Fair Subtyping for Multi-Party Session Types

Under consideration for publication in Math. Struct. in Comp. Science

Fair Subtyping for Multi-Party Session Types

LUCA PADOVANI

email: luca.padovani@unito.it

Dipartimento di Informatica, Università di Torino, Italy.

Received 14 January 2013

The subtyping relation defined for dyadic session type theories may compromise the liveness of multi-party sessions. In this paper we define a *fair* subtyping relation for multi-party session types that preserves liveness, we relate it with the subtyping relation for dyadic session types, and we provide coinductive, axiomatic, and algorithmic characterizations for it.

1. Introduction

Session types have been introduced by Honda [1993] and Honda *et al.* [1998] as a type discipline for structuring inter-process communications: a *session* is a private place of interaction between processes that communicate messages; each process plays a *role* within the scope of the session and behaves according to a *session type* that describes the sequence of messages the process is willing to send/capable to receive within the session through one of the session endpoints. In a sense, session types generalize traditional channel types [Pierce and Sangiorgi, 1996; Castagna *et al.*, 2008] by allowing messages of different types to travel along the same endpoint.

Dyadic session type theories require the two endpoints of a session to be used by exactly two processes and in complementary ways. This requirement enforces session correctness, which comprises both *communication safety* (no message of unexpected type is ever sent) and *liveness* (whenever a message is exchanged, all of the processes involved in the session make progress). For example, the session

$$\texttt{buyer} : T \mid \texttt{seller} : R \tag{1.1}$$

where T and R are the session types defined by the equations

$$\begin{array}{rcl} T & = & \texttt{seller!Buy}.T \oplus \texttt{seller!Done}.\mathsf{end} \\ R & = & \texttt{buyer?Buy}.R + \texttt{buyer?Done}.\mathsf{end} \end{array}$$

describes a conversation between two processes playing roles `buyer` and `seller`: `buyer` sends either a `Buy` message or a `Done` message to `seller`; the decision as to which type of message is sent is taken by `buyer`, whence the *internal choice* operator $\oplus$; `seller` must be ready to receive either a `Buy` or a `Done` message from `buyer`, whence the *external choice*

operator +. If a `Buy` message is exchanged, the two processes repeat this pattern; as soon as a `Done` message is exchanged, the session terminates. The correctness of session (1.1) can be easily established by noticing that T and R specify *complementary behaviors*: any message sent by `buyer` is accepted by `seller`; any internal choice taken by `buyer` is matched by an external choice offered by `seller`.

The shift from dyadic to *multi-party sessions* [Honda *et al.*, 2008] makes the definition of session correctness more subtle. First, it is no longer obvious what it means to use the endpoints of the session "in complementary ways" if the session involves more than two participants. Second, it is unreasonable to pretend that *all* of the involved participants make progress whenever a message is exchanged if communications are point-to-point. Still, one would like to make sure that no participant is left behind trying to deliver a message that no other process in the session is willing to accept, or waiting for a message that no other process in the session sends. A natural and practically reasonable formalization of correctness for multi-party sessions requires that the session must preserve the possibility to reach a terminal configuration where all of its participants no longer use the session endpoints. For example, in the session

$$\texttt{buyer} : T' \mid \texttt{seller} : R \mid \texttt{bank} : \texttt{buyer?Pay.end}$$

where T' is defined by the equation

$$T' = \texttt{seller!Buy}.T' \oplus \texttt{seller!Done.bank!Pay.end}\,,$$

the processes `buyer` and `seller` may exchange an arbitrary number of `Buy` messages and, during their interaction, the process `bank` does not make any progress. However, as long as `Buy` messages are exchanged, it is always *possible* that `buyer` sends a `Done` message to `seller` followed by a `Pay` message to `bank`. If this happens, all of the involved participants reach a terminal state and the session terminates. This potential for termination is sometimes called *fair termination* because it relies on a *fairness assumption* on the behavior of `buyer`: even though `buyer` can choose to send an arbitrary number of `Buy` messages, we exclude the "unfair" behavior in which `buyer` never chooses to send `Done`. Therefore, under a fairness assumption we conclude that `bank` will eventually receive `Pay` and make progress.

The adoption of fair termination to characterize the correctness of multi-party sessions has dramatic effects on the subtyping relation for session types [Gay and Hole, 2005; Castagna *et al.*, 2009]. Subtyping is a compatibility relation between types such that, when T is a subtype of S, it is harmless to replace an endpoint with type S with another one with type T or, equivalently, it is harmless to replace a process that behaves according to T with another one that behaves according to S. For example, the session type T defined above is a subtype of `seller!Done.end`: using an endpoint of type `seller!Done.end` means sending a `Done` message to process `seller`. Since the session type T permits sending both a `Buy` message and a `Done` message, using an endpoint with type T in place of another one with type `seller!Done.end` does not compromise the correctness of the session. Intuitively, we say that T is a subtype of S if S is a variant of T where some branches of some internal choices have been pruned. According to this

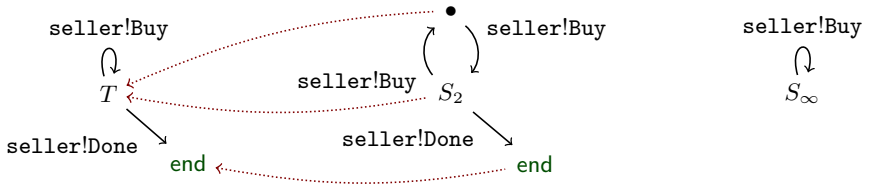

Fig. 1. Relation between $T = \texttt{seller!Buy}.T \oplus \texttt{seller!Done}.\mathsf{end}$ and $S_2 = \texttt{seller!Buy}.\texttt{seller!Buy}.S_2 \oplus \texttt{seller!Done}.\mathsf{end}$.

intuition every session type in the family

$$\begin{aligned} S_2 &= \texttt{seller!Buy}.\texttt{seller!Buy}.S_2 \oplus \texttt{seller!Done}.\mathsf{end} \\ &\vdots \\ S_n &= (\texttt{seller!Buy}.)^n S_n \oplus \texttt{seller!Done}.\mathsf{end} \\ &\vdots \\ S_\infty &= \texttt{seller!Buy}.S_\infty \end{aligned}$$

is a supertype of T. The type S_n describes the behavior of a `buyer` that always buys a number of items that is multiple of n. The type S_∞ is somewhat the limit of the sequence $\{S_i\}_{i\geq 2}$ and describes a `buyer` that only sends `Buy` messages. The fact that T is a subtype of S_∞ may be questionable, because the sessions $\texttt{buyer} : S_i \mid \texttt{seller} : R$ for $i \geq 2$ all have the potential to terminate (it is always possible that a `Done` message is sent), while the session $\texttt{buyer} : S_\infty \mid \texttt{seller} : R$ is doomed to loop forever. In a dyadic session like $\texttt{buyer} : S_\infty \mid \texttt{seller} : R$ this shortcoming is mitigated by the fact that *every* participant of the session makes indefinite progress anyway. However, this may no longer be the case when multi-party sessions are considered. Indeed, using the same arguments we might also deduce that S_∞ is a supertype of T', and now in the session

$$\texttt{buyer} : S_\infty \mid \texttt{seller} : R \mid \texttt{bank} : \texttt{buyer?Pay}.\mathsf{end}$$

participant `buyer` keeps interacting with `seller` while `bank` starves waiting for a `Pay` message that is never sent. We conclude that the well-known subtyping relation for dyadic session types is unsound in multi-party theories because it may compromise the liveness of multi-party sessions.

In this paper we study a subtyping relation, which we call *fair subtyping*, that preserves liveness also in multi-party sessions. Understanding when two session types are related by fair subtyping is a surprisingly complex business, for essentially two reasons: first of all, the differences between standard and fair subtyping emerge only when recursive session types are involved, while the two relations coincide for finite session types; second, deciding whether some branch of an internal choice can be safely pruned may involve a non-local check on the structure of the session types being compared.

To illustrate the subtleties of fair subtyping, consider again the session types T, S_2, and S_∞ depicted as the three automata in Figure 1, where the initial states have been labeled with the name of the session type and the solid arcs with the actions performed by

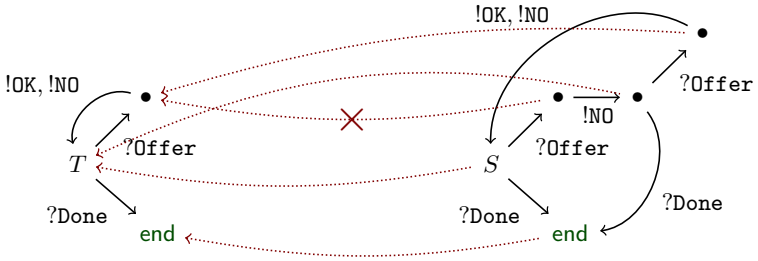

Fig. 2. Relation between $T = ?\texttt{Offer}.(!\texttt{OK}.T \oplus !\texttt{NO}.T) + ?\texttt{Done}.\mathsf{end}$ and $S = ?\texttt{Offer}.!\texttt{NO}.(?\texttt{Offer}.(!\texttt{OK}.S \oplus !\texttt{NO}.S) + ?\texttt{Done}.\mathsf{end}) + ?\texttt{Done}.\mathsf{end}$.

the processes that behave according to these types. The subtyping relation establishes a correspondence between states of two session types which is represented, in the figure, by the three dotted arrows showing, for each state of S_2, the corresponding state of T. The fact that S_∞ *is not* a supertype of T is blatantly obvious, since no end state is reachable from S_∞, but this does not explain why S_2 *is* a supertype of T. Observe that S_2 has an intermediate state $\bullet$ which lacks the outgoing $\texttt{seller}!\texttt{Done}$-labeled transition that T has. In a manner of speaking, a participant $\texttt{seller} : R$ interacting with $\texttt{buyer} : S_2$ has fewer chances to terminate than if it were interacting with $\texttt{buyer} : T$ because $\texttt{buyer} : S_2$ sends `Done` messages less frequently. The point is that every participant $\texttt{seller} : R$ such that $\texttt{buyer} : T \mid \texttt{seller} : R$ is correct must be ready to accept a `Done` message from `buyer` *at any time*. Therefore, even if `buyer` behaves according to S_2 and cannot send a `Done` message because it is in the $\bullet$ state, `buyer` can always send a `Buy` message followed by a `Done` one and drive the session into a successfully terminated state. Stated in other words, there is no $\texttt{seller} : R$ participant that terminates successfully only if it observes a `Done` message after an odd number of `Buy` messages has been received, because it is `buyer` – not `seller` – that internally chooses how many items to buy and when to move into the terminal state. We express this property saying that S_2 *rules over* (every context, like $\texttt{seller} : R$, that completes) T, which we denote by $T \prec S_2$.

An even subtler example is depicted in Figure 2 where T and S describe the behavior of two hypothetical `seller`s engaged into a bargain with a `buyer` (for the sake of readability, the role name '`buyer`' has been omitted from the figure and its caption). The `buyer` participant (not shown in the figure) sends either `Offer` or `Done` messages depending on whether it wants to make an offer regarding the purchase of an item or quit the bargain. The `seller` participant answers to each offer with either an `OK` or a `NO` message, respectively indicating that the offer was accepted or rejected. The only difference between T and S is that S lacks one outgoing $\texttt{buyer}!\texttt{OK}$-labeled transition that T has. Basically, $\texttt{seller} : S$ systematically rejects the first offer (more precisely, every odd-indexed offer) and it may send back an `OK` message only after an odd number of `Offer`s have been received. Consider now the session type

$$R = !\texttt{Offer}.(?\texttt{OK}.!\texttt{Done}.\mathsf{end} + ?\texttt{NO}.!\texttt{Offer}.(?\texttt{OK}.R + ?\texttt{NO}.R)) \tag{1.2}$$

describing the behavior of a `buyer` process that terminates its bargain only when the `seller` it is interacting with accepts the first offer (in fact, an odd-indexed offer). We have that $\texttt{seller} : T | \texttt{buyer} : R$ is correct while $\texttt{seller} : S | \texttt{buyer} : R$ loops through state S. What happens is that `buyer` forces $\texttt{seller} : S$ to go through state S in hopes that an `OK` message is received. This was possible with $\texttt{seller} : T$, but not with $\texttt{seller} : S$. The fact that a participant like $\texttt{buyer} : R$ exists means that S does not rule over T ($T \not\prec S$), and therefore T is not a subtype of S.

Higher-Order Session Types. So far we have discussed examples of session types where messages are abstracted as atomic tags such as `Offer` or `OK`. In practice, one is usually interested in providing more structured information about the type of messages being exchanged, since such information is key in assessing the correctness of processes with respect to the session types of the channels they use. As a particular but paradigmatic case, we want to define *higher-order session types* so as to promote channels to messages, just like higher-order arrow types promote functions to values in functional languages. For instance, the session type

$$\mathtt{p}!\mathtt{a}\langle T\rangle.\mathsf{end}$$

describes a channel on which it is possible to send an `a`-tagged message with an argument which is itself a channel described by the session type T. In addition to these practical considerations, the reason why we are interested in this refinement of session types is that it has an interesting impact on the theory of fair subtyping. To see why, consider the session

$$\mathtt{p} : \mathtt{q}?\mathtt{a}\langle S\rangle.\mathsf{end} \mid \mathtt{q} : \mathtt{p}!\mathtt{a}\langle T\rangle.\mathsf{end}$$

involving two participants exchanging a message that is itself a channel. In order for this session to be correct, it must be the case that T is a (fair) subtype of S, following the usual intuition that a channel of type T can be safely used where a channel of type S is expected. That is, the notion of session correctness *depends on* that of fair subtyping. At the same time, fair subtyping is just defined as the relation that does not compromise session correctness. To break this circularity we use an original technique: we parameterize session correctness on a generic subtyping relation and define subtyping as the solution of a fixpoint equation.

Contributions. We summarize the contributions of the present work as follows:

— We define a language of higher-order, multi-party session types along with semantically defined notions of session correctness and fair subtyping. While analogous notions can be found in the current literature, this is the first time that fair subtyping is extended to a higher-order language.
— We provide a complete coinductive characterization of fair subtyping based on the "ruled by" relation. We show that there is an intimate connection between the "ruled by" relation and the semantic notion of "type emptyness" for session types.
— We give a complete axiomatization of fair subtyping that is entirely syntax directed.

To the best of our knowledge, this is the first complete axiomatization of a liveness-preserving refinement for a non-trivial, higher-order process language.
— We give algorithms for deciding session type emptyness and fair subtyping.

Origin of the material. A technical report with an early draft of this work was uploaded on HAL in December 2010.[†] The same work has been presented in Lisbon at the Behavioral Types Workshop in April 2011, and it was subsequently published in the proceedings of COORDINATION 2011 [Padovani, 2011a]. The current article corresponds to a thoroughly revised and improved version of [Padovani, 2011a] whose most prominent novelties, aside from the presentation of the full proofs of all the results, are the treatment of higher-order session types and the syntax-directed axiomatization of fair subtyping and of session type emptyness.

Structure of the paper. We devote Section 2 to defining syntax and semantics of session types. The section ends with the definitions of fair subtyping and of session correctness. In Section 3 we show that fair and standard subtyping are incomparable. Then we introduce a normal form for session types that allows us to define fair subtyping as a refinement of standard subtyping. The difference between the two relations is fully captured by the "ruled by" relation we have informally introduced earlier. Section 4 provides a complete axiomatization of fair subtyping, as well as algorithms for deciding it. We discuss related work in Section 5 and conclude in Section 6 illustrating some intriguing challenges posed by fair subtyping that we aim at addressing in future work. For the sake of readability, proofs and supplementary technical material corresponding to Sections 2, 3, and 4 have been moved to Appendixes A, B, and C respectively.

2. Syntax and Semantics of Session Types

2.1. *Syntax*

We assume given: an infinite set $\mathscr{R}$ of *role tags* ranged over by $\mathtt{p}$, $\mathtt{q}$, ...; an infinite set $\mathscr{N}$ of *message tags* ranged over by $\mathtt{a}$, $\mathtt{b}$, ...; a set $\mathscr{V}$ of *recursion variables* ranged over by x, y, ... Table 1 defines the syntax of sessions and session types. *Sessions*, ranged over by M, N, ..., are finite compositions $\mathtt{p}_1 : T_1 \mid \cdots \mid \mathtt{p}_n : T_n$ representing a fixed number of participants that communicate with each other within a protected scope. Each participant is uniquely identified by a role $\mathtt{p}_i$ and behaves according to the session type T_i. We work exclusively with well-formed sessions, where $i \neq j$ implies $\mathtt{p}_i \neq \mathtt{p}_j$.

Session types, ranged over by T, S, ..., are the closed terms generated by the grammar in Table 1 such that:

— every recursion variable is guarded by at least one (input or output) prefix, and
— in every subterm $\sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i$ or $\bigoplus_{i\in I} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i$ the set I is finite and non-empty and the $\mathtt{a}_i$'s are pairwise distinct.

[†] See `http://hal.archives-ouvertes.fr/hal-00546531/fr/`.

Table 1. Syntax of types, session types, and sessions.

Type	t	::=	top	(top type)
		|	T	(session type)
Session Type	T	::=	bot	(bottom type)
		|	end	(termination)
		|	x	(variable)
		|	$\sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i$	(input)
		|	$\bigoplus_{i\in I} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i$	(output)
		|	$\mu x.T$	(recursion)
Session	M	::=	$\mathtt{p} : T$	(participant)
		|	$M \mid M$	(composition)

The first condition forbids non-contractive session types such as $\mu x.x$, while the second condition ensures that the tag $\mathtt{a}_i$ in an internal/external choice uniquely determines a continuation T_i. The session types end and bot both describe the behavior of a participant that performs no input/output actions. In the former case, the participant has successfully terminated, while in the latter case it has not. We use bot to represent an unrecoverable error state. The session type $\bigoplus_{i\in I} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i$ describes the behavior of a participant that sends a message to participant $\mathtt{p}$. The message is made of a tag $\mathtt{a}_i$ and carries an argument of type t_i. According to the tag $\mathtt{a}_i$ of the message, the participant then behaves as specified by T_i. In a dual manner, the session type $\sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i$ describes the behavior of a participant that waits for a message from participant $\mathtt{p}$. As for outputs, $\mathtt{a}_i$ specifies the tag of the message, t_i is the type of the message argument and T_i the behavior that the receiver conforms to after having received the message. For the sake of technical simplicity, we restrict ourselves to messages with a single argument, the extension to the general case not posing substantial issues. Terms x and $\mu x.T$ can be used to describe recursive behaviors, as usual. Since μ is the only binder for recursion variables, the notions of free and bound variables are defined as expected.

Types are either the top type top or any session type. Other data types can be easily accommodated within this syntactic category, but we keep the formal type language as simple as possible to avoid unnecessary clutter. The fact that bot is a session type while top is not is motivated by the observation that there are many session types – those describing "bad" behaviors – that are syntactically different from bot but *semantically equivalent* to it (the session type S_∞ we have introduced in Section 1 is one example). Therefore, the bot type provides a handy normal form for all these "bad" behaviors. Conversely, there is no session type that is equivalent to top, and for good reasons: it would be a behavior capable of satisfying any kind of request, which is clearly impossible to achieve.

In what follows, we adopt the following conventions:

— We write $\mathscr{T}$ for the set of session types.

— Whenever convenient we use an infix notation for choices and write

$$\mathtt{p}!\mathtt{a}_1\langle t_1\rangle.T_1 \oplus \cdots \oplus \mathtt{p}!\mathtt{a}_n\langle t_n\rangle.T_n \qquad \text{in place of} \qquad \bigoplus_{i=1..n} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i$$

and

$$\mathtt{p}?\mathtt{a}_1\langle t_1\rangle.T_1 + \cdots + \mathtt{p}?\mathtt{a}_n\langle t_n\rangle.T_n \qquad \text{in place of} \qquad \sum_{i=1..n} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i\,.$$

Note that mixed choices like $\mathtt{p}?\mathtt{a}\langle t\rangle.T + \mathtt{q}?\mathtt{a}\langle t\rangle.S$ and $\mathtt{p}!\mathtt{a}\langle t\rangle.T \oplus \mathtt{q}!\mathtt{a}\langle t\rangle.S$ are forbidden. In particular, the source participant $\mathtt{p}$ and the destination participant $\mathtt{q}$ in $\sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i$ and $\bigoplus_{i\in I} \mathtt{q}!\mathtt{a}_i\langle t_i\rangle.T_i$ must be the same in all branches (all the examples in the introduction are consistent with these conventions). While slightly redundant, the syntax for inputs and outputs allows us to conveniently switch between the prefix and the infix forms.

— Sometimes we omit the argument type specification in input prefixes when it is top and in output prefixes when it is bot. Therefore, we may write $\mathtt{p}?\mathtt{a}.T$ in place of $\mathtt{p}?\mathtt{a}\langle\mathsf{top}\rangle.T$ and $\mathtt{p}!\mathtt{a}.T$ in place of $\mathtt{p}!\mathtt{a}\langle\mathsf{bot}\rangle.T$.

— We take an equirecursive point of view and do not distinguish between a recursive session type and its unfolding. That is, we assume $\mu x.T = T\{\mu x.T/x\}$ where $T\{\mu x.T/x\}$ denotes the capture-avoiding substitution of the free occurrences of x in T with $\mu x.T$.

Note that all the session types defined in the introduction can be finitely represented as possibly recursive terms generated by the grammar in Table 1 ([Courcelle, 1983] is the standard reference for the formal treatment of regular trees and their finite representations). Here are a few more examples:

— $\mu x.\mathtt{q}!\mathtt{a}.x$ is analogous to S_∞ in Figure 1 and describes a process that repeatedly sends $\mathtt{a}$-tagged messages to participant $\mathtt{q}$;

— $\mu x.(\mathtt{q}?\mathtt{a}.(\mathtt{q}!\mathtt{c}.x \oplus \mathtt{q}!\mathtt{d}.x) + \mathtt{q}?\mathtt{b}.\mathsf{end})$ is analogous to session type T in Figure 2 and describes a process that waits for either an $\mathtt{a}$-tagged or a $\mathtt{b}$-tagged message. If it receives an $\mathtt{a}$-tagged message it sends either a $\mathtt{c}$-tagged or a $\mathtt{d}$-tagged message and then repeats this pattern. If it receives a $\mathtt{b}$-tagged message it terminates successfully.

— $T \stackrel{\text{def}}{=} \mu x.\mathtt{q}!\mathtt{a}\langle x\rangle.\mathsf{end}$ describes a process that sends a single $\mathtt{a}$-tagged message to $\mathtt{q}$ with an argument that has type T. Note that T satisfies the equation $T = \mathtt{q}!\mathtt{a}\langle T\rangle.\mathsf{end}$.

2.2. *Transition System for Sessions*

Intuitively, the participants of a session behave as described by the corresponding session type and the session evolves by means of internal choices taken by the participants and by synchronizations occurring between them. A session is correct if, no matter how it evolves, it preserves the possibility to reach a state in which *every* participant has successfully terminated. The most natural way to formalize correctness is to express the evolution of a session by means of a transition system, whose definition is the subject of this subsection.

Table 2. Transition system of sessions.

$$\text{(T-End)}\quad \mathtt{p} : \mathsf{end} \xrightarrow{\checkmark}_{\mathscr{S}} \mathtt{p} : \mathsf{end} \qquad \text{(T-Output)}\quad \mathtt{p} : \mathtt{q}!\mathtt{a}\langle t\rangle.T \xrightarrow{\mathtt{p}:\mathtt{q}!\mathtt{a}\langle t\rangle}_{\mathscr{S}} \mathtt{p} : T$$

$$\text{(T-Choice)}\quad \frac{k \in I}{\mathtt{p} : \bigoplus_{i\in I} \mathtt{q}!\mathtt{a}_i\langle t_i\rangle.T_i \xrightarrow{\tau}_{\mathscr{S}} \mathtt{p} : \mathtt{q}!\mathtt{a}_k\langle t_k\rangle.T_k} \qquad \text{(T-Input)}\quad \frac{k \in I \qquad (t, t_k) \in \mathscr{S}}{\mathtt{p} : \sum_{i\in I} \mathtt{q}?\mathtt{a}_i\langle t_i\rangle.T_i \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}_k\langle t\rangle}_{\mathscr{S}} \mathtt{p} : T_k}$$

$$\text{(T-Par Action)}\quad \frac{M \xrightarrow{\ell}_{\mathscr{S}} M' \qquad \ell \neq \checkmark}{M \mid N \xrightarrow{\ell}_{\mathscr{S}} M' \mid N} \qquad \text{(T-Par Comm)}\quad \frac{M \xrightarrow{\overline{\alpha}}_{\mathscr{S}} M' \qquad N \xrightarrow{\alpha}_{\mathscr{S}} N'}{M \mid N \xrightarrow{\tau}_{\mathscr{S}} M' \mid N'}$$

$$\text{(T-Par End)}\quad \frac{M \xrightarrow{\checkmark}_{\mathscr{S}} M \qquad N \xrightarrow{\checkmark}_{\mathscr{S}} N}{M \mid N \xrightarrow{\checkmark}_{\mathscr{S}} M \mid N}$$

Labels of the transition system, ranged over by ℓ, are generated by the grammar below:

Label	ℓ	::=	τ	(internal action)	**Action**	α	::=	$\mathtt{p} : \mathtt{p}?\mathtt{a}\langle t\rangle$ (input)
		|	$\checkmark$	(termination)			|	$\mathtt{p} : \mathtt{p}!\mathtt{a}\langle t\rangle$ (output)
		|	α	(visible action)				

The label τ denotes an internal action, which may result from a participant performing an internal choice or from the synchronization between two participants. The label $\checkmark$ denotes the successful termination of a participant or of the whole session. Visible actions, ranged over by α, denote input/output operations performed by participants: the label $\mathtt{p} : \mathtt{q}?\mathtt{a}\langle t\rangle$ states that participant $\mathtt{p}$ receives from participant $\mathtt{q}$ an $\mathtt{a}$-tagged message whose argument has type t; dually, the label $\mathtt{p} : \mathtt{q}!\mathtt{a}\langle t\rangle$ states that participant $\mathtt{p}$ sends to participant $\mathtt{q}$ an $\mathtt{a}$-tagged message whose argument has type t. Note that, as session types are abstract descriptions of the behavior of processes, input and output actions only describe the *type* of the arguments in messages, not the actual arguments themselves. We write $\overline{\alpha}$ for the complement of action α, where

$$\overline{\mathtt{p} : \mathtt{q}?\mathtt{a}\langle t\rangle} = \mathtt{q} : \mathtt{p}!\mathtt{a}\langle t\rangle \qquad \text{and} \qquad \overline{\mathtt{p} : \mathtt{q}!\mathtt{a}\langle t\rangle} = \mathtt{q} : \mathtt{p}?\mathtt{a}\langle t\rangle$$

(note the switching of roles). We let φ, ψ, ... range over finite strings of visible actions, and we extend complementation $\overline{\cdot}$ to strings of visible actions.

Table 2 defines the transition system (symmetric rules omitted) in terms of a family of labeled relations $\xrightarrow{\ell}$. Since the transition system depends on subtyping (see rule (T-Input)), we parameterize transitions with a *pre-subtyping relation* $\mathscr{S}$ and postpone the problem of defining subtyping concretely to Section 2.3.

Definition 2.1. A preorder $\mathscr{S}$ is a *pre-subtyping relation* if $(\mathsf{bot}, t) \in \mathscr{S}$ and $(t, \mathsf{top}) \in \mathscr{S}$ for every $t \in \mathscr{T}$.

In what follows we let $\mathscr{S}$, $\mathscr{R}$, ... range over pre-subtyping relations.

An informal explanation of the axioms and rules of Table 2 is given in the following paragraphs. Rule (T-End) states that $\mathtt{p} : \mathsf{end}$ performs a $\checkmark$ action that flags successful termination of $\mathtt{p}$ and reduces to itself. Rules (T-Output) and (T-Choice) deal with

outputs. The former one shows that a participant $\mathtt{p}$ willing to send an $\mathtt{a}$-tagged message to participant $\mathtt{q}$ performs a $\mathtt{p} : \mathtt{q}!\mathtt{a}\langle t\rangle$ action. The latter one states that a participant that is ready to send any message from a set internally and irrevocably chooses one particular message to send. In both rules we use the abbreviation $\mathtt{q}!\mathtt{a}_0\langle t_0\rangle.T_0$ for $\bigoplus_{i\in\{0\}} \mathtt{q}!\mathtt{a}_i\langle t_i\rangle.T_i$. Rule (T-Input) deals with inputs and states that a participant $\mathtt{p}$ performs $\mathtt{p} : \mathtt{q}?\mathtt{a}\langle t\rangle$ actions according to the tags of messages it is willing to receive and the participant $\mathtt{q}$ from which it expects these messages to come. The session type states that an $\mathtt{a}_k$-tagged message carries an argument of type t_k. However, following the intuition that a value of type t can be used wherever a value of type t_k is expected if t is a subtype of t_k, the rule yields a (possibly infinite) number of transitions of the form $\mathtt{p} : \mathtt{q}?\mathtt{a}_k\langle t\rangle$ for every t that is a subtype of t_k according to $\mathscr{S}$. The transition relation remains image finite, since the residual session type can only be one of the T_i for $i \in I$. Note the fundamental asymmetry between inputs and outputs: a participant autonomously commits to sending *one* particular message by means of rule (T-Choice), while it retains the ability to receive *any* message from a given set by means of rule (T-Input). This asymmetry lies at the heart of the variance properties of the subtyping relation we are going to define. Rule (T-Par Action) propagates transitions through compositions, provided that the label is different from $\checkmark$. Rule (T-Par Comm) describes the usual synchronization between participants performing complementary actions. Finally, (T-Par End) states that the session type bot has no transitions. It plays a similar role as the δ atom in process algebras with explicit deadlock (see Aceto and Hennessy [1992]).

In the following we adopt these conventions: we write $\overset{\tau}{\Longrightarrow}_{\mathscr{S}}$ for the reflexive, transitive closure of $\overset{\tau}{\longrightarrow}_{\mathscr{S}}$; we write $\overset{\ell}{\Longrightarrow}_{\mathscr{S}}$ for the composition $\overset{\tau}{\Longrightarrow}_{\mathscr{S}}\overset{\ell}{\longrightarrow}_{\mathscr{S}}\overset{\tau}{\Longrightarrow}_{\mathscr{S}}$ and $\overset{\alpha_1\cdots\alpha_n}{\Longrightarrow}_{\mathscr{S}}$ for the composition $\overset{\alpha_1}{\Longrightarrow}_{\mathscr{S}}\cdots\overset{\alpha_n}{\Longrightarrow}_{\mathscr{S}}$; we write $M \overset{\ell}{\longrightarrow}_{\mathscr{S}}$ (respectively, $M \overset{\ell}{\Longrightarrow}_{\mathscr{S}}$) if there exists N such that $M \overset{\ell}{\longrightarrow}_{\mathscr{S}} N$ (respectively, $M \overset{\ell}{\Longrightarrow}_{\mathscr{S}} N$); we write $M \overset{\ell}{\not\longrightarrow}_{\mathscr{S}}$ (respectively, $M \overset{\ell}{\not\Longrightarrow}_{\mathscr{S}}$) if there exists no N such that $M \overset{\ell}{\longrightarrow}_{\mathscr{S}} N$ (respectively, $M \overset{\ell}{\Longrightarrow}_{\mathscr{S}} N$). We also extend the labeled transition relation and the above notation to session types so that, for example, $T \overset{\ell}{\longrightarrow}_{\mathscr{S}} S$ if $\mathtt{p} : T \overset{\ell}{\longrightarrow}_{\mathscr{S}} \mathtt{p} : S$ for some $\mathtt{p}$.

We now have all the ingredients for defining correct sessions formally. Just like the transition system is parametric over some subtyping relation $\mathscr{S}$, the notion of correctness is parametric over the same relation.

Definition 2.2 ($\mathscr{S}$-correct session)**.** We say that M is *$\mathscr{S}$-correct* if $M \overset{\tau}{\Longrightarrow}_{\mathscr{S}} N$ implies $N \overset{\checkmark}{\Longrightarrow}_{\mathscr{S}}$.

The definition states that a correct session preserves the possibility of reaching a state in which all of its participant have successfully terminated. Note in particular that $\mathscr{S}$-correctness is preserved by $\overset{\tau}{\longrightarrow}_{\mathscr{S}}$ reductions, that is if M is $\mathscr{S}$-correct and $M \overset{\tau}{\Longrightarrow}_{\mathscr{S}} N$, then N is also $\mathscr{S}$-correct.

A few examples of correct and incorrect sessions follow:

— $\mathtt{p} : \mathsf{end}$ is the simplest $\mathscr{S}$-correct session and $\mathtt{p} : \mathsf{end} \mid M$ is $\mathscr{S}$-correct if and only if so is M.

— The session $M \overset{\text{def}}{=} \mathtt{p} : T \mid \mathtt{q} : S$ where $T = \mu x.(\mathtt{q}!\mathtt{a}.x \oplus \mathtt{q}!\mathtt{b}.\mathsf{end})$ and $S = \mu y.(\mathtt{p}?\mathtt{a}.y +$

$\mathsf{p}?\mathsf{b}.\mathsf{end}$) is $\mathscr{S}$-correct, because either

$$M \xrightarrow{\tau}_{\mathscr{S}} \mathsf{p} : \mathsf{q}!\mathsf{b}.\mathsf{end} \mid \mathsf{q} : S \xrightarrow{\tau}_{\mathscr{S}} \mathsf{p} : \mathsf{end} \mid \mathsf{q} : \mathsf{end} \xrightarrow{\checkmark}_{\mathscr{S}}$$

or

$$M \xrightarrow{\tau}_{\mathscr{S}} \mathsf{p} : \mathsf{q}!\mathsf{a}.T \mid \mathsf{q} : S \xrightarrow{\tau}_{\mathscr{S}} M$$

and no other transitions are possible. Note that the notion of $\mathscr{S}$-correctness does not exclude the existence of infinite interactions, provided that a path to successful termination does exist.

— The session $\mathsf{p} : \mathsf{bot} \mid M$ is not $\mathscr{S}$-correct no matter of M and $\mathscr{S}$, because $\mathsf{p} : \mathsf{bot} \xrightarrow{\checkmark}\!\!\!\!\!\!\rightarrow_{\mathscr{S}}$. In general, a necessary condition for session M to be correct is that the session types associated with all the participants in M must have the potential to terminate successfully. This condition is not sufficient. For example, a participant $M \mid \mathsf{p} : (\mathsf{q}!\mathsf{a}.\mathsf{end} \oplus \mathsf{q}!\mathsf{b}.\mathsf{bot})$ terminates successfully if M is willing to receive an a-tagged message from p, but we also have $M \mid \mathsf{p} : (\mathsf{q}!\mathsf{a}.\mathsf{end} \oplus \mathsf{q}!\mathsf{b}.\mathsf{bot}) \xrightarrow{\tau}_{\mathscr{S}} M \mid \mathsf{p} : \mathsf{q}!\mathsf{b}.\mathsf{bot}$ which is doomed to fail.

— Session correctness may crucially depend on the subtyping relation being considered. For example, the session $\mathsf{p} : \mathsf{q}!\mathsf{a}\langle t\rangle.\mathsf{end} \mid \mathsf{q} : \mathsf{p}?\mathsf{a}\langle s\rangle.\mathsf{end}$ is $\mathscr{S}$-correct provided that $(t, s) \in \mathscr{S}$. Conversely, both $\mathsf{p} : \mathsf{q}!\mathsf{a}\langle\mathsf{bot}\rangle.\mathsf{end} \mid \mathsf{q} : \mathsf{p}?\mathsf{a}\langle s\rangle.\mathsf{end}$ and $\mathsf{p} : \mathsf{q}!\mathsf{a}\langle t\rangle.\mathsf{end} \mid \mathsf{q} : \mathsf{p}?\mathsf{a}\langle\mathsf{top}\rangle.\mathsf{end}$ are correct no matter of $\mathscr{S}$ (recall that $\mathscr{S}$ is a pre-subtyping relation, meaning that $(\mathsf{bot}, s) \in \mathscr{S}$ and $(t, \mathsf{top}) \in \mathscr{S}$).

2.3. *Fair Subtyping*

Now that we have a notion of session correctness we should be able to define subtyping semantically as the relation that preserves correctness. More formally, we would like to define fair subtyping as the largest pre-subtyping $\mathscr{S}$ that satisfies the equation:

$$\begin{aligned}\mathscr{S} = \{(t, \mathsf{top}) \mid t \in \mathscr{T}\} \cup \{(\mathsf{bot}, t) \mid t \in \mathscr{T}\} \\ \cup \{(T, S) \mid \forall M, \mathsf{p} : (M \mid \mathsf{p} : T)\ \mathscr{S}\text{-correct implies } (M \mid \mathsf{p} : S)\ \mathscr{S}\text{-correct}\}\end{aligned} \tag{2.1}$$

At this stage we do not know whether an $\mathscr{S}$ satisfying (2.1) does exist, but before we address this issue it is worth reasoning on *why* such an $\mathscr{S}$ would serve as a subtyping relation. What is surprising about equation (2.1) is that it speaks about left-to-right substitutability (of behaviors), while subtyping is concerned with right-to-left substitutability (of endpoints). The mismatch is only apparent, however, and is due to the fact that session types are behavioral types (they describe the behavior of processes using session endpoints). To clarify this point, suppose that S is the type associated with an endpoint c and that some process P uses c as indicated by S. By replacing endpoint c in P with another endpoint d with type T such that $(T, S) \in \mathscr{S}$, we are changing the set of processes that P is interacting with, which together behave according to some M such that $M \mid \mathsf{p} : T$ is $\mathscr{S}$-correct. In particular, replacing c with d does *not* affect the way P behaves: P uses endpoint d (whose type is T) as if it were endpoint c (thus according to S). This means that the actual implemented session is $M \mid \mathsf{p} : S$. Since $(T, S) \in \mathscr{S}$, we know that this session is $\mathscr{S}$-correct from the very definition of $\mathscr{S}$.

To solve equation (2.1), we define an operator $\mathbf{F}$ that computes the fair subtyping relation induced by a generic pre-subtyping $\mathscr{S}$, thus:

Definition 2.3 ($\mathscr{S}$-subtyping)**.** The subtyping relation induced by $\mathscr{S}$, denoted by $\mathbf{F}(\mathscr{S})$, is defined as:

$$\begin{aligned}\mathbf{F}(\mathscr{S}) \stackrel{\text{def}}{=}\ & \{(t, \mathsf{top}) \mid t \in \mathscr{T}\} \cup \{(\mathsf{bot}, t) \mid t \in \mathscr{T}\} \\ & \cup \{(T, S) \mid \forall M, \mathtt{p} : (M \mid \mathtt{p} : T)\ \mathscr{S}\text{-correct implies } (M \mid \mathtt{p} : S)\ \mathscr{S}\text{-correct}\}\end{aligned}$$

Note that $\mathbf{F}(\mathscr{S})$ is itself a pre-subtyping relation. Also, the set of pre-subtyping relations forms a complete lattice: given an arbitrary family $\{\mathscr{S}_i\}_{i \in I}$ of pre-subtyping relations, $\bigvee_{i \in I} \mathscr{S}_i$ is the smallest pre-subtyping that includes all the $\mathscr{S}_i$ and $\bigwedge_{i \in I} \mathscr{S}_i$ is the largest pre-subtyping included in all the $\mathscr{S}_i$. Therefore, we can conclude that $\mathbf{F}$ has a greatest fixpoint thanks to the Knaster-Tarski theorem provided that $\mathbf{F}$ is monotone. This property does indeed hold (its proof, which involves a number of technical results, can be found in Appendix A).

Theorem 2.4. $\mathbf{F}$ *is monotone.*

At last we can define fair subtyping as the greatest fixpoint of $\mathbf{F}$:

Definition 2.5 (fair subtyping)**.** The *fair subtyping* relation, denoted by $\leqslant$, is the greatest fixpoint of $\mathbf{F}$. We write $\lessgtr$ for the equivalence relation induced by $\leqslant$, namely $\lessgtr = \leqslant \cap \leqslant^{-1}$.

Having closed the circularity between session correctness (Definition 2.2) and subtyping (Definition 2.3), we can dispense with the annotations in transition relations. We will therefore write $\xrightarrow{\ell}$ (respectively, $\stackrel{\ell}{\Longrightarrow}$) in place of $\xrightarrow{\ell}_{\leqslant}$ (respectively, $\stackrel{\ell}{\Longrightarrow}_{\leqslant}$). We can also define the notion of session correctness that uses (and induces) $\leqslant$ as subtyping:

Definition 2.6 (correct session)**.** We say that M is *correct* if M is $\leqslant$-correct.

A thorough study of the subtyping relation that solely relies on Definitions 2.3 and 2.5 is hard, because of the universal quantification over an infinite set of contexts M and the fact that $\leqslant$ is the solution of a fixpoint equation. Nonetheless, a few basic properties of $\leqslant$ are easy to establish.

Proposition 2.7. *The following relations hold:*

(1) $\mathtt{p}?\mathtt{a}\langle t\rangle.T \leqslant \mathtt{p}?\mathtt{a}\langle t\rangle.T + \mathtt{p}?\mathtt{b}\langle s\rangle.S$;
(2) $\mathtt{p}!\mathtt{a}\langle t\rangle.T \oplus \mathtt{p}!\mathtt{b}\langle s\rangle.S \leqslant \mathtt{p}!\mathtt{a}\langle t\rangle.T$;
(3) $\mathtt{p}?\mathtt{a}\langle t\rangle.T \leqslant \mathtt{p}?\mathtt{a}\langle s\rangle.T$ *if and only if* $t \leqslant s$;
(4) $\mathtt{p}!\mathtt{a}\langle t\rangle.T \leqslant \mathtt{p}!\mathtt{a}\langle s\rangle.T$ *if and only if* $s \leqslant t$;
(5) $\mathsf{bot} \lessgtr \mu x.\mathtt{p}?\mathtt{a}\langle t\rangle.x \lessgtr \mu y.\mathtt{q}!\mathtt{b}\langle s\rangle.y$.

To get some acquaintance with $\leqslant$, we conclude this section with an informal discussion on the reasons that make the relations in Proposition 2.7 hold. The rest of the paper is then entirely devoted to providing alternative characterizations of $\leqslant$.

Regarding item (1), observe that any session M such that $M \mid \mathtt{q} : \mathtt{p}?\mathtt{a}\langle t\rangle.T$ is correct must eventually send an $\mathtt{a}$-tagged message to $\mathtt{q}$ and the argument of the message must be some $t' \leqslant t$. Therefore, the same session will interact successfully with $\mathtt{q} : \mathtt{p}?\mathtt{a}\langle t\rangle.T + \mathtt{p}?\mathtt{b}\langle s\rangle.S$, which accepts $\mathtt{b}$-tagged messages in addition to $\mathtt{a}$-tagged ones. Item (2) is the dual of item (1): any session M such that $M \mid \mathtt{q} : (\mathtt{p}!\mathtt{a}\langle t\rangle.T \oplus \mathtt{p}!\mathtt{b}\langle s\rangle.S)$ is correct must be

willing to accept *both* $\mathtt{a}$-tagged messages with an argument of type t as well as $\mathtt{b}$-tagged messages with an argument of type s. This is because of the two reductions

$$\mathtt{q} : \mathtt{p}!\mathtt{a}\langle t\rangle.T \oplus \mathtt{p}!\mathtt{b}\langle s\rangle.S \xrightarrow{\tau} \mathtt{q} : \mathtt{p}!\mathtt{a}\langle t\rangle.T \qquad \text{and} \qquad \mathtt{q} : \mathtt{p}!\mathtt{a}\langle t\rangle.T \oplus \mathtt{p}!\mathtt{b}\langle s\rangle.S \xrightarrow{\tau} \mathtt{q} : \mathtt{p}!\mathtt{b}\langle s\rangle.S$$

where $\mathtt{q}$ may autonomously choose to send either kind of message. In particular, we have that $M \mid \mathtt{q} : \mathtt{p}!\mathtt{a}\langle t\rangle.T$ must be correct.

Items (3) and (4) focus on the variance properties of $\leqslant$ with respect to the type of arguments in input and output actions. In item (3), just like in item (1), a session M such that $M \mid \mathtt{q} : \mathtt{p}?\mathtt{a}\langle t\rangle.T$ is correct can send to $\mathtt{q}$ messages whose argument has *at most* type t. This is because $\mathtt{q} : \mathtt{p}?\mathtt{a}\langle t\rangle.T$ accepts $\mathtt{a}$-tagged messages whose argument is t *or* $\leqslant$*-smaller* (c.f. rule (T-Input) in Table 2). If we replace the behavior associated with $\mathtt{q}$ with $\mathtt{p}?\mathtt{a}\langle s\rangle.T$, then it must necessarily be the case that $t \leqslant s$, namely that the new behavior be willing to accept a broader range of messages than those accepted by $\mathtt{p}?\mathtt{a}\langle t\rangle.T$. The dual scenario occurs in item (4). Overall, we see that $\leqslant$ is *covariant* with respect to input actions and *contravariant* with respect to output actions.

Item (5) illustrates the possibly most peculiar feature of the $\leqslant$ relation, which does not distinguish between apparently unrelated behaviors. In fact, all these behaviors lack the possibility of successful termination (no end subterm occurs in them). Consequently, there is no M that, combined with any of them, yields a correct session and they all happen to be the $\leqslant$-smallest elements of fair subtyping.

3. Coinductive Characterization of Fair Subtyping

3.1. *Relationship with Unfair Subtyping*

We begin our study of $\leqslant$ by recalling the traditional subtyping relation for session types [Gay and Hole, 2005], which we denote by $\leqslant_{\mathsf{U}}$ and dub "unfair" subtyping to distinguish it from $\leqslant$. We show that $\leqslant_{\mathsf{U}}$ and $\leqslant$ are incomparable and we identify the class of session types for which $\leqslant$ is a refinement of $\leqslant_{\mathsf{U}}$.

Definition 3.1 (unfair subtyping [Gay and Hole, 2005])**.** We say that $\mathscr{U}$ is a *coinductive unfair subtyping relation* if $(t, s) \in \mathscr{U}$ implies either:

(1) $t = \mathsf{bot}$, or
(2) $s = \mathsf{top}$, or
(3) $t = s = \mathsf{end}$, or
(4) $t = \sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i$ and $s = \sum_{i\in I\cup J} \mathtt{p}?\mathtt{a}_i\langle s_i\rangle.S_i$ and $(t_i, s_i) \in \mathscr{U}$ and $(T_i, S_i) \in \mathscr{U}$ for every $i \in I$, or
(5) $t = \bigoplus_{i\in I\cup J} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i$ and $s = \bigoplus_{i\in I} \mathtt{p}!\mathtt{a}_i\langle s_i\rangle.S_i$ and $(s_i, t_i) \in \mathscr{U}$ and $(T_i, S_i) \in \mathscr{U}$ for every $i \in I$.

Unfair subtyping, denoted by $\leqslant_{\mathsf{U}}$, is the largest coinductive unfair subtyping relation.

Items (1) and (2) state the expected properties of bot and top as the smallest and biggest types, respectively. Item (3) states that the only subtype of end is end. Item (4) is the standard *covariant* rule for input actions and states that it is safe for a process that is capable of receiving any message from the set $\{\mathtt{p}?\mathtt{a}_i\langle t\rangle \mid i \in I \cup J, t \leqslant_{\mathsf{U}} s_i\}$ to wait for messages from a channel from which a subset of messages $\{\mathtt{p}?\mathtt{a}_i\langle t\rangle \mid i \in I, t \leqslant_{\mathsf{U}} t_i\}$ can

Table 3. Axiomatization of unfair subtyping (coinductive definition).

$$\text{(S-Bot)}\quad \mathsf{bot} \leqslant_{\mathsf{U}} t \qquad \text{(S-Top)}\quad t \leqslant_{\mathsf{U}} \mathsf{top} \qquad \text{(S-End)}\quad \mathsf{end} \leqslant_{\mathsf{U}} \mathsf{end}$$

$$\text{(S-Input)}\quad \frac{\forall i \in I : t_i \leqslant_{\mathsf{U}} s_i \qquad \forall i \in I : T_i \leqslant_{\mathsf{U}} S_i}{\sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i \leqslant_{\mathsf{U}} \sum_{i\in I\cup J} \mathtt{p}?\mathtt{a}_i\langle s_i\rangle.S_i}$$

$$\text{(S-Output)}\quad \frac{\forall i \in I : s_i \leqslant_{\mathsf{U}} t_i \qquad \forall i \in I : T_i \leqslant_{\mathsf{U}} S_i}{\bigoplus_{i\in I\cup J} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i \leqslant_{\mathsf{U}} \bigoplus_{i\in I} \mathtt{p}!\mathtt{a}_i\langle s_i\rangle.S_i}$$

be received. Item (5) is dual of item (4) and deals with outputs. It states that a process can safely use a channel to send any message from the set $\{\mathtt{p}!\mathtt{a}_i\langle t\rangle \mid i \in I, t \leqslant_{\mathsf{U}} s_i\}$ if the channel can accept any message from the set $\{\mathtt{p}!\mathtt{a}_i\langle t_i\rangle \mid i \in I \cup J\}$ where $s_i \leqslant_{\mathsf{U}} t_i$ for every $i \in I$. Observe that item (4) generalizes Propositions 2.7(1,3) and item (5) generalizes Propositions 2.7(2,4).

As shown by Gay and Hole [2005], $\leqslant_{\mathsf{U}}$ is the largest preorder that satisfies the axioms and rules in Table 3.

Remark 3.2. It is clear from Definition 3.1 and Table 3 that liveness plays a major role in the definition of session correctness also in dyadic session type theories. Indeed, if safety were the only concern, then it would be feasible (and technically easy) to allow for the subtyping laws

$$\mathsf{end} \leqslant_{\mathsf{U}} \sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i \qquad \text{and} \qquad \bigoplus_{i\in I} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i \leqslant_{\mathsf{U}} \mathsf{end}\,.$$

In the former case, a process waiting for a message is allowed to use a channel from which no message ever arrives. In the latter case, a process not sending any message is allowed to use a channel on which it is possible to send a message. In both cases the safety of the session is preserved (no message of unexpected type is ever received, no message of unexpected type is ever sent), but liveness is not. Therefore, our focus on liveness in this work is not really a novelty for session type theories. Rather, it is the shift from dyadic to multi-party session that makes liveness preservation more subtle. ■

Unfair subtyping is appealing because of its simple and intuitive definition. Unfortunately, it is easy to see that $\leqslant$ and $\leqslant_{\mathsf{U}}$ are incomparable relations:

— On the one hand, we have $T_1 \not\leqslant_{\mathsf{U}} S_1$ and $T_1 \leqslant S_1$ by taking $T_1 = \mu x.\mathtt{p}?\mathtt{a}.x$ and $S_1 = \mu y.\mathtt{q}!\mathtt{b}.y$. Indeed, T_1 and S_1 are $\leqslant$-equivalent, because neither of them can be part of a correct session, but they are unrelated by $\leqslant_{\mathsf{U}}$ because, according to Definition 3.1, related session types must be syntactically similar.

— On the other hand, we have $T_2 \leqslant_{\mathsf{U}} S_2$ and $T_2 \not\leqslant S_2$ by taking $T_2 = \mu x.\mathtt{p}?\mathtt{a}.(\mathtt{p}!\mathtt{b}.x \oplus \mathtt{p}!\mathtt{c}.\mathsf{end})$ and $S_2 = \mu y.(\mathtt{p}?\mathtt{a}.\mathtt{p}!\mathtt{b}.y + \mathtt{p}?\mathtt{b}.\mathsf{end})$. A behavior that distinguishes T_2 from S_2 is $R = \mu z.\mathtt{q}!\mathtt{a}.(\mathtt{q}?\mathtt{b}.z + \mathtt{q}?\mathtt{c}.\mathsf{end})$ because $\mathtt{p} : R \,|\, \mathtt{q} : T_2$ is correct but $\mathtt{p} : R \,|\, \mathtt{q} : S_2$ is not. This example shows that $\leqslant_{\mathsf{U}}$ allows the "unfair" behavior in which $\mathtt{q}$ never sends the $\mathtt{p}!\mathtt{c}$ message on which $\mathtt{p}$ relies to terminate.

In this section we show how to amend the definition of $\leqslant_{\mathsf{U}}$ so as to completely characterize $\leqslant$. We do so in three steps:

(1) We introduce a normal form for session types that allows us to focus on the subclass of *viable* session types, those that can be part of correct sessions. We show that $T \leqslant_{\mathsf{U}} S$ is a necessary condition for $T \leqslant S$ when T and S are in normal form.
(2) We express $T \leqslant S$ as the combination of the familiar $T \leqslant_{\mathsf{U}} S$ unfair subtyping for session types and a $T \prec S$ relation that holds when the paths leading to successful termination in T that have disappeared in S do not endanger correctness.
(3) We show that the $T \prec S$ relation can be reduced to checking the viability of a suitably defined $T - S$ session type, somehow representing the "behavioral difference" between T and S.

3.2. *Viability and Normal Form*

We have seen that there exist flawed session types that cannot occur in any correct session and that these session types are all equivalent according to fair subtyping regardless of their syntax. We reserve a name for session types that *can* occur in correct sessions.

Definition 3.3 (viability)**.** We say that T is *viable* if there exist M and $\mathtt{p}$ such that $M \mid \mathtt{p} : T$ is correct. We write $\mathscr{T}_{\mathsf{v}}$ for the set of viable session types.

A session type T is *not* viable if and only if $T \leqslant \mathsf{bot}$. That is, being not viable means being ($\leqslant$-smaller than) the bottom type. In a sense, viability and non-viability correspond to the notions of inhabited and empty types in traditional type theories, with one important caveat: there exist processes that behave according to non-viable session types, but there are no contexts that can successfully interact (according to Definition 2.2) with these processes. The existence of non-viable session types hinders the coinductive characterization of the subtyping relation in the style of Definition 3.1 because this characterization is based on the intuition that semantically related session types must be syntactically similar, while we have seen that this is not necessarily true when non-viable session types are involved.

We now define a normal form that makes non-viable session types readily detectable and the syntax of viable ones meaningful. Intuitively, we want to focus on those session types such that every subterm contained in them has an end subterm. This property is motivated by our definition of session correctness, which imposes the reachability of a successfully terminated state. Therefore, we formalize the set of *trees* of a (session) type t as the set of all the closed subterms of t. In fact, we will distinguish between *continuation trees* and *prefix trees* of t: the former ones are the subterms that can be reached "at the top level" of t without entering any prefix of t; the latter ones are the subterms of t occurring within a prefix of t. Formally:

Definition 3.4 (trees of a type)**.** For every type t we define the sets of *continuation trees* and *prefix trees* of t as the smallest sets $\mathtt{ctrees}(t)$ and $\mathtt{ptrees}(t)$ such that:

— $T \in \mathtt{ctrees}(T)$;
— $\sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i \in \mathtt{ctrees}(T)$ or $\bigoplus_{i\in I} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i \in \mathtt{ctrees}(T)$ implies $t_i \in \mathtt{ptrees}(T)$ and $T_i \in \mathtt{ctrees}(T)$ for every $i \in I$.

A few examples to clarify the definitions are in order:

— We have $\mathtt{ctrees}(\mathsf{end}) = \{\mathsf{end}\}$ and $\mathtt{ptrees}(\mathsf{end}) = \emptyset$.

— We have $\mathtt{ctrees}(\mathsf{bot}) = \{\mathsf{bot}\}$ and $\mathtt{ctrees}(\mathsf{top}) = \mathtt{ptrees}(\mathsf{bot}) = \mathtt{ptrees}(\mathsf{top}) = \emptyset$.
— Given $T = \mu x.\mathtt{p}?\mathtt{a}\langle t\rangle.x$ we have $\mathtt{ctrees}(T) = \{T\}$ and $\mathtt{ptrees}(T) = \{t\}$.
— Given $T = \mu x.(\mathtt{q}?\mathtt{a}.(\mathtt{q}!\mathtt{c}.x \oplus \mathtt{q}!\mathtt{d}.x) + \mathtt{q}?\mathtt{b}.\mathsf{end})$ we have $\mathtt{ctrees}(T) = \{T, \mathtt{q}!\mathtt{c}.T \oplus \mathtt{q}!\mathtt{d}.T, \mathsf{end}\}$ and $\mathtt{ptrees}(T) = \{\mathsf{bot}, \mathsf{top}\}$.
— Given $T = \mu x.\mathtt{p}!\mathtt{a}\langle x\rangle.\mathsf{end}$ we have $\mathtt{ctrees}(T) = \{T, \mathsf{end}\}$ and $\mathtt{ptrees}(T) = \{T\}$.

Note that the sets $\mathtt{ctrees}(t)$ and $\mathtt{ptrees}(t)$ are always finite because t is a regular tree and therefore it is made of a finite number of distinct subtrees [Courcelle, 1983]. We are now ready to define the class of session types in normal form.

Definition 3.5 (normal form)**.** The set of types in *normal form* is the largest set $\mathscr{T}_{\mathsf{nf}} \subseteq \mathscr{T}$ such that $t \in \mathscr{T}_{\mathsf{nf}}$ implies either:

(1) $t = \mathsf{bot}$, or
(2) $t = \mathsf{top}$, or
(3) $\mathtt{ptrees}(t) \subseteq \mathscr{T}_{\mathsf{nf}}$ and $\mathsf{end} \in \mathtt{ctrees}(T)$ for every $T \in \mathtt{ctrees}(t)$.

In words, the types bot and top are in normal form and the session type T is in normal form if so is every type in $\mathtt{ptrees}(T)$ and if every continuation tree of T contains end. For example $T = \mu x.(\mathtt{q}?\mathtt{a}.(\mathtt{q}!\mathtt{c}.x \oplus \mathtt{q}!\mathtt{d}.x) + \mathtt{q}?\mathtt{b}.\mathsf{end})$ is in normal form but $S = \mu x.(\mathtt{q}?\mathtt{a}.(\mathtt{q}!\mathtt{c}.x \oplus \mathtt{q}!\mathtt{d}.\mathsf{bot}) + \mathtt{q}?\mathtt{b}.\mathsf{end})$ is not because $\mathsf{bot} \in \mathtt{ctrees}(S)$ and $\mathsf{end} \notin \mathtt{ctrees}(\mathsf{bot})$.

Now:

Theorem 3.6. *For every $t \in \mathscr{T}$ there exists $s \in \mathscr{T}_{\mathsf{nf}}$ such that $s \lesssim t$.*

We postpone the algorithm for computing the normal form of a type to Section 4.2. Intuitively, the normal form of any non-viable session type is bot and that of a viable session type T can be obtained from T by recursively computing the normal form of all the types in $\mathtt{ptrees}(T)$ and by pruning those branches of external choices whose continuation is non-viable. For example, in the session type $S = \mu x.(\mathtt{q}?\mathtt{a}.(\mathtt{q}!\mathtt{c}.x \oplus \mathtt{q}!\mathtt{d}.\mathsf{bot}) + \mathtt{q}?\mathtt{b}.\mathsf{end})$ defined above the $\mathtt{a}$-tagged branch of the external choice is useless because followed by a non-viable continuation. That branch can be pruned without changing the semantics of S, hence we have $S \lesssim \mathtt{q}?\mathtt{b}.\mathsf{end}$ where the latter session type is in normal form.

We now turn our attention to the properties of the normal form that are relevant to the characterization of fair subtyping. The first one is that every type in normal form other than bot and top is a viable session type. Therefore, the normal form effectively makes the flawed/impossible behaviors easily recognizable from their syntax.

Theorem 3.7. *For every $t \in \mathscr{T}_{\mathsf{nf}} \setminus \{\mathsf{bot}, \mathsf{top}\}$ we have $t \in \mathscr{T}_{\mathsf{v}}$.*

The second property is that, when restricted to types in normal form, fair subtyping implies unfair subtyping.

Theorem 3.8. *Let $t, s \in \mathscr{T}_{\mathsf{nf}}$. Then $t \leqslant s$ implies $t \leqslant_{\mathsf{U}} s$.*

Remark 3.9. Given the notion of session correctness that we are considering, requiring the reachability of an end state for all of the session types describing the behavior of the participants of the session, it would appear natural to *define* session types as those terms generated by the grammar in Table 1 that satisfy condition (3) of Definition 3.5. The reason why we did not do so in the first place is that in general we need to be able to write and reason on terms according to that grammar in its full generality. In particular,

in Section 3.4 we will define an operator that may produce session types that are not in normal form. ■

Note that the normal form imposes constraints only on the subterms in $\mathtt{ctrees}(t)$, while the ones in $\mathtt{ptrees}(t)$ are only required to be in normal form. This observation tells us that the viability of a session type T does *not* depend in any way on the types that occur in the prefixes of T. To see why this is the case, consider a session M such that $M \,|\, \mathtt{p} : T$ is $\mathscr{S}$-correct. Then we can always define a "gentler" and "more accommodating" session $[M]$ from M such that $[M] \mid \mathtt{p} : T$ is $\mathscr{R}$-correct for every $\mathscr{R}$. The session $[M]$ is defined thus:

Definition 3.10 (ground session / session type)**.** We say that a session (type) is *ground* if every type in its output prefixes is bot and every type in its input prefixes is top. We write $[M]$ (respectively, $[T]$) for the ground session (type) corresponding to M (respectively, T).

Intuitively, $[M]$ is gentler than M because it only sends message arguments of type bot (which T can always receive since $(\mathsf{bot}, t) \in \mathscr{R}$ for every pre-subtyping $\mathscr{R}$) and it is more accommodating than M because it accepts message arguments of any type t (which is granted from $(t, \mathsf{top}) \in \mathscr{R}$). We therefore have the following proposition:

Proposition 3.11. *T is viable if and only if $[T]$ is viable.*

Proof. Trivial by definition of ground session (type). □

The fact that the viability of T is independent of the types in $\mathtt{ptrees}(T)$ has important consequences in the following. In particular, it allows us to characterize non-viability in a purely inductive way (see Section 4.1).

3.3. *Characterization of the Termination Property*

Now that we work with session types with a "meaningful" syntax (Theorem 3.8) we can more easily focus on the reasons why $\leqslant_{\mathsf{U}}$ is unsound with respect to $\leqslant$. To begin with, we see that $\leqslant_{\mathsf{U}}$ cannot introduce deadlocks and it can introduce livelocks only when recursive session types are involved:

Proposition 3.12. *Let $T, S \in \mathscr{T}_{\mathsf{nf}}$ and $T \leqslant_{\mathsf{U}} S$. The following properties hold:*

(1) *If T and S are finite, then $T \leqslant S$;*

(2) *If $M \mid \mathtt{p} : T$ is correct and $M \mid \mathtt{p} : S \stackrel{\tau}{\Longrightarrow} N \stackrel{\tau}{\not\longrightarrow}$, then $N \xrightarrow{\checkmark}$.*

Proposition 3.12 shows that $\leqslant_{\mathsf{U}}$ is not too far away from being a characterization of $\leqslant$. Therefore, we attempt at characterizing $T \leqslant S$ as the combination of two relations: $T \leqslant_{\mathsf{U}} S$, expressing a *safety* property (S does not introduce deadlocks), and $T \prec S$, expressing a *liveness* property (S does not preclude the successful termination of any context M that completes T). The "ruled by" relation $\prec$ is defined thus:

Definition 3.13. Let $T, S \in \mathscr{T}_{\mathsf{nf}}$ and $T \leqslant_{\mathsf{U}} S$. We say that T is *ruled by* S, written $T \prec S$, if $M \mid \mathtt{p} : T$ correct implies $M \mid \mathtt{p} : S \stackrel{\checkmark}{\Longrightarrow}$ for every M ground.

When $T \leqslant_{\mathsf{U}} S$, the behavior S may preclude successful termination of a context M that completes T only when some outputs in T have disappeared in S. The additional

property $T \prec S$ prevents this from happening. Observe that $T \leqslant S$ implies $T \prec S$, but $T \prec S$ is much weaker than $T \leqslant S$. For example, we have $\mathsf{p!a.end} \prec \mathsf{p!a.end} \oplus \mathsf{p!b.end}$ even though $\mathsf{p!a.end} \not\leqslant \mathsf{p!a.end} \oplus \mathsf{p!b.end}$. In fact, $\prec$ precisely captures the difference between $\leqslant_{\mathsf{U}}$ and $\leqslant$, in the sense that the largest relation included in $\leqslant_{\mathsf{U}}$ that is coherent with $\prec$ coincides with $\leqslant$.

Theorem 3.14. *Let $\leqslant_{\mathsf{C}}$ be the largest coinductive unfair subtyping such that $T \leqslant_{\mathsf{C}} S$ implies $T \prec S$. Then $t \leqslant s$ if and only if $t \leqslant_{\mathsf{C}} s$ for every $t, s \in \mathscr{T}_{\mathsf{nf}}$.*

3.4. *Characterization of $\prec$ and Behavioral Difference*

Although $\prec$ is a much weaker relation than $\leqslant$, it still is defined in terms of an infinite set of (ground) contexts M. We devote the last part of this section to seeking an alternative characterization of it.

Assume for the sake of discussion that $T \leqslant_{\mathsf{U}} S$ and $T \not\prec S$, meaning that there exists some context M such that the correctness of $M \mid \mathtt{p} : T$ crucially depends on the outputs that T emits and that S does not. To find M, we define a session type $T - S$ that somehow represents the "difference" between T and S and that is viable if (and only if) such M does exist. The intuition is for $T - S$ to be the same as T, except that every end that lies on a path shared by T and S is turned to a bot in $T - S$. Therefore, any hypothetical context M such that $M \mid \mathtt{p} : (T - S)$ is correct can only count on those end leaves found in T that have disappeared in S. Also, $T - S$ performs no more inputs than those performed by T. In this way we stay assured that, if M exists, it does not use any additional input capability provided by S that is not provided by T.

Formally:

Definition 3.15 (session type difference)**.** Let $T \leqslant_{\mathsf{U}} S$. The *difference* of T and S, denoted by $T - S$, is coinductively defined by the following equations:

$$\begin{aligned} \mathsf{bot} - S &= \mathsf{end} - \mathsf{end} = \mathsf{bot} \\ \textstyle\sum_{i \in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i - \sum_{i \in I \cup J} \mathtt{p}?\mathtt{a}_i\langle s_i\rangle.S_i &= \sum_{i \in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.(T_i - S_i) \\ \textstyle\bigoplus_{i \in I \cup J} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i - \bigoplus_{i \in I} \mathtt{p}!\mathtt{a}_i\langle s_i\rangle.S_i &= \bigoplus_{i \in J \setminus I} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i \oplus \bigoplus_{i \in I} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.(T_i - S_i) \end{aligned}$$

The notation $T - S$ is justified by the following properties:

Proposition 3.16. *Let $T \leqslant_{\mathsf{U}} S$. The following properties hold:*

(1) *$T - S \overset{\varphi}{\Longrightarrow}$ if and only if $T \overset{\varphi}{\Longrightarrow}$;*

(2) *$[T - S] \overset{\varphi\checkmark}{\Longrightarrow}$ if and only if $[T] \overset{\varphi\checkmark}{\Longrightarrow}$ and $[S] \overset{\varphi}{\not\Longrightarrow}$.*

Proof. Straightforward consequence of the definition of $T - S$. □

In words, $T - S$ performs all traces of actions performed by T and the *completed traces* of $[T]$ that are not completed traces of $[S]$. If we let $\mathtt{traces}(T) = \{\varphi \mid T \overset{\varphi\checkmark}{\Longrightarrow}\}$ denote the set of completed traces of T, then we may reformulate Proposition 3.16(2) as

$$\mathtt{traces}([T - S]) = \mathtt{traces}([T]) \setminus \mathtt{traces}([S])\,.$$

The fact that Proposition 3.16(2) holds for ground session types $[T - S]$, $[T]$, and $[S]$ is because, in general, T and S may input/output messages with different argument types

in accordance with the covariance/contravariance of $\leqslant_{\mathsf{U}}$ with respect to input/output prefixes.

Example 3.17. Consider the session types T, S_2, and S_∞ from Figure 1. We can express their corresponding set of completed traces using regular expressions over the alphabet of actions of the form `seller!Buy` and `seller!Done`, in this way:

$$\begin{aligned}\mathtt{traces}(T) &= \mathtt{seller!Buy}^*\ \mathtt{seller!Done}\\ \mathtt{traces}(S_2) &= (\mathtt{seller!Buy}\ \mathtt{seller!Buy})^*\ \mathtt{seller!Done}\\ \mathtt{traces}(S_\infty) &= \emptyset\end{aligned}$$

In particular, we have: $\mathtt{traces}(T)$ is the language of strings made of an arbitrary number of `seller!Buy` actions followed by exactly on `seller!Done` action; $\mathtt{traces}(S_2)$ is similar, but the number of `seller!Buy` actions is always even; $\mathtt{traces}(S_\infty)$ is empty because S_∞ has no end subterm.

Now, according to Definition 3.15 we have:

$$\begin{aligned}T - S_2 &= \mathtt{seller!Buy}.(\mathtt{seller!Buy}.(T - S_2) \oplus \mathtt{seller!Done}.\mathsf{end}) \oplus \mathtt{seller!Done}.\mathsf{bot}\\ T - S_\infty &= T\end{aligned}$$

from which we verify that $\mathtt{traces}(T - S_\infty) = \mathtt{traces}(T)$ and

$$\mathtt{traces}(T - S_2) = \mathtt{seller!Buy}\ (\mathtt{seller!Buy}\ \mathtt{seller!Buy})^*\ \mathtt{seller!Done}$$

which corresponds to the language of strings made of an odd number of `seller!Buy` actions followed by exactly on `seller!Done` action, that is the difference of $\mathtt{traces}(T)$ and $\mathtt{traces}(S_2)$. ■

Example 3.18. Consider the session types T and S from Figure 2. We have:

$$\begin{aligned}\mathtt{traces}(T) &= (?\mathtt{Offer}\ (!\mathtt{OK} + !\mathtt{NO}))^*\ ?\mathtt{Done}\\ \mathtt{traces}(S) &= (?\mathtt{Offer}\ !\mathtt{NO}\ ?\mathtt{Offer}\ (!\mathtt{OK} + !\mathtt{NO}))^*\ (\varepsilon + ?\mathtt{Offer}\ !\mathtt{NO})\ ?\mathtt{Done}\end{aligned}$$

showing that `seller` : S systematically rejects any odd-indexed offer from `buyer`. Now we have:

$$T - S = ?\mathtt{Offer}.(!\mathtt{OK}.T \oplus !\mathtt{NO}.(?\mathtt{Offer}.(!\mathtt{OK}.(T - S) \oplus !\mathtt{NO}.(T - S)) + ?\mathtt{Done}.\mathsf{bot})) + ?\mathtt{Done}.\mathsf{bot}$$

and

$$\mathtt{traces}(T - S) = (?\mathtt{Offer}\ !\mathtt{NO}\ ?\mathtt{Offer}\ (!\mathtt{OK} + !\mathtt{NO}))^*\ ?\mathtt{Offer}\ !\mathtt{OK}\ \mathtt{traces}(T)$$

where every trace has at least one odd-indexed offer that is accepted. This suggests the existence of a behavior for `buyer` such as (1.2). ■

The general relation between $T \prec S$ and $T - S$ is stated by the following result, which allows us to reduce $T \prec S$ to the viability of $T - S$.

Theorem 3.19. *Let $T, S \in \mathscr{T}_{\mathsf{nf}}$ and $T \leqslant_{\mathsf{U}} S$. Then $T \prec S$ if and only if $T - S$ is not viable.*

The basic idea of the proof, which is detailed in Section B, is that any session M such that $M \mid \mathtt{p} : (T - S)$ is correct crucially relies on the paths to end that are in T but not in S. In particular, it can be shown that $M \mid \mathtt{p} : T$ is correct and $M \mid \mathtt{p} : S \overset{\checkmark}{\not\Longrightarrow}$, which means $T \not\prec S$.

Table 4. Termination paths of a session type (inductive definition).

$$\text{(P-End)}\quad \mathsf{end} \downarrow \{\mathsf{end}\}$$

$$\text{(P-Input)}\quad \frac{T = \sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i \qquad \exists k \in I : T_k \downarrow \pi}{T \downarrow \{T\} \cup \pi}$$

$$\text{(P-Output)}\quad \frac{T = \bigoplus_{i\in I} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i \qquad \exists k \in I : T_k \downarrow \pi}{T \downarrow \{T\} \cup \pi}$$

4. Deduction Systems and Algorithms

4.1. *Axiomatization of Fair Subtyping*

The viability of a session type T is related to the reachability of end subtrees occurring in it. The first notion we formalize is that of *termination path* of a session type T, which is defined as a set of subterms of T denoting a path from T to one of its end subterms. If we think of a session type T as a possibly infinite, regular tree, a termination path of T is just a set of nodes traversed following any path from T to one of its leaves.

Definition 4.1 (termination paths)**.** The relation $T \downarrow \pi$, read T has *termination path* π, is inductively defined by the axiom and rules in Table 4. We write $\mathtt{paths}(T)$ for the set of termination paths of T, that is $\mathtt{paths}(T) \stackrel{\text{def}}{=} \{\pi \mid T \downarrow \pi\}$.

According to axiom (P-End), the session type end has one termination path only, which contains end. Rules (P-Input) and (P-Output) simply state that branching session types T have as many termination paths as the overall number of termination paths of their branches, each path enriched with the T node.

The following ones are straightforward properties of termination paths:

Proposition 4.2. *The following properties hold:*

(1) $\mathtt{paths}(T)$ *is finite for every* T*;*

(2) $T \stackrel{\varphi\checkmark}{\Longrightarrow}$ *implies* $T \downarrow \pi$ *and* $\pi \subseteq \{S \mid \exists \psi \leq \varphi : T \stackrel{\psi}{\Longrightarrow} S\}$ *for some* π*.*

Proof. Regarding item (1), observe that $T \downarrow \pi$ implies $\pi \subseteq \mathtt{ctrees}(T)$ and $\mathtt{ctrees}(T)$ is finite. Item (2) follows from a simple induction on φ. □

Item (2) states that, whenever there is a derivation $T \stackrel{\varphi\checkmark}{\Longrightarrow}$, then this derivation goes through every node of some termination path of T and possibly other nodes. The reason why the derivation $T \stackrel{\varphi\checkmark}{\Longrightarrow}$ may go through nodes that are not in a termination path of T is that the traversal of internal choices may yield subterms that are not in $\mathtt{ctrees}(T)$ because of rule (T-Choice). For example, a session type such as $T = \mathtt{p}!\mathtt{a}.\mathsf{end} \oplus \mathtt{p}!\mathtt{b}.\mathsf{end}$ has only the termination path $\{T, \mathsf{end}\}$, but T has two derivations

$$T \xrightarrow{\tau} \mathtt{p}!\mathtt{a}.\mathsf{end} \xrightarrow{\mathtt{q}:\mathtt{p}!\mathtt{a}} \mathsf{end}$$
$$T \xrightarrow{\tau} \mathtt{p}!\mathtt{b}.\mathsf{end} \xrightarrow{\mathtt{q}:\mathtt{p}!\mathtt{b}} \mathsf{end}$$

that traverse the nodes $\mathtt{p}!\mathtt{a}.\mathsf{end}$ and $\mathtt{p}!\mathtt{a}.\mathsf{end}$ not in $\mathtt{ctrees}(T)$.

We can now present the complete axiomatization of non-viability as the least predicate

Table 5. Axiomatization of non-viability (inductive definition).

$$\frac{\forall T \downarrow \pi : \exists S \in \pi : \mathtt{NonViable}(S)}{\mathtt{NonViable}(T)}\ (\text{B-Path}) \qquad \frac{\exists k \in I : \mathtt{NonViable}(T_k)}{\mathtt{NonViable}(\bigoplus_{i \in I} \mathtt{p!a}_i\langle t_i \rangle.T_i)}\ (\text{B-Output})$$

$\mathtt{NonViable}(T)$ defined by the rules in Table 5. Rule (B-Path) serves as an axiom when T has no termination paths. In this case T has no end subterm and therefore is trivially non-viable (reachability of end is a necessary condition for viability). In general, the rule says that T is non-viable if *every* termination path of T goes through some node S (different from T) that is itself non-viable. Rule (B-Output) states that an internal choice T is non-viable if *some* of its branches are non-viable. This follows from the fact that the participant behaving as T may autonomously choose that particular branch, which leads the whole session to an unrecoverable error state.

The presented axiomatization is correct and complete with respect to non-viability:

Theorem 4.3. $\mathtt{NonViable}(T)$ *if and only if* T *is not viable.*

Example 4.4. Consider the session type $T = \mu x.(\mathtt{p?a}.x + \mathtt{p?b}.(\mathtt{q!c}.\mathsf{end} \oplus \mathtt{q!d}.\mathsf{bot}))$ and observe that it has only the termination path $\pi = \{T, \mathtt{q!c}.\mathsf{end} \oplus \mathtt{q!d}.\mathsf{bot}, \mathsf{end}\}$. We have the following derivation showing that T is not viable:

$$\dfrac{\mathtt{q!c}.\mathsf{end} \oplus \mathtt{q!d}.\mathsf{bot} \in \pi \qquad \dfrac{\dfrac{}{\mathtt{NonViable}(\mathsf{bot})}\ (\text{B-Path})}{\mathtt{NonViable}(\mathtt{q!c}.\mathsf{end} \oplus \mathtt{q!d}.\mathsf{bot})}\ (\text{B-Output})}{\mathtt{NonViable}(T)}\ (\text{B-Path})$$

The topmost application of rule (B-Path) has no premises because bot has no termination paths. Note that rule (B-Output) is essential for proving $\mathtt{NonViable}(T)$ (and more generally for completeness of $\mathtt{NonViable}(\cdot)$) because $\mathtt{q!c}.\mathsf{end} \oplus \mathtt{q!d}.\mathsf{bot}$ has a termination path, but it also has a non-viable branch. ■

Example 4.5. Consider the session types T and S from Figure 2. We have:

$$\begin{aligned} T - S &= \mathtt{?Offer}.R_1 + \mathtt{?Done}.\mathsf{bot} \\ R_1 &= \mathtt{!OK}.T \oplus \mathtt{!NO}.R_2 \\ R_2 &= \mathtt{?Offer}.R_3 + \mathtt{?Done}.\mathsf{bot} \\ R_3 &= \mathtt{!OK}.(T - S) \oplus \mathtt{!NO}.(T - S) \end{aligned}$$

Every termination path of $T - S$ has either the form $\{T - S, R_1\} \cup \pi$ or the form $\{T - S, R_1, R_2, R_3\} \cup \pi$ where π is a termination path of T. None of the session types in π is non-viable because T is in normal form and different from bot. As a consequence, R_1 is non-viable only if R_2 is non-viable, R_2 is non-viable only if R_3 is non-viable, and R_3 is non-viable only if $T - S$ is non-viable. In conclusion, we are not able to find a finite derivation proving $\mathtt{NonViable}(T - S)$, as expected. ■

Now that we have a complete axiomatization for non-viability, we can easily derive one for fair subtyping. We define $\leqslant_{\mathsf{F}}$ as the largest relation that satisfies the axioms and rules of Table 6, which basically are the same axioms and rules for unfair subtyping (Table 3)

Table 6. Axiomatization of fair subtyping for session types in normal form (coinductive definition).

$$\text{(F-Bottom)}\quad \mathsf{bot} \leqslant_{\mathsf{F}} t \qquad \text{(F-Top)}\quad t \leqslant_{\mathsf{F}} \mathsf{top} \qquad \text{(F-End)}\quad \mathsf{end} \leqslant_{\mathsf{F}} \mathsf{end}$$

$$\text{(F-Input)}\quad \frac{\forall i \in I : t_i \leqslant_{\mathsf{F}} s_i \qquad \forall i \in I : T_i \leqslant_{\mathsf{F}} S_i}{\sum_{i \in I} \mathtt{p?a}_i\langle t_i\rangle.T_i \leqslant_{\mathsf{F}} \sum_{i \in I \cup J} \mathtt{p?a}_i\langle s_i\rangle.S_i}$$

$$\text{(F-Output)}\quad \frac{\forall i \in I : s_i \leqslant_{\mathsf{F}} t_i \qquad \forall i \in I : T_i \leqslant_{\mathsf{F}} S_i \qquad \mathtt{NonViable}(T - S)}{T = \bigoplus_{i \in I \cup J} \mathtt{p!a}_i\langle t_i\rangle.T_i \leqslant_{\mathsf{F}} \bigoplus_{i \in I} \mathtt{p!a}_i\langle s_i\rangle.S_i = S}$$

except for the additional premise $\mathtt{NonViable}(T - S)$ for rule (F-Output) relating T and S when these are internal choices. We have:

Theorem 4.6. *Let $t, s \in \mathscr{T}_{\mathsf{nf}}$. Then $t \leqslant_{\mathsf{C}} s$ if and only if $t \leqslant_{\mathsf{F}} s$.*

Note that the axiomatization is complete with respect to the set of session types in normal form. In Section 4.2 we will see an algorithm for deciding the viability of a session type. This will give us an effective way of computing the normal form of a session type according to the equations (4.1).

Example 4.7. Consider once again the session types T and S from Example 3.17. The infinite derivation

$$\dfrac{\dfrac{\dfrac{\vdots}{T \leqslant_{\mathsf{F}} S} \qquad \mathtt{NonViable}(T - \mathtt{seller!Buy}.S)}{T \leqslant_{\mathsf{F}} \mathtt{seller!Buy}.S} \qquad \mathsf{end} \leqslant_{\mathsf{F}} \mathsf{end} \qquad \mathtt{NonViable}(T - S)}{T \leqslant_{\mathsf{F}} S}$$

shows that $T \leqslant S$. Indeed, we have

$$T - S = \mathtt{seller!Buy}.(T - \mathtt{seller!Buy}.S) \oplus \mathtt{seller!Done}.\mathsf{bot}$$
$$T - \mathtt{seller!Buy}.S = \mathtt{seller!Buy}.(T - S) \oplus \mathtt{seller!Done}.\mathsf{end}$$

and the derivations

$$\dfrac{\dfrac{}{\mathtt{NonViable}(\mathsf{bot})}\ \text{(B-Path)}}{\mathtt{NonViable}(T - S)}\ \text{(B-Output)}$$

and

$$\dfrac{\dfrac{\dfrac{}{\mathtt{NonViable}(\mathsf{bot})}\ \text{(B-Path)}}{\mathtt{NonViable}(T - S)}\ \text{(B-Output)}}{\mathtt{NonViable}(T - \mathtt{seller!Buy}.S)}\ \text{(B-Output)}$$

easily follow from Table 5. ■

4.2. *Algorithms*

We now present algorithms for deciding viability, for computing the normal form of viable session types, and for deciding subtyping. As we have seen, viability is a crucial notion

of our theory since it characterizes those behaviors that can successfully cooperate in at least one session. From a practical point of view, the algorithm for viability gives us the tool for computing the normal form of session types and for deciding fair subtyping (see Table 6).

The algorithm for deciding viability assumes initially that every subtree of some session type T is viable and iteratively discards those subtrees for which this assumption is disproved. Each iteration performs two checks, corresponding to the two rules of Table 5: a subtree $S \in \mathtt{trees}(T)$ is viable provided that end can be reached from it via a termination path whose nodes are all viable; output nodes are viable provided that every branch is viable. Formally, let the *viability sequence* for T be the sequence $\{\mathbf{V}_i^T\}_{i\in\mathbb{N}}$ of sets of session types defined in this way:

$$\begin{aligned}
\mathbf{V}_0^T &= \mathtt{ctrees}(T)\\
\mathbf{V}_{2i+1}^T &= \{S \in \mathbf{V}_{2i}^T \mid \exists \pi \in \mathtt{paths}(S) : \pi \subseteq \mathbf{V}_{2i}^T\}\\
\mathbf{V}_{2i+2}^T &= \{\mathsf{end} \in \mathbf{V}_{2i+1}^T\} \cup \{\textstyle\sum_{j\in I} \mathsf{p}?\mathsf{a}_j\langle t_j\rangle.T_j \in \mathbf{V}_{2i+1}^T\}\\
&\qquad\qquad \cup \{\textstyle\bigoplus_{j\in I} \mathsf{p}!\mathsf{a}_j\langle t_j\rangle.T_j \in \mathbf{V}_{2i+1}^T \mid \forall j \in I : T_j \in \mathbf{V}_{2i+1}^T\}
\end{aligned}$$

Every set in the sequence is finite and the sequence is decreasing. Therefore, there exists $k \in \mathbb{N}$ such that $\mathbf{V}_k^T = \mathbf{V}_{k+1}^T = \mathbf{V}_{k+2}^T$ and we denote the fixpoint of the sequence with $\mathtt{viables}(T)$. We have:

Theorem 4.8 (viability)**.** *$T \in \mathscr{T}_{\mathsf{v}}$ if and only if $T \in \mathtt{viables}(T)$.*

Example 4.9. In Example 4.4 we have shown that the session type $T = \mu x.(\mathsf{p}?\mathsf{a}.x + \mathsf{p}?\mathsf{b}.(\mathsf{q}!\mathsf{c}.\mathsf{end} \oplus \mathsf{q}!\mathsf{d}.\mathsf{bot}))$ is not viable by building a derivation for $\mathtt{NonViable}(T)$. This is the viability sequence for T:

$$\begin{aligned}
\mathbf{V}_0^T &= \{\mathsf{end}, \mathsf{bot}, \mathsf{q}!\mathsf{c}.\mathsf{end} \oplus \mathsf{q}!\mathsf{d}.\mathsf{bot}, T\}\\
\mathbf{V}_1^T &= \{\mathsf{end}, \mathsf{q}!\mathsf{c}.\mathsf{end} \oplus \mathsf{q}!\mathsf{d}.\mathsf{bot}, T\}\\
\mathbf{V}_2^T &= \{\mathsf{end}, T\}\\
\mathbf{V}_3^T = \mathbf{V}_4^T &= \mathbf{V}_5^T = \{\mathsf{end}\}
\end{aligned}$$

Note that T is in $\mathbf{V}_1^T$ because it has a termination path whose nodes are all in $\mathbf{V}_0^T$. It is only at step 3 that the algorithm discards T because all of its termination paths go through some non-viable node. ■

Once we know how to identify viable session types effectively, computing their normal form is only a matter of pruning away those subtrees that are not viable. The normal form of a type t, denoted by $\mathtt{nf}(t)$, is defined coinductively according to the following equation:

$$\mathtt{nf}(t) = \begin{cases}
\mathsf{top} & \text{if } t = \mathsf{top}\\
\mathsf{bot} & \text{if } t = T \notin \mathtt{viables}(T)\\
\mathsf{end} & \text{if } t = \mathsf{end}\\
\textstyle\sum_{i\in I, T_i\in\mathtt{viables}(T_i)} \mathsf{p}?\mathsf{a}_i\langle\mathtt{nf}(t_i)\rangle.\mathtt{nf}(T_i) & \text{if } t = \textstyle\sum_{i\in I} \mathsf{p}?\mathsf{a}_i\langle t_i\rangle.T_i\\
\textstyle\bigoplus_{i\in I} \mathsf{p}!\mathsf{a}_i\langle\mathtt{nf}(t_i)\rangle.\mathtt{nf}(T_i) & \text{if } t = \textstyle\bigoplus_{i\in I} \mathsf{p}!\mathsf{a}_i\langle t_i\rangle.T_i
\end{cases} \tag{4.1}$$

where the cases from the third to the last one apply when $t \in \mathtt{viables}(t)$. Note that,

Table 7. Algorithmic subtyping (inductive definition).

$$\text{(A-Axiom)}\quad \Gamma, (t,s) \vdash t \leqslant_{\mathsf{A}} s \qquad \text{(A-Bottom)}\quad \Gamma \vdash \mathsf{bot} \leqslant_{\mathsf{A}} t \qquad \text{(A-Top)}\quad \Gamma \vdash t \leqslant_{\mathsf{A}} \mathsf{top} \qquad \text{(A-End)}\quad \Gamma \vdash \mathsf{end} \leqslant_{\mathsf{A}} \mathsf{end}$$

$$\text{(A-Input)}\quad \frac{\forall i \in I : \Gamma, (t,s) \vdash t_i \leqslant_{\mathsf{A}} s_i \qquad \forall i \in I : \Gamma, (t,s) \vdash T_i \leqslant_{\mathsf{A}} S_i}{\Gamma \vdash t = \sum_{i \in I} \mathtt{p?a}_i\langle t_i\rangle.T_i \leqslant_{\mathsf{A}} \sum_{i \in I \cup J} \mathtt{p?a}_i\langle s_i\rangle.S_i = s}$$

$$\text{(A-Output)}\quad \frac{\forall i \in I : \Gamma, (T,S) \vdash s_i \leqslant_{\mathsf{A}} t_i \qquad \forall i \in I : \Gamma, (T,S) \vdash T_i \leqslant_{\mathsf{A}} S_i \qquad T - S \notin \mathtt{viables}(T-S)}{T = \bigoplus_{i \in I \cup J} \mathtt{p!a}_i\langle t_i\rangle.T_i \leqslant_{\mathsf{A}} \bigoplus_{i \in I} \mathtt{p!a}_i\langle s_i\rangle.S_i = S}$$

since t is a regular tree, $\mathtt{nf}(t)$ can always be expressed by means of a finite number of equations that can be folded into a finite term using μ and recursion variables (see Example 4.11 below).

Theorem 4.10 (normal form)**.** *For every $t \in \mathscr{T}$ we have $\mathtt{nf}(t) \in \mathscr{T}_{\mathsf{nf}}$ and $t \leqslant\!\!\!\leqslant \mathtt{nf}(t)$.*

Example 4.11. Consider the session type $T = \mu x.(\mathsf{q!a}.(\mathsf{q?c}.x + \mathsf{q?d}.\mathsf{bot}) \oplus \mathsf{q!b}.\mathsf{end})$ and observe that it is viable. According to equation (4.1), we have:

$$\begin{aligned} \mathtt{nf}(T) &= \mathsf{q!a}.\mathtt{nf}(\mathsf{q?c}.T + \mathsf{q?d}.\mathsf{bot}) \oplus \mathsf{q!b}.\mathtt{nf}(\mathsf{end}) \\ \mathtt{nf}(\mathsf{q?c}.T + \mathsf{q?d}.\mathsf{bot}) &= \mathsf{q?c}.\mathtt{nf}(T) \\ \mathtt{nf}(\mathsf{end}) &= \mathsf{end} \end{aligned}$$

from which we obtain $\mathtt{nf}(T) = \mu y.(\mathsf{q!a}.\mathsf{q?c}.y \oplus \mathsf{q!b}.\mathsf{end})$. ■

We conclude this section with the presentation of algorithmic fair subtyping: we write $t \leqslant_{\mathsf{A}} s$ if $\emptyset \vdash t \leqslant_{\mathsf{A}} s$ is inductively derivable by the axioms and rules in Table 7. Beside the fact that $\leqslant_{\mathsf{A}}$ is defined inductively, rather than coinductively, there are only two differences with respect to $\leqslant_{\mathsf{F}}$ presented in Table 6:

— Judgments have the form $\Gamma \vdash t \leqslant_{\mathsf{A}} s$ where Γ is a memoization context recording pairs of types that are assumed to be related by fair subtyping. The context is enriched in rules (A-Input) and (A-Output) with the pair of types being related and the new rule (A-Axiom) allows one to deduce $t \leqslant_{\mathsf{A}} s$ whenever this pair of types occurs in the memoization context.

— The premise $\mathtt{NonViable}(T - S)$ in rule (F-Output) has been replaced by its algorithmic variant $T - S \notin \mathtt{viables}(T - S)$, which corresponds to the non-viability of $T - S$ as by Theorem 4.8.

Algorithmic subtyping is complete with respect to $\leqslant_{\mathsf{F}}$ and, therefore, with respect to $\leqslant\!\!\!\leqslant$ for session types in normal form. Termination of the algorithm is trivially guaranteed by the memoization context: for every pair of types t and s, the memoization context of a derivation for $t \leqslant_{\mathsf{A}} s$ is bounded by the set $(\mathtt{trees}(t) \cup \mathtt{trees}(s)) \times (\mathtt{trees}(t) \cup \mathtt{trees}(s))$, which is finite (the details can be found in Appendix C).

Theorem 4.12. *Let $t, s \in \mathscr{T}_{\mathsf{nf}}$. Then $t \leqslant_{\mathsf{F}} s$ if and only if $t \leqslant_{\mathsf{A}} s$.*

One can now combine Theorems 4.8, 4.10, and 4.12 to define a complete algorithm for deciding $t \leqslant s$ in the general case:

Corollary 4.13. *$t \leqslant s$ if and only if $\mathtt{nf}(t) \leqslant_{\mathsf{A}} \mathtt{nf}(s)$.*

5. Related Work

Notions that are similar (if not equal) to session correctness and fair subtyping can be found in several different contexts. The works by Malik *et al.* [2004, 2006] define a notion of *conflict* for concurrent systems as a state of the system from which no terminal state can be reached anymore. Therefore, systems that are free from conflicts, which are called *nonblocking*, correspond to correct sessions and the induced preorder relation, called *conflict preorder*, has the same properties of fair subtyping. In the context of Web services and service replaceability, Bravetti and Zavattaro [2008, 2009] define notions of *correct composition* and of *subcontract* for Web services contracts that correspond to correct sessions and fair subtyping. Contracts are represented as terms of a CCS-like process algebra and describe the observable behavior of Web services. Mooij *et al.* [2010] also work in the context of Web services, although they describe the behavior of services by means of labeled transition systems and do not resort to a concrete language. Their *accordance* relation corresponds to our fair subtyping. Finally, Baldoni *et al.* [2009] define notions of *interoperability* and *conformance* for finite-state automata describing the behavior of multi-agent systems. Again, these two notions present many similarities with session correctness and fair subtyping in our setting.

Aside from the terminology, we can identify two major differences between the present work and the aforementioned ones. First of all, we work with a language of *higher-order* session types where it is possible to describe the type of the exchanged messages, while in all the mentioned approaches messages are abstracted as atomic names. As we have seen in Section 2, this aspect has a substantial impact on the theory, because it requires to address a circularity between the labeled transition system of sessions (which is defined in terms of subtyping) and fair subtyping (which is defined in terms of the labeled transition system). We are aware of just two ways to break this circularity: one possibility is to impose a stratification on session types so that the type of messages in a session type is "simpler" than the session type itself. A shortcoming of this approach, pursued for example in Castagna *et al.* [2009]; Barbanera and de'Liguoro [2010]; Padovani [2012], is that it prevents the usage of session types of the form $\mu x.\mathtt{p?a}\langle x\rangle.\mathtt{end}$ or $\mu y.\mathtt{p!a}\langle y\rangle.\mathtt{end}$, denoting the behavior of processes that communicate messages having the same type of the channels over which they are communicated. The second approach, which we have pursued here for the first time, is to define subtyping as the fixpoint of suitable (i.e., monotone) operator.

The second substantial difference between the present work and the ones cited above is the degree of characterization of the fair subtyping relation. Bravetti and Zavattaro [2009] only present a reduction of the subcontract relation to the should testing preorder [Natarajan and Cleaveland, 1995; Rensink and Vogler, 2007]; Mooij *et al.* [2010]

provide conditions under which the two relations coincide; Ware and Malik [2011] provide a characterization of the conflict preorder by comparing corresponding states on the behaviors being related. The characterization relies on a notion of "less conflicting states" and of language of "certain conflicts". Bugliesi *et al.* [2010] provide a coinductive characterization that is sound but not complete. The coinductive characterization given by Baldoni *et al.* [2009] is unsound in our setting, since it relies on a notion of interoperability that is coarser than session correctness. For example, the session

$$\mathtt{p} : \mu x.(\mathtt{q}?\mathtt{a}.\mathtt{q}!\mathtt{c}.x + \mathtt{q}?\mathtt{b}.\mathsf{end}) \mid \mathtt{q} : \mu y.\mathtt{p}!\mathtt{a}.(\mathtt{p}?\mathtt{c}.y + \mathtt{p}?\mathtt{d}.\mathsf{end})$$

is incorrect but interoperable according to Baldoni *et al.* [2009]. In summary, to the best of our knowledge we have provided the first complete coinductive and axiomatic characterizations of a liveness-preserving refinement relation. Remarkably, both these characterizations are variations of corresponding characterizations for "unfair" subtyping relations (see Gay and Hole [2005]), which makes them easy to understand.

The framework we have depicted is similar and closely related to *fair testing* [Natarajan and Cleaveland, 1995; Rensink and Vogler, 2007]. Testing is a general technique for defining refinement relations $\sqsubseteq$ between processes so that, when $P \sqsubseteq Q$ holds, the process Q can be safely used in place of process P because every test that P passes is passed also by Q. Fair testing adds a fairness assumption to standard testing: if a system goes infinitely often through a state from which some action is possible, a component of the system may rely upon the eventual observation of that action to terminate successfully. In the present paper, we instantiate fair testing to a context where processes are session types describing the behavior of participants of a multi-party session and the test is given by the correctness of a session. Unlike the standard fair testing framework, our notion of test is therefore symmetric, in the sense that *all* of the session participants must be able to terminate successfully, whereas in the traditional fair testing only the "tester" process has to. This difference has deep consequences on the properties of fair subtyping. In particular, the existence of equivalent, non-viable session types is the distinguishing feature between fair subtyping and the should testing preorder and, interestingly enough, both the coinductive and the axiomatic characterizations of fair subtyping heavily rely on non-viability.

Alternative characterizations of the should testing preorder have already been given in the literature. Natarajan and Cleaveland [1995] present a characterization based on sets of infinite strings, while Rensink and Vogler [2007] rely on a denotational model of processes. In both cases the characterizations are quite complex, if compared to ours, because they are semantically – rather than syntactically – based. In fact, as pointed out in Rensink and Vogler [2007], no complete axiomatization of the should testing preorder is known at the present time. We are currently investigating whether the insight gained with our theory can help solving this open problem and we conjecture that the restriction of should-testing to finite-state processes (where parallel composition is forbidden underneath recursions) shares the same trace-related properties (and possibly the behavioral characterization) of fair subtyping. A major obstacle to extending the characterization of fair subtyping to a more general process language, in particular one that includes parallel composition, is that a normal form for such processes with respect to liveness-preserving refinements is

not known. Traditionally, parallel composition is dealt with with suitable *expansion laws* that rephrase it in terms of sequential and choice operators. In our case, however, it is not clear whether such law can be defined.

In [Padovani, 2013] we report on some preliminary results regarding various extensions of the present work. First of all, in [Padovani, 2013] we have weakened the notion of session correctness so as to take into account non-terminating behaviors. The fair subtyping relation induced in this way is finer than $\leqslant$ and all session types turn out to be viable. On the one hand, this makes the technique described in the present paper, which is based on behavioral difference (Definition 3.15), no longer applicable. On the other hand, we have been able to generalize the "ruled by" relation appropriately. Second, in [Padovani, 2013] we show how to extend fair subtyping to open session types (see Section 6 for the reasons why this extension is important). The obtained relation turns out to be an original precongruence not investigated elsewhere and for which we are able to provide complete coinductive and axiomatic characterizations. The axiomatic characterization, in particular, is more involved than the one given in Section 4, because the syntactic structure of recursive terms impacts subtyping. In [Padovani, 2013] we have also chosen to work with atomic messages not distinguishing the identity of the participants of a session. When restricted to closed session types, and modulo the differences induced by non-terminating behaviors, the fair subtyping relation in [Padovani, 2013] coincides with $\leqslant$. This seems to suggest that fair subtyping defined in the present paper can be easily adapted to more general session type languages like for example that described in [Castagna *et al.*, 2012], where choices are *not* directed (different branches of the same internal/external choice may regard different participants).

6. Conclusion and Future Work

The subtyping relation for session types has been originally introduced by Gay and Hole [2005] and later explored in a semantic framework by Castagna *et al.* [2009], Padovani [2011b], and by Barbanera and de'Liguoro [2010]. The subtyping relations defined in these works do not guarantee liveness in multi-party sessions (see Section 1). In the present work we have proposed an alternative theory of (higher-order) session types with a stronger notion of session correctness in which the session preserves the possibility to reach a successfully terminated state for all of its participants. We have thoroughly studied the subtyping relation semantically induced by this notion of session correctness and provided coinductive and axiomatic characterizations for it. In both cases, the characterizations are variants of the corresponding ones for "unfair" subtyping relations.

Beside its liveness preserving property, it may be argued that preserving the possibility of termination is a general, desirable property of sessions in session-oriented computing. This consideration is supported by everyday experience, whereby sessions established with online services (such as electronic auctions and commerce, home banking and postal services, social networks) always envisage an implicit or explicit "log out" action. In this respect our theory makes sense (and is applicable) also in the dyadic case, when only two communicating parties are involved. However, the study of fair subtyping pursued in this work is just the first step to make the theory profitable in practice. We devote the

rest of this section to illustrating a few interesting problems that may deserve further investigations.

A first problem has to do with the fact that the standard coinductive techniques for type checking recursive processes are unsound when used in conjunction with fair subtyping. For example, consider the process

$$P = \mathtt{rec}\ X.x!\mathtt{a}\langle e\rangle.X$$

that repeatedly sends an $\mathtt{a}$-tagged message with argument e (a generic expression) on channel x, and consider the session type

$$T = \mu x.(\mathtt{p}!\mathtt{a}\langle t\rangle.x \oplus \mathtt{p}!\mathtt{b}\langle s\rangle.\mathsf{end})$$

associated with channel x. Clearly, P cannot fulfill its obligation to (eventually) send a $\mathtt{b}$-tagged message and yet P would be declared well typed according to a derivation like the following one:

$$\dfrac{\dfrac{\dfrac{\vdash e : t \qquad \dfrac{}{\{X \mapsto x : T\}; x : T \vdash X}\ \text{(T-Rec Var)}}{\{X \mapsto x : T\}; x : \mathtt{p}!\mathtt{a}\langle t\rangle.T \vdash x!\mathtt{a}\langle e\rangle.X}\ \text{(T-Output)} \qquad T \leqslant \mathtt{p}!\mathtt{a}\langle t\rangle.T}{\dfrac{\{X \mapsto x : T\}; x : T \vdash x!\mathtt{a}\langle e\rangle.X}{x : T \vdash \mathtt{rec}\ X.x!\mathtt{a}\langle e\rangle.X}\ \text{(T-Rec)}}\ \text{(T-Sub)}$$

Rule (T-Rec) records the environment $x : T$ used for type checking P, so that an occurrence of the process variable X within the same environment can be declared well typed. Rule (T-Sub) is a standard subsumption rule asserting that the process correctly uses channel x with type T even though the actual behavior of P on x is $\mathtt{p}!\mathtt{a}\langle t\rangle.T$. The reason why $\mathtt{p}!\mathtt{a}\langle t\rangle.T \leqslant T$ holds is because the two behaviors only differ for just one output action. Then, (T-Output) simply verifies that the process behaves correctly with respect to the channel it uses, and (T-Rec Var) closes the deduction. The problem of this derivation is that the subsumption rule is applied only once, but since it occurs within a recursion it is used to enforce the relation $T \leqslant \mu x.\mathtt{p}!\mathtt{a}\langle t\rangle.x$ which does not hold. An easy way to circumvent the problem, at the expense of a quite rigid type discipline, is to forbid (T-Sub) applications altogether or within the scope of recursions. Alternatively, one may attempt at extending the semantics of session types to open terms and identify the recursive contexts for which $\leqslant$ is sound. This extension, however, is far from being trivial. Indeed, in the two main bodies of work on fair testing, proper handling of open terms is either not addressed [Natarajan and Cleaveland, 1995] or shown to have considerable effects on the properties of the relation (by inducing trace equivalence) [Rensink and Vogler, 2007]. As discussed in Section 5, some preliminary results concerning fair subtyping for open session types can be found in [Padovani, 2013].

A second problem that concerns the static analysis in conjunction with the expressiveness of the session type language is related to the fairness assumption. In the present work, as well as in all the other works where liveness-preserving refinements are presented, the so-called "fairness assumption" holds globally: *every* internal choice that is taken infinitely often must infinitely often enable all of its branches. This assumption implies that the internal choice is taken by the process performing it independently of the environment. There are cases, however, where the outcome of an internal choice depends

on data that the process has received from the environment. As an example, consider the (correct) session

$$\mathtt{p} : \mu x.\mathtt{q}!\mathtt{Login}\langle \mathit{string}\rangle.(\mathtt{q}?\mathtt{OK}.\mathsf{end} + \mathtt{q}?\mathtt{NO}.x) \mid \mathtt{q} : \mu y.\mathtt{p}?\mathtt{Login}\langle \mathit{string}\rangle.(\mathtt{p}!\mathtt{OK}.\mathsf{end} \oplus \mathtt{p}!\mathtt{NO}.y)$$

describing a participant p trying to log into a service provided by participant q. The internal choice performed by q depends on the credentials sent by p: if the process implementing p insists on sending the same invalid credentials to the process implementing q, q will systematically answer with a NO-tagged message. Therefore, we have an inconsistency between the fair termination of the session as abstracted in terms of session types and the fact that one of its possible implementations fails to terminate. In this example it may be argued that the internal choice performed by q is actually an external choice in disguise: it is true that q performs a test to verify the credentials sent by p, but ultimately it is p that determines the outcome of the internal choice. At the same time, the choice cannot be advertised as external in the session type, if only because this would require the publication of the valid credentials that make q answer with an OK-tagged message! One way to address this issue could be to distinguish between "fair" and "unfair" internal choices within the same language of session types. In the example above, the internal choice performed by q would be "unfair" because utterly dependent on some decision (which credentials to send) that is external to q. As a consequence, the session would be declared incorrect because, among the admitted behaviors of p, is the one where p repeatedly sends the same credentials. We are not aware of any work considering both "fair" and "unfair" internal choices within the same process language, whose semantics appears to be an intriguing challenge.

On a more theoretical side, we wish to recall an interesting line of work developed around an interpretation of session types as formulas of linear logic [Caires and Pfenning, 2010]. Under this interpretation, the choice operators found in session types can be naturally explained as standard connectives of linear logic. In accordance with other interpretations of types as formulas, subtyping relations can be usually understood as logical entailment. An intriguing research problem, then, is to understand the logical characterization of the liveness preserving property of fair subtyping, in addition to the coinductive and axiomatic (but behavioral) characterizations we have studied here.

Acknowledgments. This work has been partially supported by two visiting professor positions at the Laboratoire Preuves, Programmes et Systèmes of the Université Paris Diderot. The author is grateful to the MSCS reviewers for their insightful comments which helped resolving inconsistencies and improving presentation of the article. The author wishes to thank also Ugo de' Liguoro, Daniele Varacca and Gianluigi Zavattaro with whom he had enlightening discussions on the topics covered in this article.

References

Luca Aceto and Matthew Hennessy. Termination, deadlock, and divergence. *Journal of the ACM*, 39:147–187, 1992.

Matteo Baldoni, Cristina Baroglio, Amit K. Chopra, Nirmit Desai, Viviana Patti, and

Munindar P. Singh. Choice, interoperability, and conformance in interaction protocols and service choreographies. In *Proceedings of AAMAS'09*, volume 2, pages 843–850. ACM, 2009.

Franco Barbanera and Ugo de'Liguoro. Two notions of sub-behaviour for session-based client/server systems. In *Proceedings of PPDP'10*, pages 155–164. ACM, 2010.

Mario Bravetti and Gianluigi Zavattaro. A foundational theory of contracts for multi-party service composition. *Fundamenta Informaticae*, 89(4):451–478, 2008.

Mario Bravetti and Gianluigi Zavattaro. A theory of contracts for strong service compliance. *Mathematical Structures in Computer Science*, 19(3):601–638, 2009.

Michele Bugliesi, Damiano Macedonio, Luca Pino, and Sabina Rossi. Compliance preorders for Web Services. In *Proceedings of WS-FM'09*, LNCS 6194, pages 76–91. Springer, 2010.

Luís Caires and Frank Pfenning. Session types as intuitionistic linear propositions. In *Proceedings of CONCUR'10*, LNCS 6269, pages 222–236. Springer, 2010.

Giuseppe Castagna, Rocco De Nicola, and Daniele Varacca. Semantic subtyping for the pi-calculus. *Theoretical Computer Science*, 398(1-3):217–242, 2008.

Giuseppe Castagna, Mariangiola Dezani-Ciancaglini, Elena Giachino, and Luca Padovani. Foundations of session types. In *Proceedings of PPDP'09*, pages 219–230. ACM, 2009.

Giuseppe Castagna, Mariangiola Dezani-Ciancaglini, and Luca Padovani. On Global Types and Multi-Party Sessions. *Logical Methods in Computer Science*, 8:1–45, 2012.

Bruno Courcelle. Fundamental properties of infinite trees. *Theoretical Computer Science*, 25:95–169, 1983.

Simon Gay and Malcolm Hole. Subtyping for session types in the π-calculus. *Acta Informatica*, 42(2-3):191–225, 2005.

Kohei Honda, Vasco T. Vasconcelos, and Makoto Kubo. Language primitives and type disciplines for structured communication-based programming. In *Proceedings of ESOP'98*, LNCS 1381, pages 122–138. Springer, 1998.

Kohei Honda, Nobuko Yoshida, and Marco Carbone. Multiparty asynchronous session types. In *Proceedings of POPL'08*, pages 273–284. ACM, 2008.

Kohei Honda. Types for dyadic interaction. In *Proceedings of CONCUR'93*, LNCS 715, pages 509–523. Springer, 1993.

Robi Malik, David Streader, and Steve Reeves. Fair testing revisited: A process-algebraic characterisation of conflicts. In *Proceedings of ATVA'04*, pages 120–134. Springer, 2004.

Robi Malik, David Streader, and Steve Reeves. Conflicts and fair testing. *International Journal of Foundations of Computer Science*, 17(4):797–814, 2006.

Arjan J. Mooij, Christian Stahl, and Marc Voorhoeve. Relating fair testing and accordance for service replaceability. *Journal of Logic and Algebraic Programming*, 79(3-5):233–244, 2010.

V. Natarajan and Rance Cleaveland. Divergence and fair testing. In *Proceedings of ICALP'95*, LNCS 944, pages 648–659. Springer, 1995.

Luca Padovani. Fair Subtyping for Multi-Party Session Types. In *Proceedings of COORDINATION'11*, volume LNCS 6721, pages 127–141. Springer, 2011.

Luca Padovani. Session Types = Intersection Types + Union Types. In *Proceedings of ITRS'10*, volume EPTCS 45, pages 71–89, 2011.

Luca Padovani. On projecting processes into session types. *Mathematical Structures in Computer Science*, to appear, 2012.

Luca Padovani. Fair subtyping for open session types. Technical Report 146/13, Dipartimento di Informatica, Università di Torino, 2013. Available at `http://www.di.unito.it/~padovani/Papers/OpenFairSubtyping.pdf`.

Benjamin Pierce and Davide Sangiorgi. Typing and subtyping for mobile processes. *Mathematical Structures in Computer Science*, 6(5):409–453, 1996.

Arend Rensink and Walter Vogler. Fair testing. *Information and Computation*, 205(2):125–198, 2007.

Simon Ware and Robi Malik. A state-based characterisation of the conflict preorder. In *Proceedings of FOCLASA'11*, volume EPTCS 58, pages 34–48, 2011.

Appendix A. Supplement to Section 2

For the sake of readability, in the propositions and proofs that follow we use $\leqslant_{\mathscr{S}}$ to denote the relation $\mathbf{F}(\mathscr{S})$.

We begin by proving a number of results showing the relation between $T \leqslant_{\mathscr{S}} S$ and the transitions of T and S. These results pave the way to the proofs of Lemmas A.5 and A.6 that form the core of the monotonicity proof for $\mathbf{F}$.

Proposition A.1. *$T \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} T'$ implies $T \leqslant_{\mathscr{S}} T'$.*

Proof. Let $M \mid \mathtt{p} : T$ be $\mathscr{S}$-correct. Then $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} M \mid \mathtt{p} : T'$, hence $M \mid \mathtt{p} : T'$ must be $\mathscr{S}$-correct as well. □

In what follows we will sparingly use some abbreviated notation for denoting repeated sequences of input/output actions. In particular:

— We will write $\{\mathtt{p}_1, \ldots, \mathtt{p}_n\}?\mathtt{a}.T$ in place of some $\mathtt{p}_1?\mathtt{a} \cdots \mathtt{p}_n?\mathtt{a}.T$.
— We will write $\{\mathtt{p}_1, \ldots, \mathtt{p}_n\}!\mathtt{a}.T$ in place of some $\mathtt{p}_1!\mathtt{a} \cdots \mathtt{p}_n!\mathtt{a}.T$.

We will make sure that the order in which the various input/output actions are performed is irrelevant. Also, we will write $\prod_{i=1..n} \mathtt{p}_i : T_i$ in place of the session $M = \mathtt{p}_1 : T_1 \mid \cdots \mid \mathtt{p}_n : T_n$ and we will write $M(\mathtt{p}_i)$ for T_i (that is, the session type associated with role $\mathtt{p}_i$ in M).

Proposition A.2. *Let $\mathscr{S} \subseteq \mathscr{R}$ and $T \leqslant_{\mathscr{S}} S$ and $T \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t\rangle}_{\mathscr{R}} T'$ and T' be viable. Then $S \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t\rangle}_{\mathscr{R}} S'$ and $T' \leqslant_{\mathscr{S}} S'$.*

Proof. From the hypothesis $T \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t\rangle}_{\mathscr{R}} T'$ we deduce $T \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle s\rangle}_{\mathscr{S}} T'$ for some s such that $(t, s) \in \mathscr{R}$. Let $M \mid \mathtt{p} : T'$ be a $\mathscr{S}$-correct session where $\mathtt{dom}(M) = \{\mathtt{q}_1, \ldots, \mathtt{q}_m\}$. We can assume, without loss of generality, that $\mathtt{q} \in \{\mathtt{q}_1, \ldots, \mathtt{q}_m\}$. Consider

$$N \stackrel{\text{def}}{=} \mathtt{q} : \mathtt{p}!\mathtt{a}\langle s\rangle.\{\mathtt{q}_1, \ldots, \mathtt{q}_m\} \setminus \{\mathtt{q}\}?\mathtt{ack}.M(\mathtt{q}) \mid \prod_{i=1..m, \mathtt{q} \neq \mathtt{q}_i} \mathtt{q}_i : \mathtt{q}!\mathtt{ack}.M(\mathtt{q}_i)$$

where `ack` is an arbitrary tag. We have that $N \mid \mathtt{p} : T$ is $\mathscr{S}$-correct by construction of N and $N \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} M \mid \mathtt{p} : T'$. From the hypothesis $T \leqslant_{\mathscr{S}} S$ we deduce that $N \mid \mathtt{q} : S$ is

$\mathscr{S}$-correct, therefore $S \xrightarrow{\mathtt{q?a}\langle s\rangle}_{\mathscr{S}} S'$ for some S' and $M \mid \mathtt{p} : S'$ is $\mathscr{S}$-correct. We deduce $T' \leqslant_{\mathscr{S}} S'$ because M is arbitrary. From $(t, s) \in \mathscr{R}$ we conclude $S \xrightarrow{\mathtt{q?a}\langle t\rangle}_{\mathscr{R}} S'$. □

Proposition A.3. *Let* $T = \bigoplus_{i\in I} \mathtt{r!a}_i\langle t_i\rangle.T_i$ *be viable and* $T \leqslant_{\mathscr{S}} S$. *Then* $S = \bigoplus_{i\in J} \mathtt{r!a}_i\langle s_i\rangle.S_i$ *and* $J \subseteq I$ *and* $(s_i, t_i) \in \mathscr{S}$ *and* $T_i \leqslant_{\mathscr{S}} S_i$ *for every* $i \in J$.

Proof. From the hypothesis T viable we deduce that every T_i is viable. Let $\{M_i\}_{i\in I}$ be a family of sessions such that $M_i \mid \mathtt{p} : T_i$ is $\mathscr{S}$-correct for every $i \in I$. Without loss of generality we may assume that $\mathtt{dom}(M_i) = \mathtt{dom}(M_j) = \{\mathtt{q}_1, \ldots, \mathtt{q}_m\}$ for every $i, j \in I$ (if this is not the case, appropriate $\mathtt{q}_i : \mathsf{end}$ participants can be added wherever necessary). For every $j \in \{1, \ldots, m\}$, let

$$N_j \stackrel{\text{def}}{=} \mathtt{q}_j : \begin{cases} \sum_{i\in I} \mathtt{p?a}_i\langle t_i\rangle.\{\mathtt{q}_1, \ldots, \mathtt{q}_m\} \setminus \{\mathtt{r}\}\mathtt{!a}_i.M_i(\mathtt{r}) & \text{if } \mathtt{r} = \mathtt{q}_j \\ \sum_{i\in I} \mathtt{r?a}_i.M_i(\mathtt{q}_j) & \text{otherwise} \end{cases}$$

and let $N \stackrel{\text{def}}{=} \prod_{i=1}^{m} N_i$. We have that $N \mid \mathtt{p} : T$ is $\mathscr{S}$-correct and from the hypothesis $T \leqslant_{\mathscr{S}} S$ we deduce that $N \mid \mathtt{q} : S$ is $\mathscr{S}$-correct. Therefore $S = \bigoplus_{i\in J} \mathtt{r!a}_i\langle s_i\rangle.S_i$ and $J \subseteq I$ and $(s_i, t_i) \in \mathscr{S}$ for every $i \in J$. Furthermore, $N \mid \mathtt{p} : S \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} M_i \mid \mathtt{p} : S_i$ for every $i \in J$, hence $T_i \leqslant_{\mathscr{S}} S_i$ for every $i \in J$ because each M_i is arbitrary. □

Proposition A.4. *Let* **(1)** T *be viable and* **(2)** $T \leqslant_{\mathscr{S}} S$. *Then:*

(i) $\mathscr{S} \subseteq \mathscr{R}$ *and* $S \xrightarrow{\mathtt{p:q?a}\langle s\rangle}_{\mathscr{R}}$ *imply* $T \xrightarrow{\mathtt{p:q?b}\langle t\rangle}_{\mathscr{R}}$ *for some* $\mathtt{b}$ *and* t.

(ii) $S \xrightarrow{\mathtt{p:q!a}\langle s\rangle}_{\mathscr{S}} S'$ *implies* $T \xRightarrow{\mathtt{p:q!a}\langle t\rangle}_{\mathscr{S}} T'$ *for some* t *and* T' *such that* $(s, t) \in \mathscr{S}$ *and* $T' \leqslant_{\mathscr{S}} S'$.

Proof. From **(1)** we deduce $T \neq \mathsf{bot}$ and $M \mid \mathtt{p} : T$ is $\mathscr{S}$-correct for some M and $\mathtt{p}$.

Regarding item (i), if $T = \mathsf{end}$, then no participant in M will ever send a message to $\mathtt{p}$, which contradicts **(2)**. If $T = \bigoplus_{i\in I} \mathtt{r!a}_i\langle t_i\rangle.T_i$, then from Proposition A.3 we deduce that S must perform an output, which contradicts the hypothesis that it performs an input.

Regarding item (ii), we have $S = \bigoplus_{i\in J} \mathtt{q!a}_i\langle s_i\rangle.S_i$ and $\mathtt{a} = \mathtt{a}_k$ and $s = s_k$ and $S' = S_k$ for some $k \in J$. If $T = \mathsf{end}$, then no participant in M will ever wait for a message from $\mathtt{p}$, which contradicts **(2)**. If $T = \sum_{i\in I} \mathtt{r?a}_i\langle t_i\rangle.T_i$, then T_k must be $\mathscr{S}$-viable for some $k \in I$ and from Proposition A.2 we deduce that S must perform an input, which contradicts the hypothesis that it performs an output. So it must be the case that $T = \bigoplus_{i\in I} \mathtt{r!a}_i\langle t_i\rangle.T_i$. By Proposition A.3 we deduce $J \subseteq I$ and $(s_i, t_i) \in \mathscr{S}$ and $T_i \leqslant_{\mathscr{S}} S_i$ for every $i \in J$. We conclude by taking $t = t_k$ and $T' = T_k$. □

In the following two lemmas, the crucial hypothesis that makes them non-trivial is $M \mid \mathtt{p} : T$ $\mathscr{R}$-correct where $T \leqslant_{\mathscr{S}} S$ and $\mathscr{R}$ is an arbitrary pre-subtyping relation including $\mathscr{S}$. In particular, $T \leqslant_{\mathscr{S}} S$ only guarantees that $M \mid \mathtt{p} : S$ is $\mathscr{S}$-correct whenever $M \mid \mathtt{p} : T$ is also $\mathscr{S}$-correct, but there can be sessions N such that $N \mid \mathtt{p} : T$ is $\mathscr{R}$-correct while $N \mid \mathtt{p} : T$ is not $\mathscr{S}$-correct.

Lemma A.5. *Let* **(1)** $\mathscr{S} \subseteq \mathscr{R}$ *and* **(2)** $T \leqslant_{\mathscr{S}} S$ *and* **(3)** $M \mid \mathtt{p} : T$ *be* $\mathscr{R}$*-correct and* **(4)** $M \mid \mathtt{p} : S \stackrel{\tau}{\Longrightarrow}_{\mathscr{R}} M' \mid \mathtt{p} : S'$. *Then* $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow}_{\mathscr{R}} M' \mid \mathtt{p} : T'$ *for some* $T' \leqslant_{\mathscr{S}} S'$.

Proof. By unzipping the derivation **(4)** we deduce that $M \overset{\overline{\varphi}}{\Longrightarrow}_{\mathscr{R}} M'$ and $S \overset{\varphi}{\Longrightarrow}_{\mathscr{R}} S'$ for some φ. We proceed by induction on φ to find ψ and $T' \leqslant_{\mathscr{S}} S'$ such that $M \overset{\overline{\psi}}{\Longrightarrow}_{\mathscr{R}} M'$ and $T \overset{\psi}{\Longrightarrow}_{\mathscr{R}} T'$:

— $(\varphi = \varepsilon)$ We conclude by taking $\psi = \varepsilon$ and $T' = T$ and by Proposition A.1.

— $(\varphi = \varphi'\mathtt{p} : \mathtt{q}?\mathtt{a}\langle t\rangle)$ Then $M \overset{\overline{\varphi'}}{\Longrightarrow}_{\mathscr{R}} M'' \xrightarrow{\mathtt{q}:\mathtt{p}!\mathtt{a}\langle t\rangle}_{\mathscr{R}} M'$ and $S \overset{\varphi'}{\Longrightarrow}_{\mathscr{R}} S'' \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t\rangle}_{\mathscr{R}} S''' \overset{\tau}{\Longrightarrow}_{\mathscr{R}} S'$. By induction hypothesis we deduce that $M \overset{\overline{\psi'}}{\Longrightarrow}_{\mathscr{R}} M''$ and $T \overset{\psi'}{\Longrightarrow}_{\mathscr{R}} T''$ for some ψ' and $T'' \leqslant_{\mathscr{S}} S''$. From Proposition A.4$(i)$ and hypothesis **(3)** we deduce that $T'' \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t\rangle}_{\mathscr{R}} T'$ for some T' that is viable, because $M' \mid \mathtt{p} : T'$ is $\mathscr{R}$-correct. By Proposition A.2 we deduce $T' \leqslant_{\mathscr{S}} S'''$, and we conclude by taking $\psi = \psi'\mathtt{p} : \mathtt{q}?\mathtt{a}\langle t\rangle$ and because $T' \leqslant_{\mathscr{S}} S'$ by Proposition A.1.

— $(\varphi = \varphi'\mathtt{p} : \mathtt{q}!\mathtt{a}\langle t\rangle)$ Then $M \overset{\overline{\varphi'}}{\Longrightarrow}_{\mathscr{R}} M'' \xrightarrow{\mathtt{q}:\mathtt{p}?\mathtt{a}\langle t\rangle}_{\mathscr{R}} M'$ and $S \overset{\varphi'}{\Longrightarrow}_{\mathscr{R}} S'' \xrightarrow{\mathtt{p}:\mathtt{q}!\mathtt{a}\langle t\rangle}_{\mathscr{R}} S''' \overset{\tau}{\Longrightarrow}_{\mathscr{R}} S'$. By induction hypothesis we deduce that $M \overset{\overline{\psi'}}{\Longrightarrow}_{\mathscr{R}} M''$ and $T \overset{\psi'}{\Longrightarrow}_{\mathscr{R}} T''$ for some ψ' and $T'' \leqslant_{\mathscr{S}} S''$. Observe that T'' is viable because $M'' \mid \mathtt{p} : T''$ is $\mathscr{R}$-correct. By Proposition A.4(ii) we deduce $T'' \xrightarrow{\mathtt{p}:\mathtt{q}!\mathtt{a}\langle s\rangle}_{\mathscr{R}} T'$ for some s and T' such that $(t, s) \in \mathscr{S}$ and $T' \leqslant_{\mathscr{S}} S'''$. From **(1)** we deduce $(t, s) \in \mathscr{R}$ hence $M'' \xrightarrow{\mathtt{q}:\mathtt{p}?\mathtt{a}\langle s\rangle}_{\mathscr{R}} M'$. We conclude by taking $\psi = \psi'\mathtt{p} : \mathtt{q}!\mathtt{a}\langle s\rangle$ and because $T' \leqslant_{\mathscr{S}} S'$ by Proposition A.1. □

Lemma A.6. *Let* **(1)** $\mathscr{S} \subseteq \mathscr{R}$ *and* **(2)** $T \leqslant_{\mathscr{S}} S$ *and* **(3)** $M \mid \mathtt{p} : T$ *be* $\mathscr{R}$*-correct. Then* $M \mid \mathtt{p} : S \overset{\checkmark}{\Longrightarrow}_{\mathscr{R}}$.

Proof. From **(3)** we deduce that $[M] \mid \mathtt{p} : T$ is $\mathscr{S}$-correct. Then from **(2)** we deduce that $[M] \mid \mathtt{p} : S$ is also $\mathscr{S}$-correct and, in particular, $[M] \mid \mathtt{p} : S \overset{\checkmark}{\Longrightarrow}_{\mathscr{S}}$. By unzipping this derivation we deduce that $[M] \overset{\overline{\varphi}\checkmark}{\Longrightarrow}_{\mathscr{S}}$ and $S \overset{\varphi\checkmark}{\Longrightarrow}_{\mathscr{S}}$ for some φ. We proceed by induction on φ to show that $M \overset{\overline{\psi}\checkmark}{\Longrightarrow}_{\mathscr{R}}$ and $S \overset{\psi\checkmark}{\Longrightarrow}_{\mathscr{R}}$ for some ψ:

— $(\varphi = \varepsilon)$ Then we conclude by taking $\psi = \varepsilon$.

— $(\varphi = \mathtt{p} : \mathtt{q}?\mathtt{a}\langle \mathsf{bot}\rangle\varphi')$ Then $[M] \xrightarrow{\mathtt{q}:\mathtt{p}!\mathtt{a}\langle \mathsf{bot}\rangle}_{\mathscr{S}} [M'] \overset{\overline{\varphi'}\checkmark}{\Longrightarrow}_{\mathscr{S}}$ and $S \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle \mathsf{bot}\rangle}_{\mathscr{S}} S' \overset{\varphi'\checkmark}{\Longrightarrow}_{\mathscr{S}}$, meaning that $M \xrightarrow{\mathtt{q}:\mathtt{p}!\mathtt{a}\langle t\rangle}_{\mathscr{R}} M'$ for some t. From **(3)** we deduce $T \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t\rangle}_{\mathscr{R}} T'$ for some T' such that $M' \mid \mathtt{p} : T'$ is $\mathscr{R}$-correct. From **(2)** and Proposition A.2 we deduce $S \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t\rangle}_{\mathscr{R}} S'$ and $T' \leqslant_{\mathscr{S}} S'$. By induction hypothesis we have $M' \overset{\overline{\psi'}\checkmark}{\Longrightarrow}_{\mathscr{R}}$ and $S' \overset{\psi'\checkmark}{\Longrightarrow}_{\mathscr{R}}$ for some ψ'. We conclude by taking $\psi = \mathtt{p} : \mathtt{q}?\mathtt{a}\langle t\rangle\psi'$.

— $(\varphi = \mathtt{p} : \mathtt{q}!\mathtt{a}\langle s\rangle\varphi')$ We have $[M] \xrightarrow{\mathtt{q}:\mathtt{p}?\mathtt{a}\langle s\rangle}_{\mathscr{S}} [M'] \overset{\overline{\varphi'}\checkmark}{\Longrightarrow}_{\mathscr{S}}$ and $S \overset{\tau}{\Longrightarrow}_{\mathscr{S}} S' \xrightarrow{\mathtt{p}:\mathtt{q}!\mathtt{a}\langle s\rangle}_{\mathscr{S}} S'' \overset{\varphi'\checkmark}{\Longrightarrow}_{\mathscr{S}}$. From **(2)** and by Proposition A.1 we deduce $T \leqslant_{\mathscr{S}} S \leqslant_{\mathscr{S}} S'$. From Proposition A.3 we deduce $T \xrightarrow{\mathtt{p}:\mathtt{q}!\mathtt{a}\langle t\rangle}_{\mathscr{S}} T'$ for some t and T' such that $(s, t) \in \mathscr{S}$ and $T' \leqslant_{\mathscr{S}} S''$. From **(3)** we deduce $M \xrightarrow{\mathtt{q}:\mathtt{p}?\mathtt{a}\langle t\rangle}_{\mathscr{R}} M'$ where $M' \mid \mathtt{p} : T'$ is $\mathscr{R}$-correct. From **(1)** we also have $M \xrightarrow{\mathtt{q}:\mathtt{p}?\mathtt{a}\langle s\rangle}_{\mathscr{R}} M'$. By induction hypothesis we deduce that $M' \overset{\overline{\psi'}\checkmark}{\Longrightarrow}_{\mathscr{R}}$ and $S'' \overset{\psi'\checkmark}{\Longrightarrow}_{\mathscr{R}}$ for some ψ'. We conclude by taking $\psi = \mathtt{p} : \mathtt{q}!\mathtt{a}\langle s\rangle\psi'$. □

Theorem 2.4. **F** *is monotone.*

Proof. Suppose $\mathscr{S} \subseteq \mathscr{R}$ and $T \leqslant_{\mathscr{S}} S$ and let $M \,|\, \mathtt{p} : T$ be $\mathscr{R}$-correct. Consider a derivation $M \mid \mathtt{p} : S \stackrel{\tau}{\Longrightarrow}_{\mathscr{R}} M' \mid \mathtt{p} : S'$. By Lemma A.5 we deduce that $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow}_{\mathscr{R}} M' \mid \mathtt{p} : T'$ for some $T' \leqslant_{\mathscr{S}} S'$. Also, $M' \mid \mathtt{p} : T'$ is $\mathscr{R}$-correct because correctness is preserved by τ moves. By Lemma A.6 we conclude $M' \mid \mathtt{p} : S' \stackrel{\checkmark}{\Longrightarrow}_{\mathscr{R}}$. □

Appendix B. Supplement to Section 3

Theorem 3.7. *For every $t \in \mathscr{T}_{\mathsf{nf}} \setminus \{\mathsf{bot}, \mathsf{top}\}$ we have $t \in \mathscr{T}_{\mathsf{v}}$.*

Proof. Let $T \in \mathscr{T}_{\mathsf{nf}} \setminus \{\mathsf{bot}\}$. We must define a system M such that $M \,|\, \mathtt{p} : T$ is correct under the hypothesis that T is in normal form. Let $\{\mathtt{p}_1, \ldots, \mathtt{p}_n\}$ be the set of roles occurring in T and let $\mathtt{p}$ be different from them. The basic idea is to define a session where every participant other than $\mathtt{p}$ is always informed about the internal decisions taken by any other participant. Therefore, the participants that communicate with $\mathtt{p}$ propagate to all the other participants any message received from or sent to $\mathtt{p}$. The behavior associated with each $\mathtt{p}_k$ is defined by projecting T according to the following equations:

$$\mathsf{end} \downarrow \mathtt{p}_k = \mathsf{end}$$

$$\textstyle\sum_{i \in I} \mathtt{q}?\mathtt{a}_i\langle t_i \rangle.T_i \downarrow \mathtt{p}_k = \begin{cases} \bigoplus_{i \in I} \mathtt{p}!\mathtt{a}_i\langle \mathsf{bot} \rangle.\{\mathtt{p}_1, \ldots, \mathtt{p}_n\} \setminus \{\mathtt{p}_k\}!\mathtt{a}_i\langle \mathsf{bot} \rangle.(T_i \downarrow \mathtt{p}_k) & \mathtt{q} = \mathtt{p}_k \\ \sum_{i \in I} \mathtt{q}?\mathtt{a}_i\langle \mathsf{top} \rangle.(T_i \downarrow \mathtt{p}_k) & \mathtt{q} \neq \mathtt{p}_k \end{cases}$$

$$\textstyle\bigoplus_{i \in I} \mathtt{q}!\mathtt{a}_i\langle t_i \rangle.T_i \downarrow \mathtt{p}_k = \begin{cases} \sum_{i \in I} \mathtt{p}?\mathtt{a}_i\langle \mathsf{top} \rangle.\{\mathtt{p}_1, \ldots, \mathtt{p}_n\} \setminus \{\mathtt{p}_k\}!\mathtt{a}_i\langle \mathsf{bot} \rangle.(T_i \downarrow \mathtt{p}_k) & \mathtt{q} = \mathtt{p}_k \\ \sum_{i \in I} \mathtt{q}?\mathtt{a}_i\langle \mathsf{top} \rangle.(T_i \downarrow \mathtt{p}_k) & \mathtt{q} \neq \mathtt{p}_k \end{cases}$$

Now consider $M \stackrel{\text{def}}{=} \prod_{i=1}^{n} \mathtt{p}_i : T \downarrow \mathtt{p}_i$. It is a simple exercise to verify that $M \mid \mathtt{p} : T$ is correct and this concludes the proof. □

Theorem 3.8. *Let $t, s \in \mathscr{T}_{\mathsf{nf}}$. Then $t \leqslant s$ implies $t \leqslant_{\mathsf{U}} s$.*

Proof. We show that $\leqslant$ satisfies the conditions of Definition 3.1. Suppose $t \leqslant s$. We distinguish four possibilities according to the shape of t and s:

— ($t = \mathsf{bot}$ or $s = \mathsf{top}$) Then either item (1) or item (2) of Definition 3.1 is satisfied.
— ($t = \mathsf{end}$) We have $\mathtt{p} : \mathsf{end} \mid \mathtt{q} : t$ correct, hence $\mathtt{p} : \mathsf{end} \mid \mathtt{q} : s$ is also correct. We conclude $s = \mathsf{end}$, therefore item (3) of Definition 3.1 is satisfied.
— ($t = \sum_{i \in I} \mathtt{p}?\mathtt{a}_i\langle t_i \rangle.T_i$) Then every T_i is in normal form for $i \in I$. By Theorem 3.7 we deduce that every T_i is viable for $i \in I$. From Proposition A.2 we deduce that $s = \sum_{i \in J} \mathtt{p}?\mathtt{a}_i\langle s_i \rangle.S_i$ with $I \subseteq J$ and $t_i \leqslant s_i$ and $T_i \leqslant S_i$ for all $i \in I$. We conclude that item (4) of Definition 3.1 is satisfied.
— ($t = \bigoplus_{i \in I} \mathtt{p}!\mathtt{a}_i\langle t_i \rangle.T_i$) Then every T_i is in normal form for $i \in I$. By Theorem 3.7 we deduce that every T_i is viable for $i \in I$. From Proposition A.3 we deduce that $s = \bigoplus_{i \in J} \mathtt{p}!\mathtt{a}_i\langle s_i \rangle.S_i$ with $J \subseteq I$ and $s_i \leqslant t_i$ and $T_i \leqslant S_i$ for all $i \in J$. We conclude that item (3) of Definition 3.1 is satisfied. □

Lemma B.1 (simulation)**.** *Let $\mathscr{S}$ be a coinductive unfair subtyping relation such that $(T, S) \in \mathscr{S}$ and $M \mid \mathtt{q} : T$ be $\mathscr{S}$-correct and $M \overset{\overline{\varphi}}{\Longrightarrow}_{\mathscr{S}} M'$ and $S \overset{\varphi}{\Longrightarrow}_{\mathscr{S}} S'$. Then there exist ψ and T' such that $M \overset{\overline{\psi}}{\Longrightarrow}_{\mathscr{S}} M'$ and $T \overset{\psi}{\Longrightarrow}_{\mathscr{S}} T'$ and $(T', S') \in \mathscr{S}$.*

Proof. We proceed by induction on φ:

— In the base case we have $\varphi = \varepsilon$ and we conclude by taking $\psi = \varepsilon$ and $T' = T$ (note that S' may be a residual of S after an application of rule (T-Choice) but even in this case $(T', S') \in \mathscr{S}$ still holds by definition of coinductive unfair subtyping relation).

— Suppose $\varphi = \mathtt{q} : \mathtt{p}?\mathtt{a}\langle t\rangle\varphi'$. Then $M \overset{\mathtt{p}:\mathtt{q}!\mathtt{a}\langle t\rangle}{\Longrightarrow}_{\mathscr{S}} M'' \overset{\overline{\varphi}'}{\Longrightarrow}_{\mathscr{S}} M'$ and $S \xrightarrow{\mathtt{q}:\mathtt{p}?\mathtt{a}\langle t\rangle}_{\mathscr{S}} S'' \overset{\varphi'}{\Longrightarrow}_{\mathscr{S}} S'$ for some M'' and S''. We deduce $S = \sum_{i\in J} \mathtt{p}?\mathtt{a}_i\langle s_i\rangle.S_i$ and $\mathtt{a} = \mathtt{a}_k$ and $(t, s_k) \in \mathscr{S}$ and $S'' = S_k$ for some $k \in J$. From item (2) of Definition 3.1 we deduce $T = \sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i$ with $I \subseteq J$ and $(t_i, s_i) \in \mathscr{S}$ and $(T_i, S_i) \in \mathscr{S}$ for every $i \in I$. It must be the case that $k \in I$ and $(t, t_k) \in \mathscr{S}$, for otherwise $M \mid \mathtt{q} : T$ would not be $\mathscr{S}$-correct (only $\mathtt{q}$ can consume the $\mathtt{q}!\mathtt{a}\langle t\rangle$ message coming from $\mathtt{p}$). By induction hypothesis we deduce that there exist ψ' and T' such that $M'' \overset{\overline{\psi}'}{\Longrightarrow}_{\mathscr{S}} M'$ and $T_k \overset{\psi'}{\Longrightarrow}_{\mathscr{S}} T'$ and $(T', S') \in \mathscr{S}$. We conclude by taking $\psi = \mathtt{q} : \mathtt{p}?\mathtt{a}\langle t\rangle\psi'$.

— Suppose $\varphi = \mathtt{q} : \mathtt{p}!\mathtt{a}\langle s\rangle\varphi'$. Then $M \overset{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle s\rangle}{\Longrightarrow}_{\mathscr{S}} M'' \overset{\overline{\varphi}'}{\Longrightarrow}_{\mathscr{S}} M'$ and $S \overset{\tau}{\Longrightarrow}_{\mathscr{S}} \xrightarrow{\mathtt{q}:\mathtt{p}!\mathtt{a}\langle s\rangle}_{\mathscr{S}} S'' \overset{\varphi'}{\Longrightarrow}_{\mathscr{S}} S'$ for some M'' and S''. We deduce $S = \bigoplus_{i\in J} \mathtt{p}!\mathtt{a}_i\langle s_i\rangle.S_i$ and $\mathtt{a} = \mathtt{a}_k$ and $s = s_k$ and $S'' = S_k$ for some $k \in J$. From item (3) of Definition 3.1 we deduce $T = \bigoplus_{i\in I} \mathtt{p}!\mathtt{a}_i\langle t_i\rangle.T_i$ and $J \subseteq I$ and $(s_i, t_i) \in \mathscr{S}$ and $(T_i, S_i) \in \mathscr{S}$ for every $i \in J$. It must be the case that $M \overset{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t_k\rangle}{\Longrightarrow}_{\mathscr{S}} M''$ for otherwise $M \mid \mathtt{p} : T$ would not be $\mathscr{S}$-correct. By induction hypothesis we deduce that there exist ψ' and T' such that $M'' \overset{\overline{\psi}'}{\Longrightarrow}_{\mathscr{S}} M'$ and $T_k \overset{\psi'}{\Longrightarrow}_{\mathscr{S}} T'$ and $(T', S') \in \mathscr{S}$. We conclude by taking $\psi = \mathtt{q} : \mathtt{p}!\mathtt{a}\langle t_k\rangle\psi'$. □

Proposition B.2. *Let $T \leqslant_{\mathsf{U}} S$ and $M \mid \mathtt{q} : T$ be correct and $M \overset{\overline{\varphi}}{\Longrightarrow} M' \overset{\tau}{\not\longrightarrow}$ and $S \overset{\varphi}{\Longrightarrow} S' \overset{\tau}{\not\longrightarrow}$. Then there exist ψ and $T' \leqslant_{\mathsf{U}} S'$ such that $M \overset{\overline{\psi}}{\Longrightarrow} M'$ and $T \overset{\psi}{\Longrightarrow} T'$.*

Proof. The proof is structurally the same as that of Lemma B.1. The one critical aspect of this results resides in the fact that $\leqslant \not\subseteq \leqslant_{\mathsf{U}}$, so it is not obvious that the hypothesis $T \leqslant_{\mathsf{U}} S$ is enough for granting the same synchronizations between M and S to occur also between M and T. Unlike Lemma B.1, however, the two extra hypotheses $M' \overset{\tau}{\not\longrightarrow}$ and $S' \overset{\tau}{\not\longrightarrow}$ allow us to deduce that the interaction between M and S has stopped without having any pending output actions because, by rule (T-Choice), a such an action always performs τ moves. We sketch the case $\varphi = \mathtt{q} : \mathtt{p}?\mathtt{a}\langle t\rangle\varphi'$ in more detail. In this case $M \overset{\mathtt{p}:\mathtt{q}!\mathtt{a}\langle t\rangle}{\Longrightarrow} M'' \overset{\overline{\varphi}'}{\Longrightarrow} M'$ and $S \xrightarrow{\mathtt{q}:\mathtt{p}?\mathtt{a}\langle t\rangle} S'' \overset{\varphi'}{\Longrightarrow} S'$ for some M'' and S''. We deduce $S = \sum_{i\in J} \mathtt{p}?\mathtt{a}_i\langle s_i\rangle.S_i$ and $\mathtt{a} = \mathtt{a}_k$ and $t \leqslant_{\mathsf{U}} s_k$ and $S'' = S_k$ for some $k \in J$. From item (2) of Definition 3.1 we deduce $T = \sum_{i\in I} \mathtt{p}?\mathtt{a}_i\langle t_i\rangle.T_i$ with $I \subseteq J$ and $t_i \leqslant_{\mathsf{U}} s_i$ and $T_i \leqslant_{\mathsf{U}} S_i$ for every $i \in I$. It must be the case that $k \in I$ and $t \leqslant_{\mathsf{U}} t_k$, for otherwise $M \mid \mathtt{q} : T$ would not be correct. It must also be the case that $t \leqslant t_k$, for otherwise the

$\mathtt{q}!\mathtt{a}\langle t\rangle$ message coming from M would remain pending and $\mathtt{q} : T$ is the only participant that is able to receive it. The proof continues as the one of Lemma B.1. □

Proposition 3.12. *Let $T, S \in \mathscr{T}_{\mathsf{nf}}$ and $T \leqslant_{\mathsf{U}} S$. The following properties hold:*

(1) *If T and S are finite, then $T \leqslant S$;*

(2) *If $M \mid \mathtt{p} : T$ is correct and $M \mid \mathtt{p} : S \stackrel{\tau}{\Longrightarrow} N \stackrel{\tau}{\not\longrightarrow}$, then $N \stackrel{\checkmark}{\longrightarrow}$.*

Proof. Regarding item (1), it suffices to show that

$$\mathscr{S} \stackrel{\text{def}}{=} \{(t, s) \mid \ t, s \text{ finite} \wedge t \leqslant_{\mathsf{U}} s \ \}$$

is $\mathbf{F}$-consistent, namely that $\mathscr{S} \subseteq \mathbf{F}(\mathscr{S})$. Suppose $(T, S) \in \mathscr{S}$ and $M \mid \mathtt{p} : T$ is $\mathscr{S}$-correct. By Lemma B.1 it is enough to show that $M \mid \mathtt{p} : S \stackrel{\checkmark}{\Longrightarrow}_{\mathscr{S}}$. We do so by induction on T and by cases on its shape.

— ($T = \mathsf{end}$) Then $M \stackrel{\checkmark}{\Longrightarrow}_{\mathscr{S}}$. From item (1) of Definition 3.1 we deduce $S = \mathsf{end}$. We conclude $M \mid \mathtt{p} : S \stackrel{\checkmark}{\Longrightarrow}_{\mathscr{S}}$.

— ($T = \sum_{i\in I} \mathtt{q}?\mathtt{a}_i\langle t_i\rangle.T_i$) From the hypothesis that $M \mid \mathtt{p} : T$ is $\mathscr{S}$-correct we deduce $M \stackrel{\mathtt{q}:\mathtt{p}!\mathtt{a}_k\langle t\rangle}{\Longrightarrow}_{\mathscr{S}} M'$ for some $k \in I$ and t where $(t, t_k) \in \mathscr{S}$ and $M' \mid \mathtt{p} : T_k$ is $\mathscr{S}$-correct. From item (2) of Definition 3.1 we deduce $S = \sum_{i\in I\cup J} \mathtt{q}?\mathtt{a}_i\langle s_i\rangle.S_i$ and $(t_i, s_i) \in \mathscr{S}$ and $(T_i, S_i) \in \mathscr{S}$ for every $i \in I$. By induction hypothesis we obtain $M' \mid \mathtt{p} : S_k \stackrel{\checkmark}{\Longrightarrow}_{\mathscr{S}}$. We conclude by observing that $M \mid \mathtt{p} : S \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} M' \mid \mathtt{p} : S_k$.

— ($T = \bigoplus_{i\in I} \mathtt{q}!\mathtt{a}_i\langle t_i\rangle.T_i$) From the hypothesis that $M \mid \mathtt{p} : T$ is $\mathscr{S}$-correct we deduce $M \stackrel{\mathtt{q}:\mathtt{p}?\mathtt{a}_i\langle t_i\rangle}{\Longrightarrow}_{\mathscr{S}}$ for every $i \in I$ and $M \stackrel{\mathtt{q}:\mathtt{p}?\mathtt{a}_i\langle t_i\rangle}{\Longrightarrow}_{\mathscr{S}} M'$ implies that $M' \mid \mathtt{p} : T_i$ is $\mathscr{S}$-correct for every $i \in I$. From item (3) of Definition 3.1 we deduce $S = \bigoplus_{i\in J} \mathtt{q}!\mathtt{a}_i\langle s_i\rangle.S_i$ for some $J \subseteq I$ and $(s_i, t_i) \in \mathscr{S}$ and $(T_i, S_i) \in \mathscr{S}$ for every $i \in J$. Let $M \mid \mathtt{p} : S \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} M' \mid \mathtt{p} : S_k$ for some M' and $k \in J$. By induction hypothesis we conclude $M' \mid \mathtt{p} : S_k \stackrel{\checkmark}{\Longrightarrow}_{\mathscr{S}}$.

Regarding item (2), we have $N = M' \mid \mathtt{p} : S'$ for some M' and S'. We deduce that there exists a string φ of actions such that $M \stackrel{\overline{\varphi}}{\Longrightarrow} M' \stackrel{\tau}{\not\longrightarrow}$ and $S \stackrel{\varphi}{\Longrightarrow} S' \stackrel{\tau}{\not\longrightarrow}$. By Proposition B.2 we deduce that $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow} M' \mid \mathtt{p} : T'$ for some $T' \leqslant_{\mathsf{U}} S'$. Now we argue that $T' = \mathsf{end}$, which implies $S' = \mathsf{end}$:

— T' cannot be an output, because from $S' \leqslant_{\mathsf{U}} T'$ we know that S' would be an output as well and output actions always perform τ moves because of (T-Choice). This contradicts the hypothesis $N \stackrel{\tau}{\not\longrightarrow}$.

— T' cannot be an input, because from $M \mid \mathtt{p} : T$ correct we deduce $M' \mid \mathtt{p} : T'$ correct hence M' would contain an output operation which contradicts $M' \stackrel{\tau}{\not\longrightarrow}$.

— T' cannot be bot, because $T' \in \mathtt{ctrees}(T)$ and T is viable and in normal form.

From the hypothesis $M \mid \mathtt{p} : T$ correct we have $M' \stackrel{\checkmark}{\longrightarrow}$, therefore we conclude $N \stackrel{\checkmark}{\longrightarrow}$. □

We report here the definition of $\leqslant_{\mathsf{C}}$ which we included in the statement of Theorem 3.14 and which is the subject of the next two results.

Definition B.3. We denote by $\leqslant_{\mathsf{C}}$ the largest coinductive unfair subtyping such that $T \leqslant_{\mathsf{C}} S$ implies $T \prec S$.

Lemma B.4. *Let* **(1)** $T \leqslant_{\mathsf{C}} S$ *and* **(2)** $\mathscr{S} \supseteq \leqslant_{\mathsf{C}}$ *and* **(3)** $M \mid \mathtt{p} : T$ *be* $\mathscr{S}$*-correct. Then* $M \mid \mathtt{p} : S \overset{\checkmark}{\Longrightarrow}_{\mathscr{S}}$.

Proof. From the hypothesis **(3)** we deduce that $[M] \mid \mathtt{p} : T$ is correct. From the hypothesis **(1)** we deduce $T \prec S$, hence $[M] \mid \mathtt{p} : S \overset{\checkmark}{\Longrightarrow}$, namely $M \overset{\overline{\varphi}\checkmark}{\Longrightarrow}$ and $S \overset{\varphi\checkmark}{\Longrightarrow}$ for some string φ of actions. We reason by induction on φ to find some ψ such that $M \overset{\overline{\psi}\checkmark}{\Longrightarrow}_{\mathscr{S}}$ and $S \overset{\psi\checkmark}{\Longrightarrow}_{\mathscr{S}}$:

— $(\varphi = \varepsilon)$ Then $S = \mathsf{end}$ and from the hypothesis **(1)** we deduce $T = \mathsf{end}$. We conclude by taking $\psi = \varepsilon$.

— $(\varphi = \mathtt{p} : \mathtt{q}?\mathtt{a}\langle\mathsf{bot}\rangle\varphi')$ Then $M \overset{\mathtt{q}:\mathtt{p}!\mathtt{a}\langle t\rangle}{\Longrightarrow}_{\mathscr{S}} N$ and $[N] \overset{\overline{\varphi}'\checkmark}{\Longrightarrow}$ and $S = \sum_{i\in J} \mathtt{q}?\mathtt{a}_i\langle s_i\rangle.S_i \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle\mathsf{bot}\rangle}$ $S_k \overset{\varphi'\checkmark}{\Longrightarrow}$ where $\mathtt{a} = \mathtt{a}_k$ for some $k \in J$. From the hypothesis **(1)** we deduce $T = \sum_{i\in I} \mathtt{q}?\mathtt{a}_i\langle t_i\rangle.T_i$ where $I \subseteq J$ and $t_i \leqslant_{\mathsf{C}} s_i$ and $T_i \leqslant_{\mathsf{C}} S_i$ for every $i \in I$. From the hypothesis **(3)** we deduce $k \in I$ and $(t, t_k) \in \mathscr{S}$ and $N \mid \mathtt{p} : T_k$ is $\mathscr{S}$-correct. By induction hypothesis we deduce that $N \overset{\overline{\psi}'\checkmark}{\Longrightarrow}_{\mathscr{S}}$ and $S_k \overset{\psi'\checkmark}{\Longrightarrow}_{\mathscr{S}}$ for some ψ'. Since $(t, s_k) \in \mathscr{S}$ we have $S \xrightarrow{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t\rangle}_{\mathscr{S}} S_k$ and we conclude by taking $\psi = \mathtt{p} : \mathtt{q}?\mathtt{a}\langle t\rangle\psi'$.

— $(\varphi = \mathtt{p} : \mathtt{q}!\mathtt{a}\langle s\rangle\varphi')$ Then $[M] \overset{\mathtt{q}:\mathtt{p}?\mathtt{a}\langle s\rangle}{\Longrightarrow} N \overset{\overline{\varphi}'\checkmark}{\Longrightarrow}$ and $S = \bigoplus_{i\in J} \mathtt{q}!\mathtt{a}_i\langle s_i\rangle.S_i \overset{\mathtt{p}:\mathtt{q}!\mathtt{a}\langle s\rangle}{\Longrightarrow} S_k \overset{\varphi'\checkmark}{\Longrightarrow}$ and $\mathtt{a} = \mathtt{a}_k$ and $s = s_k$ for some $k \in J$. From the hypothesis **(1)** we deduce $T = \bigoplus_{i\in I} \mathtt{q}!\mathtt{a}_i\langle t_i\rangle.T_i$ where $J \subseteq I$ and $s_i \leqslant_{\mathsf{C}} t_i$ and $T_i \leqslant_{\mathsf{C}} S_i$ for every $i \in J$. From the hypothesis **(3)** we deduce $M \overset{\mathtt{q}:\mathtt{p}?\mathtt{a}\langle t_k\rangle}{\Longrightarrow}_{\mathscr{S}} M'$ and $M' \mid \mathtt{p} : T_k$ is $\mathscr{S}$-correct and $[M'] = N$. By induction hypothesis we deduce that $M' \overset{\overline{\psi}'\checkmark}{\Longrightarrow}_{\mathscr{S}}$ and $S_k \overset{\psi'\checkmark}{\Longrightarrow}_{\mathscr{S}}$ for some ψ'. Since $(s_k, t_k) \in \mathscr{S}$ we conclude by taking $\psi = \mathtt{p} : \mathtt{q}!\mathtt{a}\langle s_k\rangle\psi'$. □

Theorem 3.14. *Let* $t, s \in \mathscr{T}_{\mathsf{nf}}$. *Then* $t \leqslant s$ *if and only if* $t \leqslant_{\mathsf{C}} s$.

Proof. ($\Rightarrow$) Immediate consequence of Theorem 3.8.

($\Leftarrow$) Let $\mathscr{S}$ be the smallest pre-subtyping that includes $\leqslant_{\mathsf{C}}$ and observe that it is a coinductive unfair subtyping relation because $\leqslant_{\mathsf{U}}$ is transitive. Suppose $T \leqslant_{\mathsf{C}} S$ and that $M \mid \mathtt{p} : T$ is $\mathscr{S}$-correct. Consider a derivation $M \mid \mathtt{p} : S \overset{\tau}{\Longrightarrow}_{\mathscr{S}} M' \mid \mathtt{p} : S'$. By Lemma B.1 we deduce $M \mid \mathtt{p} : T \overset{\tau}{\Longrightarrow}_{\mathscr{S}} M' \mid \mathtt{p} : T'$ for some $T' \leqslant_{\mathsf{C}} S'$. In particular, $T' \prec S'$. From the hypothesis that $M \mid \mathtt{p} : T$ is $\mathscr{S}$-correct we deduce that $M' \mid \mathtt{p} : T'$ is also $\mathscr{S}$-correct. From Lemma B.4 we conclude $M' \mid \mathtt{p} : S' \overset{\checkmark}{\Longrightarrow}_{\mathscr{S}}$. □

Proposition B.5. $T \prec S$ *if and only if* $[T] \prec [S]$.

Proof. We only show the "only if" part, the "if" part being symmetric. Let $M \mid \mathtt{p} : [T]$ be correct where M is ground. Then $M \mid \mathtt{p} : T$ is also correct and, from the hypothesis $T \prec S$, we deduce $M \mid \mathtt{p} : S \overset{\checkmark}{\Longrightarrow}$. We conclude $M \mid \mathtt{p} : [S] \overset{\checkmark}{\Longrightarrow}$ since M is ground. □

Theorem 3.19. *Let* $T, S \in \mathscr{T}_{\mathsf{nf}}$ *and* $T \leqslant_{\mathsf{U}} S$. *Then* $T \prec S$ *if and only if* $T - S$ *is not viable.*

Proof. We prove the equivalent property that $T \not\prec S$ if and only if $T - S$ is viable. Without loss of generality we assume that both T and S are ground, since $T - S$ is viable if and only if $[T - S]$ is (Proposition 3.11) and $T \prec S$ if and only if $[T] \prec [S]$ (Proposition B.5).

($\Rightarrow$) Then there exists M ground such that (*) $M \mid \mathtt{p} : T$ is correct and (**) $M \mid \mathtt{p} : S \stackrel{\checkmark}{\Longrightarrow}$. If we prove that $M \mid \mathtt{p} : (T - S)$ is correct we are done. Consider a derivation $M \mid \mathtt{p} : (T - S) \stackrel{\tau}{\Longrightarrow} M' \mid \mathtt{p} : R$. By unzipping this derivation we deduce that there exists a string φ of actions such that $M \stackrel{\overline{\varphi}}{\Longrightarrow} M'$ and $T - S \stackrel{\varphi}{\Longrightarrow} R$. By Proposition 3.16(1) and from the definition of $T - S$ we have $T \stackrel{\varphi}{\Longrightarrow} T'$ for some T' such that either $T' = R$ or there exists S' such that $S \stackrel{\varphi}{\Longrightarrow} S'$ and $R = T' - S'$. From (*) we also deduce that $M' \mid \mathtt{p} : T'$ is correct, therefore $M' \stackrel{\overline{\varphi}'\checkmark}{\Longrightarrow}$ and $T' \stackrel{\varphi'\checkmark}{\Longrightarrow}$ for some string φ' of actions. Now we have $\varphi\varphi' \in \mathtt{traces}(T)$ and, from (**), $\varphi\varphi' \notin \mathtt{traces}(S)$. By Proposition 3.16(2) we deduce $\varphi\varphi' \in \mathtt{traces}(T - S)$, that is $R \stackrel{\varphi'\checkmark}{\Longrightarrow}$.

($\Leftarrow$) Then there exist M and $\mathtt{p}$ such that $M \mid \mathtt{p} : (T - S)$ is correct. Without loss of generality we may assume that M is ground. We deduce that $M \mid \mathtt{p} : T$ is also correct. Suppose, by contradiction, that $M \mid \mathtt{p} : S \stackrel{\checkmark}{\Longrightarrow}$. Then there exists φ such that $M \stackrel{\overline{\varphi}\checkmark}{\Longrightarrow}$ and $S \stackrel{\varphi}{\Longrightarrow} \mathsf{end}$. By Lemma B.1 we deduce that $T \stackrel{\varphi}{\Longrightarrow} \mathsf{end}$. We deduce $\varphi \in \mathtt{traces}(T) \cap \mathtt{traces}(S)$. From Proposition 3.16 we derive $T - S \stackrel{\varphi}{\Longrightarrow} \mathsf{bot}$, which is absurd since $M \mid \mathtt{p} : (T - S)$ is correct. We must conclude $M \mid \mathtt{p} : S \stackrel{\checkmark}{\Longrightarrow}$. □

Appendix C. Supplement to Section 4

Proposition 4.2. *The following properties hold:*

(1) $\mathtt{paths}(T)$ *is finite for every* T*;*

(2) $T \stackrel{\varphi\checkmark}{\Longrightarrow}$ *implies* $T \downarrow \pi$ *and* $\pi \subseteq \{S \mid \exists \psi \leq \varphi : T \stackrel{\psi}{\Longrightarrow} S\}$ *for some* π*.*

Proof. Regarding item (1), it is sufficient to observe that $T \downarrow \pi$ implies $\pi \subseteq \mathtt{ctrees}(T)$ and that $\mathtt{ctrees}(T)$ is finite. Regarding item (2) we can decompose the derivation $T \stackrel{\varphi\checkmark}{\Longrightarrow}$ as

$$T = T_0 \stackrel{\tau}{\Longrightarrow} \stackrel{\alpha_1}{\longrightarrow} T_1 \stackrel{\tau}{\Longrightarrow} \stackrel{\alpha_2}{\longrightarrow} \cdots \stackrel{\tau}{\Longrightarrow} \stackrel{\alpha_n}{\longrightarrow} T_n = \mathsf{end}$$

where $\varphi = \alpha_1 \cdots \alpha_n$. We prove that $T \downarrow \{T_0, \ldots, T_n\}$ by induction on n.

— ($n = 0$) Then $T_0 = \mathsf{end}$ and we conclude with an application of rule (P-End).

— ($n > 0$ and $\alpha_1 = \mathtt{p} : \mathtt{q}?\mathtt{a}\langle t\rangle$) Then $T_0 = \sum_{i \in I} \mathtt{q}?\mathtt{a}_i\langle t_i\rangle.S_i$ and $\mathtt{a} = \mathtt{a}_k$ and $t \leqslant t_k$ and $T_1 = S_k$ for some $k \in I$. By induction hypothesis we deduce $T_1 \downarrow \{T_1, \ldots, T_n\}$. We conclude $T \downarrow \{T_0, \ldots, T_n\}$ with an application of rule (P-Input).

— ($n > 0$ and $\alpha_1 = \mathtt{p} : \mathtt{q}!\mathtt{a}\langle t\rangle$) Symmetric of the previous case. □

Lemma C.1. $T \in \mathscr{T}_{\mathsf{v}}$ *implies* $T \in \mathtt{viables}(T)$.

Proof. We prove that $T \notin \mathtt{viables}(T)$ implies $T \notin \mathscr{T}_{\mathsf{v}}$. More precisely, we prove that for every $i \in \mathbb{N}$ and $S \in \mathbf{V}_i^T \setminus \mathbf{V}_{i+1}^T$ we have S non-viable by induction on i.

— ($i = 0$) Then $\mathtt{paths}(S) = \emptyset$ meaning $\mathsf{end} \notin \mathtt{ctrees}(S)$. We conclude that S is not viable.

— ($i > 0$ and i is even) Then for every $\pi \in \mathtt{paths}(S)$ there exists $S' \in \pi \setminus \mathbf{V}_i^T$. Suppose, by contradiction, that S is viable. Then there exists M such that $M \,|\, \mathtt{p} : S$ is correct, hence $M \mid \mathtt{p} : S \stackrel{\tau}{\Longrightarrow} N \mid \mathtt{p} : \mathtt{end}$ where $N \stackrel{\checkmark}{\longrightarrow}$. We deduce that $M \stackrel{\overline{\varphi}}{\Longrightarrow} N$ and $S \stackrel{\varphi}{\Longrightarrow} \mathtt{end}$ for some finite string φ of actions. Then there exists $\pi \in \mathtt{paths}(S)$ such that $\pi \subseteq \{S' \mid \exists \psi \leq \varphi : S \stackrel{\psi}{\Longrightarrow} S'\}$ and, in particular, there exist $\psi \leq \varphi$ and $S' \in \pi \setminus \mathbf{V}_i^T$ such that $S \stackrel{\psi}{\Longrightarrow} S'$. By induction hypothesis we deduce that S' is not viable. This contradicts the hypothesis that $M \,|\, \mathtt{p} : S$ is correct, hence that S is viable.

— ($i > 0$ and i is odd) The session type S cannot be $\mathtt{bot}$, for otherwise it would have been removed at step 0. Furthermore, S cannot be $\mathtt{end}$ or an external choice because these session types are always preserved across even-indexed steps. Then $S = \bigoplus_{j \in I} \mathtt{q}!\mathtt{a}_j\langle t_k \rangle.S_j$ and $S_k \notin \mathbf{V}_i^T$ for some $k \in I$. By induction hypothesis we deduce that S_k is not viable, therefore we conclude that S is not viable either. □

The next lemma establishes a strong correspondence between the traces of a session type and those of its normal form and is used in the proof of Theorem 4.10.

Lemma C.2. *Let $\mathscr{S}$ be the least pre-subtyping relation that includes both $\leqslant$ and $\{(t, \mathtt{nf}(t)) \mid t \in \mathscr{T}\} \cup \{(\mathtt{nf}(t), t) \mid t \in \mathscr{T}\}$. The following properties hold:*

(1) *if $M \mid \mathtt{p} : T$ is $\mathscr{S}$-correct and $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N \mid \mathtt{p} : T'$, then $M \mid \mathtt{p} : \mathtt{nf}(T) \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N \mid \mathtt{p} : \mathtt{nf}(T')$;*

(2) *$M \mid \mathtt{p} : \mathtt{nf}(T) \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N \mid \mathtt{p} : S$ implies $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N \mid \mathtt{p} : T'$ and $S = \mathtt{nf}(T')$.*

Proof. We prove the result for a single τ reduction, the general statements follow by a simple induction. We only prove item (1) since item (2) is easier (it follows from the fact that the traces of $\mathtt{nf}(T)$ are traces of T). Regarding item (1), suppose $M \mid \mathtt{p} : T \stackrel{\tau}{\longrightarrow}_{\mathscr{S}} N \mid \mathtt{p} : T'$. We reason by cases on the derivation of this reduction:

— ($M \stackrel{\tau}{\longrightarrow}_{\mathscr{S}} N$) Then $T' = T$ and there is nothing left to prove.

— ($T \stackrel{\tau}{\longrightarrow}_{\mathscr{S}} T'$) Then $T = \bigoplus_{i \in I} \mathtt{q}!\mathtt{a}_i\langle t_i \rangle.T_i$ and $T' = \mathtt{q}!\mathtt{a}_k\langle t_k \rangle.T_k$ for some $k \in I$. From the hypothesis that $M \,|\, \mathtt{p} : T$ is $\mathscr{S}$-correct we deduce that $[M] \,|\, \mathtt{p} : T$ is correct, hence T is viable. From Lemma C.1 we deduce $T \in \mathtt{viables}(T)$. We conclude observing that $\mathtt{nf}(T) = \bigoplus_{i \in I} \mathtt{q}!\mathtt{a}_i\langle \mathtt{nf}(t_i) \rangle.\mathtt{nf}(T_i) \stackrel{\tau}{\longrightarrow}_{\mathscr{S}} \mathtt{q}!\mathtt{a}_k\langle \mathtt{nf}(t_k) \rangle.\mathtt{nf}(T_k)$ by definition of $\mathtt{nf}(\cdot)$.

— ($M \stackrel{\mathtt{q}:\mathtt{p}!\mathtt{a}\langle t \rangle}{\longrightarrow}_{\mathscr{S}} N$ and $T \stackrel{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t \rangle}{\longrightarrow}_{\mathscr{S}} T'$) Then $T = \sum_{i \in I} \mathtt{q}?\mathtt{a}_i\langle t_i \rangle.T_i$ and $\mathtt{a} = \mathtt{a}_k$ and $(t, t_k) \in \mathscr{S}$ and $T' = T_k$ for some $k \in I$. From the hypothesis $M \mid \mathtt{p} : T$ $\mathscr{S}$-correct we deduce that both T and T_k are viable. By Lemma C.1 we deduce $T \in \mathtt{viables}(T)$ and $T_k \in \mathtt{viables}(T_k)$. By definition of $\mathtt{nf}(\cdot)$ we have $\mathtt{nf}(T) = \sum_{i \in J} \mathtt{q}?\mathtt{a}_i\langle \mathtt{nf}(t_i) \rangle.\mathtt{nf}(T_i)$ with $k \in J \subseteq I$. By definition of $\mathscr{S}$ we have $(t, \mathtt{nf}(t_k)) \in \mathscr{S}$ therefore we conclude $\mathtt{nf}(T) \stackrel{\mathtt{p}:\mathtt{q}?\mathtt{a}\langle t \rangle}{\longrightarrow}_{\mathscr{S}} \mathtt{nf}(T')$.

— ($M \stackrel{\mathtt{q}:\mathtt{p}?\mathtt{a}\langle t \rangle}{\longrightarrow}_{\mathscr{S}} N$ and $T \stackrel{\mathtt{p}:\mathtt{q}!\mathtt{a}\langle t \rangle}{\longrightarrow}_{\mathscr{S}} T'$) Symmetric of the previous case. □

Theorems 4.3, 4.8, and 4.10 are proved in reverse order with respect to their occurrences in Section 4 to honor their dependencies.

Theorem 4.10. *For every $t \in \mathscr{T}$ we have $\mathtt{nf}(t) \in \mathscr{T}_{\mathsf{nf}}$ and $t \lesssim \mathtt{nf}(t)$.*

Proof. Regarding $\mathtt{nf}(t) \in \mathscr{T}_{\mathsf{nf}}$, the only interesting fact to observe is that from Theorem 3.7 we know that whenever $\mathtt{nf}(t) = T \neq \mathsf{bot}$ we have $T \in \mathscr{T}_{\mathsf{v}}$. Then there exists φ such that $T \stackrel{\varphi}{\Longrightarrow} \mathsf{end}$, that is $\mathsf{end} \in \mathtt{ctrees}(T)$.

Regarding the equivalence $t \lesssim \mathtt{nf}(t)$, it suffices to show that the relation $\mathscr{S}$ of Lemma C.2 is $\mathbf{F}$-consistent, that is $\mathscr{S} \subseteq \mathbf{F}(\mathscr{S})$. The only interesting cases are $(T, \mathtt{nf}(T)) \in \mathscr{S}$ and $(\mathtt{nf}(T), T) \in \mathscr{S}$, which we prove in order.

Suppose that $(T, \mathtt{nf}(T)) \in \mathscr{S}$, that $M \mid \mathtt{p} : T$ is $\mathscr{S}$-correct and consider a derivation $M \mid \mathtt{p} : \mathtt{nf}(T) \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N \mid \mathtt{p} : S$. From Lemma C.2(2) we deduce $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N \mid \mathtt{p} : T'$ and $S = \mathtt{nf}(T')$. From the hypothesis that $M \mid \mathtt{p} : T$ is $\mathscr{S}$-correct we deduce $N \mid \mathtt{p} : T' \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N' \mid \mathtt{p} : \mathsf{end}$. From Lemma C.2(1) we conclude $N \mid \mathtt{p} : S \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N' \mid \mathtt{p} : \mathsf{end}$.

Suppose now that $(\mathtt{nf}(T), T) \in \mathscr{S}$ and that $M \mid \mathtt{p} : \mathtt{nf}(T)$ is $\mathscr{S}$-correct. Then it is easy to see that $M \mid \mathtt{p} : T$ is also $\mathscr{S}$-correct (T and $\mathtt{nf}(T)$ may only differ because some unusable branches in T have been removed in $\mathtt{nf}(T)$). Consider now a derivation $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N \mid \mathtt{p} : T'$. From Lemma C.2(1) we deduce $M \mid \mathtt{p} : \mathtt{nf}(T) \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N \mid \mathtt{p} : \mathtt{nf}(T')$. From the hypothesis that $M \mid \mathtt{p} : \mathtt{nf}(T)$ is $\mathscr{S}$-correct we deduce $N \mid \mathtt{p} : \mathtt{nf}(T') \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N' \mid \mathtt{p} : \mathsf{end}$. From Lemma C.2(2) we conclude $N \mid \mathtt{p} : T' \stackrel{\tau}{\Longrightarrow}_{\mathscr{S}} N' \mid \mathtt{p} : \mathsf{end}$. □

Theorem 4.8. *$T \in \mathscr{T}_{\mathsf{v}}$ if and only if $T \in \mathtt{viables}(T)$.*

Proof. This is a corollary of Lemma C.1 and Theorem 4.10. □

Theorem 4.3. *$\mathtt{NonViable}(T)$ if and only if T is not viable.*

Proof. ($\Rightarrow$) Suppose by contradiction that T is viable and that $M \mid \mathtt{p} : T$ is correct. We prove that there exist N and S such that $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow} N \mid \mathtt{p} : S \stackrel{\checkmark}{\not\Longrightarrow}$ by induction on the derivation of $\mathtt{NonViable}(T)$ and by cases on the last rule applied.

— (B-Path) Then for every $\pi \in \mathtt{paths}(T)$ there exists $T' \in \pi$ such that $\mathtt{NonViable}(T')$. From the hypothesis $M \mid \mathtt{p} : T$ correct we deduce $M \mid \mathtt{p} : T \stackrel{\checkmark}{\Longrightarrow}$, namely $M \stackrel{\overline{\varphi}\checkmark}{\Longrightarrow}$ and $T \stackrel{\varphi\checkmark}{\Longrightarrow}$ for some string φ of actions. Then there exists π such that $T \downarrow \pi$ and $\pi \subseteq \{T' \mid \exists \psi \leq \varphi : T \stackrel{\psi}{\Longrightarrow} T'\}$ and $T' \in \pi$ and $\mathtt{NonViable}(T')$. We have $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow} M' \mid \mathtt{p} : T'$ correct for some M'. By induction hypothesis we conclude that $M' \mid \mathtt{p} : T' \stackrel{\tau}{\Longrightarrow} N \mid \mathtt{p} : S \stackrel{\checkmark}{\not\Longrightarrow}$ for some N and S, which is absurd.

— (B-Output) Then $T = \bigoplus_{i \in I} \mathtt{p}!\mathtt{a}_i\langle t_i \rangle.T_i$ and $\mathtt{NonViable}(T_k)$ for some $k \in I$. From the hypothesis $M \mid \mathtt{p} : T$ correct we deduce $M \mid \mathtt{p} : T \stackrel{\tau}{\Longrightarrow} M \mid \mathtt{p} : T_k$ where $M \mid \mathtt{p} : T_k$ is correct. By induction hypothesis we conclude $M \mid \mathtt{p} : T_k \stackrel{\tau}{\Longrightarrow} N \mid \mathtt{p} : S \stackrel{\checkmark}{\not\Longrightarrow}$ for some N and S, which is absurd.

($\Leftarrow$) From Theorem 4.8 we deduce $T \notin \mathtt{viables}(T)$. Let $\{\mathbf{V}_i^T\}_{i \in \mathbb{N}}$ be the viability sequence for T. We prove that for every $i \in \mathbb{N}$ and $S \in \mathbf{V}_i^T \setminus \mathbf{V}_{i+1}^T$ we have $\mathtt{NonViable}(S)$ by induction on i.

— ($i = 0$) Then $\mathtt{paths}(S) = \emptyset$ and we conclude with an application of rule (B-Path).

— ($i > 0$ and i is even) Then for every $\pi \in \mathtt{paths}(S)$ there exists $S' \in \pi \setminus \mathbf{V}_i^T$. By induction hypothesis we deduce that for every $\pi \in \mathtt{paths}(S)$ there exists $S' \in \pi$ such that $\mathtt{NonViable}(S')$. We conclude $\mathtt{NonViable}(S)$ with an application of rule (B-Path).

— ($i > 0$ and i is odd) The session type S cannot be bot, for otherwise it would have been removed at step 0. Furthermore, S cannot be end or an external choice because these session types are always preserved across even-indexed steps. Therefore $S = \bigoplus_{j \in I} \mathsf{p}!\mathsf{a}_j\langle t_j \rangle.T_j$ and we deduce $T_k \in \mathbf{V}_{i-1}^T \setminus \mathbf{V}_i^T$ for some $k \in I$. By induction hypothesis we obtain $\mathtt{NonViable}(T_k)$ and we conclude $\mathtt{NonViable}(S)$ with an application of rule (B-Output). □

Lemma C.3. *Let $t \leqslant_{\mathsf{A}} s$ and $(t, s) \vdash t' \leqslant_{\mathsf{A}} s'$. Then $\vdash t' \leqslant_{\mathsf{A}} s'$.*

Proof. A simple structural induction on the derivation of $(t, s) \vdash t' \leqslant_{\mathsf{A}} s'$, replacing every instance of (A-Axiom) of the form $\Gamma, (t, s) \vdash t \leqslant_{\mathsf{A}} s$ with a copy of the proof tree for $t \leqslant_{\mathsf{A}} s$. □

Theorem 4.12. *Let $t, s \in \mathscr{T}_{\mathsf{nf}}$. Then $t \leqslant_{\mathsf{F}} s$ if and only if $t \leqslant_{\mathsf{A}} s$.*

Proof. ($\Rightarrow$) Let $\Delta = (\mathtt{trees}(t) \times \mathtt{trees}(s)) \cup (\mathtt{trees}(s) \times \mathtt{trees}(t))$. We show how to build a derivation for $\Gamma \vdash t' \leqslant_{\mathsf{A}} s'$ by induction on $\Delta \setminus \Gamma$ whenever $t', s' \in \mathtt{trees}(t) \cup \mathtt{trees}(s)$ and $t' \leqslant_{\mathsf{F}} s'$. The statement follows by taking $\Gamma = \emptyset$ and $t' = t$ and $s' = s$. In the base case we have $\Delta \setminus \Gamma = \emptyset$, namely $(t', s') \in \Gamma$. We conclude with an application of rule (A-Axiom). In the inductive case we reason by cases on the shape of t' and s', taking into account the hypothesis $t' \leqslant_{\mathsf{F}} s'$:

— ($t' = \mathsf{bot}$) We conclude with an application of rule (A-Bottom).
— ($s' = \mathsf{top}$) We conclude with an application of rule (A-Top).
— ($t' = s' \equiv \mathsf{end}$) We conclude with an application of rule (A-End).
— ($t' = \sum_{i \in I} \mathsf{p}?\mathsf{a}_i\langle t_i \rangle.T_i$ and $s' \equiv \sum_{i \in I \cup J} \mathsf{p}?\mathsf{a}_i\langle s_i \rangle.S_i$) Then $t_i \leqslant_{\mathsf{F}} s_i$ and $T_i \leqslant_{\mathsf{F}} S_i$ for every $i \in I$. By induction hypothesis we deduce $\Gamma, (t', s') \vdash t_i \leqslant_{\mathsf{A}} s_i$ and $\Gamma, (t', s') \vdash T_i \leqslant_{\mathsf{A}} S_i$ for every $i \in I$. We conclude with an application of rule (A-Input).
— ($t' = \bigoplus_{i \in I \cup J} \mathsf{p}!\mathsf{a}_i\langle t_i \rangle.T_i$ and $s' \equiv \bigoplus_{i \in I} \mathsf{p}!\mathsf{a}_i\langle s_i \rangle.S_i$) Similar to the previous case, but concluded with an application of rule (A-Output).

($\Leftarrow$) Trivial by definition of $\leqslant_{\mathsf{A}}$ and $\leqslant_{\mathsf{F}}$. □