HAL open science

A Game Interpretation of Retractable Contracts

Franco Barbanera, Ugo De' Liguoro

To cite this version:

Franco Barbanera, Ugo De' Liguoro. A Game Interpretation of Retractable Contracts. 18th International Conference on Coordination Languages and Models (COORDINATION), Jun 2016, Heraklion, Greece. pp.18-34, 10.1007/978-3-319-39519-7_2 . hal-01631714

HAL Id: hal-01631714

https://inria.hal.science/hal-01631714v1

Submitted on 9 Nov 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

A game interpretation of retractable contracts*

Franco Barbanera[1] and Ugo de' Liguoro[2]

[1] Dipartimento di Matematica e Informatica, University of Catania
barba@dmi.unict.it
[2] Dipartimento di Informatica, University of Torino
ugo.deliguoro@unito.it

Abstract. In the setting of contract theory, retractable contracts have been defined to formalize binary session protocols where the partners can go back to certain particular synchronization points when the session gets stuck, looking for a successful state, if any.
In the present paper we propose a three-party game-theoretic interpretation of client/server systems of retractable contracts. In particular, we show that a client is retractable-compliant with a server if and only if there exists a winning strategy for a particular player in a game-theoretic model of contracts. Such a player can be looked at as a *mediator*, driving the choices in the retractable points. We show that winning strategies for the mediator player correspond to orchestrators in a system of orchestrated client/server sessions, and vice versa.

The notion of *contract* has been proposed as an abstraction to formally specify and check the behaviour of software systems, and especially of web services. In particular, in the setting of service-oriented architectures the concept of agreement, often called *compliance*, is of paramount importance while searching components and ensuring that they will properly collaborate with each other. The main challenge is that compliance has to meet the contrasting requirements of guaranteeing correctness of interactions w.r.t. certain safety and liveness conditions, while remaining coarse enough to maximize the possibilities of finding compliant components in a library or services through the web.

The main conceptual tool to face the issue is that of relaxing the constraint of a perfect correspondence among contracts through *contract refinement*, also called sub-contract [9,8] and sub-behaviour [3] relations, that is pre-order relations such that processes conforming to more demanding contracts (which are lower in the pre-order) can be safely substituted in contexts allowing more permissive ones. Indeed contract refinement closely resembles subtyping, as it is apparent in the case of session types [10,3], and it is related to (but doesn't coincide with) observational pre-orders and *must-testing* in process algebra [11,6].

However, since the first contributions to the theory of contracts [9], a rather different approach has been followed, based on the idea of filtering out certain actions that, althought unmatched on both sides of a binary interaction, can be

* This work was partially supported by the COST Action IC1405 of the European Union and by the Project FIR 1B8C1 of the University of Catania.

neglected or prevented by the action of a mediating process called the *orchestrator* [14,13], without compromising the reaching of the goals of the participants, like the satisfaction of all client requests in a client-server architecture.

An alternative route for the same purpose is to change the semantics of contracts so that interacting processes can adapt each other by means of a rollback mechanism: these are the *retractable contracts* proposed in [4]. Although compliance can be decided in advance, interaction among processes exposing retractable contracts undergoes a sequence of failures and backtracks that might be avoided by extracting information from the compliance check.

The contribution of the present paper is to show that the two approaches of orchestrated and retractable compliance are indeed equivalent, at least in the case of *session contracts* (see [2,3], where they are dubbed "session behaviours"), which are contracts that limit the non-determinism by constraining both external and internal choices to a more regular form. More precisely, we consider contracts that are syntactically the same as retractable ones, but instead of adding rollback to the ususal contract semantics, we abstractly define outputs in an external choice as *affectible* actions: their actual sent can be influenced by the partner in a binary session or by some entity external to the system. Affectible actions correspond to retractable actions in [4].

The essence of the construction is that (an appropriate restriction of) orchestrators correspond to winning strategies in certain concurrent games that naturally model retractable contracts. In [5] the theory of contracts has been grounded on games over event structures among multiple players; applying this framework to retractable contracts, the interaction among a client and a server can be seen as a play in a three-party game. Player A moves according to the unaffectible actions of the client; player B moves according to the unaffectible actions of the server, whereas moves by player C correspond to affectible actions on both sides, namely the retractable agreement points of the system. The client ρ is hence affectible-compliant with the server σ whenever C has a winning strategy in the game with players A and B, where player C wins when she succeeds to lead the system $\rho \| \sigma$ to a successful state (the client terminates) or the interaction proceeds indefinitely without deadlocking.

The payoff of the game theoretic interpretation is that there is a precise correspondence between winning strategies for player C and elements of a class of orchestrators in the sense of [14]. Such a correspondence is of interest on its own, since strategies are abstract entities while orchestrators are terms of a process algebra and concrete witnesses of the agreement among participants of a session. Moreover, we can decide whether a client-server pair is reversible-compliant by means of an algorithm that synthesizes an orchestrator if any, or reports failure.

1 Affectible contracts and retractable compliance

Affectible session contracts (affectible contracts for short) are a variant of *retractable contracts* in [4]; they are syntactically the same, but affectible session

contracts have a different, and more abstract semantics. Nonetheless compliance coincides in both settings as we show in this section.

Definition 1 (Affectible session contracts). *Let* $\mathcal{N}$ (set of names) *be some countable set of symbols and let* $\overline{\mathcal{N}} = \{\, \overline{a} \mid a \in \mathcal{N} \,\}$ (set of conames), *with* $\mathcal{N} \cap \overline{\mathcal{N}} = \emptyset$. *The set* ASC *of* **affectible session contracts** *is defined as the set of the* **closed** *(with respect to the binder* rec *) expressions generated by the following grammar,*

$$
\begin{array}{rll}
\sigma, \rho := \mid & \mathbf{1} & \textit{success} \\
\mid & \sum_{i \in I} a_i.\sigma_i & \textit{input} \\
\mid & \sum_{i \in I} \overline{a}_i.\sigma_i & \textit{affectible output} \\
\mid & \bigoplus_{i \in I} \overline{a}_i.\sigma_i & \textit{unaffectible output} \\
\mid & x & \textit{variable} \\
\mid & \mathsf{rec}\, x.\sigma & \textit{recursion}
\end{array}
$$

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

Affectible as well as retractable contracts stem from *session behaviours* of [3] also called *session contracts* in [6]. With respect to session behaviors, affectible contracts add the affectible output construct, which is called retractable output in [4]. The affectible output represents points where the client-server interaction can be influenced by the partner process, or can be guided by a third party; consequently they are represented by the CCS external choice operator as it is the case of the input branching (which is always affectible). Outputs in an internal choice are regarded as unaffectible actions and treated as unretractable in the setting of retractable contracts. The transitions representing an internal choice have no label; note that any $\bigoplus_{i \in I} \overline{a}_i.\sigma_i$ just reduces to one of its summands.
In the following we consider recursion up-to unfolding, that is we equate $\mathsf{rec}\, x.\sigma$ with $\sigma\{x/\mathsf{rec}\, x.\sigma\}$. The symbol α will be used as a variable ranging over $\mathcal{N} \cup \overline{\mathcal{N}}$.

Definition 2 (LTS for ASC). *Let* $\mathbf{Act} = \mathcal{N} \cup \overline{\mathcal{N}} \cup \{\, \overline{a}^+ \mid \overline{a} \in \overline{\mathcal{N}} \,\}$.

$$
\begin{array}{lll}
(+)\ a.\sigma + \sigma' \xrightarrow{a} \sigma & & (\overline{+})\ \overline{a}.\sigma + \sigma' \xrightarrow{\overline{a}^+} \sigma \\
(\oplus)\ \overline{a}.\sigma \oplus \sigma' \longrightarrow \overline{a}.\sigma & & (\alpha) \qquad \alpha.\sigma \xrightarrow{\alpha} \sigma
\end{array}
$$

A client/server system (system for short) is a pair of contracts in ASC that we denote by $\rho \parallel \sigma$.

Definition 3 (LTS for systems). *Let* $\mathbf{csAct} = \{\, +, \tau \,\}$.

$$
\frac{\rho \to \rho'}{\rho \parallel \sigma \to \rho' \parallel \sigma}
\qquad
\frac{\sigma \to \sigma'}{\rho \parallel \sigma \to \rho \parallel \sigma'}
$$

$$
\frac{\rho \xrightarrow{a} \rho' \qquad \sigma \xrightarrow{\overline{a}^+} \sigma'}{\rho \parallel \sigma \xrightarrow{+} \rho' \parallel \sigma'}
\qquad
\frac{\rho \xrightarrow{\overline{a}^+} \rho' \qquad \sigma \xrightarrow{a} \sigma'}{\rho \parallel \sigma \xrightarrow{+} \rho' \parallel \sigma'}
$$

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

We define $\Longrightarrow = \rightarrow^* \circ \xrightarrow{\tau}$ and $\overset{+}{\Longrightarrow} = \rightarrow^* \circ \xrightarrow{+}$. In the last rule, $\overline{\alpha}$ is the CCS involution of names and co-names.

The semantics of $\rho \parallel \sigma$ is reminiscent of CCS parallel composition as used to define testing preorders in [12], but for the usage of the labels $+$ and τ and for the absence of a success marker (there is a set of success states instead: see below). We use labels $+$ and τ to distinguish among affectible and unaffectible communications respectively, altough they are both unobservable as the only observable facts are termination and the resulting state.

Lemma 1. *Let $\rho, \sigma \in \mathsf{ASC}$. $\rho \parallel \sigma \Longrightarrow$ and $\rho \parallel \sigma \overset{+}{\Longrightarrow}$ can never both occur.*

The affectible compliance relation can be now coinductively defined as follows.

Definition 4 (Affectible Compliance Relation $\dashv^{\mathsf{A}}$).

i) Let $\mathcal{H} : \mathcal{P}(\mathsf{ASC} \times \mathsf{ASC}) \rightarrow \mathcal{P}(\mathsf{ASC} \times \mathsf{ASC})$ be such that, for any $\mathcal{R} \subseteq \mathsf{ASC} \times \mathsf{ASC}$, we get $(\rho, \sigma) \in \mathcal{H}(\mathcal{R})$ if the following conditions hold:

1) $[\ \rho \parallel \sigma \not\Longrightarrow$ and $\rho \parallel \sigma \overset{+}{\not\Longrightarrow}\ ]$ implies $\rho = \mathbf{1}$;
2) $\forall \rho', \sigma'.\ [\ \rho \parallel \sigma \Longrightarrow \rho' \parallel \sigma'$ implies $\rho' \ \mathcal{R}\ \sigma'\ ]$;
3) $\rho \parallel \sigma \overset{+}{\Longrightarrow}$ implies $\exists \rho', \sigma'.\ [\ \rho \parallel \sigma \overset{+}{\Longrightarrow} \rho' \parallel \sigma'$ and $\rho' \ \mathcal{R}\ \sigma'\ ]$.

ii) A relation $\mathcal{R} \subseteq \mathsf{ASC} \times \mathsf{ASC}$ is an affectible compliance relation *if $\mathcal{R} \subseteq \mathcal{H}(\mathcal{R})$. $\dashv^{\mathsf{A}}$ is the greatest solution of the equation $X = \mathcal{H}(X)$, that is $\dashv^{\mathsf{A}} = \nu\mathcal{H}$.*

In words the client ρ is affectible-compliant with the server σ if either ρ and σ cannot communicate because $\rho = \mathbf{1}$, namely all client requirements have been satisfied; or all unaffectible communications of the system $\rho \parallel \sigma$ lead to compliant systems; or there exists an affectible communication leading to a compliant system. By Lemma 1 the last two conditions cannot be simultaneously satisfied.

Because of conditions *i2)* and *i3)*, the affectible compliance relation is an abstract concept; but it can be made concrete via the characterization in terms of retractable computations, provided in section 1.

Let us consider the following example from [4]. A Buyer is looking for a bag ($\overline{\mathtt{bag}}$) or a belt ($\overline{\mathtt{belt}}$); she will decide how to pay, either by credit card ($\overline{\mathtt{card}}$) or by cash ($\overline{\mathtt{cash}}$), after knowing the $\mathtt{price}$ from the Seller.

$$\mathsf{Buyer} = \overline{\mathtt{bag}}.\mathtt{price}.(\overline{\mathtt{card}} \oplus \overline{\mathtt{cash}}) + \overline{\mathtt{belt}}.\mathtt{price}.(\overline{\mathtt{card}} \oplus \overline{\mathtt{cash}})$$

The Seller does not accept credit card payments for items of low price, like belts, but only for more expensive ones, like bags:

$$\mathsf{Seller} = \mathtt{belt}.\overline{\mathtt{price}}.\mathtt{cash} + \mathtt{bag}.\overline{\mathtt{price}}.(\mathtt{card} + \mathtt{cash})$$

From the previous definition it is not difficult to check that $\mathsf{Buyer} \dashv^{\mathsf{A}} \mathsf{Seller}$.

Retractable contracts. Let us recall the formalism of retractable contracts; the following definitions and Theorem 1 below are from [4]. As said before, retractable and affectible contracts are syntactically the same, but the operational semantics of the formers is based on a rollback operation, acting on the recording of certain discarded branches of an interaction. The notion of *contracts with histories* is defined as follows:

Definition 5 (Contracts with histories). *Let* Histories *be the set of expressions (referred to also as* stacks*) generated by the grammar:*

$$\boldsymbol{\gamma} ::= [\,] \mid \boldsymbol{\gamma}{:}\sigma \qquad \textit{where } \sigma \in \mathsf{ASC} \cup \{\circ\}.$$

Then the set of contracts with histories *is defined by:*

$$\mathsf{RCH} = \{\boldsymbol{\gamma} \prec \sigma \mid \boldsymbol{\gamma} \in \mathsf{Histories}, \sigma \in \mathsf{ASC} \cup \{\circ\}\,\}.$$

Histories are finite lists of contracts representing the branches which have been discarded because of a retractable synchronization action. The effect of retracting such an action is modeled by restoring the last contract on the history as the actual contract and by trying a different branch, if any. This is formalised by the operational semantics of contracts with histories that is defined as follows.

Definition 6 (LTS of Contracts with Histories).

$$(+)\ \boldsymbol{\gamma} \prec \alpha.\sigma + \sigma' \xrightarrow{\alpha} \boldsymbol{\gamma}{:}\sigma' \prec \sigma \qquad (\oplus)\ \boldsymbol{\gamma} \prec \overline{a}.\sigma \oplus \sigma' \xrightarrow{\tau} \boldsymbol{\gamma} \prec \overline{a}.\sigma$$

$$(\alpha)\ \boldsymbol{\gamma} \prec \alpha.\sigma \xrightarrow{\alpha} \boldsymbol{\gamma}{:}\circ \prec \sigma \qquad (\mathsf{rb})\ \boldsymbol{\gamma}{:}\sigma' \prec \sigma \xrightarrow{\mathsf{rb}} \boldsymbol{\gamma} \prec \sigma'$$

When selecting a branch of an external choice, the discarded branches are memorised on top of the new stack (the last contract of the history) in the right-hand side of rule $(+)$; on the contrary, when an internal choice occurs, the stack remains unchanged in rule $(\oplus)$. When a single action is executed, the history is modified by adding a '$\circ$', meaning that the only available branch has been tried and no alternative is left. Rule (rb) recovers the contract on the top of the stack (if the stack is different than $[\,]$) by replacing the current one with it. Note that the combined effect of rules $(\oplus)$ and (α) is that the alternative branches of an internal choice are unrecoverable.

The interaction of a client with a server is modeled by the reduction of their parallel composition, that can be either forward, consisting of CCS style synchronisations and single internal choices, or backward if there is no possible forward reduction, the client is different than $\mathbf{1}$ (the fulfilled contract) and rule (rb) is applicable on both sides.

Definition 7 (TS of Client/Server Pairs). *We define the relation* $\longrightarrow$ *over pairs of retractable contracts with histories by the following rules:*

$$\frac{\boldsymbol{\delta} \prec \rho \xrightarrow{\alpha} \boldsymbol{\delta'} \prec \rho' \qquad \boldsymbol{\gamma} \prec \sigma \xrightarrow{\overline{\alpha}} \boldsymbol{\gamma'} \prec \sigma'}{\boldsymbol{\delta} \prec \rho \,\|\, \boldsymbol{\gamma} \prec \sigma \longrightarrow \boldsymbol{\delta'} \prec \rho' \,\|\, \boldsymbol{\gamma'} \prec \sigma'}\ (\mathsf{comm})$$

$$\frac{\boldsymbol{\delta} \prec \rho \xrightarrow{\tau} \boldsymbol{\delta} \prec \rho'}{\boldsymbol{\delta} \prec \rho \,\|\, \boldsymbol{\gamma} \prec \sigma \longrightarrow \boldsymbol{\delta} \prec \rho' \,\|\, \boldsymbol{\gamma} \prec \sigma}\ (\tau)$$

$$\frac{\boldsymbol{\gamma} \prec \rho \xrightarrow{\mathsf{rb}} \boldsymbol{\gamma'} \prec \rho' \qquad \boldsymbol{\delta} \prec \sigma \xrightarrow{\mathsf{rb}} \boldsymbol{\delta'} \prec \sigma' \qquad \rho \neq \mathbf{1}}{\boldsymbol{\gamma} \prec \rho \,\|\, \boldsymbol{\delta} \prec \sigma \longrightarrow \boldsymbol{\gamma'} \prec \rho' \,\|\, \boldsymbol{\delta'} \prec \sigma'}\ (\mathsf{rbk})$$

plus the rule symmetric to (τ) *w.r.t.* $\|$. *Moreover, rule* (rbk) *applies only if neither* (comm) *nor* (τ) *do.*

Up to the rollback mechanism, compliance in the retractable setting is defined as usually done with client/server contracts.

Definition 8 (Retractable Compliance, $\dashv^{\mathsf{tdk}}$).

i) The relation $\dashv^{\mathsf{tdk}}$ on contracts with histories is defined as follows: for any $\boldsymbol{\delta'}, \rho', \boldsymbol{\gamma'}, \sigma'$, $\boldsymbol{\delta} \prec \rho \dashv^{\mathsf{tdk}} \boldsymbol{\gamma} \prec \sigma$ holds whenever

$$\boldsymbol{\delta} \prec \rho \parallel \boldsymbol{\gamma} \prec \sigma \xrightarrow{*} \boldsymbol{\delta'} \prec \rho' \parallel \boldsymbol{\gamma'} \prec \sigma' \not\rightarrow \quad \textit{implies } \rho' = \mathbf{1}$$

ii) The relation $\dashv^{\mathsf{tdk}}$ on contracts is defined by: $\rho \dashv^{\mathsf{tdk}} \sigma$ if $[\,] \prec \rho \dashv^{\mathsf{tdk}} [\,] \prec \sigma$.

In Buyer/Seller example we have that, in case a belt is agreed upon and the buyer decides to pay using her credit card, the system gets stuck in an unsuccessful state. This causes a rollback enabling a successful state to be reached. So Buyer $\dashv^{\mathsf{tdk}}$ Seller.

Retractable compliance can be axiomatised in terms of derivability in a formal system whose statements do not mention histories.

Definition 9 (Formal System $\rhd$ for Retractable Compliance).

$$(\textsc{Ax}) : \frac{}{\Gamma \rhd \mathbf{1} \dashv^{\prec} \sigma} \qquad (\textsc{Hyp}) : \frac{}{\Gamma, \rho \dashv^{\prec} \sigma \rhd \rho \dashv^{\prec} \sigma}$$

$$(+\cdot+) : \frac{\Gamma, \alpha.\rho + \rho' \dashv^{\prec} \overline{\alpha}.\sigma + \sigma' \rhd \rho \dashv^{\prec} \sigma}{\Gamma \rhd \alpha.\rho + \rho' \dashv^{\prec} \overline{\alpha}.\sigma + \sigma'}$$

$$(\oplus\cdot+) : \frac{\forall i \in I.\ \Gamma, \bigoplus_{i\in I}\overline{a}_i.\rho_i \dashv^{\prec} \sum_{j\in I\cup J} a_j.\sigma_j \rhd \rho_i \dashv^{\prec} \sigma_i}{\Gamma \rhd \bigoplus_{i\in I}\overline{a}_i.\rho_i \dashv^{\prec} \sum_{j\in I\cup J} a_j.\sigma_j}$$

$$(+\cdot\oplus) : \frac{\forall i \in I.\ \Gamma, \sum_{j\in I\cup J} a_j.\sigma_j \dashv^{\prec} \bigoplus_{i\in I}\overline{a}_i.\rho_i \rhd \rho_i \dashv^{\prec} \sigma_i}{\Gamma \rhd \sum_{j\in I\cup J} a_j.\sigma_j \dashv^{\prec} \bigoplus_{i\in I}\overline{a}_i.\rho_i}$$

Let us formally show that $\emptyset \rhd \mathsf{Buyer} \dashv^{\prec} \mathsf{Seller}$

$$\dfrac{\dfrac{\dfrac{}{\Gamma'' \rhd \mathbf{1} \dashv^{\prec} \mathbf{1}}(\textsc{Ax}) \qquad \dfrac{}{\Gamma'' \rhd \mathbf{1} \dashv^{\prec} \mathbf{1}}(\textsc{Ax})}{\Gamma' \rhd \overline{\mathtt{card}} \oplus \overline{\mathtt{cash}} \dashv^{\prec} \mathtt{card} + \mathtt{cash}}(\oplus,+)}{\mathsf{Buyer}' \dashv^{\prec} \mathsf{Seller} \rhd \mathtt{price}.(\overline{\mathtt{card}} \oplus \overline{\mathtt{cash}}) \dashv^{\prec} \overline{\mathtt{price}}.(\mathtt{card} + \mathtt{cash})}(+,+)}{\rhd \mathsf{Buyer} \dashv^{\prec} \mathsf{Seller}}(+,+)$$

where $\Gamma' = \mathsf{Buyer} \dashv^{\prec} \mathsf{Seller},\ \mathtt{price}.(\overline{\mathtt{card}} \oplus \overline{\mathtt{cash}}) \dashv^{\prec} \overline{\mathtt{price}}.(\mathtt{card} + \mathtt{cash})$
and $\Gamma'' = \Gamma',\ \overline{\mathtt{card}} \oplus \overline{\mathtt{cash}} \dashv^{\prec} \mathtt{card} + \mathtt{cash}$

The formal system $\rhd$ completely axiomatises retractable compliance:

Theorem 1 (Soundness and Completeness of system $\rhd$ w.r.t $\dashv^{\mathsf{tdk}}$).
$\rho \dashv^{\mathsf{tdk}} \sigma$ *if and only if* $\rhd \rho \dashv^{\prec} \sigma$.

Equivalence of $\dashv^{\mathsf{A}}$ and $\dashv^{\mathsf{tbk}}$. As previously observed, the judgements of system $\rhd$ abstract away from histories, which are essential in the definition of rollback. This is possible because rollback is just a backtracking mechanism, which is however limited to the exploration of alternative branches of the reduction tree of a system rooted at retractable communications. Since affectible and retractable communications are the same, it is natural to look at system $\rhd$ to establish the equivalence among $\dashv^{\mathsf{A}}$ and $\dashv^{\mathsf{tbk}}$.

Lemma 2. *If $\rho \dashv^{\mathsf{A}} \sigma$, then 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* $\exists h \in I \cap J.\ \rho_h \dashv^{\mathsf{A}} \sigma_h$;
3. $\rho = \bigoplus_{i\in I} \overline{a}_i.\rho_i$, $\sigma = \sum_{j\in J} a_j.\sigma_j$, $I \subseteq J$ *and* $\forall h \in I.\ \rho_h \dashv^{\mathsf{A}} \sigma_h$;
4. $\rho = \sum_{i\in I} a_i.\rho_i$, $\sigma = \bigoplus_{j\in J} \overline{a}_j.\sigma_j$, $I \supseteq J$ *and* $\forall h \in J.\ \rho_h \dashv^{\mathsf{A}} \sigma_h$.

In Theorem 1, soundness and completeness of system $\rhd$ has been proved when the symbol $\dashv^{\prec}$ is interpreted as the retractable compliance relation $\dashv^{\mathsf{tbk}}$. We now show that system $\rhd$ is sound and complete also when the symbol $\dashv^{\prec}$ is interpreted as the affectible compliance relation $\dashv^{\mathsf{A}}$. The equivalence of the relations $\dashv^{\mathsf{tbk}}$ and $\dashv^{\mathsf{A}}$ follows then as an immediate corollary.

Definition 10 (A $\dashv^{\mathsf{A}}$-semantics for system $\rhd$).
Let Γ be a set of statements of the form $\rho \dashv^{\prec} \sigma$. We define

i) $\models^{\mathsf{A}} \Gamma$ *if* $\forall(\rho' \dashv^{\prec} \sigma') \in \Gamma.\ [\rho' \dashv^{\mathsf{A}} \sigma']$;
ii) $\Gamma \models^{\mathsf{A}} \rho \dashv^{\prec} \sigma$ *if* $\models^{\mathsf{A}} \Gamma \Rightarrow \rho \dashv^{\mathsf{A}} \sigma$.

The proof of the following Lemma is inspired to [7].

Lemma 3 (Soundness of $\rhd$ w.r.t $\dashv^{\mathsf{A}}$). *If $\Gamma \rhd \rho \dashv^{\prec} \sigma$, then $\Gamma \models^{\mathsf{A}} \rho \dashv^{\prec} \sigma$.*

We write $\mathcal{D} :: \Gamma \rhd \rho \dashv^{\prec} \sigma$ when $\mathcal{D}$ is a derivation in the system $\rhd$ with conclusion $\Gamma \rhd \rho \dashv^{\prec} \sigma$. We can easily implement a backward proof search (from conclusion to premises) in the formal system $\rhd$ by means of a procedure **Prove**.

Lemma 4. *i)* $\mathbf{Prove}(\Gamma \rhd \rho \dashv^{\prec} \sigma) = \mathcal{D} \neq \mathbf{fail}$ *implies* $\mathcal{D} :: \Gamma \rhd \rho \dashv^{\prec} \sigma$;
ii) $\mathbf{Prove}(\Gamma \rhd \rho \dashv^{\prec} \sigma)$ *terminates for all judgments* $\Gamma \rhd \rho \dashv^{\prec} \sigma$.

Lemma 5 (Completeness of $\rhd$ w.r.t $\dashv^{\mathsf{A}}$). *If $\rho \dashv^{\mathsf{A}} \sigma$, then $\rhd \rho \dashv^{\prec} \sigma$.*

Proof. (Sketch) If $\rho \dashv^{\mathsf{A}} \sigma$ then by Lemma 2 there are four possibilities; disregarding the contexts Γ's, we see that each of these cases corresponds exactly to one rule in system $\rhd$, where **Prove** is recursively applied to the respective premises, but for rule (Hyp), that corresponds to an exit clause in **Prove**. It follows that $\mathbf{Prove}(\rhd \rho \dashv^{\prec} \sigma) \neq \mathbf{fail}$, so that the thesis follows by Lemma 4, since **Prove** always terminates either returning a correct derivation or **fail**.

Corollary 1. $\dashv^{\mathsf{tbk}} \quad = \quad \dashv^{\mathsf{A}}$

Proof. By Lemmas 3 and 5 and Theorem 1

2 Game-theoretic interpretation of retractable contracts

Following [5] we interpret affectible contracts as certain games over event structures. This yields a game-theoretic interpretation of affectible contracts, and hence of retractable contracts by Corollary 1. For the reader's convenience we briefly recall the basic notions of event structure and game associated to an LTS.

Definition 11 (Event structure [15]). *Let* $\mathbf{E}$ *be a denumerable universe of* events *and let* $\mathbf{A}$ *be a universe of* action labels. *Besides, let* $\# \subseteq E \times E$ *be an irreflexive and symmetric relation (called* conflict relation*).*

i) The predicate CF on sets $X \subseteq E$ *and the set Con of finite* conflict-free *sets are defined by* $CF(X) = \forall e, e' \in X. \neg(e \# e')$ $\quad Con = \{\, X \subseteq_{fin} E \mid CF(X) \,\}$

ii) An event structure *is a quadruple* $\mathcal{E} = (E, \#, \vdash, l)$ *where*
- $\vdash \subseteq Con \times E$ *is a relation such that* $sat(\vdash) = \vdash$ *(i.e.* $\vdash$ *is* saturated*), where* $sat(\vdash) = \{\, (Y, e) \mid X \vdash e \ \ \&\ \ X \subseteq Y \in Con \,\}$*;*
- $l : E \to \mathbf{A}$ *is a labelling function.*

Given a set E of events, E^∞ denotes the set of sequences (both finite and infinite) of its elements. We denote by $\boldsymbol{e} = \langle e_0 e_1 \cdots \rangle$ a sequence of events[3]. Given $\boldsymbol{e}$, we denote by $\widehat{\boldsymbol{e}}$ the set of its elements, by $|\boldsymbol{e}|$ its length (either a natural number or ∞) and by $\boldsymbol{e}_{/i}$ for $i < |\boldsymbol{e}|$ the subsequence $\langle e_0 e_1 \cdots e_{i-1} \rangle$ of its first i elements. Given a set X we denote by $|X|$ its cardinality. $\mathbb{N}$ is the set of natural numbers.

Definition 12 (LTS over configurations [5]). *Given an event structure* $\mathcal{E} = (E, \#, \vdash, l)$*, we define the LTS* $(\mathcal{P}_{fin}(E), E, \to_{\mathcal{E}})$ *as follows:*

$$C \xrightarrow{e} C \cup \{\, e \,\} \qquad \textit{if} \qquad C \vdash e,\ e \notin C \ \ \textit{and}\ \ CF(C \cup \{\, e \,\})$$

Given an LTS $(S, \to)$ and a state $s \in S$, we denote by $(s, \to)$ the restriction of $\to$ to the transitions starting with the state s, and by $\mathsf{Tr}(s, \to)$ the set of the (finite or infinite) traces in $(s, \to)$ out of s.

Multi-player games. All the subsequent definitions and terminology are from [5], except in the case of games that we call multi-player instead of "contracts", which would be confusing in the present setting.

A set of partecipants (players) to a game will be denoted by $\mathfrak{P}$, whereas the universe of partecipants is denoted by $\mathfrak{P}_{\mathfrak{U}}$. We shall use A, B,... as variables ranging over $\mathfrak{P}$ or $\mathfrak{P}_{\mathfrak{U}}$. The symbols A, B, ... will denote particular elements of $\mathfrak{P}$ or $\mathfrak{P}_{\mathfrak{U}}$. We assume that each event is associated to a player by means of a function $\pi : \mathbf{E} \to \mathfrak{P}_{\mathfrak{U}}$. Moreover, given $A \in \mathfrak{P}_{\mathfrak{U}}$ we define $\mathbf{E}_A = \{\, e \in \mathbf{E} \mid \pi(e) = A \,\}$.

Definition 13 (Multi-player game).

i) A game $\mathcal{G}$ *is a pair* $(\mathcal{E}, \Phi)$ *where* $\mathcal{E} = (E, \#, \vdash, l)$ *is an event structure and* $\Phi : \mathfrak{P}_{\mathfrak{U}} \rightharpoonup E^\infty \to \{\, -1, 0, 1 \,\}$ *associates each participant and trace with a* payoff*. Moreover, for all* $X \vdash e$ *in* $\mathcal{E}$*,* $\Phi(\pi(e))$ *is defined. We say that* $\mathcal{G}$ *is a game with partecipants* $\mathfrak{P}$ *whenever* ΦA *is defined for any player* A *in* $\mathfrak{P}$*.*

[3] Differently than in [5], we use the notation $\boldsymbol{e}$ for sequences instead of σ, which refers to a contract here.

ii) A play *of a game* $\mathcal{G} = (\mathcal{E}, \Phi)$ *is a (finite or infinite) trace of* $(\emptyset, \rightarrow_{\mathcal{E}})$ *i.e. an element of* $\mathsf{Tr}(\emptyset, \rightarrow_{\mathcal{E}})$.

Definition 14 (Strategy and conformance). *A* strategy Σ *for a partecipant* A *in a game* $\mathcal{G}$ *is a function which maps each finite play* $\boldsymbol{e} = \langle e_0 \cdots e_n \rangle$ *to a (possibly empty) subset of* $\mathbf{E}_A$ *such that:* $e \in \Sigma(\boldsymbol{e}) \;\Rightarrow\; \boldsymbol{e}e$ *is a play of* $\mathcal{G}$.
A play $\boldsymbol{e} = \langle e_0 e_1 \cdots \rangle$ conforms *to a strategy* Σ *for a partecipant* A *in* $\mathcal{G}$ *if, for all* $i \geq 0$, $e_i \in \mathbf{E}_A \;\Rightarrow\; e_i \in \Sigma(\boldsymbol{e}_{/i})$.

Although events, namely moves, are associated to players via the map π, this is not injective in general, so that players can share moves. In general there are neither a turn rule nor alternation of players, similarly to concurrent games in [1]. A strategy Σ provides "suggestions" to some player on how to legally move continuing finite plays (also called "positions" in game-theoretic literature). But Σ may be ambiguous at some places, since $\Sigma(\boldsymbol{e})$ may contain more than an event; in fact it can be viewed as a partial mapping which is undefined when $\Sigma(\boldsymbol{e}) = \emptyset$.

We refer to [5] for the general definition of winning strategy for multi-player games (briefly recalled also in Remark 1 below), since it involves the conditions of fairness and innocence, which will be trivially satisfied in our interpretation of affectible client/server systems, where the notion of winning strategy corresponds to the one given in Def. 19.

Turn-based operational semantics and compliance. Toward the game theoretic interpretation of a client/server system $\rho \parallel \sigma$, we introduce a slightly different description of the semantics of affectible contracts, making explicit the idea of a three-player game. We interpret the internal choices and the input actions of the client as moves of a player A and the internal choices and the input actions of the server as moves of a player B. The synchronisations due to affectible choices are instead interpreted as moves of the third player C.

From a technical point of view this is a slight generalization and adaptation to our scenario of the turn-based semantics of "session types" in [5], §5.2. The changes are needed both because we have three players instead of two, and because session types are just session contracts, that is affectible contracts without affectible outputs.

Definition 15 (Single-buffered ASC). *The set* $\mathsf{ASC}^{[]}$ *of* single-buffered *affectible contracts is defined by* $\mathsf{ASC}^{[]} = \mathsf{ASC} \cup \{\, \mathbf{0} \,\} \cup \{\, [\overline{a}_k]\sigma_k \mid \oplus_{i \in I} \overline{a}_i.\sigma_i \in \mathsf{ASC}, k \in I \,\}$

We use the symbols $\tilde{\rho}, \tilde{\sigma}, \tilde{\rho}', \tilde{\sigma}' \ldots$ to denote elements of $\mathsf{ASC}^{[]}$. A *turn-based configuration* (configuration for short) is a pair $\tilde{\rho} \,\|\!\| \, \tilde{\sigma}$, where $\tilde{\rho}, \tilde{\sigma} \in \mathsf{ASC}^{[]}$.

As in [5], we have added the "single buffered" contracts $[\overline{a}]\sigma$ to represent the situation in which $\overline{a}$ is the only output offered after an internal choice. Since the actual synchronization takes place in a subsequent step, $\overline{a}$ is "buffered" in front of the continuation σ.

$$\oplus_{i\in I}\overline{a}_i.\rho_i \,|||\, \tilde{\sigma} \xrightarrow{\mathsf{A}:\overline{a}_k} [\overline{a}_k]\rho_k \,|||\, \tilde{\sigma} \qquad \Sigma_{i\in I}a_i.\rho_i \,|||\, [\overline{a}_k]\sigma \xrightarrow{\mathsf{A}:a_k} \rho_k \,|||\, \sigma$$

$$\tilde{\rho} \,|||\, \oplus_{i\in I}\overline{a}_i.\sigma_i \xrightarrow{\mathsf{B}:\overline{a}_k} \tilde{\rho} \,|||\, [\overline{a}_k]\sigma_k \qquad [\overline{a}_k]\rho \,|||\, \Sigma_{i\in I}a_i.\sigma_i \xrightarrow{\mathsf{B}:a_k} \rho \,|||\, \sigma_k$$

$$\overline{a}.\rho + \rho' \,|||\, a.\sigma + \sigma' \xrightarrow{\mathsf{C}:a} \rho \,|||\, \sigma \qquad a.\rho + \rho' \,|||\, \overline{a}.\sigma + \sigma' \xrightarrow{\mathsf{C}:a} \rho \,|||\, \sigma$$

$$\mathbf{1} \,|||\, \tilde{\rho} \xrightarrow{\mathsf{C}:\checkmark} \mathbf{0} \,|||\, \tilde{\rho}$$

where $(k \in I)$

Fig. 1. Turn-based operational semantics of turn-based configurations

Definition 16 (Turn-based operational semantics of configurations). *Let* $\mathbf{tbAct} = \{\mathsf{A}, \mathsf{B}, \mathsf{C}\} \times (\mathbf{Act} \cup \{\checkmark\})$. *In Figure 1 we define the LTS* $\longrightarrow$ *over turn-based configurations, with labels in* $\mathbf{tbAct}$.

Comparing $\longrightarrow$ with the LTS for affectible contracts, we observe that $[\overline{a}]\sigma$ is a duplicate of $\overline{a}.\sigma$, with the only difference that now there is a redundant step in $\oplus_{i\in I}\overline{a}_i.\rho_i \,|||\, \tilde{\sigma} \xrightarrow{\mathsf{A}:\overline{a}_k} [\overline{a}_k]\rho_k \,|||\, \tilde{\sigma}$ when I is the singleton $\{k\}$. Also we have the new reduction $\mathbf{1} \,|||\, \tilde{\rho} \xrightarrow{\mathsf{C}:\checkmark} \mathbf{0} \,|||\, \tilde{\rho}$ to signal when player C wins.
Let $\boldsymbol{\beta} = \langle \beta_1 \cdots \beta_n \rangle \in \mathbf{tbAct}^*$. We shall use the notation $\xrightarrow{\boldsymbol{\beta}} \;=\; \xrightarrow{\beta_1} \circ \cdots \circ \xrightarrow{\beta_n}$

Definition 17 (Turn-Based Compliance Relation $\dashv^{\mathsf{tb}}$).

i) Let $\mathcal{H} : \mathcal{P}(\mathsf{ASC}^{[]} \times \mathsf{ASC}^{[]}) \to \mathcal{P}(\mathsf{ASC}^{[]} \times \mathsf{ASC}^{[]})$ *be such that, for any* $\mathcal{R} \subseteq \mathsf{ASC}^{[]} \times \mathsf{ASC}^{[]}$, *we get* $(\tilde{\rho}, \tilde{\sigma}) \in \mathcal{H}(\mathcal{R})$ *if:*
1) $\tilde{\rho} \,|||\, \tilde{\sigma} \not\longrightarrow$ *implies* $\rho = \mathbf{0}$;
2) $\forall \tilde{\rho}', \tilde{\sigma}'.\ [\ \tilde{\rho} \,|||\, \tilde{\sigma} \xrightarrow{\beta} \tilde{\rho}' \,|||\, \tilde{\sigma}'$ *implies* $\tilde{\rho}' \;\mathcal{R}\; \tilde{\sigma}'\ ]$,
where $\beta \in \{\mathsf{A}{:}a, \mathsf{A}{:}\overline{a}, \mathsf{B}{:}a, \mathsf{B}{:}\overline{a} \mid a \in \mathcal{N}\}$;
3) $\exists a \in \mathcal{N}.\tilde{\rho} \,|||\, \tilde{\sigma} \xrightarrow{\mathsf{C}:a}$ *implies* $\exists \tilde{\rho}', \tilde{\sigma}', a.\ [\tilde{\rho} \,|||\, \tilde{\sigma} \xrightarrow{\mathsf{C}:a} \tilde{\rho}' \,|||\, \tilde{\sigma}'$ *and* $\tilde{\rho}' \;\mathcal{R}\; \tilde{\sigma}']$;

ii) A relation $\mathcal{R} \subseteq \mathsf{ASC}^{[]} \times \mathsf{ASC}^{[]}$ *is a* turn-based compliance relation *if* $\mathcal{R} \subseteq \mathcal{H}(\mathcal{R})$. $\dashv^{\mathsf{tb}}$ *is the greatest solution of the equation* $X = \mathcal{H}(X)$, *that is* $\dashv^{\mathsf{tb}} \;=\; \nu\mathcal{H}$.

iii) For $\rho, \sigma \in \mathsf{ASC}$, *we say that* ρ *is* turn-based compliant *with* σ *if* $\rho \dashv^{\mathsf{tb}} \sigma$.

Turn-based compliance is equivalent to affectible compliance

Theorem 2. *Let* $\rho, \sigma \in \mathsf{ASC}$. $\quad \rho \dashv^{\mathsf{tb}} \sigma \quad \Leftrightarrow \quad \rho \dashv^{\mathsf{A}} \sigma$.

Three-player game interpretation for ASC client/server systems. Using the turn-based semantics, we associate to any client/server system an event structure, and then a three-player game[4], extending the treatment of session types with two-player games in [5]. For our purposes we just consider the LTS of a given client/server system instead of an arbitrary one.

[4] Such interpretation is called *semantic-based* in [5] and it applies quite naturally to our context. Instead the *syntax-based* approach (which is equivalent to the semantic-based one in a two-players setting; see [5] §5.3.2) cannot be straightforwardly extended to a three-player game.

Definition 18 (ES of affectible-contracts systems).
Let $\rho \parallel \sigma$ be a client/server system of affectible contracts. We define the event structure $[\![\rho \parallel \sigma]\!] = (E, \#, \vdash, l)$, where

- $E = \{\, (n, \beta) \mid n \in \mathbb{N}, \beta \in \mathbf{tbAct} \,\}$
- $\# = \{\, ((n, \beta_1), (n, \beta_2)) \mid n \in \mathbb{N}, \beta_1, \beta_2 \in \mathbf{tbAct}, \beta_1 \neq \beta_2 \,\}$
- $\vdash = \mathit{sat}(\vdash_{\rho\|\sigma})$
  where $\vdash_{\rho\|\sigma} = \{\, (X, (n, \beta)) \mid \rho \,\|\!|\, \sigma \xrightarrow{snd(X)} \tilde{\rho}' \,\|\!|\, \tilde{\sigma}' \xrightarrow{\beta} \quad \text{and } n = |X| + 1 \,\}$
- $l(n, \beta) = \beta$

where the partial function $snd(\text{-})$ maps any $X = \{\, (i, \beta_i) \,\}_{i=1..n}$ to $\langle \beta_1 \cdots \beta_n \rangle$, and it is undefined over sets not of the shape of X.

Events in $[\![\rho \parallel \sigma]\!]$ are actions in **tbAct** paired with time stamps. Two events are in conflict if different actions should be performed at the same time, so that configurations must be linearly ordered w.r.t. time. The relation $X \vdash_{\rho\|\sigma} (n, \beta)$ holds if X is a trace in the LTS of $\rho \parallel \sigma$ of length $n - 1$; therefore the enabling $Y \vdash (n, \beta)$ holds if and only if Y includes a trace of length $n - 1$ that can be prolonged by β, possibly including (n, β) itself and any other action that might occur after β in the LTS.

So, by the above, $\vdash_{\mathsf{Buyer}\|\mathsf{Seller}}$ in $[\![\mathsf{Buyer} \parallel \mathsf{Seller}]\!]$ corresponds to

$$\begin{aligned}
\{\ & \emptyset \vdash_{\mathsf{Buyer}\|\mathsf{Seller}} (1, (\mathsf{C} : \mathtt{belt})),\ \emptyset \vdash_{\mathsf{Buyer}\|\mathsf{Seller}} (1, (\mathsf{C} : \mathtt{bag})),\\
& \{(1, (\mathsf{C}{:}\mathtt{belt}))\} \vdash_{\mathsf{Buyer}\|\mathsf{Seller}} (2, (\mathsf{B}{:}\overline{\mathtt{price}})), \{(1, (\mathsf{C}{:}\mathtt{bag}))\} \vdash_{\mathsf{Buyer}\|\mathsf{Seller}} (2, (\mathsf{B}{:}\overline{\mathtt{price}})),\\
& \{(1, (\mathsf{C}{:}\mathtt{belt})), (2, (\mathsf{Seller}{:}\overline{\mathtt{price}}))\} \vdash_{\mathsf{Buyer}\|\mathsf{Seller}} (3, (\mathsf{A}{:}\mathtt{price})), \ldots\\
& \ldots X_1 \vdash_{\mathsf{Buyer}\|\mathsf{Seller}} (6, (\mathsf{C}, \checkmark)) \qquad \}
\end{aligned}$$

where $X_1 = \{(1, (\mathsf{C}{:}\mathtt{bag})), (2, (\mathsf{B}{:}\overline{\mathtt{price}})), (3, (\mathsf{A}{:}\mathtt{price})), (4, (\mathsf{A}{:}\overline{\mathtt{cash}})), (5, (\mathsf{B}{:}\mathtt{cash}))\,\}$
The $\vdash_{\rho\|\sigma}$ of this simple example is finite . It is not so in general for systems with recursive contracts.

The following definition is a specialisation of Definitions 4.6 and 4.7 in [5]. We use $\mathsf{MaxTr}(s, \rightarrow)$ and $\mathsf{FinMaxTr}(s, \rightarrow)$ to denote the set of maximal traces and finite maximal traces, respectively, of $\mathsf{Tr}(s, \rightarrow)$.

Definition 19. *Given $\rho, \sigma \in \mathsf{ASC}$, we define the game $\mathcal{G}_{\rho\|\sigma}$ as $([\![\rho \parallel \sigma]\!], \Phi)$, where $\pi(n, \beta) = A$ if $\beta = A{:}\alpha$, ΦA is defined only if $A \in \{\, \mathsf{A}, \mathsf{B}, \mathsf{C} \,\}$ and*

$$\Phi A \boldsymbol{e} = \begin{cases} 1 & \text{if } \ \mathbf{P}(A, \boldsymbol{e}) \\ -1 & \text{otherwise} \end{cases}$$

where $\mathbf{P}(A, \boldsymbol{e})$ holds whenever

$$\boldsymbol{e} \in \mathsf{Tr}(\emptyset, \rightarrow_{[\rho\|\sigma]})\ \&\ [\boldsymbol{e} \in \mathsf{FinMaxTr}(\emptyset, \rightarrow_{[\rho\|\sigma]}) \ \Rightarrow\ \exists \boldsymbol{e}', n.\, \boldsymbol{e} = \boldsymbol{e}'(n, (A{:}\checkmark))]$$

A player A wins *in the sequence of events $\boldsymbol{e}$ if $\Phi A\, \boldsymbol{e} > 0$. A strategy Σ for player A is* winning *if A wins in all plays conforming to Σ.*

Note that, $\mathbf{P}(A, \boldsymbol{e})$ holds for any A and infinite element $\boldsymbol{e}$ of $\mathsf{Tr}(\emptyset, \rightarrow_{[\rho\|\sigma]})$.

For the game $\mathcal{G}_{\mathsf{Buyer} \parallel \mathsf{Seller}}$, it is possible to check that, for instance,

$$\Phi \mathsf{C} \boldsymbol{s_1} = 1, \quad \Phi \mathsf{A} \boldsymbol{s_1} = -1, \quad \Phi \mathsf{B} \boldsymbol{s_2} = -1, \quad \Phi \mathsf{C} \boldsymbol{s_3} = -1$$

where $\boldsymbol{s_1} = (1, (\mathsf{C}{:}\mathtt{bag}))(2, (\mathsf{B}{:}\overline{\mathtt{price}}))(3, (\mathsf{A}{:}\mathtt{price}))(4, (\mathsf{A}{:}\overline{\mathtt{cash}}))(5, (\mathsf{B}{:}\mathtt{cash}))(6, (\mathsf{C}, \checkmark))$,
$\boldsymbol{s_2} = (4, (\mathsf{A}{:}\mathtt{bag}))(1, (\mathsf{C}{:}\overline{\mathtt{price}}))$
$\boldsymbol{s_3} = (1, (\mathsf{C}{:}\mathtt{bag}))(2, (\mathsf{B}{:}\overline{\mathtt{price}}))(3, (\mathsf{A}{:}\mathtt{price}))(4, (\mathsf{A}{:}\overline{\mathtt{cash}}))(5, (\mathsf{B}{:}\mathtt{cash}))$

Let us define a particular strategy $\widetilde{\Sigma}$ for C in $\mathcal{G}_{\mathsf{Buyer} \parallel \mathsf{Seller}}$ as follows:

$$\widetilde{\Sigma}(\boldsymbol{s}) = \begin{cases} \{\, (1, (\mathsf{C}{:}\mathsf{bag})) \,\} & \text{if } \boldsymbol{s} = \langle\rangle \\ \{\, (6, (\mathsf{C}, \checkmark)) \,\} & \text{if } \boldsymbol{s} = \boldsymbol{s_3} \\ \emptyset & \text{for any other play} \end{cases}$$

The strategy $\widetilde{\Sigma}$ for C in $\mathcal{G}_{\mathsf{Buyer} \parallel \mathsf{Seller}}$ is winning.

Remark 1. According to [5], A wins in a play if $\mathcal{W}A\boldsymbol{e} > 0$, where $\mathcal{W}A\boldsymbol{e} = \Phi A\boldsymbol{e}$ if all players are "innocent" in $\boldsymbol{e}$, while if A is "culpable", $\mathcal{W}A\boldsymbol{e} = -1$, and if A is innocent and someone else culpable, $\mathcal{W}A\boldsymbol{e} = +1$. A strategy Σ of A is winning if A wins in all *fair* plays conforming to Σ. A play $\boldsymbol{e}$ is "fair" for a strategy Σ of a player A if any event in E_A which is infinitely often enabled is eventually performed. Symmetrically A is "innocent" in $\boldsymbol{e}$ if she eventually plays all persistently enabled moves of her in $\boldsymbol{e}$, namely if she is fair to the other players, since the lack of a move by A might obstacle the moves by others; she is "culpable" otherwise. As said above, Definition 19 is a particularisation of the general definitions in [5]. In fact in a game $\mathcal{G}_{\rho\|\sigma}$ no move of any player can occur more than once in a play $\boldsymbol{e}$ because of time stamps. Therefore no move can be "persistently enabled", nor it can be prevented since it can be enabled with a given time stamp only if there exists a legal transition in the LTS with the same label. Hence any player is innocent in a play $\boldsymbol{e}$ of $\mathcal{G}_{\rho\|\sigma}$ and all plays are fair. Therefore $\mathcal{W}$ coincides with Φ.

It is possible to characterize affectible and retractable compliance in terms of the existstence of a winning strategy for C in $\mathcal{G}_{\rho\|\sigma}$.

Theorem 3. *$\rho \dashv^{\mathsf{A}} \sigma$ (or, equivalently, $\rho \dashv^{\mathsf{tk}} \sigma$) if and only if player C has a winning strategy in the three-player game $\mathcal{G}_{\rho\|\sigma}$.*

3 Strategies as orchestrators

In the present section we show that a client ρ is retractable-compliant with a server σ if and only if their interactions can be led to a successful state by means of the mediation of an orchestrator. To do that we show how an orchestrator can be obtained out of a "univocal" winning strategy (see Def. 24 below) for player C in the game $\mathcal{G}_{\rho\|\sigma}$, and vice versa. For a detailed discussion on orchestrators for contracts and orchestrators for session-contracts, we refer to [14,13] and [2] respectively. In the present setting, our orchestrators, that we dub *strategy-orchestrators*, are defined as a variant of the session-orchestrators of [2], which in turn are a restriction of orchestrators in [14]. The task of a strategy orchestrator is to mediate the interactions between two affectible session contracts by selecting one of the possible affectible choices and constraining non-affectible ones.
We consider two sorts of orchestration actions, having the following shapes:
$\langle \alpha, \overline{\alpha} \rangle$, enabling the unaffectible synchronization $\rho \parallel \sigma \xrightarrow{\tau} \rho' \parallel \sigma'$;
$\langle \alpha, \overline{\alpha} \rangle^{+}$, enabling the affectible synchronization $\rho \parallel \sigma \xrightarrow{+} \rho' \parallel \sigma'$.

Definition 20 (Strategy Orchestrators).

i) The set **OrchAct** *of* strategy-orchestration actions *is defined by*

$$\mathbf{OrchAct} = \{\, \langle \alpha, \overline{\alpha} \rangle \mid \alpha \in \mathcal{N} \cup \overline{\mathcal{N}} \,\} \cup \{\, \langle \alpha, \overline{\alpha} \rangle^{+} \mid \alpha \in \mathcal{N} \cup \overline{\mathcal{N}} \,\}$$

We let $\mu, \mu', \ldots$ *range over elements of* **OrchAct** *with the shape* $\langle \alpha, \overline{\alpha} \rangle$*, and* $\mu^{+}, \mu'^{+}, \ldots$ *range over elements of* **OrchAct** *with the shape* $\langle \alpha, \overline{\alpha} \rangle^{+}$ *.*

ii) We define the set Orch *of* strategy orchestrators*, ranged over by* $f, g, \ldots$*, as the* closed *(with respect to the binder* rec *) terms generated by the following grammar:*

$$\begin{array}{rll} f, g ::= & \mathtt{1} & \textit{idle} \\ & \mid \mu^{+}.f & \textit{prefix} \\ & \mid \mu_1.f_1 \vee \ldots \vee \mu_n.f_n & \textit{disjunction} \\ & \mid x & \textit{variable} \\ & \mid \mathsf{rec}\, x.f & \textit{recursion} \end{array}$$

where the μ_i *in a disjunction are pairwise distinct. Moreover, we impose strategy orchestrators to be* contractive*, i.e. the* f *in* $\mathsf{rec}\, x.f$ *is assumed not to be a variable.*

We write $\bigvee_{i \in I} \mu_i.f_i$ as short for $\mu_1.f_1 \vee \ldots \vee \mu_n.f_n$, where $I = \{\, 1, \ldots, n \,\}$. If not stated otherwise, we consider recursive orchestrators up-to unfolding, that is we equate $\mathsf{rec}\, x.f$ with $f\{x/\mathsf{rec}\, x.f\}$. We omit trailing $\mathtt{1}$'s.

Strategy orchestrators are "simple orchestrators" in [14] and "synchronous orchestrators" in [13], but for the kind of prefixes which are allowed in a single prefix or in a disjunction. In fact a prefix $\langle \alpha, \overline{\alpha} \rangle^{+}$ cannot occur in disjunctions, where all the orchestrators must be prefixed by $\langle \alpha, \overline{\alpha} \rangle$ actions.

Definition 21 (Strategy orchestrators LTS).
We define the labelled transition system $(\mathsf{Orch}, \mathbf{OrchAct}, \mapsto)$ *by*

$$\mu^{+}.f \overset{\mu^{+}}{\mapsto} f \qquad\qquad (\textstyle\bigvee_{i \in I} \mu_i.f_i) \overset{\mu_k}{\mapsto} f_k \quad (k \in I)$$

An orchestrated system, represented by $\rho \parallel_f \sigma$, is client/server system whose interaction is mediated by an orchestrator.

Definition 22 (LTS for orchestrated-systems).
Let $\rho, \sigma \in \mathsf{ASC}$ *and* $f \in \mathsf{Orch}$*.*

$$\frac{\rho \rightarrow \rho'}{\rho \parallel_f \sigma \rightarrow \rho' \parallel_f \sigma} \qquad\qquad \frac{\sigma \rightarrow \sigma'}{\rho \parallel_f \sigma \rightarrow \rho \parallel_f \sigma'}$$

$$\frac{\rho \xrightarrow{a} \rho' \quad f \overset{\langle \overline{a}, a \rangle^{+}}{\mapsto} f' \quad \sigma \xrightarrow{\overline{a}^{+}} \sigma'}{\rho \parallel_f \sigma \xrightarrow{+} \rho' \parallel_{f'} \sigma'} \qquad\qquad \frac{\rho \xrightarrow{\overline{a}^{+}} \rho' \quad f \overset{\langle a, \overline{a} \rangle^{+}}{\mapsto} f' \quad \sigma \xrightarrow{a} \sigma'}{\rho \parallel_f \sigma \xrightarrow{+} \rho' \parallel_{f'} \sigma'}$$

$$\frac{\rho \xrightarrow{\alpha} \rho' \quad f \overset{\langle \overline{\alpha}, \alpha \rangle}{\mapsto} f' \quad \sigma \xrightarrow{\overline{\alpha}} \sigma'}{\rho \parallel_f \sigma \xrightarrow{\tau} \rho' \parallel_f \sigma'} \; (\alpha \in \mathcal{N} \cup \overline{\mathcal{N}})$$

Moreover, we define $\Longrightarrow = \rightarrow^{*} \circ (\xrightarrow{\tau} \cup \xrightarrow{+})$*.*

In both transitions $\xrightarrow{+}$ and $\xrightarrow{\tau}$ synchronization may happen only if the orchestrator has a transition with the appropriate pair of actions. This is because in an orchestrated interaction both client and server are committed to the synchronizations allowed by the orchestrator only. It is then clear that an orchestrator always selects one synchronisation of affectible actions on client and server side, while the disjunction of orchestrators represents the constraint that only certain synchronisations of unaffectible actions are permitted.

Definition 23 (Strategy-orchestrated Compliance).

i) $f : \rho \dashv^{\mathsf{Orch}} \sigma$ *if for any* ρ' *and* σ'*, the following holds:*

$$\rho \parallel_f \sigma \Longrightarrow^* \rho' \parallel_{f'} \sigma' \not\Longrightarrow \quad \textit{implies} \quad \rho' = \mathbf{1}.$$

ii) $\rho \dashv^{\mathsf{Orch}} \sigma$ *if* $\exists f.\ [\, f : \rho \dashv^{\mathsf{Orch}} \sigma \,]$.

Definition 24 (Univocal strategies). Σ *is* univocal *if* $\forall \boldsymbol{e}.\ |\Sigma(\boldsymbol{e})| \leq 1$.

The strategy $\widetilde{\Sigma}$ for C in $\mathcal{G}_{\mathsf{Buyer} \parallel \mathsf{Seller}}$, defined in the previous section, is univocal.

The proof of the following theorem relies on the fact that any orchestrator f such that $f : \rho \dashv^{\mathsf{Orch}} \sigma$ corresponds to a univocal winning strategies for player C in $\mathcal{G}_{\rho \parallel \sigma}$. Vice versa a univocal winning strategy Σ for C always induces an orchestrator f_Σ. It is not restrictive to look at univocal strategies only, as established in the next lemma.
We say that Σ *refines* Σ', written $\Sigma \leq \Sigma'$, if and only if $\Sigma(\boldsymbol{e}) \subseteq \Sigma'(\boldsymbol{e})$ for all $\boldsymbol{e}$.

Lemma 6. *If* C *has a winning strategy* Σ*, then* C *has a univocal winning strategy* Σ' *such that* $\Sigma' \leq \Sigma$.

Theorem 4.
$\exists f.\ [\, f : \rho \dashv^{\mathsf{Orch}} \sigma \,] \ \Leftrightarrow$ *there exists a winning strategy for player* C *in* $\mathcal{G}_{\rho \parallel \sigma}$.
In particular, a winning strategy for player C *in* $\mathcal{G}_{\rho \parallel \sigma}$ *can be obtained out of an orchestrator* f *such that* $f : \rho \dashv^{\mathsf{Orch}} \sigma$*, and vice versa.*

The orchestrator that can be obtained out of the strategy $\widetilde{\Sigma}$ is

$$\langle \mathtt{bag}, \overline{\mathtt{bag}} \rangle^+ . \langle \overline{\mathtt{price}}, \mathtt{price} \rangle (\langle \mathtt{cash}, \overline{\mathtt{cash}} \rangle \vee \langle \mathtt{card}, \overline{\mathtt{card}} \rangle).$$

Remark 2. Univocal strategies correspond to strategy-orchestrators and are technically easier to work with. On the other hand, we can recover a full correspondence among C strategies and orchestrators by allowing disjunctions of affectible synchronization actions $\langle \alpha, \overline{\alpha} \rangle^+$. In a session-based scenario, however, we expect any nondeterminism to depend solely on either the client or the server. By allowing $f = \langle \overline{a}, a \rangle^+ . f_1 \vee \langle \overline{b}, b \rangle^+ . f_2$ in the system $a.\rho_1 + b.\rho_2 \parallel_f \overline{a}.\sigma_1 + \overline{b}.\sigma_2$, the nondeterminism would depend on the orchestrator too.

Based on the formal system of Definition 9, the algorithm **Synth** in Figure 3 takes a (initially empty) set of assumptions Γ, and the affectible contracts ρ and σ, and it returns a set O of orchestrators (and hence a set of strategies by the above) if any, such that for any $f \in O$ we have $f : \rho \dashv^{\mathsf{Orch}} \sigma$; the algorithm returns the empty set otherwise. In the algorithm **Synth** we consider orchestrators as explicit terms, that is we do not consider recursion up-to unfolding.

$\mathbf{Synth}(\Gamma, \rho, \sigma) =$

$\quad$ **if** $x : \rho \dashv^{\mathsf{Orch}} \sigma \in \Gamma$ **then** $\{\, x \,\}$

$\quad$ **else** **if** $\rho = \mathbf{1}$ **then** $\{\, \mathbf{1} \,\}$

$\quad$ **else** **if** $\rho = \sum_{i \in I} \overline{\alpha}_i.\rho_i$ **and** $\sigma = \sum_{j \in J} \alpha_j.\sigma_j$ (where $\alpha \in \mathcal{N} \cup \overline{\mathcal{N}}$) **then**
$\quad$ **let** $\Gamma' = \Gamma,\ x{:}\rho \dashv^{\mathsf{Orch}} \sigma$ **in**
$\quad\quad \bigcup_{i \in I} \{\, \mathsf{rec}\, x.\langle \alpha_i, \overline{\alpha}_i \rangle^+.f \mid f \in \mathbf{Synth}\,(\Gamma', \rho_i, \sigma_i) \,\}$

$\quad$ **else** **if** $\rho = \bigoplus_{i \in I} \overline{a}_i.\rho_i$ **and** $\sigma = \sum_{j \in I \cup J} a_j.\sigma_j$ **then**
$\quad$ **let** $\Gamma' = \Gamma,\ x{:}\rho \dashv^{\mathsf{Orch}} \sigma$ **in**
$\quad\quad \{\, \mathsf{rec}\, x. \bigvee_{i \in I} \langle a_i, \overline{a}_i \rangle.f_i \mid \forall i \in I. f_i \in \mathbf{Synth}(\Gamma', \rho_i, \sigma_i \,\}$

$\quad$ **else** **if** $\rho = \sum_{j \in I \cup J} \overline{a}_j.\rho_i$ **and** $\sigma = \bigoplus_{i \in I} a_i.\sigma_i$ **then**
$\quad$ **let** $\Gamma' = \Gamma,\ x{:}\rho \dashv^{\mathsf{Orch}} \sigma_i$ **in**
$\quad\quad \{\, \mathsf{rec}\, x. \bigvee_{i \in I} \langle a_i, \overline{a}_i \rangle.f_i \mid \forall i \in I. f_i \in \mathbf{Synth}(\Gamma', \rho_i, \sigma_i \,\}$

$\quad$ **else** $\emptyset$

Fig. 2. The algorithm **Synth**.

Theorem 5 (Soundness and Completeness of Synth).
The algorithm ***Synth*** *is correct and complete in the following sense:*

i) ***Synth***(Γ, ρ, σ) *terminates for any* Γ, ρ *and* σ.
ii) If $f \in$ ***Synth***$(\emptyset, \rho, \sigma) \neq \emptyset$ *then* $f : \rho \dashv^{\mathsf{Orch}} \sigma$.
iii) If $f : \rho \dashv^{\mathsf{Orch}} \sigma$ *then there exists* $g \in$ ***Synth***$(\emptyset, \rho, \sigma) \neq \emptyset$ *such that the (possibly infinite) unfolding of* f *and* g *yields the same regular tree.*

It is not difficult to check that by computing $\mathbf{Synth}(\emptyset, \mathsf{Buyer}, \mathsf{Seller})$ we get a set just consisting of the orchestrator corresponding to the strategy $\widetilde{\Sigma}$, namely
$\mathbf{Synth}(\emptyset, \mathsf{Buyer}, \mathsf{Seller}) = \big\{ \langle \mathtt{bag}, \overline{\mathtt{bag}} \rangle^+.\langle \overline{\mathtt{price}}, \mathtt{price} \rangle (\langle \mathtt{cash}, \overline{\mathtt{cash}} \rangle \vee \langle \mathtt{card}, \overline{\mathtt{card}} \rangle) \big\}$
Using the previous results and Lemma 6 we get the following:

Corollary 2. *i) The relation* $\dashv^{\mathsf{Orch}}$ *is decidable.*
ii) For any $\rho, \sigma \in \mathsf{ASC}$*, it is decidable whether there exists a winning strategy for player* C *in* $\mathcal{G}_{\rho \parallel \sigma}$.
Moreover, in case a winning strategy exists, it is possible to effectively compute a univocal winning strategy.

4 Conclusion and Future Work

We have studied two approaches to loosening compliance among a client and a server in contract theory, based on the concepts of dynamic adaptation and of mediated interaction respectively. We have seen that these induce equivalent notions of compliance, which can be shown via the abstract concept of winning strategy in a suitable class of games.

The byproduct is that the existence of the agreement among two contracts specifying adaptive behaviours is established by statically synthesizing the proper

orchestrator, hence avoiding any trial and error mechanism at run time. The study in this paper has been limited to the case of binary sessions since this is the setting in which both orchestrators and retractable contracts have been introduced. However strategy based concepts of agreement have been developed in the more general scenario of multiparty interaction, which seems a natural direction for future work.

Acknowledgments. The authors wish to thank Massimo Bartoletti for an insightful discussion, Mariangiola Dezani for her everlasting support and the three anonymous referees for their help in improving the final version of the paper.

References

1. Abramsky, S., Mellies, P.A.: Concurrent games and full completeness. In: Logic in Computer Science, 1999. Proceedings. 14th Symposium on. pp. 431–442 (1999)
2. Barbanera, F., van Bakel, S., de'Liguoro, U.: Orchestrated session compliance. In: Proceedings ICE'15. EPTCS, vol. 189, pp. 21–36 (2015)
3. Barbanera, F., de'Liguoro, U.: Sub-behaviour relations for session-based client/server systems. MSCS 25(6), 1339–1381 (2015)
4. Barbanera, F., Dezani-Ciancaglini, M., Lanese, I., de'Liguoro, U.: Retractable contracts. In: PLACES. EPTCS, vol. 203, pp. 61–72. Open Publishing Ass. (2015)
5. Bartoletti, M., Cimoli, T., Pinna, G.M., Zunino, R.: Contracts as games on event structures. J.of Logical and Algebraic Methods in Progr. 85(3), 399 – 424 (2016)
6. Bernardi, G., Hennessy, M.: Compliance and testing preorders differ. In: Software Engineering and Formal Methods - SEFM 2013. LNCS, vol. 8368, pp. 69–81 (2013)
7. Brandt, M., Henglein, F.: Coinductive axiomatization of recursive type equality and subtyping. Fundam. Inform. 33(4), 309–338 (1998)
8. Bravetti, M., Zavattaro, G.: A theory of contracts for strong service compliance. Mathematical Structures in Computer Science 19(3), 601–638 (2009)
9. Castagna, G., Gesbert, N., Padovani, L.: A theory of contracts for web services. ACM Trans. on Prog. Lang. and Sys. 31(5), 19:1–19:61 (2009)
10. Gay, S., Hole, M.: Subtyping for Session Types in the Pi-Calculus. Acta Informatica 42(2/3), 191–225 (2005)
11. Laneve, C., Padovani, L.: The Must Preorder Revisited: An Algebraic Theory for Web Services Contracts. In: CONCUR'07. LNCS, vol. 4703, pp. 212–225. Springer (2007)
12. Nicola, R.D., Hennessy, M.: Testing equivalence for processes. In: ICALP. LNCS, vol. 154, pp. 548–560. Springer (1983)
13. Padovani, L.: Contract-based discovery and adaptation of web services. In: SFM. LNCS, vol. 5569, pp. 213–260. Springer (2009)
14. Padovani, L.: Contract-Based Discovery of Web Services Modulo Simple Orchestrators. Theoretical Computer Science 411, 3328–3347 (2010)
15. Winskel, G.: Event structures. In: Advances in Petri Nets 1986, Part II (1987)