Retractable and Speculative Contracts

Franco Barbanera[1], Ivan Lanese[2], and Ugo de'Liguoro[3]

[1] Dipartimento di Matematica e Informatica, University of Catania
`barba@dmi.unict.it`
[2] Dipartimento di Informatica - Scienza e Ingegneria, University of Bologna/INRIA
`ivan.lanese@gmail.com`
[3] Dipartimento di Informatica, University of Torino
`ugo.deliguoro@unito.it`

Abstract. *Behavioral contracts* are abstract descriptions of the communications that clients and servers perform. Behavioral contracts come naturally equipped with a notion of *compliance*: when a client and a server follow compliant contracts, their interaction is guaranteed to progress or successfully complete. We study two extensions of contracts, dealing respectively with *backtracking* and with *speculative execution*. We show that the two extensions give rise to *the same notion of compliance*. As a consequence, they also give rise to the same *subcontract relation*, which determines when one server can be replaced by another preserving compliance. Moreover, compliance and subcontract relation are both decidable in polynomial time.

1 Introduction

Binary behavioral contracts [13,26,14] and binary session types [21] are abstractions of programs used to statically ensure that a client and a server interact successfully (see the survey in [23]). Along the years, the basic theory has been extended to deal with many features of clients and servers, such as exceptions [11], time [8], and so on. We consider here two new features: *backtracking*, allowing one to go back to previous stages of the interaction, and *speculative execution* [29], allowing one to try different alternatives concurrently. These two features have quite different origin and aims. Backtracking is used to avoid failures due to wrong past decisions in a wide range of settings, from the undo button in web browsers, to the execution model of Prolog, to techniques for rollback-recovery [1]. Speculative execution is used for efficiency reasons in different areas, from simulation [12], to thread-level optimization [30], to web services [15].

We present two extensions of binary contracts (Section 2): *retractable contracts* capturing backtracking, and *speculative contracts* capturing speculative execution. The two extensions are based on the same syntax, but naturally have different semantics. Essentially, they add to the session contracts of [3,9] (called first-order session behaviors in [3]) an operator of *external choice among output* operations. The most interesting case is when an external choice among outputs and an external choice among inputs interact. In the retractable semantics, the client and the server agree on which option to explore, but they rollback and try a

different possibility if the computation gets stuck. In the speculative semantics all the possibilities are explored concurrently, and it is enough for one of them to succeed to guarentee the success of the whole computation.

This paper defines retractable and speculative contracts, and studies the related theory, considering the notions of *compliance* (Section 3), guaranteeing that the interaction progresses or successfully completes, *subcontract relation* (Section 4), determining when a server (resp. client) can be replaced by another server (resp. client) preserving compliance, and *dual contract* (Section 4), that is the most general contract (in terms of the subcontract relation) compliant with a given contract. Our analysis provides two main insights:

- Even if retractable contracts and speculative contracts have different semantics and give rise to different client-server interactions, the relations of compliance, subcontract and duality in the two settings do coincide. While surprising at first sight, this can be explained by noticing that in both the cases different alternatives are explored (sequentially for retractable contracts, in parallel for speculative contracts) and the success of one of them guarantees the success of the whole computation. In other terms, the two semantics provide different implementations of *angelic nondeterminism*, first described by Hoare [20].
- While retractable/speculative contracts are strictly more expressive than session contracts (indeed they are a conservative extension, see Section 3.1), their theory preserves the main good properties of the theory of session contracts. In particular, compliance and subcontract relations are both decidable (Section 3) in polynomial time (Section 5), and the dual of a contract always exists and has a simple syntactic characterization (Section 4).

To ensure the existence of the dual contract, one needs to introduce an operator of internal choice among inputs. While this operator has limited practical impact, it makes the model more symmetric and the mathematical treatment simpler.

The results above make us confident in the fact that our semantics correctly captures the interaction patterns we are interested in. As further element supporting this, we show (Section 6) that the backtracking mechanism of retractable contracts can be seen as an application to behavioral contracts of the general theory proposed in [28] to define reversible extensions of process calculi.

A few preliminary results on the topic of this paper have been presented in a workshop paper [7], which considers *retractable session contracts*, i.e., retractable contracts without internal choice among inputs. The main result of [7] is the decidability of the compliance relation (while we study here also the complexity), which was obtained via an algorithm that we now know to be exponential. Here we present a more refined, polynomial one (Figure 10). In [7] the subcontract relation and the dual contract were not studied, and indeed the dual contract did not exist due to the absence of internal choice among inputs.

Proofs missing from the main part are collected in Appendix A.

2 Contracts for Retractable and Speculative Interactions

We present below a uniform syntax for retractable and speculative contracts, with two semantics. It can be obtained from the syntax of session contracts of [3,9] (called first-order session behaviors in [3]), that we dub here SC, just adding external retractable/speculative choice among outputs and internal choice among inputs. As a matter of fact our contracts can also be seen as an extension of the retractable session contracts of [7], that we dub here rC, simply adding internal choice among inputs. As a reference, session contracts and retractable session contracts are recalled in Appendix A.1.

Definition 1 (Retractable/Speculative Contracts). *Let $\mathcal{N}$ (set of names) be some countable set of symbols and let $\overline{\mathcal{N}}$ (set of conames) be $\{\overline{a} \mid a \in \mathcal{N}\}$, with $\mathcal{N} \cap \overline{\mathcal{N}} = \emptyset$. The set* rsC *of retractable/speculative contracts is defined as the set of the* closed *expressions generated by the following grammar,*

$$
\sigma, \rho := \mid \mathbf{1} \qquad\qquad \text{SUCCESS}
$$
$$
\mid \sum_{i \in I} a_i.\sigma_i \qquad \text{EXTERNAL INPUT CHOICE}
$$
$$
\mid \sum_{i \in I} \overline{a}_i.\sigma_i \qquad \text{EXTERNAL OUTPUT CHOICE}
$$
$$
\mid \bigoplus_{i \in I} a_i.\sigma_i \qquad \text{INTERNAL INPUT CHOICE}
$$
$$
\mid \bigoplus_{i \in I} \overline{a}_i.\sigma_i \qquad \text{INTERNAL OUTPUT CHOICE}
$$
$$
\mid x \qquad\qquad \text{VARIABLE}
$$
$$
\mid \mathsf{rec}\, x.\sigma \qquad \text{RECURSION}
$$

where I is non-empty and finite, the names and the conames in choices are pairwise distinct and σ is not a variable in $\mathsf{rec}\, x.\sigma$.

Recursion in rsC is guarded and hence contractive in the usual sense. We take an equi-recursive view of recursion by equating $\mathsf{rec}\, x.\sigma$ with $\sigma[\mathsf{rec}\, x.\sigma/x]$. We use α to range over $\mathcal{N} \cup \overline{\mathcal{N}}$, with the convention $\overline{\alpha} = \overline{a}$ if $\alpha = a$, and $\overline{\alpha} = a$ if $\alpha = \overline{a}$. We write $\alpha_1.\sigma_1 + \alpha_2.\sigma_2$ for binary external input/output choice and $\alpha_1.\sigma_1 \oplus \alpha_2.\sigma_2$ for binary internal input/output choice. They are both commutative by definition. Also, $\alpha.\sigma$ denotes both internal and external unary choice. This is not a source of confusion since internal and external choices do coincide in the unary case. We also write $\alpha_k.\sigma_k + \sigma'$ for $\sum_{i \in I} \alpha_i.\sigma_i$ where $k \in I$ and $\sigma' = \sum_{i \in (I \setminus \{k\})} \alpha_i.\sigma_i$ (and similarly for internal choices). When no ambiguity can arise, we call just *contracts* the expressions in rsC. They are written by omitting all trailing $\mathbf{1}$'s.

We discuss below the two interpretations and the two semantics for our contracts: the retractable one, and the speculative one.

2.1 Retractable semantics

The main novelty of the retractable semantics is that when an external choice among outputs and an external choice among inputs interact, the client and the server agree on which option to explore, but they rollback and try a different possibility if the computation gets stuck.

In order to deal with rollbacks, we decorate contracts with their history, which memorizes, for past choices, the alternatives that have been discharged and that can be tried upon rollback. We use '$\circ$' to stand for no-remaining-alternatives.

Definition 2 (Contracts with History). *Let* Histories *be the expressions generated by the grammar* $H ::= \langle \rangle \mid H\!:\!\sigma$, *where* $\sigma \in$ **rsC** $\cup \{\circ\}$ *and* $\circ \notin$ **rsC**. Histories *are hence stacks of contracts and* $\circ$. *Then the set of* contracts with history *is defined by:* $\quad$ **rsCH** $= \{H \ltimes \sigma \mid H \in$ Histories$, \sigma \in$ **rsC** $\cup \{\circ\}\,\}$.

We write just $\sigma_1\!:\!\cdots\!:\!\sigma_k$ for the stack $(\cdots(\langle\rangle\!:\!\sigma_1)\!:\!\cdots)\!:\!\sigma_k$.

As standard for contracts, the definition of the retractable semantics is in two stages: we first define a labeled transition system (LTS) for contracts with history (Definition 3), and then we use it to define a reduction semantics for pairs of contracts representing one client and one server (Definition 4).

Definition 3 (Semantics of Contracts with History).

$(+)\ H \ltimes \alpha.\sigma + \sigma' \xrightarrow{\alpha} H\!:\!\sigma' \ltimes \sigma \qquad\qquad (\oplus)\ H \ltimes \alpha.\sigma \oplus \sigma' \xrightarrow{\tau} H \ltimes \alpha.\sigma$

$(\alpha)\ H \ltimes \alpha.\sigma \xrightarrow{\alpha} H\!:\!\circ \ltimes \sigma \qquad\qquad\quad (\mathsf{rb})\ H\!:\!\sigma' \ltimes \sigma \xrightarrow{\mathsf{rb}} H \ltimes \sigma'$

In the transition rule for external choice $(+)$, the action α is executed, and the discharged branches in σ' are memorized. In internal choice $(\oplus)$, instead, the selection of one branch is represented by a label τ, and the history H is unchanged. When a single action is executed (α), a '$\circ$' is added to the history, meaning that the only possible branch has been tried and no alternative is left. Rule (rb) pops the contract at the top of the stack, replacing the current one with it.

The client/server interaction is modeled by the reduction of their parallel composition, that can be either *forward*, consisting of CCS-style synchronizations and single internal choices, or *backward*, only when there is no possible forward reduction, and the client is not satisfied, i.e., it is different from $\mathbf{1}$.

Definition 4 (Semantics of Retractable Client/Server Pairs).
The following rules, plus the rule symmetric to (τ) *w.r.t.* $\parallel$, *define the relation* $\longrightarrow$ *over pairs of contracts with history:*

(comm)

$$\frac{H_1 \ltimes \rho \xrightarrow{\alpha} H'_1 \ltimes \rho' \qquad H_2 \ltimes \sigma \xrightarrow{\overline{\alpha}} H'_2 \ltimes \sigma'}{H_1 \ltimes \rho \parallel H_2 \ltimes \sigma \longrightarrow H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma'}$$

(τ)

$$\frac{H_1 \ltimes \rho \xrightarrow{\tau} H_1 \ltimes \rho'}{H_1 \ltimes \rho \parallel H_2 \ltimes \sigma \longrightarrow H_1 \ltimes \rho' \parallel H_2 \ltimes \sigma}$$

(rbk)

$$\frac{H_1 \ltimes \rho \xrightarrow{\mathsf{rb}} H'_1 \ltimes \rho' \qquad H_2 \ltimes \sigma \xrightarrow{\mathsf{rb}} H'_2 \ltimes \sigma' \qquad \rho \neq \mathbf{1}}{H_1 \ltimes \rho \parallel H_2 \ltimes \sigma \longrightarrow H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma'}$$

Rule (rbk) *applies only if neither* (comm) *nor* (τ) *do.*

The *forward reduction* $\longrightarrow_f$ is the relation generated by rules (τ) and (comm).

$H_1 \ltimes a.c + b.d$	$\| \ H_2 \ltimes \overline{a}.\overline{c} + \overline{b}.\overline{e}$	agreement point
$\longrightarrow \quad H_1{:}a.c \ltimes d$	$\| \ H_2{:}\overline{a}.\overline{c} \ltimes \overline{e}$	synchr. and state memorizations
$\longrightarrow \quad H_1 \ltimes a.c$	$\| \ H_2 \ltimes \overline{a}.\overline{c}$	synchr. failure and rollback to the last agreement point
$\longrightarrow \qquad \dots$		etc.
$H_1 \ltimes a.c + b.d$	$\| \ H_2 \ltimes \overline{a}.\overline{c} \oplus \overline{b}.\overline{e}$	normal interaction point
$\longrightarrow \quad H_1 \ltimes a.c + b.d$	$\| \ H_2 \ltimes \overline{b}.\overline{e}$	internal choice
$\longrightarrow \quad H_1{:}a.c \ltimes d$	$\| \ H_2{:} \circ \ltimes \overline{e}$	synchr. and client-state memorization
$\longrightarrow \quad H_1 \ltimes a.c$	$\| \ H_2 \ltimes \circ$	synchr. failure and rollback
$\longrightarrow \qquad \dots$		synchr. failure due to $\circ$, rollback continues to the previous agreement point

Fig. 1. Role of $\circ$ during rollbacks

Example 1. In order to get a better insight on the role of '$\circ$' in the rollback mechanism, observe that, for a client like $a.c + b.d$, rule $(+)$ in Definition 3 forces the memorization of a "rollback state" independently from the shape of the server, which could be, for instance, $\overline{a}.\overline{c} + \overline{b}.\overline{e}$ or $\overline{a}.\overline{c} \oplus \overline{b}.\overline{e}$. In the first case we are in presence of an agreement point, hence the memorized state is the one the client has to rollback to in case of a synchronization failure. In the second case, instead, we are not in presence of an agreement point, since the server decides in isolation which alternative to select, so a future synchronization failure must not make the client roll back to this point. One could hence wonder whether rule $(+)$ could produce some rollback to states which are not agreement points. Indeed, what happens is that, when such a state is reached, at least one of the partners has '$\circ$' as contract. Since '$\circ$' cannot synchronize with anything, the client/server pair is forced to recover an *older* past (if any). This is exemplified in Figure 1.

Remark 1. The semantics defined above for retractable contracts can be seen as an instantiation to contracts of the standard reversible semantics for process calculi, see, e.g., [16,28,24,25]. In particular, by removing:
1. the fact that not all reductions are retractable, but only external choices;
2. the side condition $\rho \neq \mathbf{1}$ in rule (*rbk*), which disallows backtrack after success;
3. the fact that rule (*rbk*) can be applied only if no other rule applies, ensuring that backtrack is performed only when forward computation is stuck;
4. the fact that in external choices the chosen path is not stored in the history, avoiding to retry the same path multiple times;

the semantics would be a classic uncontrolled semantics according to the terminology of [25]. The mechanisms above provide a semantic control of reversibility [25], specifying which rollback steps are allowed, and when. We discuss in Remark 2 the impact that removing the control mechanisms above would have on retractable contracts and on their theory.

$$
\begin{array}{ccc}
\langle\,\rangle \bowtie
\begin{array}{c}
\overline{\mathsf{QoSday}}.(\mathsf{priceMed}.\overline{\mathsf{ok}} \\
\hline
+\ \mathsf{priceLow}.\overline{\mathsf{ok}}) \\
\hline
+\ \overline{\mathsf{QoSnight}}.\mathsf{priceLow}.\overline{\mathsf{ok}}
\end{array}
& \| &
\langle\,\rangle \bowtie \textstyle\sum_{\mathsf{QoS}} \mathsf{QoS}.\overline{\mathsf{price}_{\mathsf{QoS}}}.\mathsf{ok}
\end{array}
$$

$$
\longrightarrow \quad
\begin{array}{c}
\langle\,\rangle : \overline{\mathsf{QoSnight}}.\mathsf{priceLow}.\overline{\mathsf{ok}} \\
\bowtie \mathsf{priceMed}.\overline{\mathsf{ok}} + \mathsf{priceLow}.\overline{\mathsf{ok}}
\end{array}
\ \| \
\begin{array}{c}
\langle\,\rangle : \textstyle\sum_{\mathsf{QoS}\neq\mathsf{QoSday}} \mathsf{QoS}.\overline{\mathsf{price}_{\mathsf{QoS}}}.\mathsf{ok} \\
\bowtie\ \mathsf{priceHigh}.\mathsf{ok}
\end{array}
$$

$$
\longrightarrow \quad
\langle\,\rangle \bowtie \overline{\mathsf{QoSnight}}.\mathsf{priceLow}.\overline{\mathsf{ok}}
\ \| \
\langle\,\rangle \bowtie \textstyle\sum_{\mathsf{QoS}\neq\mathsf{QoSday}} \mathsf{QoS}.\overline{\mathsf{price}_{\mathsf{QoS}}}.\mathsf{ok}
$$

$$
\longrightarrow \quad
\langle\,\rangle : \circ \bowtie \mathsf{priceLow}.\overline{\mathsf{ok}}
\ \| \
\begin{array}{c}
\langle\,\rangle : \textstyle\sum_{\mathsf{QoS}\neq\mathsf{QoSday},\mathsf{QoSnight}} \mathsf{QoS}.\overline{\mathsf{price}_{\mathsf{QoS}}}.\mathsf{ok} \\
\bowtie\ \mathsf{priceLow}.\mathsf{ok}
\end{array}
$$

$$
\longrightarrow \quad
\langle\,\rangle : \circ : \circ \bowtie \overline{\mathsf{ok}}
\ \| \
\begin{array}{c}
\langle\,\rangle : \textstyle\sum_{\mathsf{QoS}\neq\mathsf{QoSday},\mathsf{QoSnight}} \mathsf{QoS}.\overline{\mathsf{price}_{\mathsf{QoS}}}.\mathsf{ok} : \circ \\
\bowtie\ \mathsf{ok}
\end{array}
$$

$$
\longrightarrow \quad
\langle\,\rangle : \circ : \circ : \circ \bowtie \mathbf{1}
\ \| \
\begin{array}{c}
\langle\,\rangle : \textstyle\sum_{\mathsf{QoS}\neq\mathsf{QoSday},\mathsf{QoSnight}} \mathsf{QoS}.\overline{\mathsf{price}_{\mathsf{QoS}}}.\mathsf{ok}: \circ : \circ \\
\bowtie\ \mathbf{1}
\end{array}
$$

Fig. 2. An example of retractable interaction

Example 2. Retractable contracts allow one to first try a preferred alternative, but to accept also another alternative if the first one proves to be impossible to obtain. In cloud computing settings, companies may hire virtual machines and storing facilities from cloud providers with some agreed Quality of Service (QoS). A company is willing to hire at some medium or low price a certain amount of machines for online elaboration during day time, but, if the price is too high, it is also willing to switch to offline night elaboration. In this last case it is only willing to pay a low price.

A retractable contract with this behavior may be written as:
$$\mathsf{cloudClient} = \overline{\mathsf{QoSday}}.(\mathsf{priceMed}.\overline{\mathsf{ok}} + \mathsf{priceLow}.\overline{\mathsf{ok}}) + \overline{\mathsf{QoSnight}}.\mathsf{priceLow}.\overline{\mathsf{ok}}$$
Notice that the contract does not specify which alternative the client prefers: this aspect of the client behavior is abstracted away. A sample server is:
$$\mathsf{cloudServer} = \textstyle\sum_{\mathsf{QoS}\in\{\mathsf{QoSday},\mathsf{QoSnight},\dots\}} \mathsf{QoS}.\overline{\mathsf{price}_{\mathsf{QoS}}}.\mathsf{ok}$$
A sample interaction is described in Figure 2, where we assume that
$$\mathsf{price}_{\mathsf{QoSday}} = \mathsf{priceHigh} \quad \text{and} \quad \mathsf{price}_{\mathsf{QoSnight}} = \mathsf{priceLow}.$$

2.2 Speculative semantics

The main idea of the speculative semantics is that in an external output choice all the options are tried concurrently: if at least one of them succeeds, then the whole computation succeeds. In order to represent concurrent trials we need runtime contracts featuring multiple threads.

Definition 5 (Contracts with Threads). Contracts with threads **C**, *used as runtime syntax for contracts, are parallel compositions of* threads **T**. *Each thread is a contract prefixed by a sequence (possibly empty) of actions uniquely identifying it.*

$$\mathbf{C} ::= \mathbf{T} \mid (\mathbf{C} \mid \mathbf{T}) \mid (\mathbf{T} \mid \mathbf{C}) \qquad \mathbf{T} ::= \sigma \mid \alpha@\mathbf{T}$$

We assume the operator '$\mid$' to be associative and commutative.

As for the retractable semantics, the definition of the speculative semantics is in two stages: we first define an LTS for contracts with threads (Definition 6), and then we use it to define a reduction semantics for pairs of contracts with threads representing one client and one server (Definition 7).

Definition 6 (Semantics of Contracts with Threads).
In the LTS below, we use as labels actions $\alpha ::= a \mid \bar{a}$, sequences of actions $\beta ::= \alpha \mid \alpha\beta$, and complex labels $\beta_\tau ::= \tau \mid \beta, ?\mathbf{T}$, where $?\mathbf{T}$ is either the thread $\mathbf{T}$ or nothing. Also, $\alpha@?\mathbf{T}$ is nothing if $?\mathbf{T}$ is nothing, $\alpha@\mathbf{T}$ otherwise.

$$
\begin{array}{lll}
\textit{(Fork)} & \textit{($\oplus$)} & \textit{(α)} \\[2pt]
\alpha.\sigma + \sigma' \xrightarrow{\alpha,\sigma'} \alpha@\sigma & \alpha.\sigma \oplus \sigma' \xrightarrow{\tau} \alpha.\sigma & \alpha.\sigma \xrightarrow{\alpha} \alpha@\sigma
\end{array}
$$

$$
\begin{array}{lll}
\textit{(@-α)} & \textit{(@-τ)} & \textit{(ParL)} \\[4pt]
\dfrac{\mathbf{T} \xrightarrow{\beta,?\mathbf{T}''} \mathbf{T}'}{\alpha@\mathbf{T} \xrightarrow{\alpha\beta,\alpha@?\mathbf{T}''} \alpha@\mathbf{T}'} & \dfrac{\mathbf{T} \xrightarrow{\tau} \mathbf{T}'}{\alpha@\mathbf{T} \xrightarrow{\tau} \alpha@\mathbf{T}'} & \dfrac{\mathbf{T} \xrightarrow{\beta_\tau} \mathbf{T}'}{\mathbf{T} \mid \mathbf{C} \xrightarrow{\beta_\tau} \mathbf{T}' \mid \mathbf{C}}
\end{array}
$$

In the rule for external choice ($Fork$), when an action α is executed, its continuation σ is prefixed by it. The other branches σ' need to be executed in a freshly spawned thread. Since such thread needs to be installed at top level, σ' is added to the label, and the actual installation is performed at the level of speculative client/server pairs (see rule ($comm$) in Definition 7). The rule for internal choice ($\oplus$) simply selects one of the available options. A unary choice (α) executes the action α and prefixes with it the continuation σ.

Because of rule (@-α), execution is allowed below an @ prefix. The prefix itself is added to the label β and, if present, to the thread $\mathbf{T}''$. Prefixes uniquely identify threads, and ensure that each thread interacts only with the one with dual prefix which is running on the communication partner. This is specified in Definition 7 below. No prefix is added to τ actions, propagated by rule (@-τ). Rule ($ParL$) simply allows components of a parallel composition to execute (a symmetric rule is not needed thanks to the commutativity of $\mid$).

The interaction of a client with a server is modeled by the reduction of their parallel composition.

Definition 7 (Semantics of Speculative Client/Server Pairs).
The following rules, plus the rule symmetric to (τ) w.r.t. $\parallel$, define the relation $\longrightarrow$ over pairs of contracts with threads. In the LTS below, $\mathbf{C} \mid ?\mathbf{T}$ is $\mathbf{C}$ if $?\mathbf{T}$ is nothing, $\mathbf{C} \mid \mathbf{T}$ otherwise. Also, the duality operator extends from actions to sequences: $\overline{\alpha\beta} = \bar{\alpha}\bar{\beta}$.

$$
\begin{array}{ll}
\textit{(comm)} & \textit{(τ)} \\[4pt]
\dfrac{\mathbf{C} \xrightarrow{\beta,?\mathbf{T}} \mathbf{C}' \quad \mathbf{C}'' \xrightarrow{\bar{\beta},?\mathbf{T}''} \mathbf{C}'''}{\mathbf{C} \parallel \mathbf{C}'' \longrightarrow \mathbf{C}' \mid ?\mathbf{T} \parallel \mathbf{C}''' \mid ?\mathbf{T}''} & \dfrac{\mathbf{C} \xrightarrow{\tau} \mathbf{C}'}{\mathbf{C} \parallel \mathbf{C}'' \longrightarrow \mathbf{C}' \parallel \mathbf{C}''}
\end{array}
$$

Rule (*comm*) allows threads performing dual sequences of actions to interact. This implies that both the actual actions and the prefixes of the threads performing them should be dual. Threads in the labels, if present, are installed in parallel. Rule (τ) simply propagates the τ action.

Example 3. A server provides access to multiple algorithms for SAT solving [34]. A client first sends the problem instance to be solved, then selects the algorithm, and finally sends the relevant parameters. The server computes the solution according to the received commands, and sends it back. Since the most efficient technique depends on the problem instance [33], the server supports speculative execution, to allow one to try different algorithms at the same time (this is called the portfolio approach). The server contract is described by:

$$\mathsf{SATserver} = \mathtt{inst}.\sum_i \mathtt{alg}_i.\sum_j \mathtt{par}_j.\overline{\mathtt{sol}}$$

A simple client that tries both the DPLL approach and the walksat approach can be modeled as follows:

$$\mathsf{SATclient} = \overline{\mathtt{inst}}.(\overline{\mathtt{DPLL}}.\overline{\mathtt{par}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\overline{\mathtt{par}}.\mathtt{sol})$$

A sample computation proceeds as described in Figure 3, assuming that the server supports both DPLL and walksat. To keep the example simple we drop the choice of parameters. Let us see in more details how the creation of threads is managed. The first reduction in Figure 3 is due to rule (*comm*), since

$$\overline{\mathtt{inst}}.(\overline{\mathtt{DPLL}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\mathtt{sol}) \xrightarrow{\overline{\mathtt{inst}}} \overline{\mathtt{inst}}@(\overline{\mathtt{DPLL}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\mathtt{sol})$$

and

$$\mathtt{inst}.\sum_i \mathtt{alg}_i.\overline{\mathtt{sol}} \xrightarrow{\mathtt{inst}} \mathtt{inst}@\sum_i \mathtt{alg}_i.\overline{\mathtt{sol}}.$$

The second reduction is also due to rule (*comm*), since, on the client side

$$\frac{\dfrac{}{\overline{\mathtt{DPLL}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\mathtt{sol} \xrightarrow{\overline{\mathtt{DPLL}},\,\overline{\mathtt{walksat}}.\mathtt{sol}} \overline{\mathtt{DPLL}}@\mathtt{sol}} \text{(Fork)}}{\overline{\mathtt{inst}}@(\overline{\mathtt{DPLL}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\mathtt{sol}) \xrightarrow{\overline{\mathtt{inst}\,\mathtt{DPLL}},\,\overline{\mathtt{inst}@\mathtt{walksat}.\mathtt{sol}}} \overline{\mathtt{inst}}@\overline{\mathtt{DPLL}}@\mathtt{sol}} \text{(@-}\alpha)$$

whereas, on the server side,

$$\frac{\dfrac{}{\sum_i \mathtt{alg}_i.\overline{\mathtt{sol}} \xrightarrow{\mathtt{DPLL},\,\sum_{\{i|\mathsf{A}_i \neq \mathtt{DPLL}\}} \mathtt{alg}_i.\overline{\mathtt{sol}}} \mathtt{DPLL}@\overline{\mathtt{sol}}} \text{(Fork)}}{\mathtt{inst}@\sum_i \mathtt{alg}_i.\overline{\mathtt{sol}} \xrightarrow{\mathtt{inst}\,\mathtt{DPLL},\,\mathtt{inst}@\sum_{\{i|\mathsf{A}_i \neq \mathtt{DPLL}\}} \mathtt{alg}_i.\overline{\mathtt{sol}}} \mathtt{inst}@\mathtt{DPLL}@\overline{\mathtt{sol}}} \text{(@-}\alpha)$$

3 Compliance

The compliance relation for session contracts [3,9] consists in requiring that, whenever no reduction is possible, all client's requests and offers have been satisfied, i.e. the client is in the success state **1**. For retractable contracts, thanks to the retractable operational semantics taking care of forward and backward reductions, we can adopt the same definition. We use $\xrightarrow{\ *\ }$ to denote the reflexive and transitive closure of $\longrightarrow$, and $\nrightarrow$ to specify that no $\longrightarrow$ reduction exists.

Definition 8 (Retractable Compliance Relation $\dashv\vdash^{\mathsf{R}}$).

$$\overline{\texttt{inst}}.(\overline{\texttt{DPLL}}.\texttt{sol} + \overline{\texttt{walksat}}.\texttt{sol}) \quad \| \quad \texttt{inst}.\textstyle\sum_i \texttt{alg}_i.\overline{\texttt{sol}}$$

$$\longrightarrow \quad \overline{\texttt{inst}}@(\overline{\texttt{DPLL}}.\texttt{sol} + \overline{\texttt{walksat}}.\texttt{sol}) \quad \| \quad \texttt{inst}@\textstyle\sum_i \texttt{alg}_i.\overline{\texttt{sol}}$$

$$\longrightarrow \quad \begin{array}{l} \overline{\texttt{inst}}@\overline{\texttt{DPLL}}@\texttt{sol} \\ | \ \overline{\texttt{inst}}@\overline{\texttt{walksat}}.\texttt{sol} \end{array} \quad \| \quad \begin{array}{l} \texttt{inst}@\texttt{DPLL}@\overline{\texttt{sol}} \\ | \ \texttt{inst}@\textstyle\sum_{\{i|A_i \neq \texttt{DPLL}\}} \texttt{alg}_i.\overline{\texttt{sol}} \end{array}$$

$$\longrightarrow \quad \begin{array}{l} \overline{\texttt{inst}}@\overline{\texttt{DPLL}}@\texttt{sol} \\ | \ \overline{\texttt{inst}}@\overline{\texttt{walksat}}@\texttt{sol} \end{array} \quad \| \quad \begin{array}{l} \texttt{inst}@\texttt{DPLL}@\overline{\texttt{sol}} \\ | \ \texttt{inst}@\texttt{walksat}@\overline{\texttt{sol}} \\ | \ \texttt{inst}@\textstyle\sum_{\{i|A_i \neq \texttt{DPLL},\texttt{walksat}\}} \texttt{alg}_i.\overline{\texttt{sol}} \end{array}$$

$$\longrightarrow \quad \begin{array}{l} \overline{\texttt{inst}}@\overline{\texttt{DPLL}}@\texttt{sol} \\ | \ \overline{\texttt{inst}}@\overline{\texttt{walksat}}@\texttt{sol}@1 \end{array} \quad \| \quad \begin{array}{l} \texttt{inst}@\texttt{DPLL}@\overline{\texttt{sol}} \\ | \ \texttt{inst}@\texttt{walksat}@\overline{\texttt{sol}}@1 \\ | \ \texttt{inst}@\textstyle\sum_{\{i|A_i \neq \texttt{DPLL},\texttt{walksat}\}} \texttt{alg}_i.\overline{\texttt{sol}} \end{array}$$

Fig. 3. An example of speculative interaction

i) *The relation* $\dashv\vdash^{\mathsf{R}}$ *on contracts with history is defined by:*
   $\mathrm{H}_1 \ltimes \rho \dashv\vdash^{\mathsf{R}} \mathrm{H}_2 \ltimes \sigma$ *if, for each* $\mathrm{H}_1', \mathrm{H}_2', \rho', \sigma'$ *such that*
   $\mathrm{H}_1 \ltimes \rho \parallel \mathrm{H}_2 \ltimes \sigma \overset{*}{\longrightarrow} \mathrm{H}_1' \ltimes \rho' \parallel \mathrm{H}_2' \ltimes \sigma' \not\longrightarrow,$ *we have* $\rho' = \mathbf{1}$
ii) *The relation* $\dashv\vdash^{\mathsf{R}}$ *on contracts is defined by:* $\rho \dashv\vdash^{\mathsf{R}} \sigma$ *if* $\langle\,\rangle \ltimes \rho \dashv\vdash^{\mathsf{R}} \langle\,\rangle \ltimes \sigma.$

For speculative contracts we need to take into account the fact that the whole computation succeeds if at least one of its branches succeeds.

Definition 9 (Speculative Compliance Relation $\dashv\vdash^{\mathsf{S}}$).
The relation $\dashv\vdash^{\mathsf{S}}$ *on contracts is defined by:*
   $\rho \dashv\vdash^{\mathsf{S}} \sigma$ *if for each* $\mathbf{C}_\rho, \mathbf{C}_\sigma$ *such that* $\rho \parallel \sigma \overset{*}{\longrightarrow} \mathbf{C}_\rho \parallel \mathbf{C}_\sigma \not\longrightarrow$
   there exist $\mathbf{C}, n, \alpha_1, \dots, \alpha_n$ *such that* $\mathbf{C}_\rho = \mathbf{C} \mid \alpha_1@...@\alpha_n@\mathbf{1}$

We now provide a formal system characterizing compliance on both retractable and speculative contracts.

Definition 10 (Formal System for Compliance $\rhd$).
Judgments in the formal system $\rhd$ *are expressions of the form* $\Gamma \rhd \rho \dashv \sigma$,
where the environment Γ *is a finite set of expressions of the form* $\delta \dashv \gamma$, *with*
$\rho, \sigma, \delta, \gamma \in \mathbf{rsC}$. *Axioms and rules are as in Figure 4.*

The only non standard rule of system $\rhd$ is $(+\cdot+)$, which ensures compliance of two external choices when they contain respectively (at least) *one* α and the corresponding $\overline{\alpha}$, followed by compliant contracts. This contrasts with the rules $(\oplus\cdot+)$ and $(+\cdot\oplus)$, where *each* α in an internal choice must have a corresponding $\overline{\alpha}$ in the external choice, followed by compliant contracts. No rule is provided for the case $(\oplus\cdot\oplus)$ since two internal choices are compliant only if both of them are unary choices. In such a case internal choice coincides with external choice, thus this case is taken into account by the rules we already have. Notice that rule $(+\cdot+)$ implicitly represents the fact that, in the decision procedure for two contracts made of external choices, the possible synchronizing branches have to be tried, until either a successful one is found or all fail. Looking at a derivation

$$\begin{array}{lll}
\text{(Ax)} & \text{(Hyp)} & \text{(+ } \cdot \text{ +)} \\
\Gamma \,\triangleright\, \mathbf{1} \dashv \sigma & \Gamma, \rho \dashv \sigma \,\triangleright\, \rho \dashv \sigma & \dfrac{\Gamma, \alpha.\rho + \rho' \dashv \overline{\alpha}.\sigma + \sigma' \,\triangleright\, \rho \dashv \sigma}{\Gamma \,\triangleright\, \alpha.\rho + \rho' \dashv \overline{\alpha}.\sigma + \sigma'}
\end{array}$$

$$\text{(}\oplus \cdot \text{+)} \quad \dfrac{\forall h \in I.\ \Gamma, \bigoplus_{i \in I} \overline{\alpha}_i.\rho_i \dashv \sum_{j \in I \cup J} \alpha_j.\sigma_j \,\triangleright\, \rho_h \dashv \sigma_h}{\Gamma \,\triangleright\, \bigoplus_{i \in I} \overline{\alpha}_i.\rho_i \dashv \sum_{j \in I \cup J} \alpha_j.\sigma_j}$$

$$\text{(+ } \cdot \oplus\text{)} \quad \dfrac{\forall h \in I.\ \Gamma, \sum_{j \in I \cup J} \overline{\alpha}_j.\rho_j \dashv \bigoplus_{i \in I} \alpha_i.\sigma_i \,\triangleright\, \rho_h \dashv \sigma_h}{\Gamma \,\triangleright\, \sum_{j \in I \cup J} \overline{\alpha}_j.\rho_j \dashv \bigoplus_{i \in I} \alpha_i.\sigma_i}$$

Fig. 4. System $\triangleright$

bottom-up, at each application of a rule, the considered pair of contracts is added to the environment Γ. In this way, if the same pair is reached again due to the equi-recursive view of contracts, the derivation can be closed using rule (Hyp). Rule (Ax) instead closes the derivation when the client reaches the success state **1**. We write $\triangleright \rho \dashv \sigma$ instead of $\Gamma \triangleright \rho \dashv \sigma$ when Γ is empty.

Derivability in system $\triangleright$ is decidable, since it is syntax-directed and proof reconstruction does terminate. The procedure **Prove** in Figure 5 clearly implements the formal system, namely it is straightforward to check the following

Fact 1 *i)* **Prove**$(\Gamma \triangleright \rho \dashv \sigma) \neq$ **fail** *iff* $\Gamma \triangleright \rho \dashv \sigma$.
ii) **Prove**$(\Gamma \triangleright \rho \dashv \sigma) = \mathcal{D} \neq$ **fail** *implies* $\mathcal{D} :: \Gamma \triangleright \rho \dashv \sigma$.

Theorem 1. *Derivability in the formal system $\triangleright$ is decidable.*

Proof. By Fact 1, we only need to show that the procedure **Prove** always terminates. Note that, in all recursive calls **Prove**$(\Gamma, \rho \dashv \sigma \triangleright \rho_k \dashv \sigma_k)$ inside **Prove**$(\Gamma \triangleright \rho \dashv \sigma)$, the expressions ρ_k and σ_k are subexpressions of, respectively, ρ and σ (because of the equi-recursive view of recursion they can also be ρ and σ). Since contract expressions generate regular trees, there are only finitely many such subexpressions. This implies that the number of different calls of procedure **Prove** is always finite. $\square$

We can prove the soundness and the completeness of the formal system $\triangleright$ w.r.t. both the retractable and the speculative semantics (see Appendix A.3 for the proofs).

Theorem 2 (Retractable Soundness and Completeness).

$$\triangleright \rho \dashv \sigma \quad \textit{iff} \quad \rho \dashv\!\!\vdash^{\mathsf{R}} \sigma$$

Theorem 3 (Speculative Soundness and Completeness).

$$\triangleright \rho \dashv \sigma \quad \textit{iff} \quad \rho \dashv\!\!\vdash^{\mathsf{S}} \sigma$$

$\mathbf{Prove}(\Gamma \rhd \rho \curlywedge \sigma) =$

  if $\rho = \mathbf{1}$ **then** $\dfrac{}{\Gamma \rhd \mathbf{1} \curlywedge \sigma}\ (\textsc{Ax})$

  else **if** $\rho \curlywedge \sigma \in \Gamma$ **then** $\dfrac{}{\Gamma, \rho \curlywedge \sigma \rhd \rho \curlywedge \sigma}\ (\textsc{Hyp})$

  else **if** $\rho = \sum_{i \in I} \alpha_i.\rho_i$ **and** $\sigma = \sum_{j \in J} \overline{\alpha}_j.\sigma_j$

      and **exists** $k \in I \cap J$ **s.t.** $\mathcal{D} = \mathbf{Prove}(\Gamma, \rho \curlywedge \sigma \rhd \rho_k \curlywedge \sigma_k) \neq \mathbf{fail}$

     then $\dfrac{\mathcal{D}}{\Gamma \rhd \rho \curlywedge \sigma}\ (+ \cdot +)$ **else** **fail**

  else **if** $\rho = \bigoplus_{i \in I} \overline{\alpha}_i.\rho_i$ **and** $\sigma = \sum_{j \in I \cup J} \alpha_j.\sigma_j$

      and **for all** $k \in I$ $\mathcal{D}_k = \mathbf{Prove}(\Gamma, \rho \curlywedge \sigma \rhd \rho_k \curlywedge \sigma_k) \neq \mathbf{fail}$

     then $\dfrac{\forall k \in I\ \ \mathcal{D}_k}{\Gamma \rhd \rho \curlywedge \sigma}\ (\oplus \cdot +)$

  else **if** $\rho = \sum_{j \in I \cup J} \overline{\alpha}_j.\rho_j$ **and** $\sigma = \bigoplus_{i \in I} \alpha_i.\sigma_i$

      and **for all** $k \in I$ $\mathcal{D}_k = \mathbf{Prove}(\Gamma, \rho \curlywedge \sigma \rhd \rho_k \curlywedge \sigma_k) \neq \mathbf{fail}$

     then $\dfrac{\forall k \in I\ \ \mathcal{D}_k}{\Gamma \rhd \rho \curlywedge \sigma}\ (+ \cdot \oplus)$ **else** **fail**

  else **fail**

Fig. 5. The procedure **Prove**.

We can now formally prove the client and server of Example 3 to be compliant, i.e. $\overline{\mathtt{inst}}.(\overline{\mathtt{DPLL}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\mathtt{sol}) \dashv\mathrel{\|}^{\mathsf{S}} \mathtt{inst}.\sum_i \mathtt{alg}_i.\overline{\mathtt{sol}}$, by providing a derivation for $\rhd \ \overline{\mathtt{inst}}.(\overline{\mathtt{DPLL}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\mathtt{sol}) \curlywedge \mathtt{inst}.\sum_i \mathtt{alg}_i.\overline{\mathtt{sol}}$, as shown in Figure 6.

By the soundness and completeness of system $\rhd$ w.r.t. both the relations of retractable and speculative compliance, we immediately get that the two compliance relations do coincide.

Corollary 1 (Retractable and Speculative Compliances Coincide).

$$\dashv\mathrel{\|}^{\mathsf{R}} \ = \ \dashv\mathrel{\|}^{\mathsf{S}}$$

By the above, from now on we write $\dashv\mathrel{\|}$ instead of $\dashv\mathrel{\|}^{\mathsf{R}}$ or $\dashv\mathrel{\|}^{\mathsf{S}}$. So the following also easily follows.

Corollary 2 (Compliance Decidability). *The relation* $\dashv\mathrel{\|}$ *is decidable.*

Remark 2. We now discuss the impact on the compliance relation of the four mechanisms for controlling reversibility in the semantics of retractable client/server pairs (see Remark 1). In particular, we analyze what would happen by dropping each one of them in isolation:

$$\cfrac{\cfrac{\cfrac{\cfrac{\overline{\gamma_1,\gamma_2,\gamma_3 \;\;\triangleright\;\; \mathbf{1} \dashv \mathbf{1}}}{\gamma_1,\gamma_2 \;\;\triangleright\;\; \mathtt{sol} \dashv \overline{\mathtt{sol}}}\;(\text{Ax})}{\gamma_1 \;\;\triangleright\;\; \overline{\mathtt{DPLL}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\mathtt{sol} \dashv \sum_i \mathbf{alg}_i.\overline{\mathtt{sol}}}\;(+ \cdot \oplus)}{\triangleright\;\; \overline{\mathtt{inst}}.(\overline{\mathtt{DPLL}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\mathtt{sol}) \dashv \mathtt{inst}.\sum_i \mathbf{alg}_i.\overline{\mathtt{sol}}}\;(+ \cdot +)}{}\;(\oplus \cdot +)$$

where $\gamma_1 = \overline{\mathtt{inst}}.(\overline{\mathtt{DPLL}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\mathtt{sol}) \dashv \mathtt{inst}.\sum_i \mathbf{alg}_i.\overline{\mathtt{sol}}$
$\quad\quad \gamma_2 = \overline{\mathtt{DPLL}}.\mathtt{sol} + \overline{\mathtt{walksat}}.\mathtt{sol} \dashv \sum_i \mathbf{alg}_i.\overline{\mathtt{sol}}$
$\quad\quad \gamma_3 = \mathtt{sol} \dashv \overline{\mathtt{sol}}$

and where, for some i, $\mathtt{alg}_i = \mathtt{walksat}$.

Fig. 6. A sample derivation in $\triangleright$

Drop "Not all reductions are retractable": each reduction could be undone. From the compliance point of view, all the choices would be retractable. Hence, retractable contracts would not be a conservative extension (see Subsect. 3.1) of session contracts any more. The case we consider is strictly more general, since we allow for both retractable and unretractable choices.

Drop the side condition $\rho \neq 1$ in rule (rbk) of Definition 4: any forward finite interaction would be followed by a rollback. In particular, most of the client/server pairs without recursion (except a few trivial ones, like $\langle\,\rangle \bowtie \mathbf{1} \parallel \langle\,\rangle \bowtie \sigma$) would end into $\langle\,\rangle \bowtie \circ \parallel \langle\,\rangle \bowtie \circ$. Thus all these pairs of contracts would not be compliant.

Drop "rule (rbk) can be applied only if no other rule applies": interactions could rollback before succeeding. As in the case above, most client/server pairs (except a few trivial ones, but including recursive ones) could reduce to $\langle\,\rangle \bowtie \circ \parallel \langle\,\rangle \bowtie \circ$. Again all these pairs of contracts would not be compliant.

Drop "in choices the chosen path is not memorized": any client/server pair that would not normally succeed with at least one retractable choice could diverge by undoing and redoing the choice forever, thus trivially ensuring compliance.

None of the last three scenarios provides a reasonable setting. The first one would be reasonable, but the case we consider is strictly more general.

3.1 Conservativity Results

It is possible to show that all the relations on our retractable and speculative contracts (**rsC**) are conservative extensions of corresponding notions on (first-order) session contracts (SC) as defined in [3,9], and on the retractable session contracts (**rC**) as defined in [7].

As previously said, it is not difficult to check that session contracts SC are a subset of retractable session contracts **rC**, which, in turn, are a subset of the contracts **rsC** we are presently investigating, namely: $\mathsf{SC} \subsetneq \mathbf{rC} \subsetneq \mathbf{rsC}$. Obviously the strict inclusion $\mathsf{SC} \subsetneq \mathbf{rsC}$ is not enough, by itself, to guarantee the retractable

and speculative operational semantics for **rsC** to be conservative extensions of the operational semantics of SC. We prove that it is so in the following Proposition 1. Informally, it states that both the forward retractable semantics $\longrightarrow_f$ and the speculative semantics $\longrightarrow$ of pairs of contracts in SC are annotated versions of their semantics in SC, which we recall in Appendix A.1.

Proposition 1 (Operational Semantics Conservativity). *Let $\rho, \sigma \in$ SC.*

i) $\rho \parallel \sigma \stackrel{*}{\longrightarrow}_{\mathsf{SC}} \rho' \parallel \sigma'$ *iff* $\mathrm{H}_1 \ltimes \rho \parallel \mathrm{H}_2 \ltimes \sigma \stackrel{*}{\longrightarrow}_f \mathrm{H}_1' \ltimes \rho' \parallel \mathrm{H}_2' \ltimes \sigma'$
for some $\mathrm{H}_1, \mathrm{H}_2, \mathrm{H}_1'$ *and* H_2'

ii) $\rho \parallel \sigma \stackrel{*}{\longrightarrow}_{\mathsf{SC}} \rho' \parallel \sigma'$ *iff* $\rho \parallel \sigma \stackrel{*}{\longrightarrow} \alpha_1 @ \dots \alpha_n @ \rho' \mid \mathbf{C}_\rho \parallel \overline{\alpha_1} @ \dots \overline{\alpha_n} @ \sigma' \mid \mathbf{C}_\sigma$
for some $n, \alpha_1, \dots, \alpha_n, \mathbf{C}_\rho$ *and* $\mathbf{C}_\sigma$

where $\longrightarrow_{\mathsf{SC}}$ *denotes the reduction relation on* SC *pairs in the theory of session contracts.*

Proof. See Appendix A.2. □

We do not take into account conservativity of the retractable operational semantics for **rsC** over the one for **rC** because it is quite trivial, since the rules in the two semantics are essentially the same. A conservativity result of the speculative operational semantics for **rsC** over the one for **rC** would instead consist in a rather cumbersome and uninteresting statement.

The conservativity result for the operational semantics is not enough, in itself, to guarantee the theory of retractable compliance for **rsC** to be a conservative extension of both the theory of compliance for **rC** and for SC. Also in this case, however, we can prove it to be so, that is, the compliance relation for session contracts SC is the restriction of the compliance relation $\dashv\vdash$ for our contracts to pairs of session contracts SC, and similarly for the restriction of $\dashv\vdash$ to retractable session contracts **rC**.

To prove the results above, let $\dashv\vdash_{\mathsf{SC}}$ and $\dashv\vdash_{\mathbf{rC}}$ be the compliance relations on, respectively, session contracts and retractable session contracts. Also, let $\rhd_{\mathsf{SC}}$ and $\rhd_{\mathbf{rC}}$ be, respectively, the formal systems axiomatizing them (see Appendix A.1). We first show that the logical theories of $\rhd_{\mathsf{SC}}$ and $\rhd_{\mathbf{rC}}$ are conservative extensions of the logical theory $\rhd$.

Proposition 2 (Formal Systems Conservativity).

i) *Let $\rho, \sigma \in$ SC:* $\rhd_{\mathsf{SC}} \rho \dashv \sigma$ *iff* $\rhd \rho \dashv \sigma$
ii) *Let $\rho, \sigma \in$ **rC**:* $\rhd_{\mathbf{rC}} \rho \dashv \sigma$ *iff* $\rhd \rho \dashv \sigma$

Proof. See Appendix A.2. □

From Proposition 2 and the soundness and completeness property of $\rhd_{\mathsf{SC}}$ and $\rhd_{\mathbf{rC}}$ (Theorems 7 and 8 in Appendix A.1) we immediately get what follows.

Corollary 3 (Compliances Conservativity).

i) *Let $\rho, \sigma \in$ SC:* $\rho \dashv\vdash_{\mathsf{SC}} \sigma$ *iff* $\rho \dashv\vdash \sigma$

ii) Let $\rho, \sigma \in \mathbf{rC}$: $\qquad \rho \dashv\vdash_{\mathbf{rC}} \sigma \quad iff \quad \rho \dashv\vdash \sigma$

A more direct proof of conservativity of $\dashv\vdash$ over $\dashv\vdash_{\mathsf{SC}}$, enabling to get a better insight of the differences of the compliance relations for the different formalisms, can be obtained by an analysis of the behaviors of reductions. Some care is however needed in such a case, since reductions can modify the stack even when we restrict ourselves to session contracts. This implies that, in a sequence of reductions out of a client/server system $\langle \rangle \Join \rho \parallel \langle \rangle \Join \sigma$ with $\rho, \sigma \in \mathsf{SC}$, also rollbacks can occur. In order to handle them, one has to show that, in a reduction sequence like the above, only particular stacks are actually produced, such that once a rollback procedure is started it necessarily goes on till a stuck state is reached. Details can be found in Appendix A.2.

4 Duality and the Subcontract Relation

Unlike the retractable session contracts of [7], in the present setting it is possible to get a natural notion of *duality*. The dual $\overline{\sigma}$ of an element σ of $\mathbf{rsC}$ is obtained, as for session contracts, by interchanging any name a with $\overline{a}$ and $+$ with $\oplus$.

Formally, we first define duality for (possibly open) contracts, that we dub $\mathbf{rsCo}$, and then we restrict such a definition to $\mathbf{rsC}$ (i.e., to closed expressions).

Definition 11 (Syntactic duality).

i) Let $\sigma \in \mathbf{rsCo}$. The syntactic dual $\overline{\sigma}$ of σ is defined by the following clauses:
$$\overline{\mathbf{1}} = \mathbf{1} \qquad \overline{x} = x \qquad \overline{\mathsf{rec}\, x.\sigma} = \mathsf{rec}\, x.\overline{\sigma}$$
$$\overline{\sum_{i \in I} \alpha_i.\sigma_i} = \bigoplus_{i \in I} \overline{\alpha}_i.\overline{\sigma}_i \qquad \overline{\bigoplus_{i \in I} \alpha_i.\sigma_i} = \sum_{i \in I} \overline{\alpha}_i.\overline{\sigma}_i$$

ii) We define $\overline{(\cdot)} : \mathbf{rsC} \to \mathbf{rsC}$ as the restriction to $\mathbf{rsC}$ of the duality function on $\mathbf{rsCo}$, observing that $\overline{\sigma} \in \mathbf{rsC}$ iff $\sigma \in \mathbf{rsC}$.

From now on, in order to avoid too cumbersome definitions, any time an inductive definition on elements of $\mathbf{rsC}$ is provided, it will be tacitly assumed to be actually the restriction to $\mathbf{rsC}$ of the corresponding inductive definition on $\mathbf{rsCo}$.

A first relevant property of duality is the following:

Proposition 3. *For any $\sigma \in \mathbf{rsC}$, $\quad \sigma \dashv\vdash \overline{\sigma}$.*

Proof. Since $\overline{\sigma}$ is obtained from σ by exchanging each α with $\overline{\alpha}$ and $+$ with $\oplus$, it is easy to get a derivation of $\rhd\ \overline{\sigma} \dashv\vdash \sigma$. The thesis is then an immediate consequence of soundness and completeness of $\rhd$.

The notion of dual contract allows one to combine pairs of contracts in the compliance relation, as follows:

Proposition 4. *For any $\rho, \sigma, \sigma' \in \mathbf{rsC}$, $\rho \dashv\vdash \sigma$ and $\overline{\sigma} \dashv\vdash \sigma'$ imply $\rho \dashv\vdash \sigma'$*

Proof. See Appendix A.4. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \square$

We will provide further properties of duality using the notion of subcontract relation. Indeed, the notion of compliance naturally induces a substitutability relation on servers, denoted $\preccurlyeq_s$, that we call *subcontract relation for servers*. Such a relation may be used for implementing contract-based query engines (see [27] for a detailed discussion). An analogous subcontract relation, denoted $\preccurlyeq_c$, can be defined for clients.

Definition 12 (Subcontract Relations for Servers and for Clients).
Let $\sigma, \sigma' \in \mathsf{rsC}$. We define

$$
\begin{aligned}
&i)\ \ \sigma \preccurlyeq_s \sigma' \ \ \triangleq \ \ \forall \rho \in \mathsf{rsC} \ [\, \rho \dashv\vdash \sigma \ \textit{implies} \ \rho \dashv\vdash \sigma' \,] \\
&ii)\ \ \sigma \preccurlyeq_c \sigma' \ \ \triangleq \ \ \forall \rho \in \mathsf{rsC} \ [\, \sigma \dashv\vdash \rho \ \textit{implies} \ \sigma' \dashv\vdash \rho \,]
\end{aligned}
$$

Using Proposition 4 we can characterize both $\preccurlyeq_s$ and $\preccurlyeq_c$ in terms of duality and compliance, relate them and getting their decidability.

Theorem 4. *For any $\sigma, \sigma' \in \mathsf{rsC}$:*

$$
\begin{aligned}
&i)\ \ \sigma \preccurlyeq_s \sigma' \quad \textit{iff} \quad \overline{\sigma} \dashv\vdash \sigma' \\
&ii)\ \ \sigma \preccurlyeq_c \sigma' \quad \textit{iff} \quad \sigma' \dashv\vdash \overline{\sigma} \\
&iii)\ \ \sigma \preccurlyeq_s \sigma' \quad \textit{iff} \quad \overline{\sigma'} \preccurlyeq_c \overline{\sigma} \\
&iv)\ \ \sigma \preccurlyeq_s \sigma' \ \textit{and} \ \sigma \preccurlyeq_c \sigma' \ \textit{are decidable.}
\end{aligned}
$$

Proof. (i) ($\Rightarrow$) By contraposition, assume that $\overline{\sigma} \not\dashv\vdash \sigma'$. Since $\overline{\sigma} \dashv\vdash \sigma$ by Proposition 3, then by definition of $\preccurlyeq_s$ we have $\sigma \not\preccurlyeq_s \sigma'$.
($\Leftarrow$) Let $\overline{\sigma} \dashv\vdash \sigma'$. If $\rho \dashv\vdash \sigma$, then from $\overline{\sigma} \dashv\vdash \sigma'$, we get $\rho \dashv\vdash \sigma'$ by Proposition 4, and therefore $\sigma \preccurlyeq_s \sigma'$ by definition.
(ii) ($\Rightarrow$) By contraposition, assume that $\sigma' \not\dashv\vdash \overline{\sigma}$. Since $\sigma \dashv\vdash \overline{\sigma}$ by Proposition 3, then by definition of $\preccurlyeq_c$ we have $\sigma \not\preccurlyeq_c \sigma'$.
($\Leftarrow$) Let $\sigma' \dashv\vdash \overline{\sigma}$. If $\sigma \dashv\vdash \rho$, then from $\sigma' \dashv\vdash \overline{\sigma}$, we get $\sigma' \dashv\vdash \rho$ by Proposition 4, and therefore $\sigma \preccurlyeq_c \sigma'$ by definition.
(iii) From Item (i) we have $\sigma \preccurlyeq_s \sigma'$ iff $\overline{\sigma} \dashv\vdash \sigma'$. From Item (ii) we have $\overline{\sigma'} \preccurlyeq_c \overline{\sigma}$ iff $\overline{\sigma} \dashv\vdash \overline{\overline{\sigma'}}$. The thesis follows since $\overline{\overline{\sigma'}} = \sigma'$.
(iv) From Items (i) and (ii) and decidability of $\overline{\sigma} \dashv\vdash \sigma'$. $\qquad\square$

By item iii) above, from now on we can simply concentrate on the relation $\preccurlyeq_s$.

Proposition 5 (Dual as a Least Element w.r.t. $\preccurlyeq_s$).
Let $\rho \in \mathsf{rsC}$. Then $\overline{\rho}$ is a least element in the set of the servers of ρ, that is,
$$
\rho \dashv\vdash \overline{\rho} \quad \textit{and} \quad (\forall \sigma \in \mathsf{rsC}\colon \ \rho \dashv\vdash \sigma \ \textit{implies} \ \overline{\rho} \preccurlyeq_s \sigma)
$$

Proof. Suppose that $\rho \dashv\vdash \sigma$ and take any contract τ such that $\tau \dashv\vdash \overline{\rho}$. Since $\overline{\overline{\rho}} = \rho$, by Proposition 4 we know that $\tau \dashv\vdash \sigma$; hence $\overline{\rho} \preccurlyeq_s \sigma$ by definition. $\qquad\square$

As done for the compliance relation, we characterize now the subcontract relation for servers in terms of derivability in the following formal system, where the symbol $\ll$ is used as syntactical counterpart of the relation $\preccurlyeq_s$.

$$\begin{array}{ccc}
\text{(Ax-}\preccurlyeq_s) & \text{(Hyp-}\preccurlyeq_s) & \text{(}\oplus\cdot+\text{-}\preccurlyeq_s) \\
\Gamma \blacktriangleright \mathbf{1} \ll \sigma' & \Gamma, \sigma \ll \sigma' \blacktriangleright \sigma \ll \sigma' & \dfrac{\Gamma, \alpha.\sigma_1 \oplus \sigma_2 \ll \alpha.\sigma'_1 + \sigma'_2 \blacktriangleright \sigma_1 \ll \sigma'_1}{\Gamma \blacktriangleright \alpha.\sigma_1 \oplus \sigma_2 \ll \alpha.\sigma'_1 + \sigma'_2}
\end{array}$$

$$\text{(}+\cdot+\text{-}\preccurlyeq_s)\quad \dfrac{\forall h \in I.\ \Gamma, \sum_{i \in I} \alpha_i.\sigma_i \ll \sum_{j \in I \cup J} \alpha_j.\sigma'_j \blacktriangleright \sigma_h \ll \sigma'_h}{\Gamma \blacktriangleright \sum_{i \in I} \alpha_i.\sigma_i \ll \sum_{j \in I \cup J} \alpha_j.\sigma'_j}$$

$$\text{(}\oplus\cdot\oplus\text{-}\preccurlyeq_s)\quad \dfrac{\forall h \in I.\ \Gamma, \bigoplus_{j \in I \cup J} \alpha_j.\sigma_j \ll \bigoplus_{i \in I} \alpha_i.\sigma'_i \blacktriangleright \sigma_h \ll \sigma'_h}{\Gamma \blacktriangleright \bigoplus_{j \in I \cup J} \alpha_j.\sigma'_j \ll \bigoplus_{i \in I} \alpha_i.\sigma'_i}$$

Fig. 7. The formal system $\blacktriangleright$

Definition 13 (Formal System for Subcontract $\blacktriangleright$). *Judgments in the formal system $\blacktriangleright$ are expressions of the form $\Gamma \blacktriangleright \rho \ll \sigma$, where the environment Γ is a finite set of expressions of the form $\delta \ll \gamma$, with $\rho, \sigma, \delta, \gamma \in \mathbf{rsC}$. Axioms and rules are as in Figure 7.*

The rules in system $\blacktriangleright$ can be read as a translation of the rules in system $\rhd$ via Theorem 4(i). As for $\rhd$, in $\Gamma \blacktriangleright \rho \ll \sigma$ we may drop Γ if empty.

Lemma 1. $\Gamma \blacktriangleright \sigma \ll \sigma' \quad iff \quad \widetilde{\Gamma} \rhd \overline{\sigma} \dashv \sigma'$
where $\Gamma = \{\sigma_i \ll \sigma'_i\}_{i \in I}$ and $\widetilde{\Gamma} = \{\overline{\sigma_i} \dashv \sigma'_i\}_{i \in I}$.

Proof. $(\Rightarrow)$ By induction over the derivation of $\Gamma \blacktriangleright \sigma \ll \sigma'$.
$(\Leftarrow)$ By induction over the derivation of $\widetilde{\Gamma} \rhd \sigma \dashv \sigma'$. $\qquad\square$

System $\blacktriangleright$ is sound and complete for the subcontract relation $\preccurlyeq_s$.

Theorem 5 (Soundness and Completeness of $\blacktriangleright$). $\blacktriangleright \sigma \ll \sigma' \quad iff \quad \sigma \preccurlyeq_s \sigma'$

Proof. $(\Rightarrow)$ Let $\blacktriangleright \sigma \ll \sigma'$. By Lemma 1 we get $\rhd \overline{\sigma} \dashv \sigma'$ and hence $\overline{\sigma} \dashv\vdash \sigma'$ by soundness of system $\rhd$. The thesis now descends from Theorem 4.
$(\Leftarrow)$ Let $\sigma \preccurlyeq_s \sigma'$. By Theorem 4 we have that $\overline{\sigma} \dashv\vdash \sigma'$. By completeness of system $\rhd$ we get $\rhd \overline{\sigma} \dashv \sigma'$. Now, by Lemma 1, we can obtain $\blacktriangleright \sigma \ll \sigma'$. $\qquad\square$

System $\blacktriangleright$ can be used to show that $\preccurlyeq_s$ is a partial order and hence, by antisymmetry, $\overline{\rho}$ is also the minimum server of ρ: it is minimal, hence there is no smaller server, and there is a unique minimal.

Proposition 6. $\preccurlyeq_s$ *is a partial order* $\land\ \forall \rho \in \mathbf{rsC},\ \overline{\rho}$ *is the minimum server of* ρ.

Proof (Sketch). We need to show $\preccurlyeq_s$ to be reflexive, transitive and antisymmetric. Reflexivity and transitivity immediately descend from the definition of $\preccurlyeq_s$ (Definition 12).

For the antisymmetric property, instead, we cannot rely directly on the definition of $\preccurlyeq_s$, since from $\sigma \preccurlyeq_s \sigma'$ and $\sigma' \preccurlyeq_s \sigma$ we can only infer that σ and σ'
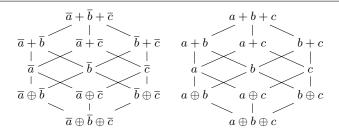
Fig. 8. Subcontract preorder: a sample

have the same set of clients. This does not trivially imply that $\sigma = \sigma'$.

We can proceed, instead, roughly as follows. Let $\sigma, \sigma' \in \mathbf{rsC}$ be such that $\sigma \preccurlyeq_s \sigma'$ and $\sigma' \preccurlyeq_s \sigma$. By completeness of $\blacktriangleright$ we get $\blacktriangleright \sigma \ll \sigma'$ and $\blacktriangleright \sigma' \ll \sigma$. By having such derivations, we can infer that in each of them no rule $(\oplus \cdot + \text{-} \preccurlyeq_s)$ can be present. Moreover, in each application of rule $(+ \cdot + \text{-} \preccurlyeq_s)$ or rule $(\oplus \cdot \oplus \text{-} \preccurlyeq_s)$, we have necessarily that $J = \emptyset$. Out of that we can infer $\sigma = \sigma'$.

For the second conjunct of the statement, suppose, towards a contradiction, that there exists $\sigma \neq \overline{\rho}$ such that $\rho \dashv\mid \sigma$ and $\sigma \preccurlyeq_s \overline{\rho}$. By Proposition 5 we have $\overline{\rho} \preccurlyeq_s \sigma$. By antisymmetry we have $\sigma = \overline{\rho}$, against the hypothesis. □

The structure of the partial order is shown in Figure 8, where the relations between terms with a unique choice among actions a, b, c, $\overline{a}$, $\overline{b}$ and $\overline{c}$ are pictured.

Remark 3. Analogously to what done in Subsect. 3.1, one can show the subcontract relation $\preccurlyeq_s$ to be a conservative extension of the corresponding notion in SC. Moreover, the restriction of $\preccurlyeq_s$ to $\mathbf{rC}$ provides a suitable notion of subcontract for $\mathbf{rC}$ (which has never been studied before).

5 Complexity Issues

The algorithm **Prove** in Figure 5 (and hence the decision procedure for compliance) is simple, but, as it is, its complexity is strictly exponential, as shown by the example below, where the exponential number of recursive calls of the decision procedure is actually reached. The example is an adaptation of the one presented in [19](§11) concerning the subtyping relation for recursive arrow and product types.

For each $n \in \mathbb{N}$ we define two contracts ρ_n and σ_n by induction, as follows.

$$\rho_0 = a + b \qquad \rho_{n+1} = \mathsf{rec}\, x.a.x + b.\rho_n$$
$$\sigma_0 = \mathsf{rec}\, x.\overline{a}.x \qquad \sigma_{n+1} = \overline{a}.\sigma_n \oplus \mathsf{rec}\, x.\overline{b}.x$$

As for the example in [19], the size of ρ_n and σ_n is linear in n, since ρ_n and σ_n appear just once in the definitions of ρ_{n+1} and σ_{n+1}, respectively. By complete induction over n it is possible to prove that, for any n, $\rho_n \dashv\mid \sigma_n$. The computation of **Prove**$(\emptyset \rhd \rho_n \dashv \sigma_n)$ builds a derivation for $\rhd \rho_n \dashv \sigma_n$ in an actual exponential

$$
\begin{array}{c}
(\mathrm{Ax}_\infty) \\
\rhd\ \mathbf{1} \dashv \sigma
\end{array}
\qquad\qquad
\begin{array}{c}
(+\cdot +_\infty) \\
\dfrac{\rho \dashv \sigma}{\alpha.\rho + \rho' \dashv \overline{\alpha}.\sigma + \sigma'}
\end{array}
$$

$$
\begin{array}{c}
(\oplus\cdot +_\infty) \\
\dfrac{\forall h \in I.\ \rho_h \dashv \sigma_h}{\bigoplus_{i\in I}\overline{\alpha}_i.\rho_i \dashv \sum_{j\in I\cup J}\alpha_j.\sigma_j}
\end{array}
\qquad\qquad
\begin{array}{c}
(+\cdot \oplus_\infty) \\
\dfrac{\forall h \in I.\ \rho_h \dashv \sigma_h}{\sum_{j\in I\cup J}\overline{\alpha}_j.\rho_j \dashv \bigoplus_{i\in I}\alpha_i.\sigma_i}
\end{array}
$$

Fig. 9. The non-well founded system $\rhd_\infty$

number of calls. Given n, the first part of the recursive-call tree looks as follows (where we abbreviate "**Prove**" by "**Pr**")

$$\mathbf{Pr}(\emptyset \rhd \rho_n \dashv \sigma_n)$$

$$\mathbf{Pr}(\Gamma_1 \rhd \rho_n \dashv \sigma_{n-1}) \qquad \mathbf{Pr}(\Gamma_2 \rhd \rho_{n-1} \dashv \sigma_n)$$

$$\mathbf{Pr}(\Gamma_3 \rhd \rho_n \dashv \sigma_{n-2})\ \underline{\mathbf{Pr}(\Gamma_4 \rhd \rho_{n-1} \dashv \sigma_{n-1})} \quad \underline{\mathbf{Pr}(\Gamma_5 \rhd \rho_{n-1} \dashv \sigma_{n-1})}\ \mathbf{Pr}(\Gamma_6 \rhd \rho_{n-2} \dashv \sigma_n)$$

$$\ldots\ldots\ \text{etc.}$$

where $\Gamma_4 = \{\rho_n \dashv \sigma_n, \rho_n \dashv \sigma_{n-1}\} \neq \{\rho_n \dashv \sigma_n, \rho_{n-1} \dashv \sigma_n\} = \Gamma_5$. So, any call of the shape $\mathbf{Pr}(\Gamma \rhd \rho_k \dashv \sigma_k)$ produces two calls $\mathbf{Pr}(\Gamma' \rhd \rho_{k-1} \dashv \sigma_{k-1})$ and $\mathbf{Pr}(\Gamma'' \rhd \rho_{k-1} \dashv \sigma_{k-1})$ with $\Gamma' \neq \Gamma''$; overall there are at least 2^n calls.

However, the complexity of the compliance decision procedure can be drastically reduced down to a polynomial complexity.

A polynomial decision algorithm. We first define a non-well founded version of system $\rhd$.

Definition 14 (The non-well founded system $\rhd_\infty$). *We write $\rhd_\infty \rho \dashv \sigma$ whenever there exists a finite or infinite derivation tree formed by the rules in Figure 9 having $\rho \dashv \sigma$ as conclusion, and such that each finite branch ends with an instance of axiom* (Ax_∞).

Because all expressions in the premises are subexpressions of those in the conclusion, and contracts are regular trees, in an infinite branch there must be at least a judgment occurring infinitely many times.

Lemma 2 (Systems $\rhd$ and $\rhd_\infty$ are equivalent). $\rhd \rho \dashv \sigma$ *iff* $\rhd_\infty \rho \dashv \sigma$

In Figure 10 we present a decision algorithm $\mathbf{Decide}_{\dashv}$, based on the procedures $\mathbf{P}$ and $\mathbf{P}^+$. A run of the proof reconstruction algorithm resembles a computation tree of an alternating Turing machine, where nodes corresponding to rules $(\oplus\cdot +_\infty)$ and $(+\cdot \oplus_\infty)$ are universal, nodes corresponding to $(+\cdot +_\infty)$ are existential; $\mathbf{P}(\mathsf{A},\mathsf{F},\mathsf{L},\mathsf{b})$ attempts to prove all statements in its goal list L, while $\mathbf{P}^+(\mathsf{A},\mathsf{F},\mathsf{L},\mathsf{b})$ succeeds if at least one goal in L is satisfiable.

The procedure $\mathbf{P}$ is an adaptation of the concrete subtyping algorithm for recursive arrow and product types of [19](§10) to the present, more complex

$\mathbf{Decide}_{\dashv\|}\,(\rho \dashv \sigma) = \mathbf{let}\quad (\mathsf{A},\mathsf{F},\mathsf{b}) = \mathbf{P}(\emptyset,\emptyset,[\rho \dashv \sigma],\mathbf{ok})$

$\qquad\qquad\qquad\qquad\quad \mathbf{in}\quad \mathsf{b} = \mathbf{ok}$

where

$\mathbf{P}(\mathsf{A},\mathsf{F},[\,],\mathsf{b}) = (\mathsf{A},\mathsf{F},\mathsf{b})$

$\mathbf{P}(\mathsf{A},\mathsf{F},(\rho \dashv \sigma)\!:\!\mathsf{xs},\mathsf{b}) =$

-1- $\quad\mathbf{if}\quad \rho = \mathbf{1}\quad \mathbf{then}\quad \mathbf{P}(\mathsf{A},\mathsf{F},\mathsf{xs},\mathsf{b})$

-2- $\qquad\mathbf{else\ if}\quad \rho \dashv \sigma \in \mathsf{A}\quad \mathbf{then}\quad \mathbf{P}(\mathsf{A},\mathsf{F},\mathsf{xs},\mathsf{b})$

-3- $\qquad\mathbf{else\ if}\quad \rho \dashv \sigma \in \mathsf{F}\quad \mathbf{then}\quad (\mathsf{A},\mathsf{F},\mathbf{fail})$

-4- $\qquad\mathbf{else\ if}\quad \rho = \sum_{i \in I} \alpha_i.\rho_i\ \ \mathbf{and}\ \ \sigma = \sum_{j \in J}\overline{\alpha}_j.\sigma_j\ \ \mathbf{and}\ \ I \cap J = \{i_1,\ldots,i_n\}$

-5- $\qquad\qquad\mathbf{then}\quad \mathbf{let}\ \ (\mathsf{A}_0,\mathsf{F}_0,\mathsf{b}_0) = \mathbf{P}^{+}(\mathsf{A}\cup\{\rho \dashv \sigma\},\mathsf{F},[\rho_{i_1} \dashv \sigma_{i_1}\ldots\rho_{i_n} \dashv \sigma_{i_n}],\mathsf{b})$

-6- $\qquad\qquad\qquad\qquad\mathbf{in}\ \ \mathbf{if}\ \ \mathsf{b}_0{=}\mathbf{fail}\ \ \mathbf{then}\ \ (\mathsf{A}_0,\mathsf{F}_0,\mathbf{fail})$

-7- $\qquad\qquad\qquad\qquad\qquad\qquad\mathbf{else}\ \ \mathbf{P}(\mathsf{A}_0,\mathsf{F}_0,\mathsf{xs},\mathsf{b}_0)$

-8- $\qquad\mathbf{else\ if}\quad \rho = \bigoplus_{i \in I}\overline{\alpha}_i.\rho_i\ \ \mathbf{and}\ \ \sigma = \sum_{j \in J}\alpha_j.\sigma_j\ \ \mathbf{and}\ \ I \subseteq J\ \ \mathbf{and}\ \ I = \{i_1,\ldots,i_n\}$

-9- $\qquad\qquad\mathbf{then}\quad \mathbf{let}\ \ (\mathsf{A}_0,\mathsf{F}_0,\mathsf{b}_0) = \mathbf{P}(\mathsf{A}\cup\{\rho \dashv \sigma\},\mathsf{F},[\rho_{i_1} \dashv \sigma_{i_1}\ldots\rho_{i_n} \dashv \sigma_{i_n}],\mathsf{b})$

-10- $\qquad\qquad\qquad\qquad\mathbf{in}\ \ \mathbf{if}\ \ \mathsf{b}_0{=}\mathbf{fail}\ \ \mathbf{then}\ \ (\mathsf{A}_0,\mathsf{F}_0,\mathbf{fail})$

-11- $\qquad\qquad\qquad\qquad\qquad\qquad\mathbf{else}\quad \mathbf{P}(\mathsf{A}_0,\mathsf{F}_0,\mathsf{xs},\mathsf{b}_0)$

-12- $\qquad\mathbf{else\ if}\quad \rho = \sum_{i \in I}\overline{\alpha}_i.\rho_i\ \ \mathbf{and}\ \ \sigma = \bigoplus_{j \in J}\alpha_j.\sigma_j\ \ \mathbf{and}\ \ I \supseteq J\ \ \mathbf{and}\ \ J = \{j_1,\ldots,j_n\}$

-13- $\qquad\qquad\mathbf{then}\quad \mathbf{let}\ \ (\mathsf{A}_0,\mathsf{F}_0,\mathsf{b}_0) = \mathbf{P}(\mathsf{A}\cup\{\rho \dashv \sigma\},\mathsf{F},[\rho_{i_1} \dashv \sigma_{i_1}\ldots\rho_{j_n} \dashv \sigma_{j_n}],\mathsf{b})$

-14- $\qquad\qquad\qquad\qquad\mathbf{in}\ \ \mathbf{if}\ \ \mathsf{b}_0{=}\mathbf{fail}\ \ \mathbf{then}\ \ (\mathsf{A}_0,\mathsf{F}_0,\mathbf{fail})$

-15- $\qquad\qquad\qquad\qquad\qquad\qquad\mathbf{else}\quad \mathbf{P}(\mathsf{A}_0,\mathsf{F}_0,\mathsf{xs},\mathsf{b}_0)$

-16- $\qquad\mathbf{else\ if}\quad \rho = \mathsf{rec}\,x.\rho'\quad \mathbf{then}\quad \mathbf{P}(\mathsf{A},\mathsf{F},(\{\mathsf{rec}\,x.\rho'/x\}\rho' \dashv \sigma)\!:\!\mathsf{xs},\mathsf{b})$

-17- $\qquad\mathbf{else\ if}\quad \sigma = \mathsf{rec}\,x.\sigma'\quad \mathbf{then}\quad \mathbf{P}(\mathsf{A},\mathsf{F},(\rho \dashv \{\mathsf{rec}\,x.\sigma'/x\}\sigma')\!:\!\mathsf{xs},\mathsf{b})$

-18- $\qquad\mathbf{else}\ \ (\mathsf{A},\mathsf{F} \cup \{\rho \dashv \sigma\},\mathbf{fail})$

and where

$\mathbf{P}^{+}(\mathsf{A},\mathsf{F},[\rho \dashv \sigma],\mathsf{b}) = \mathbf{P}(\mathsf{A},\mathsf{F},[\rho \dashv \sigma],\mathsf{b})$

$\mathbf{P}^{+}(\mathsf{A},\mathsf{F},(\rho \dashv \sigma)\!:\!\mathsf{xs},\mathsf{b})=$

-19- $\qquad\mathbf{let}\quad (\mathsf{A}_0,\mathsf{F}_0,\mathsf{b}_0) = \mathbf{P}(\mathsf{A},\mathsf{F},[\rho \dashv \sigma],\mathsf{b})\ \ \mathbf{in}$

-20- $\qquad\qquad\mathbf{if}\ \ \mathsf{b}_0 = \mathbf{fail}\ \ \mathbf{then}\ \ \mathbf{P}^{+}(\mathsf{A} \cup \mathsf{A}_0,\mathsf{F} \cup \mathsf{F}_0,\mathsf{xs},\mathbf{ok})$

-21- $\qquad\qquad\qquad\mathbf{else}\ \ (\mathsf{A}_0,\mathsf{F}_0,\mathsf{b}_0)$

Fig. 10. The polynomial decision procedure for compliance.

context. It consists of a proof reconstruction procedure for $\rhd_\infty$ using a depth-first technique. **P** accumulates in its first argument A all the judgments it encounters during the search, in order to avoid looping over the same judgments (a role similar to Γ in system $\rhd$). With respect to the algorithm in [19]($\S$10) we have two further parameters, F and b. The argument F accumulates the judgments for which it has been found that no derivation exists. When a rule $(+ \cdot +)$ is encountered, the algorithm proceeds by calling the procedure $\mathbf{P}^+$ which, in case a premise is unprovable, goes on checking the other premises. The negative information inferred about unprovable judgments is stored in F and it is carried along by the procedure $\mathbf{P}^+$ (as well as the positive information stored in A) in order not to duplicate work. The argument b, that can be either **ok** or **fail**, is used to record whether the last call was successful or not, and it is used by $\mathbf{P}^+$ to know whether it has to stop with success, or to check a new premise.

Let us note that, contrary to the previous treatment, while studying the algorithm $\mathbf{Decide}_{\dashv\vdash}$, we abandon the equi-recursive view of recursion, and we represent a contract by a particular explicit (possibly) recursive expression.

Proposition 7 (Complexity of Deciding Compliance). *Given two contracts $\rho, \sigma \in \mathbf{rsC}$, deciding whether $\rho \dashv\vdash \sigma$ has a complexity $\mathcal{O}(n^5)$, where n is the maximum size of ρ and σ.*

Proof. First, observe that:

(1) the recursive calls in lines -1-, -2-, -16-, -17-, -19- and, possibly, in lines -7-, -11-, -15-, -20- do leave unaltered the arguments A and F;
(2) in the other recursive calls the cardinality of $\mathsf{A} \cup \mathsf{F}$ strictly increases.

The number p of consecutive calls related to item 1 cannot be greater than the maximum branching of a node in the derivation tree we are trying to build, and this is bounded by the size n of the input. That is p is $\mathcal{O}(n)$. The number q of calls related to item 2, instead, is bounded by the cardinality of all the possible pairs of the subterms ρ' and σ' of, respectively, ρ and σ, that is q is $\mathcal{O}(n^2)$. This means that the overall number of calls is bounded by pq, that is $\mathcal{O}(n^3)$.

It remains to look at
  – the complexity of checking whether, given $\rho \dashv\vdash \sigma$ and $\rho' \dashv\vdash \sigma'$, the expression ρ represents the same regular tree as ρ', and σ the same as σ';
  – the complexity of checking the conditions $\rho \dashv\vdash \sigma \in \mathsf{A}$ and $\rho \dashv\vdash \sigma \in \mathsf{F}$.

The first one is $\mathcal{O}(n)$ as contracts are regular expressions. This implies that the second one is $\mathcal{O}(n^2)$. Since the above conditions are checked before each recursive call, the overall complexity is polynomial, and in particular $\mathcal{O}(n^5)$. $\qquad\square$

Corollary 4 (Complexity of Deciding Subcontract). *Given two contracts $\rho, \sigma \in \mathbf{rsC}$, deciding whether $\rho \preccurlyeq_s \sigma$ has a complexity in $\mathcal{O}(n^5)$, where n is the maximum size of ρ and σ.*

Proof. It follows from Proposition 7 using Theorem 4 to reduce the checking of subcontract to the checking of compliance. Note that building the dual of a given contract takes linear time. $\qquad\square$

Remark 4. The polynomial decision procedure **Decide**$_{\dashv\|}$ applies also to the formalism of retractable session contracts of [7]. In fact, the sound and complete formal system $\rhd_{\mathbf{rC}}$ (and the corresponding procedure **Prove**) for $\dashv\|_{\mathbf{rC}}$ (see Appendix A.1) is the restriction to elements of **rC** of the system in Figure 4. Obviously, when applied to elements of **rC**, the clauses -8- and -12- of **Decide**$_{\dashv\|}$ do not need to take into account the possibility of internal input choices.

6   Retractable Contracts vs Reversible Computing

In this section we explore the relations between our retractable contracts and calculi for reversible computation (see [25] for an overview). In [28], Phillips and Ulidowski provide an automatic technique to derive, from the forward semantics of a given calculus, its reversible semantics. In principle, we would like to start from the forward calculus underlying our retractable contracts, that is from retractable contracts equipped with the semantics obtained by replacing a and $\overline{a}$ with α in the semantics of session contracts (see Definition 20 in Appendix A.1).

Definition 15 (Retractable Contracts Underlying Semantics).

$$\alpha.\sigma \oplus \sigma' \xrightarrow{\tau}_U \alpha.\sigma \qquad \alpha.\sigma \xrightarrow{\alpha}_U \sigma \qquad \alpha.\sigma + \sigma \xrightarrow{\alpha}_U \sigma$$

$$\frac{\rho \xrightarrow{\tau}_U \rho'}{\rho \parallel \sigma \longrightarrow_U \rho' \parallel \sigma} \qquad \frac{\sigma \xrightarrow{\tau}_U \sigma'}{\rho \parallel \sigma \longrightarrow_U \rho \parallel \sigma'} \qquad \frac{\rho \xrightarrow{\alpha}_U \rho' \qquad \sigma \xrightarrow{\overline{\alpha}}_U \sigma'}{\rho \parallel \sigma \longrightarrow_U \rho' \parallel \sigma'}$$

However, the technique in [28] requires the LTS of the forward semantics to satisfy a number of conditions, and the LTS in Definition 15 does not satisfy them. Thus, in order to apply the technique, we transform the syntax and the semantics of our forward calculus as follows:

- we merge the two levels of syntax and semantics (contracts and client/server pairs) into one;
- we transform internal choice into τ-prefixed external choice;
- we separate action prefixing from internal/external choice.

The syntax of the resulting calculus, that we dub TC (transformed contracts), is:

$$\sigma := \alpha_\tau.\sigma \;\mid\; \sum_{i \in I} \sigma_i \;\mid\; x \;\mid\; \mathsf{rec}\, x.\sigma \;\mid\; \sigma \parallel \sigma' \;\mid\; \mathbf{1}$$

where α_τ denotes a, $\overline{a}$ or τ.

We use $[\![\cdot]\!]$ to denote the translation of either a contract σ or a client/server pair $\sigma \parallel \rho$ into the syntax above.

Definition 16 (Translation function). *The translation function* $[\![\cdot]\!] : \mathbf{rsCo} \cup \mathbf{rsCo} \times \mathbf{rsCo} \longrightarrow \mathsf{TC}$ *is defined inductively as follows:*

$$[\![\textstyle\sum_{i \in I} \alpha_i.\sigma_i]\!] = \textstyle\sum_{i \in I} \alpha_i.[\![\sigma_i]\!] \qquad [\![x]\!] = x \qquad [\![\mathsf{rec}\, x.\sigma]\!] = \mathsf{rec}\, x.[\![\sigma]\!]$$

$$[\![\textstyle\bigoplus_{i \in I} \alpha_i.\sigma_i]\!] = \textstyle\sum_{i \in I} \tau.\alpha_i.[\![\sigma_i]\!] \qquad [\![\mathbf{1}]\!] = \mathbf{1} \qquad [\![\sigma \parallel \rho]\!] = [\![\sigma]\!] \parallel [\![\rho]\!]$$

Transformed contracts are more general than our contracts, allowing for general parallel composition and mixed choice, however the restriction of transformed contracts to the translation of closed contracts via function $[\![\cdot]\!]$ is closed under reduction, as shown by the semantics below. Thus, from now on we consider only such transformed contracts.

Definition 17 (Semantics of Transformed Contracts).

$$\alpha_\tau.\sigma \xrightarrow{\alpha_\tau} \sigma \qquad \frac{\sigma \xrightarrow{\alpha_\tau} \sigma'}{\sigma + \rho \xrightarrow{\alpha_\tau} \sigma'}$$

$$\frac{\rho \xrightarrow{\tau} \rho'}{\rho \parallel \sigma \xrightarrow{\tau} \rho' \parallel \sigma} \qquad \frac{\rho \xrightarrow{\alpha} \rho' \quad \sigma \xrightarrow{\overline{\alpha}} \sigma'}{\rho \parallel \sigma \xrightarrow{\tau} \rho' \parallel \sigma'}$$

Symmetric rules have been omitted.

It is easy to check that the LTS for transformed contracts and the LTS underlying retractable contracts model the same client/server interactions.

Proposition 8. *Let* $\sigma, \rho, \sigma', \rho' \in \mathsf{rsC}$.

$$\sigma \parallel \rho \longrightarrow_U \sigma' \parallel \rho' \quad \textit{iff} \quad [\![\sigma \parallel \rho]\!] \xrightarrow{\tau} [\![\sigma' \parallel \rho']\!]$$

Proof. By inspection of the rules. $\qquad\qquad\qquad\qquad\qquad\square$

One can apply to the LTS in Definition 17 the technique in [28], obtaining the LTS below. In order to simplify the treatment, we replaced the keys used in [28] to annotate actions with an underline. While this is not correct in general, this is correct in the image of our contracts, since keys are used to distinguish interactions with different communication partners, but in our case for each action there is at most one possible partner.

Definition 18 (Reversible Transformed Contracts: Forward Rules).

$$\frac{\mathrm{std}(X)}{\alpha_\tau.X \xrightarrow{\alpha_\tau} \underline{\alpha_\tau}.X} \qquad \frac{X \xrightarrow{\alpha_\tau} X'}{\underline{\alpha_\tau}.X \xrightarrow{\alpha_\tau} \underline{\alpha_\tau}.X'} \qquad \frac{X \xrightarrow{\alpha_\tau} X' \quad \mathrm{std}(Y)}{X + Y \xrightarrow{\alpha_\tau} X' + Y}$$

$$\frac{X \xrightarrow{\tau} X'}{X \parallel Y \xrightarrow{\tau} X' \parallel Y} \qquad \frac{X \xrightarrow{\alpha} X' \quad Y \xrightarrow{\overline{\alpha}} Y'}{X \parallel Y \xrightarrow{\tau} X' \parallel Y'}$$

Symmetric rules have been omitted, and $\mathrm{std}(X)$ *holds if* X *does not contain underlined prefixes.*

Backward rules, denoted by arrow $\rightsquigarrow$, can be obtained simply by changing the direction of the arrows in the rules above.

In Figure 11 we show an example of how a reduction sequence in our reversible contract formalism does correspond to a reduction sequence for transformed contracts. We formalize the correspondence hinted at in the example by providing a definition of simulation between a contract with history and its encoding into reversible transformed contracts:

$[\![a.c + b.d]\!] = a.c + b.d$		$[\![\overline{a}.\overline{c} + \overline{b}.(e \oplus f)]\!] = \overline{a}.\overline{c} + \overline{b}.(\tau.e + \tau.f)$

	$\langle\,\rangle \ltimes a.c + b.d \;\;\|\;\; \langle\,\rangle \ltimes \overline{a}.\overline{c} + \overline{b}.(e \oplus f)$		$a.c + b.d \;\;\|\;\; \overline{a}.\overline{c} + \overline{b}.(\tau.e + \tau.f)$
$\longrightarrow$	$a.c \ltimes d \;\;\|\;\; \overline{a}.\overline{c} \ltimes e \oplus f$	$\longrightarrow$	$a.c + \underline{b}.d \;\;\|\;\; \overline{a}.\overline{c} + \overline{\underline{b}}.(\tau.e + \tau.f)$
$\longrightarrow$	$a.c \ltimes d \;\;\|\;\; \overline{a}.\overline{c} \ltimes e$	$\longrightarrow$	$a.c + \underline{b}.d \;\;\|\;\; \overline{a}.\overline{c} + \overline{\underline{b}}.(\underline{\tau}.e + \tau.f)$
		$\rightsquigarrow$	$a.c + \underline{b}.d \;\;\|\;\; \overline{a}.\overline{c} + \overline{\underline{b}}.(\tau.e + \tau.f)$
$\overset{\text{rb}}{\longrightarrow}$	$\langle\,\rangle \ltimes a.c \;\;\|\;\; \langle\,\rangle \ltimes \overline{a}.\overline{c}$	$\rightsquigarrow$	$a.c + b.d \;\;\|\;\; \overline{a}.\overline{c} + \overline{b}.(\tau.e + \tau.f)$
$\longrightarrow$	$\circ \ltimes c \;\;\|\;\; \circ \ltimes \overline{c}$	$\longrightarrow$	$\underline{a}.c + b.d \;\;\|\;\; \overline{\underline{a}}.\overline{c} + \overline{b}.(\tau.e + \tau.f)$
$\longrightarrow$	$\circ\!:\!\circ \ltimes \mathbf{1} \;\;\|\;\; \circ\!:\!\circ \ltimes \mathbf{1}$	$\longrightarrow$	$\underline{a}.\underline{c} + b.d \;\;\|\;\; \overline{\underline{a}}.\overline{\underline{c}} + \overline{b}.(\tau.e + \tau.f)$

Fig. 11. Contracts with history vs transformed contracts: an example

Definition 19. *Let R be a relation between contracts with history and reversible transformed contracts. R is a simulation iff for each $(\mathrm{H} \ltimes \sigma, X) \in R$:*

- *if $\mathrm{H} \ltimes \sigma \longrightarrow \mathrm{H}' \ltimes \sigma'$ with a forward move then $X \overset{\tau}{\longrightarrow} X'$ and $(\mathrm{H}' \ltimes \sigma', X') \in R$;*
- *if $\mathrm{H} \ltimes \sigma \longrightarrow \mathrm{H}' \ltimes \sigma'$ with a backward move then $X \overset{\tau}{\rightsquigarrow}^+ X'$ and $(\mathrm{H}' \ltimes \sigma', X') \in R$;*

where $\overset{\tau}{\rightsquigarrow}^+$ is the transitive closure of $\overset{\tau}{\rightsquigarrow}$.

Theorem 6. *For each $\sigma \in \mathbf{rsC}$ there is a simulation R such that $(\langle\,\rangle \ltimes \sigma, [\![\sigma]\!]) \in R$.*

Proof (Sketch). The definition of the relation R is quite convoluted, hence we will not spell out it here, but we present the main ideas below. Essentially, one starts from the inverse of the translation function $[\![\cdot]\!]$, keeping into account that:

- underlined actions are dropped;
- for branches in choices where another branch contains underlined actions two possibilities have to be considered: either the branch is dropped (corresponding to paths that have been executed and discarded, or directly discarded by an unretractable choice), or it is moved to the history (corresponding to paths starting from a retractable choice which have not been tried yet).

It is easy to check that such a relation is a simulation. □

Note that the opposite of Theorem 6 cannot hold because of the mechanisms to control reversibility discussed in Remark 1. Indeed, the technique in [28] generates an uncontrolled semantics. A sample difference is that in transformed contracts we can have an infinite reduction sequence persistently choosing the right branch after the backward reduction, as follows.

$$a.c + b.d \;\;\|\;\; \overline{a}.\overline{c} + \overline{b}.(\tau.e + \tau.f) \;\longrightarrow^+\; a.c + \underline{b}.d \;\;\|\;\; \overline{a}.\overline{c} + \overline{\underline{b}}.(\underline{\tau}.e + \tau.f)$$
$$\rightsquigarrow^+ \; a.c + b.d \;\;\|\;\; \overline{a}.\overline{c} + \overline{b}.(\tau.e + \tau.f) \;\longrightarrow^+\; a.c + \underline{b}.d \;\;\|\;\; \overline{a}.\overline{c} + \overline{\underline{b}}.(\underline{\tau}.e + \tau.f)$$
$$\rightsquigarrow^+ \; a.c + b.d \;\;\|\;\; \overline{a}.\overline{c} + \overline{b}.(\tau.e + \tau.f) \;\longrightarrow^+\; \text{etc.}$$

It is easy to check, instead, that $\langle\,\rangle \ltimes a.c + b.d \;\|\; \langle\,\rangle \ltimes \overline{a}.\overline{c} + \overline{b}.(e \oplus f)$ can perform no infinite reduction sequence since the chosen branch is discarded upon rollback.

7   Related Work and Conclusion

We have presented two conservative extensions of the session contracts of [2,3,9], a formalism interpreting session types [21] into a subset of contracts [13,26,14]. One extension deals with backtracking and one with speculative execution. We have shown that they both give rise to *the same* compliance relation, and, as a consequence, to the same subcontract (both for servers and for clients) and duality relations. For each of these relations we provided syntactic characterizations of the semantic concepts, allowing for efficient ways of checking them.

We discussed in the Introduction the improvements w.r.t. the preliminary results about retractable session contracts in [7]. Another closely related work is [5,6], where a different form of contracts with rollback is presented. Our retractable contracts depart from that model on three main aspects: (1) we use rollback in a disciplined way to tolerate failures in the interaction (in [5,6] it is an internal decision of a partecipant), thus improving compliance; (2) we embed checkpoints in the structure of contracts, avoiding explicit checkpoints; (3) we keep a stack of "pasts", instead of just a single past as in [5,6].

Reversibility, generalizing backtracking by allowing one to go back to any past state, has also been studied in the setting of binary session types [31,32]. There however the emphasis is on defining the reversible engine, based on causal-consistent reversibility [25], and not on studying compliance or subtyping (which would correspond to our subcontract relation).

Similarly to our retractable contracts, long running transactions with compensations, and in particular interacting transactions [17], allow one to undo past agreements. In interacting transactions, however, abort (which corresponds to our backtracking) can occur at any time, not only when an agreement cannot be found as in our case. Also, each transaction offers just two possibilities, and they are sorted: first the normal execution, then the compensation. Finally, compliance of interacting transactions has never been studied.

In [4] a game-theoretical interpretation of the retractable session contracts of [7] has been provided. Such an interpretation is likely to extend to the retractable contracts presented here.

We plan also to investigate whether our approach can be extended to multi-party sessions [22]. An investigation of multi-party sessions with rollbacks and named checkpoints has been already undertaken in [18]. In such a paper, however, the cause of a rollback is not a synchronization failure, but it is completely transparent to the calculus. Moreover, chosen branches are not discarded and can be retried upon rollback.

Because of the relevance of higher-order features in type systems, and of session delegation in type systems with sessions in particular, also higher-order session contracts, i.e. session contracts with delegation, have been investigated [3,10]. It is hence worth studying the integration of backtracking (or speculative execution) and session delegation.

A last line of future work is the study of how to extract retractable or speculative contracts from actual software based on backtracking or on speculative parallelism, and how to propagate the results on contracts to the original software.

References

1. A. Avizienis, J.-C. Laprie, B. Randell, and C. E. Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Trans. Dep. Sec. Comput.*, 1(1):11–33, 2004.
2. F. Barbanera and U. de'Liguoro. Two notions of sub-behaviour for session-based client/server systems. In *PPDP*, pages 155–164. ACM Press, 2010.
3. F. Barbanera and U. de'Liguoro. Sub-behaviour relations for session-based client/server systems. *MSCS*, 25(6):1339–1381, 2015.
4. F. Barbanera and U. de'Liguoro. A game interpretation of retractable contracts. In *COORDINATION*, volume 9686 of *LNCS*. Springer, 2016.
5. F. Barbanera, M. Dezani-Ciancaglini, and U. de'Liguoro. Compliance for reversible client/server interactions. In *BEAT*, volume 162 of *EPTCS*, pages 35–42, 2014.
6. F. Barbanera, M. Dezani-Ciancaglini, and U. de'Liguoro. Reversible client/server interactions. *Formal Asp. Comput.*, 28(4):697–722, 2016.
7. F. Barbanera, M. Dezani-Ciancaglini, I. Lanese, and U. de'Liguoro. Retractable contracts. In *PLACES 2015*, volume 203 of *EPTCS*, pages 61–72. Open Publishing Association, 2016.
8. M. Bartoletti et al. Compliance and subtyping in timed session types. In *FORTE*, volume 9039 of *LNCS*, pages 161–177. Springer, 2015.
9. G. T. Bernardi and M. Hennessy. Modelling session types using contracts. *Mathematical Structures in Computer Science*, 26(3):510–560, 2016.
10. G. T. Bernardi and M. Hennessy. Using higher-order contracts to model session types. *Logical Methods in Computer Science*, 12(2), 2016.
11. M. Carbone, K. Honda, and N. Yoshida. Structured interactional exceptions in session types. In *CONCUR*, volume 5201 of *LNCS*, pages 402–417. Springer, 2008.
12. C. D. Carothers, K. S. Perumalla, and R. Fujimoto. Efficient optimistic parallel simulations using reverse computation. *ACM Trans. Model. Comput. Simul.*, 9(3):224–253, 1999.
13. S. Carpineti, G. Castagna, C. Laneve, and L. Padovani. A formal account of contracts for Web Services. In *WS-FM*, number 4184 in LNCS, pages 148–162. Springer, 2006.
14. G. Castagna, N. Gesbert, and L. Padovani. A theory of contracts for web services. *ACM Trans. on Prog. Lang. and Sys.*, 31(5):19:1–19:61, 2009.
15. M. Dalla Preda et al. Graceful interruption of request-response service interactions. In *ICSOC*, volume 7084 of *LNCS*, pages 590–600. Springer, 2011.
16. V. Danos and J. Krivine. Reversible communicating systems. In *CONCUR*, volume 3170 of *LNCS*, pages 292–307. Springer, 2004.
17. E. de Vries, V. Koutavas, and M. Hennessy. Communicating transactions - (extended abstract). In *CONCUR*, volume 6269 of *LNCS*, pages 569–583. Springer, 2010.
18. M. Dezani-Ciancaglini and P. Giannini. Reversible multiparty sessions with checkpoints. In *EXPRESS/SOS'16*, volume 222 of *EPTCS*, pages 60–74, 2016.
19. V. Gapeyev, M. Y. Levin, and B. C. Pierce. Recursive subtyping revealed. *J. Funct. Program.*, 12(6):511–548, 2002.
20. C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.
21. K. Honda, V. T. Vasconcelos, and M. Kubo. Language primitives and type disciplines for structured communication-based programming. In *ESOP*, volume 1381 of *LNCS*, pages 22–138. Springer, 1998.
22. K. Honda, N. Yoshida, and M. Carbone. Multiparty asynchronous session types. In *POPL*, pages 273–284. ACM Press, 2008.

23. H. Hüttel et al. Foundations of session types and behavioural contracts. *ACM Comput. Surv.*, 49(1):3:1–3:36, 2016.
24. I. Lanese, C. A. Mezzina, and J.-B. Stefani. Reversibility in the higher-order π-calculus. *Theor. Comput. Sci.*, 625:25–84, 2016.
25. I. Lanese, C. A. Mezzina, and F. Tiezzi. Causal-consistent reversibility. *Bulletin of the EATCS*, 114, 2014.
26. C. Laneve and L. Padovani. The must preorder revisited: An algebraic theory for web services contracts. In *CONCUR*, volume 4703 of *LNCS*, pages 212–225. Springer, 2007.
27. L. Padovani. Contract-based discovery of web services modulo simple orchestrators. *Theoretical Computer Science*, 411:3328–3347, 2010.
28. I. C. C. Phillips and I. Ulidowski. Reversing algebraic process calculi. *J. of Logic and Alg. Progr.*, 73(1-2):70–96, 2007.
29. P. Prabhu, G. Ramalingam, and K. Vaswani. Safe programmable speculative parallelism. In *PLDI*, pages 50–61. ACM, 2010.
30. C. G. Quiñones et al. Mitosis compiler: An infrastructure for speculative threading based on pre-computation slices. In *PLDI*, pages 269–279. ACM, 2005.
31. F. Tiezzi and N. Yoshida. Towards reversible sessions. In *PLACES*, volume 155 of *EPTCS*, pages 17–24, 2014.
32. F. Tiezzi and N. Yoshida. Reversible session-based pi-calculus. *J. Log. Algebr. Meth. Program.*, 84(5):684–707, 2015.
33. L. Xu, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Satzilla: Portfolio-based algorithm selection for SAT. *J. Artif. Intell. Res. (JAIR)*, 32:565–606, 2008.
34. L. Zhang and S. Malik. The quest for efficient boolean satisfiability solvers. In *CAV*, volume 2404 of *LNCS*, pages 17–36. Springer, 2002.

A Appendix: Proofs

A.1 Session Contracts and Retractable Session Contracts

Session contracts, a formalism interpreting session types [21] into a subset of contracts [13,26,14], have been introduced in [3,9].

The set SC of session contracts can be seen as the subset of elements in **rsC** not containing external output choices and internal input choices, with the following operational semantics.

Definition 20 (Semantics of Session Contracts).

$$\overline{a}.\sigma \oplus \sigma' \xrightarrow{\;\tau\;}_{\mathsf{SC}} \overline{a}.\sigma \qquad\qquad \alpha.\sigma \xrightarrow{\;\alpha\;}_{\mathsf{SC}} \sigma \qquad\qquad a.\sigma + \sigma \xrightarrow{\;a\;}_{\mathsf{SC}} \sigma$$

As done for **rsC**, we can look at session contracts up-to unfolding of recursion.

The next definitions introduce the LTS for client/server pairs of session contracts, and the corresponding compliance relation.

Definition 21 (Semantics of Client/Server Pairs of Session Contracts).

$$\frac{\rho \xrightarrow{\;\tau\;}_{\mathsf{SC}} \rho'}{\rho\|\sigma \longrightarrow_{\mathsf{SC}} \rho'\|\sigma} \qquad\qquad \frac{\sigma \xrightarrow{\;\tau\;}_{\mathsf{SC}} \sigma'}{\rho\|\sigma \longrightarrow_{\mathsf{SC}} \rho\|\sigma'} \qquad\qquad \frac{\rho \xrightarrow{\;\alpha\;}_{\mathsf{SC}} \rho' \quad \sigma \xrightarrow{\;\overline{\alpha}\;}_{\mathsf{SC}} \sigma'}{\rho\|\sigma \longrightarrow_{\mathsf{SC}} \rho'\|\sigma'}$$

$$\begin{array}{ll}
\text{(Ax)} & \text{(Hyp)} \\
\Gamma \vartriangleright_{\mathsf{SC}} \mathbf{1} \mathbin{\prec\!\!\!\vdash} \sigma & \Gamma, \rho \mathbin{\prec\!\!\!\vdash} \sigma \ \vartriangleright_{\mathsf{SC}} \rho \mathbin{\prec\!\!\!\vdash} \sigma
\end{array}$$

$$\begin{array}{c}
(\oplus \cdot +) \\
\dfrac{\forall h \in I. \ \Gamma, \bigoplus_{i \in I} \overline{a}_i.\rho_i \mathbin{\prec\!\!\!\vdash} \sum_{j \in I \cup J} a_j.\sigma_j \ \vartriangleright_{\mathsf{SC}} \rho_h \mathbin{\prec\!\!\!\vdash} \sigma_h}{\Gamma \ \vartriangleright_{\mathsf{SC}} \bigoplus_{i \in I} \overline{a}_i.\rho_i \mathbin{\prec\!\!\!\vdash} \sum_{j \in I \cup J} a_j.\sigma_j}
\end{array}$$

$$\begin{array}{c}
(+ \cdot \oplus) \\
\dfrac{\forall h \in I. \ \Gamma, \sum_{j \in I \cup J} a_j.\rho_j \mathbin{\prec\!\!\!\vdash} \bigoplus_{i \in I} \overline{a}_i.\sigma_i \ \vartriangleright_{\mathsf{SC}} \rho_h \mathbin{\prec\!\!\!\vdash} \sigma_h}{\Gamma \ \vartriangleright_{\mathsf{SC}} \sum_{j \in I \cup J} a_j.\rho_j \mathbin{\prec\!\!\!\vdash} \bigoplus_{i \in I} \overline{a}_i.\sigma_i}
\end{array}$$

Fig. 12. System $\vartriangleright_{\mathsf{SC}}$

Definition 22 (Compliance Relation $\dashv\!\!\vdash_{\mathsf{SC}}$ for Session Contracts).
The relation $\dashv\!\!\vdash_{\mathsf{SC}} \subset \mathsf{SC} \times \mathsf{SC}$ is defined by:
$\rho \dashv\!\!\vdash_{\mathsf{SC}} \sigma \quad$ *if, for each* $\quad \rho', \sigma' \quad$ *such that*
$$\rho \parallel \sigma \xrightarrow{\ *\ }_{\mathsf{SC}} \rho' \parallel \sigma' \nrightarrow_{\mathsf{SC}} \ \text{we have } \rho' = \mathbf{1}$$

The sound and complete formal system $\vartriangleright_{\mathsf{SC}}$ for $\dashv\!\!\vdash_{\mathsf{SC}}$ is recalled in Figure 12.

Theorem 7. *Let $\rho, \sigma \in \mathsf{SC}$. $\rho \dashv\!\!\vdash_{\mathsf{SC}} \sigma$ iff $\vartriangleright_{\mathsf{SC}} \rho \mathbin{\prec\!\!\!\vdash} \sigma$*

Proof. In [3] a formal system $\vartriangleright_{\mathsf{H}}$ is devised which is sound and complete for compliance of higher-order session contracts (HSC), that is

$$\vartriangleright_{\mathsf{H}} \rho \mathbin{\prec\!\!\!\vdash} \sigma \ \text{ iff } \ \rho \dashv\!\!\vdash_{\mathsf{H}} \sigma$$

The set SC can be looked at as the first-order restriction of HSC. It is easy to show that, for $\rho, \sigma \in \mathsf{SC}$,

$$\vartriangleright_{\mathsf{H}} \rho \mathbin{\prec\!\!\!\vdash} \sigma \ \text{ iff } \ \vartriangleright_{\mathsf{SC}} \rho \mathbin{\prec\!\!\!\vdash} \sigma$$

It is also not difficult to show that, for $\rho, \sigma \in \mathsf{SC}$,

$$\rho \dashv\!\!\vdash_{\mathsf{SC}} \sigma \ \text{ iff } \ \rho \dashv\!\!\vdash_{\mathsf{H}} \sigma$$

From the above statement, the thesis descends immediately. $\qquad\square$

Retractable session contracts were introduced in [7]. The set $\mathbf{rC}$ of retractable session contracts can be seen as the subset of elements in $\mathbf{rsC}$ not containing internal input choices. The operational semantics of retractable session contracts is the restriction to elements in $\mathbf{rC}$ of the semantics defined in Definitions 3 and 4. The sound and complete formal system $\vartriangleright_{\mathbf{rC}}$ for $\dashv\!\!\vdash_{\mathbf{rC}}$ is the restriction to elements of $\mathbf{rC}$ of the system in Figure 4.

Theorem 8 (From [7]). *Let $\rho, \sigma \in \mathbf{rC}$. $\rho \dashv\!\!\vdash_{\mathbf{rC}} \sigma$ iff $\vartriangleright_{\mathbf{rC}} \rho \mathbin{\prec\!\!\!\vdash} \sigma$*

A.2 Conservativity Proofs

Before proving that the operational semantics of retractable and speculative contracts is a conservative extension of the operational semantics of session contracts, we need a simple technical lemma and a fact.

Lemma 3. *Let $\rho \in \mathsf{SC}$. Then either $\rho = \overline{a}.\rho_1 \oplus \rho_2$, or $\rho = \alpha.\rho'$, or $\rho = a.\rho_1 + \rho_2$, or $\rho = \mathbf{1}$. Moreover,*

i) $\overline{a}.\rho_1 \oplus \rho_2 \xrightarrow{\tau} \overline{a}.\rho_1$ if and only if $\mathrm{H} \ltimes \overline{a}.\rho_1 \oplus \rho_2 \xrightarrow{\tau} \mathrm{H} \ltimes \overline{a}.\rho_1$;
ii) $\alpha.\rho' \xrightarrow{\alpha} \rho'$ if and only if $\mathrm{H} \ltimes \alpha.\rho' \xrightarrow{\alpha} \mathrm{H}{:}\circ \ltimes \rho'$;
iii) $a.\rho_1 + \rho_2 \xrightarrow{a} \rho_1$ if and only if $\mathrm{H} \ltimes a.\rho_1 + \rho_2 \xrightarrow{a} \mathrm{H}: \rho_2 \ltimes \rho_1$
iv) $\rho = \mathbf{1}$ if and only if $(\mathrm{H} \ltimes \rho \xotimes{\alpha} $ and $\mathrm{H} \ltimes \rho \xotimes{\tau})$.

Proof. Easy, by definition of session contract and by Definitions 4 and 20. □

Fact 2 *Let $\rho, \sigma \in \mathsf{SC}$. $\mathrm{H}_1 \ltimes \rho \parallel \mathrm{H}_2 \ltimes \sigma \longrightarrow_f \mathrm{H}'_1 \ltimes \rho' \parallel \mathrm{H}'_2 \ltimes \sigma'$ implies $\rho', \sigma' \in \mathsf{SC}$*

Both the retractable operational semantics and the speculative operational semantics of contracts are conservative extension of the operational semantics of session contracts SC in the following sense:

Proposition 1 Let $\rho, \sigma \in \mathsf{SC}$.
i) $\rho \parallel \sigma \xrightarrow{*}_{\mathsf{SC}} \rho' \parallel \sigma'$ iff $\mathrm{H}_1 \ltimes \rho \parallel \mathrm{H}_2 \ltimes \sigma \xrightarrow{*}_f \mathrm{H}'_1 \ltimes \rho' \parallel \mathrm{H}'_2 \ltimes \sigma'$
for some $\mathrm{H}_1, \mathrm{H}_2, \mathrm{H}'_1$ and H'_2

ii) $\rho \parallel \sigma \xrightarrow{*}_{\mathsf{SC}} \rho' \parallel \sigma'$ iff $\rho \parallel \sigma \xrightarrow{*} \alpha_1 @ \ldots \alpha_n @ \rho' \mid \mathbf{C}_\rho \parallel \overline{\alpha_1} @ \ldots \overline{\alpha_n} @ \sigma' \mid \mathbf{C}_\sigma$
for some $n, \alpha_1, \ldots, \alpha_n, \mathbf{C}_\rho$ and $\mathbf{C}_\sigma$

Proof. (i) The inclusion $\mathsf{SC} \subsetneq \mathbf{rsC}$ holds by definition. Hence, given $\rho, \sigma \in \mathsf{SC}$ we have $\rho, \sigma \in \mathbf{rsC}$ and $\mathrm{H}_1 \ltimes \rho, \mathrm{H}_2 \ltimes \sigma \in \mathbf{rsCH}$.

($\Rightarrow$) By induction on the length of the reduction sequence $\xrightarrow{*}_{\mathsf{SC}}$, using Definitions 4 and 21 and Lemma 3 to check all the possible cases for the reductions of client/server pairs.

($\Leftarrow$) Using Fact 2 we first show that no pair of the form $\mathrm{H}_1 \ltimes \alpha.\rho_1 + \rho_2 \parallel \mathrm{H}_2 \ltimes \overline{\alpha}.\sigma_1 + \sigma_2$ can ever appear inside the reduction sequence $\xrightarrow{*}_f$. Then we proceed by induction on the length of the reduction sequence $\xrightarrow{*}_f$ using Definitions 4 and 21 and Lemma 3 to check all the possible cases for the reductions of client/server pairs with histories.

(ii) By induction on the length of the derivation. The base case is trivial. Let us consider the inductive case.

($\Rightarrow$) the actions performed by σ' and ρ' can be performed as well by $\alpha_1 @ \ldots @ \alpha_n @ \rho'$ and $\overline{\alpha_1} @ \ldots \overline{\alpha_n} @ \sigma'$, with the only possible side effects of adding complementary actions to the prefixes, and of spawning further parallel threads. Since both the effects are compatible with the thesis, we are done.

($\Leftarrow$) if the action is a τ action, then the corresponding session contract can perform it as well. If it is a synchronization, by definition of the semantics it involves two threads with complementary prefixes $\alpha_1@\ldots@\alpha_n@\rho'$ and $\overline{\alpha_1}@\ldots\overline{\alpha_n}@\sigma'$. By inductive hypothesis $\rho \parallel \sigma \xrightarrow{*}_{\mathsf{SC}} \rho' \parallel \sigma'$, hence the thesis follows since ρ' and σ' can match the synchronization. $\square$

The proof of Proposition 2(i) descends immediately from the fact that $\mathsf{SC} \subsetneq \mathbf{rC} \subsetneq \mathbf{rsC}$. and the fact that system $\rhd_{\mathsf{SC}}$ is a subsystem of system $\rhd$. Similarly for Proposition 2(ii), by taking into account $\rhd_{\mathbf{rC}}$.

A direct proof of Corollary 3(i) We provide now a direct proof of Corollary 3(i), that is of the conservativity of the retractable/speculative compliance with respect to to session-contract compliance. We do that by taking into account the definition of the relation $\dashv\vdash^{\mathsf{R}}$, since we have that $\dashv\vdash \, = \, \dashv\vdash^{\mathsf{R}}$.
The proof requires some care. In fact, as seen in Lemma 3, even if we restrict ρ in $\mathrm{H}\ltimes\rho$ to be a session contract, reductions can modify the stack. This implies that in a sequence of reductions out of a retractable client/server system $\langle\,\rangle\ltimes\rho \parallel \langle\,\rangle\ltimes\sigma$ with $\rho, \sigma \in \mathsf{SC}$, also $\xrightarrow{\mathsf{rb}}$ reductions can occur. In order to handle the presence of rollbacks, we can show that in reduction sequences out of $\langle\,\rangle\ltimes\rho \parallel \langle\,\rangle\ltimes\sigma$, with $\rho, \sigma \in \mathsf{SC}$, only particular stacks can be produced (called *incompatible* below), such that once a rollback procedure is started it necessarily goes on till a stuck state is reached.

Definition 23 (Incompatible Stacks). *Let* $\mathrm{H}_1, \mathrm{H}_2 \in \mathbf{rsCH}$ *such that* $\mathrm{H}_1 = \delta_1 : \cdots : \delta_n$ *and* $\mathrm{H}_2 = \gamma_1 : \cdots : \gamma_n$ *for some* n.
We say that H_1 *and* H_2 *are* incompatible *if for any* $1 \leq i \leq n$ *either* $\delta_i = \circ$ *or* $\gamma_i = \circ$.

The following technical lemmas describe the behaviour of reduction and reduction sequences out of contracts with histories when session contracts are taken into account.

Lemma 4. *Let* $\rho, \sigma \in \mathsf{SC}$.

i) *Let* $\mathrm{H}_1\ltimes\rho \parallel \mathrm{H}_2\ltimes\sigma \longrightarrow_f \mathrm{H}'_1\ltimes\rho' \parallel \mathrm{H}'_2\ltimes\sigma'$. *If* H_1 *and* H_2 *are incompatible, so are* H'_1 *and* H'_2.

ii) *If* $\langle\,\rangle\ltimes\rho \parallel \langle\,\rangle\ltimes\sigma \xrightarrow{*}_f \mathrm{H}'_1\ltimes\rho' \parallel \mathrm{H}'_2\ltimes\sigma'$ *then* H'_1 *and* H'_2 *are incompatible.*

iii) *Given two incompatible stacks* H_1 *and* H_2 *and given* $\rho, \sigma \in \mathbf{rsC}$, *if* $\mathrm{H}_1\ltimes\rho \parallel \mathrm{H}_2\ltimes\sigma \not\longrightarrow_f$ *then* $\mathrm{H}_1\ltimes\rho \parallel \mathrm{H}_2\ltimes\sigma \xrightarrow{\mathsf{rb}}{}^* \mathrm{H}'_1\ltimes\rho' \parallel \mathrm{H}'_2\ltimes\sigma' \not\longrightarrow$.

iv) *If* $\langle\,\rangle\ltimes\rho \parallel \langle\,\rangle\ltimes\sigma \xrightarrow{*} \mathrm{H}'_1\ltimes\rho' \parallel \mathrm{H}'_2\ltimes\sigma' \not\longrightarrow$ *then the reduction sequence* $\xrightarrow{*}$ *is actually of the form* $\xrightarrow{*}_f \xrightarrow{\mathsf{rb}}{}^*$

Proof. **(i)** If the reduction $\longrightarrow_f$ is actually a reduction (τ), the thesis trivially holds. Otherwise, $\longrightarrow_f$ is necessarily a reduction (*comm*). Since $\rho, \sigma \in \mathsf{SC}$, from $\mathrm{H}_1\ltimes\rho \parallel \mathrm{H}_2\ltimes\sigma \longrightarrow_f \mathrm{H}'_1\ltimes\rho' \parallel \mathrm{H}'_2\ltimes\sigma'$ we can infer that either $\rho = a.\rho''$ and $\sigma = \overline{a}.\sigma''$ or $\rho = a.\rho'' + \rho'''$ and $\sigma = \overline{a}.\sigma''$ or $\sigma = a.\sigma''$ and $\rho = \overline{a}.\rho''$ or $\sigma = a.\sigma'' + \sigma'''$ and $\rho = \overline{a}.\sigma''$. In all such cases, by rule (*comm*) of Definition

4, we get that either $H'_1 = H_1 : \circ$ or $H'_2 = H_2 : \circ$. This implies H'_1 and H'_2 to be incompatible if H_1 and H_2 are so.

(ii) By induction of the length of the reduction sequence, using point (i).

(iii) Since H_1 and H_2 are incompatible, the rollbacks out of $H_1 \ltimes \rho \parallel H_2 \ltimes \sigma$ can proceed until the stacks become empty. At that point, the last element of such sequence of rollbacks is $\langle \rangle \ltimes \delta_1 \parallel \langle \rangle \ltimes \gamma_1$, where $\langle \rangle \ltimes \delta_1 \parallel \langle \rangle \ltimes \gamma_1 \not\rightarrow$ since one among δ_1 and γ_1 is equal to $\circ$.

(iv) If the reduction sequence does not contain any rollback, we get the thesis immediately. Let us then consider the leftmost subsequence such that
$$\langle \rangle \ltimes \rho \parallel \langle \rangle \ltimes \sigma \xrightarrow{\;*\;}_f H''_1 \ltimes \rho'' \parallel H''_2 \ltimes \sigma'' \not\rightarrow_f$$
By point (ii) we have that H''_1 and H''_2 are incompatible. By point (iii) we hence get that $H''_1 \ltimes \rho'' \parallel H''_2 \ltimes \sigma'' \xrightarrow{\mathsf{rb}\;*} H'''_1 \ltimes \rho''' \parallel H'''_2 \ltimes \sigma''' \not\rightarrow$. So, we have necessarily that $H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma'$ does coincide with $H'''_1 \ltimes \rho''' \parallel H'''_2 \ltimes \sigma''' \not\rightarrow$ and that the reduction sequence is actually a sequence $\xrightarrow{\;*\;}_f \xrightarrow{\mathsf{rb}\;*}$. $\qquad\square$

Fact 3 *Let $\rho, \sigma \in \mathsf{SC}$.*
$$H_1 \ltimes \rho \parallel H_2 \ltimes \sigma \longrightarrow_f H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma' \not\rightarrow \text{ implies } \rho \parallel \sigma \longrightarrow_{\mathsf{SC}} \rho' \parallel \sigma'.$$

We are now ready to provide a direct proof of Corollary 3(i), that we recall below (taking into account that $\dashv\mid^{\mathsf{R}} = \dashv\mid$).

The theory of retractable compliance is a conservative extension of compliance for session contracts SC, that is, given $\rho, \sigma \in \mathsf{SC}$,

$$\rho \dashv\mid_{\mathsf{SC}} \sigma \quad \textit{iff} \quad \rho \dashv\mid^{\mathsf{R}} \sigma$$

Proof. $(\Rightarrow)$ By contradiction let us assume

$$\langle \rangle \ltimes \rho \parallel \langle \rangle \ltimes \sigma \xrightarrow{\;*\;} H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma' \not\rightarrow$$

with $\rho' \neq \mathbf{1}$. By Lemma 4(iv) just the following two cases can be taken into account:
either
$$\langle \rangle \ltimes \rho \parallel \langle \rangle \ltimes \sigma \xrightarrow{\;*\;}_f H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma' \not\rightarrow$$

or
$$\langle \rangle \ltimes \rho \parallel \langle \rangle \ltimes \sigma \xrightarrow{\;*\;}_f H''_1 \ltimes \rho'' \parallel H''_2 \ltimes \sigma'' \xrightarrow{\mathsf{rb}\;+} H'_1 \ltimes \rho' \parallel H'_1 \ltimes \sigma' \not\rightarrow$$

In the first case, by induction, using Fact 3, we get that $\rho \parallel \sigma \longrightarrow_{\mathsf{SC}} \rho' \parallel \sigma'$ / $\longrightarrow_{\mathsf{SC}}$ with $\rho' \neq \mathbf{1}$, that is $\rho \not\dashv\mid_{\mathsf{SC}} \sigma$.

In the second case, similarly to what done previously, we can get $\rho \parallel \sigma \longrightarrow_{\mathsf{SC}} \rho'' \parallel \sigma'' \not\rightarrow_{\mathsf{SC}}$. Since $H''_1 \ltimes \rho'' \parallel H''_2 \ltimes \sigma'' \xrightarrow{\mathsf{rb}}$, we have necessarily that $\rho'' \neq \mathbf{1}$. This means that $\rho \not\dashv\mid_{\mathsf{SC}} \sigma$.

$(\Leftarrow)$ By contradiction, let us assume that $\rho \parallel \sigma \longrightarrow_{\mathsf{SC}} \rho' \parallel \sigma' \not\rightarrow_{\mathsf{SC}}$ with $\rho' \neq \mathbf{1}$. It is easy now to get $\langle \rangle \ltimes \rho \parallel \langle \rangle \ltimes \sigma \xrightarrow{\;*\;}_f H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma' \not\rightarrow_f$. We distinguish now two cases: either $H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma' \xrightarrow{\mathsf{rb}}$ or $H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma' \xrightarrow{\mathsf{rb}}\!\!\!\!/\,$. In the

first case we have finished, since we get $\rho \not\parallel^{\mathsf{R}} \sigma$ by definition. In the second one, by Lemma 4(ii) we get that $H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma' \xrightarrow{\mathsf{rb}}^* H''_1 \ltimes \rho'' \parallel H''_2 \ltimes \sigma'' \not\rightarrow$. Moreover, ρ'' cannot be $\mathbf{1}$, otherwise it should have be put on the stack in one of the reductions of $\langle\,\rangle \ltimes \rho \parallel \langle\,\rangle \ltimes \sigma \xrightarrow{\;*\;}_f H'_1 \ltimes \rho' \parallel H'_2 \ltimes \sigma'$, and that is impossible. Hence we get by definition that $\rho \not\parallel^{\mathsf{R}} \sigma$. $\qquad\square$

A.3   Soundness and Completeness Proofs (Theorems 2 and 3)

We begin with the proof of Soundness and Completeness of system $\,\rhd\,$ with respect to the retractable compliance.

Retractable Soundness and Completeness It is useful to show that if a configuration is stuck, then both histories are empty. This is a consequence of the fact that the property "the histories of client and server have the same length" is preserved by reductions.

Lemma 5. *If* $\langle\,\rangle \ltimes \rho \parallel \langle\,\rangle \ltimes \sigma \xrightarrow{\;*\;} H_1 \ltimes \rho' \parallel H_2 \ltimes \sigma' \not\rightarrow$, *then* $H_1 = H_2 = \langle\,\rangle$.

Proof. Clearly $H_1 \ltimes \rho' \parallel H_2 \ltimes \sigma' \not\rightarrow$ implies either $H_1 = \langle\,\rangle$ or $H_2 = \langle\,\rangle$. Observe that:

- rule (comm) adds one element to both stacks;
- rule (τ) does not modify both stacks;
- rule (rbk) removes one element from both stacks.

Then starting from two stacks containing the same number of elements, the reduction always produces two stacks containing the same number of elements. So $H_1 = \langle\,\rangle$ implies $H_2 = \langle\,\rangle$ and vice versa. $\qquad\square$

The following lemma proves that compliance is preserved by the concatenation of histories to the left of the current histories.

Lemma 6. *If* $H_1 \ltimes \rho \dashv\parallel^{\mathsf{R}} H_2 \ltimes \sigma$, *then* $H'_1 : H_1 \ltimes \rho \dashv\parallel^{\mathsf{R}} H'_2 : H_2 \ltimes \sigma$ *for all* H'_1, H'_2.

Proof. It suffices to show that

$$H_1 \ltimes \rho \dashv\parallel^{\mathsf{R}} H_2 \ltimes \sigma \text{ implies } \rho' : H_1 \ltimes \rho \dashv\parallel^{\mathsf{R}} H_2 \ltimes \sigma \text{ and } H_1 \ltimes \rho \dashv\parallel^{\mathsf{R}} \sigma' : H_2 \ltimes \sigma$$

which we prove by contraposition.

Suppose that $\rho' : H_1 \ltimes \rho \not\parallel^{\mathsf{R}} H_2 \ltimes \sigma$; then

$$\rho' : H_1 \ltimes \rho \parallel H_2 \ltimes \sigma \xrightarrow{\;*\;} H'_1 \ltimes \rho'' \parallel H'_2 \ltimes \sigma'' \not\rightarrow \text{ and } \rho'' \neq \mathbf{1}$$

If ρ' is never used, then $H'_1 = \rho' : H''_1$ and $H'_2 = \langle\,\rangle$, so that we get

$$H_1 \ltimes \rho \parallel H_2 \ltimes \sigma \xrightarrow{\;*\;} H''_1 \ltimes \rho'' \parallel \langle\,\rangle \ltimes \sigma'' \not\rightarrow$$

Otherwise we have that

$$\rho' : H_1 \ltimes \rho \parallel H_2 \ltimes \sigma \xrightarrow{\;*\;} \rho' \ltimes \rho'' \parallel H'_2 \ltimes \sigma'' \longrightarrow \langle\,\rangle \ltimes \rho' \parallel H''_2 \ltimes \sigma'''$$

and we assume that $\overset{*}{\longrightarrow}$ is the shortest such reduction. It follows that $\rho'' \neq \mathbf{1}$. By the minimality assumption about the length of $\overset{*}{\longrightarrow}$ we know that ρ' neither has been restored by some previous application of rule (rbk), nor pushed back into the stack before. We get

$$\mathrm{H}_1 \ltimes \rho \parallel \mathrm{H}_2 \ltimes \sigma \overset{*}{\longrightarrow} \langle\,\rangle \ltimes \rho'' \parallel \mathrm{H}_2'' \ltimes \sigma'' \nrightarrow$$

In both cases we conclude that $\mathrm{H}_1 \ltimes \rho \nvdash^{\mathsf{R}} \mathrm{H}_2 \ltimes \sigma$ as desired.

Similarly we can show that $\mathrm{H}_1 \ltimes \rho \nvdash^{\mathsf{R}} \sigma' : \mathrm{H}_2 \ltimes \sigma$ implies $\mathrm{H}_1 \ltimes \rho \nvdash^{\mathsf{R}} \mathrm{H}_2 \ltimes \sigma$. $\quad\square$

The following lemma gives all possible shapes of compliant contracts. It is the key lemma for the proof of soundness and completeness.

Lemma 7. *We have $\rho \dashv\!\vdash^{\mathsf{R}} \sigma$ if and only if one of the following conditions holds:*

1. *$\rho = \mathbf{1}$;*
2. *$\rho = \sum_{i \in I} \alpha_i.\rho_i$, $\sigma = \sum_{j \in J} \overline{\alpha}_j.\sigma_j$ and $\exists k \in I \cap J.\ \rho_k \dashv\!\vdash^{\mathsf{R}} \sigma_k$;*
3. *$\rho = \bigoplus_{i \in I} \overline{\alpha}_i.\rho_i$, $\sigma = \sum_{j \in J} \alpha_j.\sigma_j$, $I \subseteq J$ and $\forall k \in I.\ \rho_k \dashv\!\vdash^{\mathsf{R}} \sigma_k$;*
4. *$\rho = \sum_{i \in I} \overline{\alpha}_i.\rho_i$, $\sigma = \bigoplus_{j \in J} \alpha_j.\sigma_j$, $I \supseteq J$ and $\forall k \in J.\ \rho_k \dashv\!\vdash^{\mathsf{R}} \sigma_k$.*

Proof. The if part is immediate. We prove the only if part by contraposition and by cases on the possible shapes of ρ and σ.

Suppose $\rho = \sum_{i \in I} \alpha_i.\rho_i$, $\sigma = \sum_{j \in J} \overline{\alpha}_j.\sigma_j$, $I \cap J = \{k_1, \ldots, k_n\}$ and $\rho_{k_i} \nvdash^{\mathsf{R}} \sigma_{k_i}$ for $1 \leq i \leq n$. Then we get

$$\langle\,\rangle \ltimes \rho_{k_i} \parallel \langle\,\rangle \ltimes \sigma_{k_i} \overset{*}{\longrightarrow} \mathrm{H}_i \ltimes \rho_i' \parallel \mathrm{H}_i' \ltimes \sigma_i' \nrightarrow$$

for $1 \leq i \leq n$, where $\rho_i' \neq \mathbf{1}$ and $\mathrm{H}_i = \mathrm{H}_i' = \langle\,\rangle$ by Lemma 5. This implies

$$\textstyle\sum_{i \in I \setminus \{k_1\}} \alpha_i.\rho_i \ltimes \rho_{k_1} \parallel \sum_{j \in J \setminus \{k_1\}} \overline{\alpha}_j.\sigma_j \ltimes \sigma_{k_1} \overset{*}{\longrightarrow} \sum_{i \in I \setminus \{k_1\}} \alpha_i.\rho_i \ltimes \rho_1' \parallel \sum_{j \in J \setminus \{k_1\}} \overline{\alpha}_j.\sigma_j \ltimes \sigma_1'$$

by Lemma 6. Let $I' = I \setminus J$ and $J' = J \setminus I$. We can reduce $\langle\,\rangle \ltimes \rho \parallel \langle\,\rangle \ltimes \sigma$ only as follows:

$$
\begin{aligned}
\langle\,\rangle \ltimes \rho \parallel \langle\,\rangle \ltimes \sigma &\longrightarrow \textstyle\sum_{i \in I \setminus \{k_1\}} \alpha_i.\rho_i \ltimes \rho_{k_1} \parallel \sum_{j \in J \setminus \{k_1\}} \overline{\alpha}_j.\sigma_j \ltimes \sigma_{k_1} \text{ by (comm)}\\
&\overset{*}{\longrightarrow} \textstyle\sum_{i \in I \setminus \{k_1\}} \alpha_i.\rho_i \ltimes \rho_1' \parallel \sum_{j \in J \setminus \{k_1\}} \overline{\alpha}_j.\sigma_j \ltimes \sigma_1'\\
&\longrightarrow \textstyle\langle\,\rangle \ltimes \sum_{i \in I \setminus \{k_1\}} \alpha_i.\rho_i \parallel \langle\,\rangle \ltimes \sum_{j \in J \setminus \{k_1\}} \overline{\alpha}_j.\sigma_j \quad \text{by (rbk)}\\
&\;\vdots \qquad\qquad\qquad\qquad\quad \vdots\\
&\overset{*}{\longrightarrow} \textstyle\sum_{i \in I'} \alpha_i.\rho_i \ltimes \rho_n' \parallel \sum_{j \in J'} \overline{\alpha}_j.\sigma_j \ltimes \sigma_n'\\
&\longrightarrow \textstyle\langle\,\rangle \ltimes \sum_{i \in I'} \alpha_i.\rho_i \parallel \langle\,\rangle \ltimes \sum_{j \in J'} \overline{\alpha}_j.\sigma_j \qquad\qquad \text{by (rbk)}
\end{aligned}
$$

and $\langle\,\rangle \ltimes \sum_{i \in I'} \alpha_i.\rho_i \parallel \langle\,\rangle \ltimes \sum_{j \in J'} \overline{\alpha}_j.\sigma_j$ is stuck since $I' \cap J' = \emptyset$.

Suppose $\rho = \bigoplus_{i \in I} \overline{\alpha}_i.\rho_i$ and $\sigma = \sum_{j \in J} \alpha_j.\sigma_j$. If $I \not\subseteq J$ let $k \in I \setminus J$; then we get

$$\langle\,\rangle \ltimes \rho \parallel \langle\,\rangle \ltimes \sigma \longrightarrow \langle\,\rangle \ltimes \overline{\alpha}_k.\rho_k \parallel \langle\,\rangle \ltimes \sigma \text{ by } (\tau)\\ \nrightarrow$$

Otherwise $I \subseteq J$ and $\rho_k \not\Vdash^{\mathsf{R}} \sigma_k$ for some $k \in I$. By reasoning as above we have

$$\langle\,\rangle\ltimes\rho_k \,\|\, \langle\,\rangle\ltimes\sigma_k \xrightarrow{\;*\;} \langle\,\rangle\ltimes\rho' \,\|\, \langle\,\rangle\ltimes\sigma' \not\rightarrow$$

and

$$\circ\ltimes\rho_k \,\|\, \textstyle\sum_{j\in J\setminus\{k\}}\alpha_j.\sigma_j\ltimes\sigma_k \xrightarrow{\;*\;} \circ\ltimes\rho' \,\|\, \textstyle\sum_{j\in J\setminus\{k\}}\alpha_j.\sigma_j\ltimes\sigma'$$

which imply

$$
\begin{aligned}
\langle\,\rangle\ltimes\rho \,\|\, \langle\,\rangle\ltimes\sigma &\longrightarrow \langle\,\rangle\ltimes\overline{\alpha}_k.\rho_k \,\|\, \langle\,\rangle\ltimes\sigma && \text{by } (\tau)\\
&\longrightarrow \circ\ltimes\rho_k \,\|\, \textstyle\sum_{j\in J\setminus\{k\}}\alpha_j.\sigma_j\ltimes\sigma_k && \text{by } (\mathsf{comm})\\
&\xrightarrow{\;*\;} \circ\ltimes\rho' \,\|\, \textstyle\sum_{j\in J\setminus\{k\}}\alpha_j.\sigma_j\ltimes\sigma' &&\\
&\longrightarrow \langle\,\rangle\ltimes\circ \,\|\, \langle\,\rangle\ltimes\textstyle\sum_{j\in J\setminus\{k\}}\alpha_j.\sigma_j && \text{by } (\mathsf{rbk})\\
&\not\rightarrow &&
\end{aligned}
$$

In both cases we conclude that $\rho \not\Vdash^{\mathsf{R}} \sigma$.

The proof for the case $\rho = \sum_{i\in I}\overline{\alpha}_i.\rho_i$, $\sigma = \bigoplus_{j\in J}\alpha_j.\sigma_j$ is similar.

Lemma 7 suggests that $\dashv\Vdash^{\mathsf{R}}$ can be coinductively defined, or equivalently that $\dashv\Vdash^{\mathsf{R}} = \bigcap_n \dashv\Vdash^{\mathsf{R}}_n$ where $\dashv\Vdash^{\mathsf{R}}_0$ is the trivial relation $\mathbf{rsC} \times \mathbf{rsC}$, and for all $n > 0$, $\mathbf{1} \dashv\Vdash^{\mathsf{R}}_n \sigma$ and we have $\rho \dashv\Vdash^{\mathsf{R}}_n \sigma$ if one of the following holds:

1. $\rho = \sum_{i\in I}\alpha_i.\rho_i$, $\sigma = \sum_{j\in J}\overline{\alpha}_j.\sigma_j$ and $\exists k \in I \cap J.\ \rho_k \dashv\Vdash^{\mathsf{R}}_{n-1} \sigma_k$;
2. $\rho = \bigoplus_{i\in I}\overline{\alpha}_i.\rho_i$, $\sigma = \sum_{j\in J}\alpha_j.\sigma_j$, $I \subseteq J$ and $\forall k \in I.\ \rho_k \dashv\Vdash^{\mathsf{R}}_{n-1} \sigma_k$;
3. $\rho = \sum_{i\in I}\overline{\alpha}_i.\rho_i$, $\sigma = \bigoplus_{j\in J}\alpha_j.\sigma_j$, $I \supseteq J$ and $\forall k \in J.\ \rho_k \dashv\Vdash^{\mathsf{R}}_{n-1} \sigma_k$.

We write:

1. $\models^{\mathsf{R}} \Gamma$ if for all $\rho' \dashv\Vdash^{\mathsf{R}} \sigma' \in \Gamma$ we have $\rho' \dashv\Vdash^{\mathsf{R}} \sigma'$
2. $\Gamma \models^{\mathsf{R}} \rho \dashv\sigma$ if $\models^{\mathsf{R}} \Gamma$ implies $\rho \dashv\Vdash^{\mathsf{R}} \sigma$

We also write $\Gamma \models^{\mathsf{R}}_n \rho \dashv\sigma$ if $\dashv\Vdash^{\mathsf{R}}$ is replaced by $\dashv\Vdash^{\mathsf{R}}_n$ in the above.

Observing that $\dashv\Vdash^{\mathsf{R}}_{n+1} \subseteq \dashv\Vdash^{\mathsf{R}}_n$, we have that $\models^{\mathsf{R}}_{n+1} \Gamma$ implies $\models^{\mathsf{R}}_n \Gamma$. Also it is immediate to verify that the following holds:

Fact 4 *If $\Gamma \models^{\mathsf{R}}_n \rho \dashv\sigma$ for all n, then $\Gamma \models^{\mathsf{R}} \rho \dashv\sigma$.*

We are ready now to prove Theorem 2, which, using the notation above, can be restated as follows.

Theorem 2 (Retractable Soundness and Completeness)

$$\triangleright \rho \dashv\sigma \ \text{ iff } \ \models^{\mathsf{R}} \rho \dashv\sigma$$

Proof. ($\Rightarrow$) For this direction we can actually prove a stronger statement, namely

$$\Gamma \triangleright \rho \dashv\sigma \ \Rightarrow \ \Gamma \models^{\mathsf{R}} \rho \dashv\sigma$$

By Fact 4 it suffices to prove that if $\Gamma \rhd \rho \dashv\vdash \sigma$ then $\Gamma \models_n^R \rho \dashv\vdash \sigma$ for all n, which we establish by simultaneous induction over n and over the derivation $\mathcal{D}$ of $\Gamma \rhd \rho \dashv\vdash \sigma$.

If $\mathcal{D}$ either ends by Ax or by Hyp then the thesis trivially holds. If $\mathcal{D}$ ends by rule:

$$
\frac{\overset{(+ \cdot +)}{\Gamma, \alpha.\rho + \rho' \dashv\vdash \overline{\alpha}.\sigma + \sigma' \ \rhd \ \rho \dashv\vdash \sigma}}{\Gamma \ \rhd \ \alpha.\rho + \rho' \dashv\vdash \overline{\alpha}.\sigma + \sigma'}
$$

then we have to show that $\models_n^R \Gamma$ implies $\alpha.\rho + \rho' \dashv\Vdash_n^R \overline{\alpha}.\sigma + \sigma'$. By induction over n we know that $\Gamma \models_{n-1}^R \alpha.\rho + \rho' \dashv\vdash \overline{\alpha}.\sigma + \sigma'$; from this and the fact that $\models_n^R \Gamma$ implies $\models_{n-1}^R \Gamma$, we obtain that $\alpha.\rho + \rho' \dashv\Vdash_{n-1}^R \overline{\alpha}.\sigma + \sigma'$, and hence that $\models_{n-1}^R \Gamma, \alpha.\rho + \rho' \dashv\vdash \overline{\alpha}.\sigma + \sigma'$. By induction over $\mathcal{D}$ it follows that $\rho \dashv\Vdash_{n-1}^R \sigma$, which implies $\alpha.\rho + \rho' \dashv\Vdash_n^R \overline{\alpha}.\sigma + \sigma'$ by Lemma 7 as desired.

The cases in which $\mathcal{D}$ ends by either $(\oplus \cdot +)$ or $(+ \cdot \oplus)$ are similar, and we conclude.

($\Leftarrow$) By Theorem 1 each computation of $\mathbf{Prove}(\ \rhd \ \rho \dashv\vdash \sigma)$ always terminates. By Lemma 7 and Fact 1, $\rho \dashv\Vdash^R \sigma$ implies that $\mathbf{Prove}(\ \rhd \ \rho \dashv\vdash \sigma) \neq \mathbf{fail}$, and hence $\rhd \ \rho \dashv\vdash \sigma$. $\qquad\square$

We proceed now with the proof of Soundness and Completeness of system $\rhd$ with respect to the speculative compliance.

Speculative Soundness and Completeness

Lemma 8. *We have $\rho \dashv\Vdash^S \sigma$ if and only if one of the following conditions holds:*

1. *$\rho = \mathbf{1}$;*
2. *$\rho = \sum_{i \in I} \alpha_i.\rho_i$, $\sigma = \sum_{j \in J} \overline{\alpha}_j.\sigma_j$ and $\exists k \in I \cap J. \ \rho_k \dashv\Vdash^S \sigma_k$;*
3. *$\rho = \bigoplus_{i \in I} \overline{\alpha}_i.\rho_i$, $\sigma = \sum_{j \in J} \alpha_j.\sigma_j$, $I \subseteq J$ and $\forall k \in I. \ \rho_k \dashv\Vdash^S \sigma_k$;*
4. *$\rho = \sum_{i \in I} \overline{\alpha}_i.\rho_i$, $\sigma = \bigoplus_{j \in J} \alpha_j.\sigma_j$, $I \supseteq J$ and $\forall k \in J. \ \rho_k \dashv\Vdash^S \sigma_k$.*

Proof. The if part is immediate. We prove the only if part by contraposition and by cases on the possible shapes of ρ and σ.

Suppose $\rho = \sum_{i \in I} \alpha_i.\rho_i$, $\sigma = \sum_{j \in J} \overline{\alpha}_j.\sigma_j$, $I \cap J = \{k_1, \ldots, k_n\}$ and $\rho_{k_i} \not\dashv\Vdash^S \sigma_{k_i}$ for $1 \leq i \leq n$. Then we get

$$
\rho_{k_i} \parallel \sigma_{k_i} \overset{*}{\longrightarrow} \mathbf{C}_i^\rho \parallel \mathbf{C}_i^\sigma \not\longrightarrow
$$

for $1 \leq i \leq n$, where $\mathbf{C}_i^\rho \neq \mathbf{C} \mid \alpha_1@...@\alpha_n@\mathbf{1}$.

This implies

$$
\begin{aligned}
&\sum_{i \in I} \alpha_i.\rho_i \parallel \sum_{j \in J} \overline{\alpha}_j.\sigma_j \\
\overset{*}{\longrightarrow} \ &\left(\textstyle\prod_{i \in I \cap J} \alpha_i@\rho_i\right) \mid \sum_{i \in I \setminus J} \alpha_i.\rho_i \parallel \left(\textstyle\prod_{j \in I \cap J} \overline{\alpha}_j@\sigma_j\right) \mid \sum_{j \in J \setminus I} \overline{\alpha}_j.\sigma_j
\end{aligned}
$$

where $\prod_{i \in I} \mathbf{C}_i$ denotes the parallel composition of $\mathbf{C}_i$ for each $i \in I$.

One can notice that terms $\sum_{i \in I \setminus J} \alpha_i.\rho_i$ and $\sum_{j \in J \setminus I} \overline{\alpha}_j.\sigma_j$ cannot interact with other terms. Instead, term $\prod_{i \in I \cap J} \alpha_i @ \rho_i$ can interact only with term $\prod_{j \in I \cap J} \overline{\alpha}_j @ \sigma_j$ and vice versa. The interaction follows the computations $\rho_{k_i} \parallel \sigma_{k_i} \xrightarrow{*} \mathbf{C}_i^\rho \parallel \mathbf{C}_i^\sigma \not\longrightarrow$, with the added prefixes α_i and $\overline{\alpha}_i$. However, none of these computations produces a thread of the form $\alpha'_1 @ \dots @ \alpha'_n @ \mathbf{1}$, hence $\rho \not\Vdash^{\mathsf{S}} \sigma$.

Suppose $\rho = \bigoplus_{i \in I} \overline{\alpha}_i.\rho_i$ and $\sigma = \sum_{j \in J} \alpha_j.\sigma_j$. If $I \not\subseteq J$ let $k \in I \setminus J$; then we get

$$\rho \parallel \sigma \longrightarrow \overline{\alpha}_k.\rho_k \parallel \sigma \text{ by } (\tau)$$
$$\not\longrightarrow$$

Otherwise $I \subseteq J$ and $\rho_k \not\Vdash^{\mathsf{S}} \sigma_k$ for some $k \in I$. By reasoning as above we have

$$\rho_k \parallel \sigma_k \xrightarrow{*} \mathbf{C}^\rho \parallel \mathbf{C}^\sigma \not\longrightarrow$$

and

$$\rho \parallel \sigma$$
$$\longrightarrow \overline{\alpha}_k.\rho_k \parallel \sigma$$
$$\longrightarrow \overline{\alpha}_k @ \rho_k \parallel \alpha_k @ \sigma_k \mid \sum_{j \in J \setminus \{k\}} \alpha_j.\sigma_j \xrightarrow{*} \overline{\alpha}_k @ \mathbf{C}^\rho \parallel \alpha_k @ \mathbf{C}^\sigma \mid \sum_{j \in J \setminus \{k\}} \alpha_j.\sigma_j$$
$$\not\longrightarrow$$

where $\alpha @ \mathbf{C}$ denotes $\prod_{i \in I} \alpha @ \mathbf{T}_i$ if $\mathbf{C} = \prod_{i \in I} \mathbf{T}_i$.
In both cases we conclude that $\rho \not\Vdash^{\mathsf{S}} \sigma$.

The proof for the case $\rho = \sum_{i \in I} \overline{\alpha}_i.\rho_i$, $\sigma = \bigoplus_{j \in J} \alpha_j.\sigma_j$ is similar. $\qquad \square$

As with $\Vdash^{\mathsf{R}}$, we define the family of relations $\Vdash_n^{\mathsf{S}}$ on the basis of Lemma 8, which are such that $\Vdash^{\mathsf{S}} = \bigcap_n \Vdash_n^{\mathsf{S}}$; similarly we define the respective notions $\Gamma \models^{\mathsf{S}} \rho \dashv \sigma$ and $\Gamma \models^{\mathsf{S}_n} \rho \dashv \sigma$.

Theorem 3 (Speculative Soundness and Completeness)

$$\rhd \rho \dashv \sigma \quad \text{iff} \quad \models^{\mathsf{S}} \rho \dashv \sigma$$

Proof. ($\Rightarrow$) This implication can be proved in the same way as Theorem 2.
($\Leftarrow$) By Theorem 1 each computation of $\mathbf{Prove}(\rhd \ \rho \dashv \sigma)$ always terminates. By Lemma 7 and Fact 1, $\rho \Vdash^{\mathsf{R}} \sigma$ implies that $\mathbf{Prove}(\rhd \ \rho \dashv \sigma) \neq \mathbf{fail}$, and hence $\rhd \ \rho \dashv \sigma$. $\qquad \square$

A.4   Proof of Proposition 4

As stated previously, Lemma 7 $\Vdash^{\mathsf{R}}$ can be coinductively defined. This holds for $\dashv\vdash$ as well, since, by Corollary 1, $\Vdash^{\mathsf{R}} = \Vdash^{\mathsf{S}} = \dashv\vdash$. So $\dashv\vdash = \bigcap_n \dashv\vdash_n$ where $\dashv\vdash_n = \Vdash_n^{\mathsf{R}}$.
We recall here the definition of $\dashv\vdash_n$ for sake of readability.
$\dashv\vdash_0$ is the trivial relation $\mathbf{rsC} \times \mathbf{rsC}$, and for all $n > 0$, $\mathbf{1} \dashv\vdash_n \sigma$ and we have $\rho \dashv\vdash_n \sigma$ if one of the following holds:

1. $\rho = \sum_{i \in I} \alpha_i.\rho_i$, $\sigma = \sum_{j \in J} \overline{\alpha}_j.\sigma_j$ and $\exists k \in I \cap J$. $\rho_k \dashv\!\parallel_{n-1} \sigma_k$;
2. $\rho = \bigoplus_{i \in I} \overline{\alpha}_i.\rho_i$, $\sigma = \sum_{j \in J} \alpha_j.\sigma_j$, $I \subseteq J$ and $\forall k \in I$. $\rho_k \dashv\!\parallel_{n-1} \sigma_k$;
3. $\rho = \sum_{i \in I} \overline{\alpha}_i.\rho_i$, $\sigma = \bigoplus_{j \in J} \alpha_j.\sigma_j$, $I \supseteq J$ and $\forall k \in J$. $\rho_k \dashv\!\parallel_{n-1} \sigma_k$.

We shall prove that

$$\forall n.[(\rho \dashv\!\parallel_n \sigma \text{ and } \overline{\sigma} \dashv\!\parallel_n \sigma') \text{ implies } \rho \dashv\!\parallel_n \sigma'] \tag{1}$$

So, from $\rho \dashv\!\parallel \sigma$ and $\overline{\sigma} \dashv\!\parallel \sigma'$ we have that $\forall n.\rho \dashv\!\parallel_n \sigma$ and $\forall n.\overline{\sigma} \dashv\!\parallel_n \sigma'$ and hence, by 1 we can get $\forall n.\rho \dashv\!\parallel_n \sigma'$, that is $\rho \dashv\!\parallel \sigma'$.

We show now 1 by induction on n.

The base case is trivial. Let then $n > 0$ with $\rho \dashv\!\parallel_n \sigma$ and $\overline{\sigma} \dashv\!\parallel_n \sigma'$. We proceed by cases according to the possible shapes of ρ and σ in the definition of $\dashv\!\parallel_n$.

$\rho = \mathbf{1}$ Immediate.

$\rho = \sum_{i \in I} \alpha_i.\rho_i$, $\sigma = \sum_{j \in J} \overline{\alpha}_j.\sigma_j$ **and** $\exists k \in I \cap J$. $\rho_k \dashv\!\parallel_{n-1} \sigma_k$
  We have then that $\overline{\sigma} = \bigoplus_{j \in J} \alpha_j.\overline{\sigma}_j$. So, by $\overline{\sigma} \dashv\!\parallel_n \sigma'$ and by definition of $\dashv\!\parallel_n$ we have that $\sigma' = \sum_{h \in H} \overline{\alpha}_h.\sigma'_h$, $J \subseteq H$ and $\forall j \in J$. $\overline{\sigma}_j \dashv\!\parallel_{n-1} \sigma'_j$. By the induction hypothesis we can hence get that $\exists k \in (I \cap J) \subseteq (I \cap H)$ such that $\rho_k \dashv\!\parallel_{n-1} \sigma'_k$, that means, by definition, that $\rho \dashv\!\parallel_n \sigma'$.

$\rho = \bigoplus_{i \in I} \overline{\alpha}_i.\rho_i$, $\sigma = \sum_{j \in J} \alpha_j.\sigma_j$, $I \subseteq J$ **and** $\forall k \in I$. $\rho_k \dashv\!\parallel_{n-1} \sigma_k$
  We have then that $\overline{\sigma} = \bigoplus_{j \in J} \overline{\alpha}_j.\overline{\sigma}_j$. So, by $\overline{\sigma} \dashv\!\parallel_n \sigma'$ and by definition of $\dashv\!\parallel_n$ we have that $\sigma' = \sum_{h \in H} \alpha_h.\sigma'_h$, $J \subseteq H$ and $\forall j \in J$. $\overline{\sigma}_j \dashv\!\parallel_{n-1} \sigma'_j$. By the induction hypothesis we can hence get that $\forall k \in I \subseteq J \subseteq H$. $\rho_k \dashv\!\parallel_{n-1} \sigma'_k$, that means, by definition, that $\rho \dashv\!\parallel_n \sigma'$.

$\rho = \sum_{i \in I} \overline{\alpha}_i.\rho_i$, $\sigma = \bigoplus_{j \in J} \alpha_j.\sigma_j$, $I \supseteq J$ **and** $\forall k \in J$. $\rho_k \dashv\!\parallel_{n-1} \sigma_k$
  We have then that $\overline{\sigma} = \sum_{j \in J} \overline{\alpha}_j.\overline{\sigma}_j$. So, by $\overline{\sigma} \dashv\!\parallel_n \sigma'$ and by definition of $\dashv\!\parallel_n$ we have to take into account two cases.

  $\sigma' = \sum_{h \in H} \alpha_h.\sigma'_h$ **and** $\exists k \in J \cap H$. $\sigma_k \dashv\!\parallel_{n-1} \sigma'_k$
    By the induction hypothesis we can get that $\exists k \in (J \cap H) \subseteq (I \cap H)$ such that $\rho_k \dashv\!\parallel_{n-1} \sigma'_k$, that means, by definition, that $\rho \dashv\!\parallel_n \sigma'$.

  $\sigma' = \bigoplus_{h \in H} \alpha_h.\sigma'_h$, $J \supseteq H$ **and** $\forall h \in H$. $\sigma_h \dashv\!\parallel_{n-1} \sigma'_h$
    By the induction hypothesis we can get that $\forall h \in H \subseteq J \subseteq I$. $\rho_h \dashv\!\parallel_{n-1} \sigma'_h$, that means, by definition, that $\rho \dashv\!\parallel_n \sigma'$.