Compliance for reversible client/server interactions*

Franco Barbanera
Dipartimento di Matematica e Informatica
University of Catania
barba@dmi.unict.it

Mariangiola Dezani-Ciancaglini
Dipartimento di Informatica
University of Torino
dezani@di.unito.it

Ugo de'Liguoro
Dipartimento di Informatica
University of Torino
deliguoro@di.unito.it

In the setting of *session behaviours*, we study an extension of the concept of compliance when a disciplined form of backtracking is present. After adding checkpoints to the syntax of session behaviours, we formalise the operational semantics via a LTS, and define a natural notion of *checkpoint compliance*. We then obtain a co-inductive characterisation of such compliance relation, and an axiomatic presentation that is proved to be sound and complete. As a byproduct we get a decision procedure for the new compliance, being the axiomatic system algorithmic.

1 Introduction. In human as well as automatic negotiations, an interesting feature is the ability of rolling back to some previous point, undoing previous choices and possibly trying a different path. *Rollbacks* are familiar to the users of web browsers, and so are also the troubles that these might cause during "undisciplined" interactions. Clicking the "back" button, or going to some previous point in the chronology when we are in the middle of a transaction, say the booking of a flight, can be as smart as dangerous. In any case it is surely a behaviour that service programmers want to discipline. Also the converse has to be treated with care: a server discovering that a service becomes available after having started a conversation could take advantage from some kind of rolling backs. However, such a server would be quite unfair if the rollbacks were completely hidden from the client.

Adding rollbacks to interaction protocols requires a sophisticated concept of client/server compliance. In this paper we investigate protocols admitting a simple, though non trivial form of reversibility in the framework of the theory of contracts introduced in [4] and developed in a series of papers, e.g. [5]. We focus here on the scenario of client/server architectures, where services stored in a repository are queried by clients to establish two-sided communications, and the central concept is that of *compliance*.

More precisely, we consider the formalism of *session behaviours* as introduced in [2, 1, 3], but without delegation. This is a formalism interpreting the session types, introduced by Honda et al. in [7], into a subset of CCS without τ. We extend the session behaviours syntax by means of markers that we call *checkpoints*; these are intended as pointers to the last place where either the client or the server can roll back at any time. We investigate which constraints must be imposed to obtain a safe notion of client/server interaction in the new scenario, by defining a model in the form of a LTS, and by characterising the resulting concept of compliance both coinductively and axiomatically. Since the axiomatic system is algorithmic that is decidable, the compliance of behaviours with checkpoints is decidable.

Before entering into the formal development of session behaviours with checkpoints, we illustrate the basic concepts by discussing a few examples. Suppose that the client is a customer willing to arrange

*This work was partially supported by EU Collaborative project ASCENS 257414, ICT COST Action IC1201 BETTY, MIUR PRIN Project CINA Prot. 2010LHT4KM and Torino University/Compagnia San Paolo Project SALT.

Submitted to:
BEAT 2014

for an holiday, while the server is the web service of a travel agency. Let the action sea represent the quest for a seaside accommodation and let mount stands for the request of a settlement in the mountains. By house we mean the request of a house, while bung stands for the request of a bungalow. Dual actions represent offers, so that e.g. the co-action $\overline{\mathsf{sea}}$ signals availability of accommodations in a seaside and $\overline{\mathsf{house}}$ that a house can be booked.

Suppose that the customer seeks a house or a bungalow at sea, but just a house in the mountains. Then the client behaviour, represented as a process algebraic term, is described by:

$$\rho = \mathsf{sea}.(\mathsf{house} + \mathsf{bung}) \;+\; \mathsf{mount}.\mathsf{house}$$

where dots are sequential compositions and sums are external choices. We say that a client ρ is *compliant* with a server σ, written $\rho \dashv \sigma$, if all client communication actions are matched by the dual actions on the server side. According to this the customer will be not compliant with a server behaving as:

$$\sigma = \overline{\mathsf{mount}}.(\overline{\mathsf{house}} \oplus \overline{\mathsf{bung}})$$

where $\oplus$ is internal choice. In fact the interaction represented by the parallel composition $\rho \parallel \sigma$, that evolves by synchronising corresponding actions and co-actions, might lead to $\mathsf{house} \parallel \overline{\mathsf{bung}}$. This means that the customer is offered a bungalow in the mountains she is not willing to reserve.

Now consider the *dual* behaviour of ρ, dubbed $\overline{\rho}$, which is obtained by exchanging actions by the respective co-actions, and external by internal choices. Then we get the server:

$$\overline{\rho} = \overline{\mathsf{sea}}.(\overline{\mathsf{house}} \oplus \overline{\mathsf{bung}}) \;\oplus\; \overline{\mathsf{mount}}.\overline{\mathsf{house}}$$

and clearly we get $\rho \dashv \overline{\rho}$. In general we expect that $\rho \dashv \overline{\rho}$, or equivalently that $\overline{\sigma} \dashv \sigma$, since duality is involutive.

Taking a further step, let us consider a server such that, after sending the offer $\overline{\mathsf{sea}}$ followed by $\overline{\mathsf{house}}$, might realise that a better offer is now available which can be issued by sending $\overline{\mathsf{bung}}$ instead of $\overline{\mathsf{house}}$; this can be achieved only by rolling back to the choice $\overline{\mathsf{house}} \oplus \overline{\mathsf{bung}}$. Rollback is however a new feature, that cannot be easily represented by usual process algebra operations [12].

To express rollback we then introduce the symbol '$\blacktriangle$' to mark the point where a session behaviour can backtrack to; we call such a marker a *checkpoint*. We suppose that a suitable mechanism keeps memory of the past, by recording the behaviour $\blacktriangle\sigma$ each time the checkpoint is traversed by synchronising on some action that σ is ready to do. For simplicity we assume that only one "past" can be recorded at any time, so that a new memorisation destroys the old one, leading to a model in which the client and the server can backtrack just to the lastly traversed checkpoint.

By adding some checkpoints to $\overline{\rho}$ we get for example $\sigma' = \blacktriangle(\overline{\mathsf{sea}}.\blacktriangle(\overline{\mathsf{house}} \oplus \overline{\mathsf{bung}}) \;\oplus\; \overline{\mathsf{mount}}.\overline{\mathsf{house}})$. With respect to $\overline{\rho}$ the new server can undo all of the internal choices, in order to keep the negotiation open as much as possible and to give to the client some better chance for booking a place, even in case it wasn't available at the beginning of the interaction. But how should the client be redesigned to interact properly? Unfortunately the most natural choice of taking the client as the dual $\overline{\sigma'} = \blacktriangle(\mathsf{sea}.\blacktriangle(\mathsf{house} + \mathsf{bung}) + \mathsf{mount}.\mathsf{house})$ fails. In fact, writing $\xrightarrow{\mathsf{fw}}$ for the forward step and $\xrightarrow{\mathsf{rollbk}}$ for the synchronous rollback, we have among the possible interactions between σ' and $\overline{\sigma'}$:

	$\blacktriangle(\mathsf{sea}.\blacktriangle(\mathsf{house} + \mathsf{bung}) \;+\; \mathsf{mount}.\mathsf{house}) \parallel \blacktriangle(\overline{\mathsf{sea}}.\blacktriangle(\overline{\mathsf{house}} \oplus \overline{\mathsf{bung}}) \;\oplus\; \overline{\mathsf{mount}}.\overline{\mathsf{house}})$	
$\xrightarrow{\mathsf{fw}}$	$\blacktriangle(\mathsf{sea}.\blacktriangle(\mathsf{house} + \mathsf{bung}) \;+\; \mathsf{mount}.\mathsf{house}) \parallel \overline{\mathsf{sea}}.\blacktriangle(\overline{\mathsf{house}} \oplus \overline{\mathsf{bung}})$	*internal choice*
$\xrightarrow{\mathsf{fw}}$	$\blacktriangle(\mathsf{house} + \mathsf{bung}) \parallel \blacktriangle(\overline{\mathsf{house}} \oplus \overline{\mathsf{bung}})$	*synchronising on* sea *and* $\overline{\mathsf{sea}}$
$\xrightarrow{\mathsf{fw}}$	$\blacktriangle(\mathsf{house} + \mathsf{bung}) \parallel \overline{\mathsf{house}}$	*internal choice*
$\xrightarrow{\mathsf{rollbk}}$	$\blacktriangle(\mathsf{sea}.\blacktriangle(\mathsf{house} + \mathsf{bung}) \;+\; \mathsf{mount}.\mathsf{house}) \parallel \blacktriangle(\overline{\mathsf{house}} \oplus \overline{\mathsf{bung}})$	*rollback to the last traversed* $\blacktriangle$

which is now in a stuck state. The mismatch between external and internal choice is the effect of the asymmetry of the respective semantics in process algebra. The selection of a branch in an external choice is just one step; on the contrary the synchronisation on $\overline{\mathsf{sea}}$ in the second step above comes *after* the internal choice has occurred. This has consequences with respect to the backtracking, since the checkpoint alignment fails.

In [1] it has been proved that the dual of a server is the minimum client that complies with the server with respect to a natural (and efficiently decidable) ordering, and vice versa the dual of a client is the minimum compliant server. This is an essential feature of the theory, since it is supposed to model a scenario in which clients look for servers through a network querying a service of a certain shape, that is easier to find if we know its minimal form. To express this precisely, let us write $\rho \dashv^{\blacktriangle} \sigma$ to denote the compliance of ρ with σ in a setting with backtracking, that we call *checkpoint compliance*; then we put the requirement that in the new theory the following holds:

$$\forall \rho. \quad \rho \dashv^{\blacktriangle} \overline{\rho} \tag{1}$$

For (1) to hold we change the operational semantics of $\oplus$ by gluing the choice and the synchronisation over a co-action, that can be formalised by the rule:

$$\overline{a}.\sigma_1 \oplus \sigma_2 \xrightarrow{\overline{a}} \sigma_1$$

This has however the unpleasant consequence that $a \dashv^{\blacktriangle} \overline{a} \oplus \overline{b}$, while we have that $a \not\dashv \overline{a} \oplus \overline{b}$, where the compliance $\dashv$ is defined according to the standard LTS [1, 2, 3]. In general, we expect the compliance of behaviours with rollback to be conservative with respect to the compliance without rollback:

$$\forall \rho, \sigma. \quad \rho \dashv^{\blacktriangle} \sigma \;\Rightarrow\; \mathit{erase}(\rho) \dashv \mathit{erase}(\sigma) \tag{2}$$

where *erase* deletes all checkpoints. We will accomplish this by asking that any co-action has a corresponding action in reducing the parallel of internal and external choices.

The essence of this change is that rolling back has to be a synchronous action, and therefore it cannot be the effect of an internal choice, since the latter is unobservable. This is a general principle. Consider the interaction

$$(\mathsf{sea}.\mathsf{house}.\mathsf{mount}.\mathsf{house}) \| (\overline{\mathsf{sea}}.{}_{\blacktriangle}\overline{\mathsf{house}}.\overline{\mathsf{mount}}.\overline{\mathsf{house}})$$

It is the pair of a client willing to book a house at sea *and* a house in the mountains, and a server that can succeed by renting twice a house at seaside! The point is that the client has no way to be aware of what happened and to react according to her own policy, which is instead the case if both are forced to backtrack at the same time. For this to be guaranteed we require that the client and the server either both can or both cannot rollback in all configurations.

We finally observe that it is not necessarily the case that compliant behaviours show some correspondence between the respective checkpoints. For example it holds that:

$${}_{\blacktriangle}(\mathsf{sea}.\mathsf{house}.\mathsf{garden} + \mathsf{house}.\mathsf{garden}) \;\dashv^{\blacktriangle}\; {}_{\blacktriangle}(\overline{\mathsf{sea}}.{}_{\blacktriangle}\overline{\mathsf{house}}.\overline{\mathsf{garden}} \oplus \overline{\mathsf{house}}.\overline{\mathsf{garden}})$$

which makes sense, since the client ${}_{\blacktriangle}(\mathsf{sea}.\mathsf{house}.\mathsf{garden} + \mathsf{house}.\mathsf{garden})$ is asking for a house with garden, either at sea or anywhere else.

2 Calculus. As explained in the Introduction, we allow checkpoints only before internal or external choices. Therefore we define session behaviours as in [2, 3] just adding checkpointed choices.

Definition 2.1 (Session Behaviours with Checkpoints)**.** *Let $\mathscr{N}$ be some countable set of symbols and $\overline{\mathscr{N}} = \{\overline{a} \mid a \in \mathscr{N}\}$, with $\mathscr{N} \cap \overline{\mathscr{N}} = \emptyset$. The set* SB *of* **session behaviours with checkpoints** *is defined by the grammar of Figure 1, where I is non-empty and finite, the names and the conames in choices are pairwise distinct and σ is not a variable in* $\mathsf{rec}x.\sigma$.

$$
\begin{array}{rcll}
\sigma,\rho & := & \mid \mathbf{1} & \text{success} \\
 & & \mid \sum_{i\in I} a_i.\sigma_i & \text{external choice} \\
 & & \mid {}_{\blacktriangle}\sum_{i\in I} a_i.\sigma_i & \text{checkpointed external choice} \\
 & & \mid \bigoplus_{i\in I} \overline{a}_i.\sigma_i & \text{internal choice} \\
 & & \mid {}_{\blacktriangle}\bigoplus_{i\in I} \overline{a}_i.\sigma_i & \text{checkpointed internal choice} \\
 & & \mid x & \text{variable} \\
 & & \mid \mathsf{rec}\, x.\sigma & \text{recursion}
\end{array}
$$

Figure 1: Syntax of session behaviours with checkpoints

Note that recursion in SB is guarded and hence contractive in the usual sense. We take an equi-recursive view of recursion by equating $\mathsf{rec}\, x.\sigma$ with $\sigma[\mathsf{rec}\, x.\sigma/x]$. Hence there is no point in considering also terms of the shape ${}_{\blacktriangle}\mathsf{rec}\, x.\sigma$.

Let us call just *behaviours* the expressions in SB. In the operational semantics of the calculus we have to record the last encountered behaviour γ that was prefixed by a checkpoint in the interaction leading to σ. Therefore we will consider configurations of the shape:

$$\gamma \prec \sigma$$

In the starting configuration or just after a rollback has occurred, there is no further point to which the behaviour might rollback, a situation we represent by writing $\circ \prec \sigma$. Let $\mathsf{SB}^{\blacktriangle}$ be the set of behaviours starting with ${}_{\blacktriangle}$; then we ask $\gamma \in \mathsf{SB}^{\blacktriangle} \cup \{\circ\}$, which is the set of the "pasts", and denote by γ, δ, possibly with superscripts, its elements. Then the LTS of clients and servers is formalised as follows.

Definition 2.2 (Reduction of Session Behaviours)**.**

$$\gamma \prec \sum_{i\in I} a_i.\sigma_i \xrightarrow{a_i} \gamma \prec \sigma_i \ (i \in I)\ (+) \qquad \gamma \prec \bigoplus_{i\in I} \overline{a}_i.\sigma_i \xrightarrow{\overline{a}_i} \gamma \prec \sigma_i \ (i \in I)\ (\oplus)$$

$$\frac{\gamma \prec \sigma \xrightarrow{\alpha} \gamma \prec \sigma' \quad \alpha \in \mathscr{N} \cup \overline{\mathscr{N}}}{\gamma \prec {}_{\blacktriangle}\sigma \xrightarrow{\alpha} {}_{\blacktriangle}\sigma \prec \sigma'}\ (\blacktriangle) \qquad \sigma \prec \sigma' \xrightarrow{\mathsf{rbk}} \circ \prec \sigma\ (\mathsf{rbk})$$

Notice that Rule $(+)$ is the standard forward computation for external choice, but for the presence of the $\gamma \prec \cdot$. Rule $(\oplus)$ glues into just one step both the internal choice and the communication of a coname, becoming very similar to the rule for external choice. The reduction of client/server parallel compositions (Definition 2.4 below) will be only possible when all internal choices can be matched by the corresponding external choices, which has the effect of saving the conservativity principle (2) of the Introduction. Rule $(\blacktriangle)$ says that in the presence of a checkpoint the forward reduction must update the behaviour at which it is possible to rollback (in this case ${}_{\blacktriangle}\sigma$). Rule (rbk) implements the rollback: the previous past behaviour is erased, the behaviour prefixed by the last traversed checkpoint becomes the new past behaviour and no further rollback is allowed in the new configuration.

When composing in parallel clients and servers we have to consider the different nature of the reductions for internal and external choices. To this aim it is handy to collect the sets of names and conames prefixing the choices, as done in the following definition. Notice that the resulting sets only contain names, since each coname is mapped to the corresponding name.

Definition 2.3 ($\mathscr{A}^+(\cdot)$, $\mathscr{A}^\oplus(\cdot)$)**.** *Let*

$$\mathscr{A}^+(\mathbf{1}) = \mathscr{A}^+(\bigoplus_{i\in I}\overline{a}_i.\sigma_i) = \emptyset \qquad \mathscr{A}^+(\sum_{i\in I}a_i.\sigma_i) = \{a_i \mid i\in I\} \qquad \mathscr{A}^+(\blacktriangle\sigma) = \mathscr{A}^+(\sigma)$$
and
$$\mathscr{A}^\oplus(\mathbf{1}) = \mathscr{A}^\oplus(\sum_{i\in I}a_i.\sigma_i) = \emptyset \qquad \mathscr{A}^\oplus(\bigoplus_{i\in I}\overline{a}_i.\sigma_i) = \{a_i \mid i\in I\} \qquad \mathscr{A}^\oplus(\blacktriangle\sigma) = \mathscr{A}^\oplus(\sigma)$$

The interaction of a client with a server is modelled by the reduction of their parallel composition, that can be either forward, consisting of CCS style synchronisations, or backward, where both behaviours synchronously go back to the respective last traversed checkpointed behaviours.

Definition 2.4 (Communication Reduction of Client and Server Pairs)**.**

$$\frac{\delta \prec \rho \xrightarrow{a} \delta' \prec \rho' \qquad \gamma \prec \sigma \xrightarrow{\overline{a}} \gamma' \prec \sigma' \qquad \mathscr{A}^\oplus(\sigma) \subseteq \mathscr{A}^+(\rho)}{\delta \prec \rho \parallel \gamma \prec \sigma \xrightarrow{\tau} \delta' \prec \rho' \parallel \gamma' \prec \sigma'}$$

$$\frac{\delta \prec \rho \xrightarrow{\overline{a}} \delta' \prec \rho' \qquad \gamma \prec \sigma \xrightarrow{a} \gamma' \prec \sigma' \qquad \mathscr{A}^\oplus(\rho) \subseteq \mathscr{A}^+(\sigma)}{\delta \prec \rho \parallel \gamma \prec \sigma \xrightarrow{\tau} \delta' \prec \rho' \parallel \gamma' \prec \sigma'}$$

$$\frac{\rho \prec \rho' \xrightarrow{\mathsf{rbk}} \circ \prec \rho \qquad \sigma \prec \sigma' \xrightarrow{\mathsf{rbk}} \circ \prec \sigma}{\rho \prec \rho' \parallel \sigma \prec \sigma' \xrightarrow{\mathsf{rbk}} \circ \prec \rho \parallel \circ \prec \sigma}$$

We denote by $\xrightarrow{*}$ the reflexive and transitive closure of forward reductions.

It is easy to verify that if $\circ \prec \rho \parallel \circ \prec \sigma \xrightarrow{*} \circ \prec \rho' \parallel \circ \prec \sigma'$, then $\rho \parallel \sigma$ reduces to $\rho' \parallel \sigma'$ in the calculi of [2, 3], by splitting in two steps each application of rule $(\oplus)$. If $\rho \parallel \sigma$ reduces to $\rho' \parallel \sigma'$ in the calculi of [2, 3] we can find ρ'', σ'' such that both $\rho' \parallel \sigma'$ reduces to $\rho'' \parallel \sigma''$ and

$$\circ \prec \rho \parallel \circ \prec \sigma \xrightarrow{*} \circ \prec \rho'' \parallel \circ \prec \sigma''.$$

We take $\rho'' \parallel \sigma''$ as different than $\rho' \parallel \sigma'$ only in case the last applied rule is an internal choice, which in rule $(\oplus)$ is fused with the communication of the coname.

The last definition makes it clear that the characterisation of compliance in the present calculus requires some care, since the last checkpointed behaviours of clients and servers must be compliant. We formalise this intuition in the next section.

3 Compliance. The compliance relation of session behaviour calculi requires that whenever there is no possible reduction, then all client requests and offers are satisfied, i.e. it is $\mathbf{1}$. In presence of backward computations we have also to require that the client and the server either both can or both cannot reverse to their last encountered checkpoints. This leads to the following definition, in which the set of configurations is denoted by $\mathsf{SB}_\prec$, i.e. $\mathsf{SB}_\prec = \{\gamma \prec \sigma \mid \gamma \in \mathsf{SB}^\blacktriangle \cup \{\circ\}, \sigma \in \mathsf{SB}\}$

Definition 3.1 (Checkpoint Compliance Relation $\dashv^\blacktriangle$)**.** *i) Let $\mathscr{H} : \mathscr{P}(\mathsf{SB}_\prec \times \mathsf{SB}_\prec) \to \mathscr{P}(\mathsf{SB}_\prec \times \mathsf{SB}_\prec)$ be such that, for any $\mathcal{R} \subseteq \mathsf{SB}_\prec \times \mathsf{SB}_\prec$, we get $(\delta \prec \rho, \gamma \prec \sigma) \in \mathscr{H}(\mathcal{R})$ if:*

1) $\delta \prec \rho \parallel \gamma \prec \sigma \not\xrightarrow{\tau}$ implies $\rho = \mathbf{1}$ and either $\delta = \gamma = \circ$ or $\delta, \gamma \in \mathsf{SB}^\blacktriangle$;

2) $\delta \prec \rho \parallel \gamma \prec \sigma \xrightarrow{\beta} \delta' \prec \rho' \parallel \gamma' \prec \sigma'$ implies $\delta' \prec \rho' \; \mathcal{R} \; \gamma' \prec \sigma'$, where $\beta \in \{\tau, \mathsf{rbk}\}$.

ii) A relation $\mathcal{R} \subseteq \mathsf{SB}_\prec \times \mathsf{SB}_\prec$ is a checkpoint compliance relation *if $\mathcal{R} \subseteq \mathscr{H}(\mathcal{R})$. The relation $\dashv^\blacktriangle$ is the greatest solution of the equation $X = \mathscr{H}(X)$:*

$$\dashv^\blacktriangle \; = \nu\mathscr{H}$$

iii) We say that ρ is checkpoint compliant *with σ (notation $\rho \dashv^\blacktriangle \sigma$) if $\circ \prec \rho \dashv^\blacktriangle \circ \prec \sigma$.*

Roughly, when $\rho \dashv^{\blacktriangle} \sigma$ holds, ρ and σ are compliant in the standard sense and they keep on being so after any possible synchronous rollback that can occur during a standard interaction. Moreover it can never be the case that one of them can perform a rollback and the other one cannot, also when ρ is in the success configuration.

It is easy to verify that Definition 3.1(iii) satisfies the requirements (1) and (2) discussed in the Introduction. Namely that each session behaviour is checkpoint compliant with its dual, and that if a client and a server are checkpoint compliant, then the client and the server obtained by erasing the checkpoints are compliant. More formally, if the $erase(\cdot)$ mapping deletes all checkpoints:

Proposition 3.2. *1.* $\forall \rho. \quad \rho \dashv^{\blacktriangle} \overline{\rho}$.

2. $\forall \rho, \sigma. \quad \rho \dashv^{\blacktriangle} \sigma \;\Rightarrow\; erase(\rho) \dashv erase(\sigma)$.

In the following we will use the notation ${}_{\triangle}\sigma$ to represent ambiguously σ and ${}_{\blacktriangle}\sigma$.
In order to give a formal system characterising checkpoint compliance it is handy to define a function $\mathbf{b} : \mathsf{SB}^{\blacktriangle} \cup \{\circ\} \times \mathsf{SB} \to \mathsf{SB}^{\blacktriangle} \cup \{\circ\}$ which returns the second argument when it is checkpointed, and the first argument otherwise. Formally:

$$\mathbf{b}(\gamma, {}_{\triangle}\sigma) = \begin{cases} {}_{\blacktriangle}\sigma & \text{if } \triangle = \blacktriangle \\ \gamma & \text{otherwise.} \end{cases}$$

Forward reduction in Definition 2.4 can be shortly written in terms of the function $\mathbf{b}$:

Lemma 3.3.

$$\gamma \prec {}_{\triangle}(\textstyle\sum_{i\in I} a_i.\sigma_i) \xrightarrow{a_k} \mathbf{b}(\gamma, {}_{\triangle}(\textstyle\sum_{i\in I} a_i.\sigma_i)) \prec \sigma_i.$$
$$\gamma \prec {}_{\triangle}(\textstyle\bigoplus_{i\in I} \overline{a}_i.\sigma_i) \xrightarrow{\overline{a}_k} \mathbf{b}(\gamma, {}_{\triangle}(\textstyle\bigoplus_{i\in I} \overline{a}_i.\sigma_i)) \prec \sigma_i.$$

We now axiomatically characterise the checkpoint compliance relation by means of a formal system, whose judgments are of the form $\Gamma \rhd \delta \prec \rho \dashv^{\prec} \gamma \prec \sigma$, where Γ is an environment, i.e. a finite set $\Gamma = \{\delta_i \prec \rho_i \dashv^{\prec} \gamma_i \prec \sigma_i\}_{i\in I}$. The rules of the formal system are given in Figure 2, where in writing $\gamma \prec \delta$ we assume that $\delta \in \mathsf{SB}$. We denote by $\dashv^{\prec}$ the formal counterpart of $\dashv^{\blacktriangle}$. We are now in place to establish the soundness and completeness of the formal system in Figure 2.

Theorem 3.4 (Soundness)**.**
If $\Gamma \rhd \delta \prec \rho \dashv^{\prec} \gamma \prec \sigma$ *and* $\delta' \prec \rho' \dashv^{\blacktriangle} \gamma' \prec \sigma'$ *for all* $\delta' \prec \rho' \dashv^{\prec} \gamma' \prec \sigma' \in \Gamma$*, then* $\delta \prec \rho \dashv^{\blacktriangle} \gamma \prec \sigma$.

Proof. (Sketch) By induction on derivations. If the last applied rule is (HYP) it is trivial.

If the last applied rule is (AX) and $\delta = \gamma = \circ$, then condition (i1) of Definition 3.1 is satisfied and condition (i2) of Definition 3.1 is trivially satisfied, since there are no reductions.

If the last applied rule is (AX) and $\Gamma \rhd \circ \prec \delta \dashv^{\prec} \circ \prec \gamma$ we get $\delta, \gamma \in \mathsf{SB}$, which implies $\delta, \gamma \in \mathsf{SB}^{\blacktriangle}$ by construction, so condition (i1) of Definition 3.1 is satisfied. In this case the only possible reduction is $\delta \prec \mathbf{1} \parallel \gamma \prec \sigma \xrightarrow{\mathsf{rbk}} \circ \prec \delta \parallel \circ \prec \gamma$. The premise $\Gamma \rhd \circ \prec \delta \dashv^{\prec} \circ \prec \gamma$ implies by induction $\circ \prec \delta \dashv^{\blacktriangle} \circ \prec \gamma$, so also condition (i2) of Definition 3.1 is satisfied.

If the last applied rule is $(+ \cdot \oplus)$, then condition (i1) of Definition 3.1 is trivially satisfied. In this case by Lemma 3.3 $\delta \prec \rho \parallel \gamma \prec \sigma \xrightarrow{\tau} \mathbf{b}(\delta, \rho) \prec \rho_j \parallel \mathbf{b}(\gamma, \sigma) \prec \sigma_j$ for all $j \in J$. The premise

$$\Gamma' \rhd \mathbf{b}(\delta, \rho) \prec \rho_j \dashv^{\prec} \mathbf{b}(\gamma, \sigma) \prec \sigma_j$$

gives $\mathbf{b}(\delta, \rho) \prec \rho_j \dashv^{\blacktriangle} \mathbf{b}(\gamma, \sigma) \prec \sigma_j$ by induction and since $\dashv^{\blacktriangle}$ is the greatest fix point. If $\delta = \gamma = \circ$ there is no rollback, otherwise

$$\delta \prec \rho \parallel \gamma \prec \sigma \xrightarrow{\mathsf{rbk}} \circ \prec \delta \parallel \circ \prec \gamma.$$

The premise $\Gamma' \rhd \circ \prec \delta \dashv^{\prec} \circ \prec \gamma$ implies by induction $\circ \prec \delta \dashv^{\blacktriangle} \circ \prec \gamma$, so also condition (i2) of Definition 3.1 is satisfied. The proof for rule $(\oplus \cdot +)$ is similar. □

$$\frac{\text{either } \delta=\gamma=\circ \text{ or } \Gamma \rhd \circ \prec \delta \dashv^{\prec} \circ \prec \gamma}{\Gamma \rhd \delta \prec \mathbf{1} \dashv^{\prec} \gamma \prec \sigma}\ (\text{Ax}) \qquad \frac{}{\Gamma, \delta \prec \rho \dashv^{\prec} \gamma \prec \sigma \rhd \delta \prec \rho \dashv^{\prec} \gamma \prec \sigma}\ (\text{Hyp})$$

$$\frac{\forall j \in J.\ \Gamma' \rhd \mathbf{b}(\delta,\rho) \prec \rho_j \dashv^{\prec} \mathbf{b}(\gamma,\sigma) \prec \sigma_j \qquad \text{either } \delta=\gamma=\circ \text{ or } \Gamma' \rhd \circ \prec \delta \dashv^{\prec} \circ \prec \gamma}{\Gamma \rhd \delta \prec \rho \dashv^{\prec} \gamma \prec \sigma}\ (+\cdot\oplus)$$

where $\Gamma'=\Gamma,\ \delta \prec \rho \dashv^{\prec} \gamma \prec \sigma$ and $\rho = {}_{\triangle 1}(\sum_{i\in I\cup J} a_i.\rho_i)$ and $\sigma = {}_{\triangle 2}(\bigoplus_{j\in J} \overline{a}_j.\sigma_j)$

$$\frac{\forall i \in I.\ \Gamma' \rhd \mathbf{b}(\delta,\rho) \prec \rho_i \dashv^{\prec} \mathbf{b}(\gamma,\sigma) \prec \sigma_i \qquad \text{either } \delta=\gamma=\circ \text{ or } \Gamma' \rhd \circ \prec \delta \dashv^{\prec} \circ \prec \gamma}{\Gamma \rhd \delta \prec \rho \dashv^{\prec} \gamma \prec \sigma}\ (\oplus\cdot+)$$

where $\Gamma'=\Gamma,\ \delta \prec \rho \dashv^{\prec} \gamma \prec \sigma$ and $\rho = {}_{\triangle 1}(\bigoplus_{i\in I} a_i.\rho_i)$ and $\sigma = {}_{\triangle 2}(\sum_{j\in I\cup J} \overline{a}_j.\sigma_j)$

Figure 2: The formal system for checkpoint compliance

Theorem 3.5 (Completeness)**.**
If $\delta \prec \rho \dashv^{\blacktriangle} \gamma \prec \sigma$ *and* $\delta' \prec \rho' \dashv^{\blacktriangle} \gamma' \prec \sigma' \in \Gamma$ *for all* $\delta' \prec \rho' \dashv^{\prec} \gamma' \prec \sigma' \in \Gamma$, *then* $\Gamma \rhd \delta \prec \rho \dashv^{\prec} \gamma \prec \sigma$.

Proof. (Sketch) By co-induction on the definition of $\dashv^{\blacktriangle}$. If $\delta \prec \rho \parallel \gamma \prec \sigma \stackrel{\tau}{\not\longrightarrow}$, then $\rho=\mathbf{1}$ and either $\delta=\gamma=\circ$ or $\delta,\gamma \in \mathsf{SB}^{\blacktriangle}$ by condition (i1) of Definition 3.1. In the second case

$$\delta \prec \mathbf{1} \parallel \gamma \prec \sigma \stackrel{\mathsf{rbk}}{\longrightarrow} \circ \prec \delta \parallel \circ \prec \gamma,$$

which implies $\circ \prec \delta \dashv^{\blacktriangle} \circ \prec \gamma$ by condition (i2) of Definition 3.1. So in all cases axiom (Ax) applies.

If $\delta \prec \rho \parallel \gamma \prec \sigma \stackrel{\tau}{\longrightarrow}$, then either $\rho = {}_{\triangle 1}(\sum_{i\in I\cup J} a_i.\rho_i)$ and $\sigma = {}_{\triangle 2}(\bigoplus_{j\in J} \overline{a}_j.\sigma_j)$ or $\rho = {}_{\triangle 1}(\bigoplus_{i\in I} \overline{a}_i.\rho_i)$ and $\sigma = {}_{\triangle 2}(\sum_{j\in I\cup J} a_j.\sigma_j)$. We consider the first case, the proof for the second case being similar. In this case $\delta \prec \rho \parallel \gamma \prec \sigma \stackrel{\tau}{\longrightarrow} \mathbf{b}(\delta,\rho) \prec \rho_j \parallel \mathbf{b}(\gamma,\sigma) \prec \sigma_j$ for all $j \in J$ by Lemma 3.3. This implies

$$\mathbf{b}(\delta,\rho) \prec \rho_j \dashv^{\blacktriangle} \mathbf{b}(\gamma,\sigma) \prec \sigma_j$$

by condition (i2) of Definition 3.1. If $\delta \prec \mathbf{1} \parallel \gamma \prec \sigma \stackrel{\mathsf{rbk}}{\longrightarrow} \circ \prec \delta \parallel \circ \prec \gamma$ we get also $\circ \prec \delta \dashv^{\blacktriangle} \circ \prec \gamma$ by condition (i2) of Definition 3.1. So in all cases rule $(+\cdot\oplus)$ applies. □

The main result of our paper is that the formal system provides a complete axiomatic characterisation of the checkpoint compliance, which leads to an decision procedure for checkpoint compliance:

Theorem 3.6 (Main Theorem)**.** *The formal system* $\rhd$ *characterises checkpoint compliance, i.e.*

$$\rho \dashv^{\blacktriangle} \sigma \quad \textit{iff} \quad \rhd \circ \prec \rho \dashv^{\prec} \circ \prec \sigma.$$

4 Related work and conclusion. Since the pioneering work by Danos and Krivine [6], reversible computations in process algebras have been widely studied. The calculus of [6] adds a distributed monitoring system to CCS [11] allowing to rewind computations. Phillips and Ulidowski [12] propose a

method for reversing process operators that are definable by SOS rules in a general format, using keys to bind synchronised actions together. A reversible variant of the higher-order π-calculus is defined in [10], using name tags for identifying threads and explicit memory processes. In [9] Lanese et al. enrich the calculus of [10] with a fine-grained rollback primitive. The closest paper to ours is [13], where Tiezzi and Yoshida study the interplay between reverse computations and session-based interactions. Their calculus uses tags and memories as previous proposals in the literature on reversibility.

As pointed out in [12], reversibility in process calculi is challenging, since we cannot distinguish between the processes $a\|a$ and $a.a$ by simply recording the past actions. For this reason both histories and unique identifiers for threads have been used to track information. A key requirement, dubbed *causal consistency* in [6], is that of undoing only actions if no other action depending on them has been executed (and not undone). Session behaviours overcome all these problems: in fact both the client and the server reduce in a sequential way. This justifies the relative simplicity of our calculus.

We plan to investigate whether our approach can be extended to multi-party sessions [8], the rational being that the parallelism is limited since the interactions must follow the communication protocols prescribed by global types. The subbehaviour relation induced by our notion of compliance is also worth being thoroughly studied.

Acknowledgements The authors gratefully thank the referees for their numerous constructive remarks.

REFERENCES.

[1] Franco Barbanera & Ugo de'Liguoro (2010): *Two notions of sub-behaviour for session-based client/server systems*. In: *PPDP*, ACM Press, pp. 155–164, doi:10.1145/1836089.1836109.

[2] Franco Barbanera & Ugo de' Liguoro (2014): *Sub-behaviour relations for session-based client/server systems*. *Math. Struct. in Comp. Science.* To appear.

[3] Giovanni Bernardi & Matthew Hennessy (2014): *Modelling session types using contracts*. *Math. Struct. in Comp. Science.* To appear.

[4] S. Carpineti, G. Castagna, C. Laneve & L. Padovani (2006): *A formal account of contracts for Web Services*. In: *WS-FM*, *LNCS* 4184, Springer, pp. 148–162, doi:10.1007/11841197_10.

[5] Giuseppe Castagna, Nils Gesbert & Luca Padovani (2009): *A theory of contracts for Web services*. *ACM Trans. on Prog. Lang. and Sys.* 31(5), pp. 19:1–19:61, doi:10.1145/1538917.1538920.

[6] Vincent Danos & Jean Krivine (2004): *Reversible Communicating Systems*. In: *CONCUR*, *LNCS* 3170, Springer, pp. 292–307, doi:10.1007/978-3-540-28644-8_19.

[7] Kohei Honda, Vasco T. Vasconcelos & Makoto Kubo (1998): *Language Primitives and Type Disciplines for Structured Communication-based Programming*. In: *ESOP*, *LNCS* 1381, Springer, pp. 22–138, doi:10.1007/BFb0053567.

[8] Kohei Honda, Nobuko Yoshida & Marco Carbone (2008): *Multiparty Asynchronous Session Types*. In: *POPL*, ACM Press, pp. 273–284, doi:10.1145/1328897.1328472.

[9] I. Lanese, C. A. Mezzina, A. Schmitt & J.-B. Stefani (2011): *Controlling Reversibility in Higher-Order Pi*. In: *CONCUR*, *LNCS* 6901, Springer, pp. 297–311, doi:10.1007/978-3-642-23217-6_20.

[10] Ivan Lanese, Claudio Antares Mezzina & Jean-Bernard Stefani (2010): *Reversing Higher-Order Pi*. In: *CONCUR*, *LNCS* 6269, Springer, pp. 478–493, doi:10.1007/978-3-642-15375-4_33.

[11] Robin Milner (1989): *Communication and concurrency*. PHI Series in computer science, Prentice Hall.

[12] Iain C. C. Phillips & Irek Ulidowski (2007): *Reversing algebraic process calculi*. *J. of Logic and Alg. Progr.* 73(1-2), pp. 70–96, doi:10.1016/j.jlap.2006.11.002.

[13] Francesco Tiezzi & Nobuko Yoshida (2014): *Towards Reversible Sessions*. In: *PLACES*, *EPTCS* 155, pp. 17–24, doi:10.4204/EPTCS.155.3.