \cap \mathsf{Client}(\tau)$, so that $\tau \dashv \rho$ and $\rho \dashv \tau$, that is $\rho \in \tau^{\perp}$. We conclude that $\preceq: \ \supseteq \ \preceq_c \cap \preceq_s$.
To prove the inclusion $\preceq: \ \subseteq \ \preceq_c \cap \preceq_s$, consider the pair $(\preceq:, \preceq:)$. Then by Lemma 4.8 it is a coinductive client/server relation, so that $\preceq: \subseteq \preceq_c$ and $\preceq: \subseteq \preceq_s$ by Lemma 4.7. □

### 4.1 Subtyping Higher Order Sessions

The (full) theory of subtyping in [11] is equivalent to the theory obtained by adding the rules in Figure 3 to those for subtyping first order session types in Figure 1, and by reading the symbol $\leq$ as $<:$. For the sake of brevity we have written $\Gamma \vdash A \leq B, C \leq D$ in place of two premises: $\Gamma \vdash A \leq B$ and $\Gamma \vdash C \leq D$.
The use of polarities in the definition of higher order session behaviours has no counterpart in the syntax of session types, neither in the standard session type systems, say those in [21], nor in [1] by the present authors. The system we proposed and dubbed "asymmetric session types" can easily accommodate types of the shape $?(A^p)B$ and $![A^p]B$ for $p = c, s$, by attributing to the sent/received type the same polarity of the private channel (where we used $+ = s$ and $- = c$), e.g.:

$$\frac{\Gamma \vdash \{\kappa_2^q/x\}P \ \triangleright\ \Delta \cdot \kappa_1^p : B \cdot \kappa_2^q : A}{\Gamma \vdash \mathsf{catch}\ \kappa_1^p(x).P \ \triangleright\ \Delta \cdot \kappa_1^p : ?(A^q)B}$$

However the fact that, by Lemma 4.7 (to be compared with Def. 4.6), both $\preceq_c$ and $\preceq_s$ are covariant w.r.t. higher order input, and contravariant w.r.t higher order output, and more importantly Theorem 4.9 allow the treatment of the theory of session subtyping without resorting to the more complex assignment system. Let us extend the type syntax by:

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

Now the type interpretation can be extended straightforwardly to denote elements in $\mathcal{HS}$:

$$[\![\, ?(A^p)B ]\!] = ?[\![A]\!]^p[\![B]\!], \ \ [\![\, ![A^p]B ]\!] = ![\![A]\!]^p[\![B]\!].$$

Now, for $p = c, s$, let $\Gamma \vdash A \leq_p B$ be the theory obtained by adding to the theories of $\leq_c$ and $\leq_s$ in Def. 3.12 the higher order rules in Figure 3, by replacing $\leq$ by $\leq_c$ and $\leq_s$, respectively. Then, the proof of Theorem 3.13 (actually of Theorem 3.11) can be extended by considering the new cases of rules T-SUB-IN and T-SUB-OUT in a similar manner to those of rules T-SUB-& and T-SUB-$\oplus$, establishing:

THEOREM 4.10 (Higher Order C/S Subtyping).
*For any higher order session types $A, B$, if $\vdash A \leq_p B$ then $[\![A]\!] \preceq_p [\![B]\!]$, for $p = c, s$.*

*Proof.* Similar to the proof of Theorem 3.11. □

Let $A$ be a higher order session type in the sense of [12], i.e. without polarities; we interpret $A$ into $\mathcal{HS}$ by arbitrarily assigning a (fixed) polarity $p$ to all higher order input/output occurring in $A$: by abusing notation we call the resulting type $A$, and $[\![A]\!]$ its interpretation into $\mathcal{HS}$. We write $A <: B$ if the judgement $\vdash A <: B$ can be derived by the rules in Figure 1 and in Figure 3.

COROLLARY 4.11 (Higher Order Subtyping).
*If $A <: B$ for any higher order session types $A$ and $B$, then $[\![A]\!] \preceq: [\![B]\!]$.*

*Proof.* By observing that if $\vdash A <: B$ is derivable, then also $\vdash A \leq_c B$ and $\vdash A \leq_s B$ are such: by Theorem 4.10 we know that $[\![A]\!] \preceq_p [\![B]\!]$ for both $p = c$ and $p = s$ so that we conclude by Theorem 4.9. □

## 5. Related Works

The starting point of the present research is the compliance relation among contracts introduced in [13], and in general the theory of contracts for web services: see [7].
As we have explained, however, we depart from it adopting a more general concept, such that even non terminating behaviours can be compliant. In fact we allow the satisfaction in the limit of the requirement that all the actions by the client should find an adequate reply by the server. This is similar to [17], and we give a definition which is literally the same as that one used in [16]. In spite of this, there are some differences indeed: the definition in [16] is given for finite behaviours, i.e. without recursion. The extension to recursive behaviours as they are defined in the same paper leads to an unreasonable treatment of divergence (see Remark 2.5 in the present paper). On the other hand, the concept of "subsession" from [17] (which is the same as that of "compliance" in [8]) is not our compliance, nor one of our sub-behaviour relations. Rather it is comparable to our orthogonality and behavioural subtyping, since for the

test to succeed it is required that both sides of a parallel combination complete (reach a final state).

The issue of comparing contracts to session types has been addressed in [5, 14], besides the quoted [8, 17]. As explained in the introduction, we have designed our behaviours (both first and higher order) to model directly session types. The system we have treated to axiomatize the subtyping relation is equivalent to that one introduced in [11] but it avoids the function *unfold* and the concept of "type simulation" which is external w.r.t. the formal system. Gay and Hole motivation for adopting that solution was that the preliminary proposal in [10] was not syntax directed, and therefore unsuitable for developing an algorithm for type reconstruction. By adapting the idea proposed in [2] for recursive functional types, we obtain a simpler system, similar to the original one, but syntax directed.

The choice of restricting to session behaviours is responsible for the neat characterization of the main concepts involved, and first of all of the the notion of the dual of a behaviour. To appreciate the advantage of the definition of session behaviours one could compare it to the difficult treatment of duality for the full set of behaviours in [14]. A restriction to contracts, producing an effect similar to the one induced by our restrictions, has been proposed in [3] where terms like $a + b$ and also like $a + \overline{b}$ are avoided by imposing any output action $\overline{b}$ to be preceded by an internal tau action; however the absence of an internal choice and the ability of mixing input (i.e. branching actions in our interpretation) and output summands (which are naturally interpreted as selection actions) make this behaviour calculus rather unsuitable for our purposes.

On the other hand, the other papers appeared so far obtain a better matching with session types at the price of departing from the original type systems and process algebra (e.g. by introducing internal choice in the term syntax). This clearly opens the question of what we actually loose w.r.t. contracts in terms of expressivity, that we leave open; however, the fact that session types can be modelled by session behaviours makes it reasonable to consider such a set of behaviours as a basis for describing interaction protocols; besides all the examples we have found in the cited literature can be written in our syntax without any essential change.

With respect to the approach followed in the above mentioned works, we have followed the alternative route: first we see session types as some abstract (specification of) behaviour, that can be fruitfully treated by means of technique from process algebra (see [6]; apparently behavioural subtyping has been developed first in the area of OO concurrent languages: see e.g. [15, 18]). Incidentally we observe that the formal description of higher-order interaction rises the intriguing issue of higher order LTS, which deserves investigation on itself. In doing that we have kept the type syntax as it is, only by decorating types and channels with polarities to accommodate the new idea of compliance, which we consider as one of the most relevant contributions of the theory of contracts.

## 6. Conclusions

We have proposed a simple theory of behaviours and a two sided sub-behaviour concept, based on the idea of compliance coming from the theory of contracts. We have studied the basic properties of these concepts, with special attention to the complementary roles and views in client/server interactions. We have obtained a behavioural model of session types which is so close and faithful that we conjecture to be complete. By this we held to have refined previous work on web interaction specification by contracts, in the sense of distilling a simpler and more handy theory, with a clear semantics of duality. We think that this is not just a technical result, but also that it could be of use in devising efficient algorithms implementing tests for sub-contract and sub-behaviour relations.

## Acknowledgments

We are grateful to Cosimo Laneve and Luca Padovani for discussions and comments on previous versions of this work.

## References

[1] F. Barbanera, S. Capecchi, and U. de'Liguoro. Typing Asymmetric Client-Server Interaction. In *Proc. of FSEN'09*, number 5961 in LNCS, page 5961. Springer, 2010.

[2] M. Brandt and F. Henglein. Coinductive axiomatization of recursive type equality and subtyping. *Fundamenta Informatica*, 33, 1998.

[3] M. Bravetti and G. Zavattaro. Contract based multi-party service composition. In *Proc. of IPM International Symposium on Fundamentals of Software Engineering (FSEN'07)*, volume 4767 of *Lecture Notes in Computer Science*, pages 207–222, 2007.

[4] S. Carpineti, G. Castagna, C. Laneve, and L. Padovani. A formal account of contracts for Web Services. In *WS-FM, 3rd Int. Workshop on Web Services and Formal Methods*, number 4184 in LNCS, pages 148–162. Springer, 2006.

[5] G. Castagna, M. Dezani-Ciancaglini, E. Giachino, and L. Padovani. Foundations of Session Types. In *PPDP'09*, pages 219–230. ACM Press, 2009.

[6] G. Castagna and A. Frisch. A gentle introduction to semantic subtyping. In *Proc. of ICALP'05*, volume 3580 of *Lecture Notes in Computer Science*, pages 30–34, 2005.

[7] G. Castagna, N. Gesbert, and L. Padovani. A theory of contracts for web services. volume 31 of *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 2009.

[8] G. Castagna and L. Padovani. Contracts for mobile processes. In *Proceedings of the 20th International Conference on Concurrency Theory (CONCUR'09)*, volume 5710 of *LNCS*, pages 211–228. Springer-Verlag, 2009.

[9] M. Dezani-Ciancaglini, U. de' Liguoro, and N. Yoshida. On Progress for Structured Communications. In G. Barthe and C. Fournet, editors, *TGC'07*, volume 4912 of *LNCS*, pages 257–275. Springer, 2008.

[10] S. Gay and M. Hole. Types and Subtypes for Client-Server Interactions. In *ESOP'99*, volume 1576 of *LNCS*, pages 74–90. Springer-Verlag, 1999.

[11] S. Gay and M. Hole. Subtyping for Session Types in the Pi-Calculus. *Acta Informatica*, 42(2/3):191–225, 2005.

[12] K. Honda, V. T. Vasconcelos, and M. Kubo. Language Primitives and Type Disciplines for Structured Communication-based Programming. In *ESOP'98*, volume 1381 of *LNCS*, pages 22–138. Springer-Verlag, 1998.

[13] C. Laneve and L. Padovani. The Must Preorder Revisited: An Algebraic Theory for Web Services Contracts. In *CONCUR'07*, volume 4703 of *LNCS*, pages 212–225. Springer-Verlag, 2007.

[14] C. Laneve and L. Padovani. The pairing of contracts and session types. In *Concurrency, Graphs and Models*, volume 5065 of *Lecture Notes in Computer Science*, pages 681–700, 2008.

[15] Leavens and Pigozzi. A complete algebraic characterization of behavioral subtyping. *ACTAINF: Acta Informatica*, 36, 2000.

[16] L. Padovani. Contract-based discovery and adaptation of web services. http://www.di.unito.it/∼padovani/publications.html, 2009.

[17] L. Padovani. Session types at the mirror. *CoRR*, abs/0911.5449, 2009.

[18] A. Ravara, P. Resende, and V. Vasconcelos. An algebra of behavioural types. Preprint, CLC, Department of Mathematics, Instituto Superior Técnico, 1049-001 Lisboa, Portugal, 2002. Submitted for publication.

[19] A. Rensink and W. Vogler. Fair testing. *INFCTRL: Information and Computation (formerly Information and Control)*, 205, 2007.

[20] D. Sangiorgi and D. Walker. *The $\pi$-Calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.

[21] N. Yoshida and V. T. Vasconcelos. Language Primitives and Type Disciplines for Structured Communication-based Programming Revisited. In *SecReT'06*, volume 171 of *ENTCS*, pages 73–93. Elsevier, 2007.