Science of Computer Programming 167 (2018) 25–50

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

A theory of retractable and speculative contracts

Franco Barbanera [a], Ivan Lanese [b,*], Ugo de'Liguoro [c]

[a] Dipartimento di Matematica e Informatica, University of Catania, Italy
[b] Dipartimento di Informatica – Scienza e Ingegneria, University of Bologna/INRIA, Italy
[c] Dipartimento di Informatica, University of Torino, Italy

A R T I C L E I N F O

Article history:
Received 31 October 2017
Received in revised form 19 April 2018
Accepted 13 June 2018
Available online xxxx

Keywords:
Behavioral contracts
Backtracking
Speculative execution
Compliance
Reversible computing

A B S T R A C T

Behavioral contracts are abstract descriptions of expected communication patterns followed by either clients or servers during their interaction. Behavioral contracts come naturally equipped with a notion of *compliance*: when a client and a server follow compliant contracts, their interaction is guaranteed to progress or successfully complete. We study two extensions of behavioral contracts, *retractable contracts* dealing with *backtracking* and *speculative contracts* dealing with *speculative execution*. We show that the two extensions give rise to *the same notion of compliance*. As a consequence, they also give rise to the same *subcontract relation*, which determines when one server can be replaced by another preserving compliance. Moreover, compliance and subcontract relation are both decidable in quadratic time. Finally, we study the relationship between retractable contracts and calculi for reversible computing.

1. Introduction

Binary behavioral contracts [1–3] and binary session types [4] are abstractions of programs used to statically ensure that a client and a server successfully interact (see the survey in [5]). Along the years, the basic theory has been extended to deal with many features of clients and servers, such as exceptions [6], time [7], and so on. We consider here two new features: *backtracking*, allowing one to go back to previous stages of the interaction, and *speculative execution* [8], allowing one to try different alternatives concurrently. These two features have quite different origin and aims. Backtracking is used to avoid failures due to wrong past decisions in a wide range of settings, from the undo button in web browsers, to the execution model of Prolog, to techniques for rollback-recovery [9]. Speculative execution is used for efficiency reasons in different areas, from simulation [10], to thread-level optimization [11], to web services [12].

We present two extensions of binary contracts (Section 2): *retractable contracts* capturing backtracking, and *speculative contracts* capturing speculative execution. The two extensions are based on the same syntax, but naturally have different semantics. Essentially, they add to the session contracts of [13,14] (called first-order session behaviors in [13]) an operator of *external choice among output* operations. Classically, external means that the participant provides a set of alternatives, and the communication partner decides which one (s)he wants to take. This is opposed to internal choice, where the participant decides in isolation which alternative is taken. The setting in our extension is slightly more complex. The most interesting case is when an external choice among outputs and an external choice among inputs interact. In the retractable semantics, the client and the server agree on which option to explore, but they rollback and try a different possibility if

* Corresponding author.
E-mail addresses: barba@dmi.unict.it (F. Barbanera), ivan.lanese@gmail.com (I. Lanese), ugo.deliguoro@unito.it (U. de'Liguoro).

https://doi.org/10.1016/j.scico.2018.06.005

the computation gets stuck. In the speculative semantics, instead, all the possibilities are explored concurrently, and it is enough for one of them to succeed in order to guarantee the success of the whole computation.

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

- Even if retractable contracts and speculative contracts have different semantics and give rise to different client–server interactions, the relations of compliance, subcontract and duality in the two settings do coincide. While surprising at first sight, this can be explained by noticing that in both the cases different alternatives are explored (sequentially for retractable contracts, in parallel for speculative contracts) and the success of one of them guarantees the success of the whole computation. In other terms, the two semantics provide different implementations of *angelic nondeterminism*, first described by Hoare [15].
- While retractable/speculative contracts are strictly more expressive than session contracts (indeed they are a conservative extension, see Section 3.1), their theory preserves the main good properties of the theory of session contracts. In particular, compliance and subcontract relations are both decidable (Section 3) in quadratic time (Section 5), and the dual of a contract always exists and has a simple syntactic characterization (Section 4).

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

The results above make us confident in the fact that our semantics correctly captures the interaction patterns we are interested in. As further element supporting this, we show (Section 6) that the backtracking mechanism of retractable contracts can be seen as an application to behavioral contracts of the general theory proposed in [16] to define reversible extensions of process calculi.

This paper is an extended and revised version of [17] (a few preliminary results had been originally presented in a workshop paper [18]). Section 6, where the relation between retractable contracts and calculi for reversible computing is investigated, is completely new. Moreover, the analysis of the complexity of deciding compliance and subcontract relation has been refined, reducing the resulting complexity from a fifth power to a square. The paper also includes additional proofs and examples, and a more detailed discussion of related work. Finally, the whole presentation has been revised and improved.

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

2. Contracts for retractable and speculative interactions

We present below a uniform syntax for retractable and speculative contracts, with two semantics. It can be obtained from the syntax of (first-order) session contracts of [13,14] – dubbed SC in the present paper, and briefly recalled in Section 3.1.1 – by just adding external retractable/speculative choice among outputs and internal choice among inputs. As a matter of fact our contracts can also be seen as an extension of the retractable session contracts of [18], that we dub here **r**C, by simply adding internal choice among inputs.

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

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

where I is non-empty and finite, the names and the conames in choices are pairwise distinct and σ is not a variable in $\mathsf{rec}\, x.\sigma$. Recursion in **rs**C is guarded and hence contractive in the usual sense. We take an equi-recursive view of recursion by equating $\mathsf{rec}\, x.\sigma$ with $\sigma[\mathsf{rec}\, x.\sigma/x]$.

Intuitively, a name a represents a communication channel, which can be used either in an input action (denoted a) or in an output action (denoted $\overline{a}$). The dot is used to denote precedence: to perform $a.\sigma$ one first performs a and then continues as specified by σ. In the syntax above, branches $a_i.\sigma_i$ and $\overline{a}_i.\sigma_i$ can be composed either in internal choice or in