Journal of Logical and Algebraic Methods in Programming ••• (••••) •••–•••

Contents lists available at ScienceDirect

Journal of Logical and Algebraic Methods in Programming

www.elsevier.com/locate/jlamp

# Contracts as games on event structures [☆]

Massimo Bartoletti [a,*], Tiziana Cimoli [a], G. Michele Pinna [a], Roberto Zunino [b]

[a] *Dipartimento di Matematica e Informatica, Università degli Studi di Cagliari, Italy*
[b] *Dipartimento di Matematica, Università degli Studi di Trento, Italy*

A R T I C L E I N F O

*Article history:*
Received 2 November 2014
Received in revised form 4 May 2015
Accepted 6 May 2015
Available online xxxx

*Keywords:*
Contracts
Event structures
Compliance
Session types

A B S T R A C T

Event structures are one of the classical models of concurrent systems. The idea is that an *enabling* $X \vdash e$ represents the fact that the event $e$ can only occur after all the events in the set $X$ have already occurred. By interpreting events as actions promised by some participants, and by associating each participant with a goal (a function on sequences of events), we use event structures as a formal model for *contracts*. The states of a contract are sequences of events; a participant has a contractual obligation (in a given state) whenever some of its events is enabled in such a state. To represent the fact that participants are mutually distrusting, we study concurrent games on event structures; there, participants may play by firing events in order to reach their goals, and eventually win, lose or tie.
A crucial notion arising in this setting is that of *agreement*: a participant agrees on a set of contracts if she has a strategy to reach her goals in all the plays conforming to her strategy (or to make another participant sanctionable for not honouring an obligation). Another relevant notion is *protection*: a participant is protected by her contract when she has a strategy to avoid losing in any contexts, even in those where she has not reached an agreement. We study conditions for obtaining agreement and protection, and we show that these properties mutually exclude each other in a certain class of contracts. We then relate the notion of agreement in contracts with that of *compliance* in session types. In particular, we show that compliance corresponds to the fact that *eager* strategies lead to agreement.

© 2015 Elsevier Inc. All rights reserved.

## 1. Introduction

Several recent papers have been devoted to the study of *contracts* as a way to formally specify abstractions of the behaviour of software systems. A common aspect that gathers together some of these studies is a notion of *compliance*. This is a relation between systems which want to interact. Before starting the interaction, contracts are statically checked for compliance: when enjoyed, it guarantees that systems respecting their contracts will interact correctly. Since distributed applications are often constructed by dynamically discovering and composing services published by different (possibly distrusting) organizations, compliance becomes relevant to protect those services from each other's misbehaviour. Indeed, the larger an application is, the greater is the probability that some of its components deviates from the expected behaviour (either because of unintentional bugs, or maliciousness).

[☆] Work partially supported by Aut. Region of Sardinia under grants L.R.7/2007 CRP-17285 (TRICS), P.I.A. 2010 Project "Social Glue", by MIUR PRIN 2010-11 project "Security Horizons", and by EU COST Action IC1201 (BETTY).
[*] Corresponding author at: Dipartimento di Matematica e Informatica, Università degli Studi di Cagliari, via Ospedale 72, 09124 Cagliari, Italy.
*E-mail address:* bart@unica.it (M. Bartoletti).

http://dx.doi.org/10.1016/j.jlamp.2015.05.001
2352-2208/© 2015 Elsevier Inc. All rights reserved.

Compliance can be modelled in many different ways. Typically, it is formalised as a fairness property, which ensures progress (possibly, until reaching a success state [1,2]), or which ensures the possibility of always reaching success [3,4]. Weaker variants of compliance allow services to discard some messages [5], or involve orchestrators which can sometimes rearrange them [6]. All these approaches express contracts as terms of some process calculus.

In this paper we study compliance in the semantic setting of event structures (ES [7]). By abstracting away from the concrete details of process calculi, this model may be used as a unifying framework for reasoning about contracts, in the same spirit that event structures are used as an underlying semantics for a variety of concrete calculi for concurrency.

In our setting, a contract specifies the behaviour promised and expected by a participant or set of participants. Contracts coming from different participants can be composed together. In our view, *agreement* (a generalisation of compliance) is a property of composed contracts, which — roughly — ensures an acceptable interaction to each participant in the composition.

Our contracts are built upon four principal notions:

*Events* are the atomic observables. For instance, "Alice gives an apple to Bob" can be modelled as an event (say, $a$) in an ES. We assume that each event is unique, i.e. it cannot occur twice in the same computation. Thus, if Alice has to give two apples to Bob, we assume two events $a_0$, $a_1$ (representing two distinct occurrences of the same action).

*Participants* are the entities which advertise contracts, and are bound to perform the events prescribed by their contracts. We assume that each event is associated with a unique participant. For instance, if both Alice and Carol have to give an apple to Bob, we use two distinct events.

*Obligations* make explicit the causal dependencies between the events performed by participants. For instance, Alice's contract clause "I will give an apple to Bob after I have received a banana" induces an obligation for her to do event $a$ after event $b$ has been performed, since she has promised to do it. Event structures are a natural model for obligations; for instance, we can interpret the above clause as the *enabling* $\{b\} \vdash a$.

*Objectives* express the degree of "satisfaction" of a participant in a contract execution. Contracts associate each participant to an objective function, which in turn associates each execution with a *payoff*, which can be "win", "lose", or "tie".

In the above setting, we provide a formal definition of contracts, by interpreting their semantics as a multi-player concurrent game on event structures. We then formalise two key notions about contracts, namely *agreement* and *protection*. Intuitively, agreement is a property of a contract which results from the composition of a number of individual contracts from a set of participants. A participant agrees with such composed contract if she has a strategy to interact with the other participants so that in each interaction she either wins, or it is possible to blame another participant who is not honouring his obligations. Instead, protection is a property of a contract of a single participant. It requires that, whenever the contract is composed with any other contracts, possibly crafted by adversaries, then the participant has a strategy to avoid losing (by instead winning or tying) in the interactions with such adversaries.

*Contributions.* The main contributions of this paper are the following:

- We provide a formal definition for the intuitive notion of *agreement*. We study conditions for reaching agreements in contracts with *Offer-Request* payoffs, where participants request some actions in exchange for an offered service. Lemma 4.17 gives a necessary condition for agreement, while Theorem 4.19 gives a sufficient one.
- We interpret binary session types [8,9] as contracts, by providing them with event structure semantics (Definitions 5.12, 5.19). We establish our semantics faithful to the original one, by proving that the associated event structure is bisimilar to the session type in its operational semantics (Theorems 5.17, 5.20). We then exploit this correspondence result to show that compliance in session types holds whenever eager strategies lead to an agreement in the associated contract (Theorem 5.23). To prove this correspondence, we establish an auxiliary result about event structures. We provide them with two notions of Labelled Transition Systems, one based on the remainder, and the other one on configurations, and we relate them by bisimilarity (Lemma A.5).
- We formalise the notion of protection, and we study necessary conditions (Lemma 6.6) and sufficient conditions (Theorem 6.7) for obtaining protection in contracts with Offer-Request payoffs. We then show that agreement and protection are mutually exclusive for contracts with Offer-Request payoffs suffering from a circularity condition (Theorem 6.11). Roughly, the problem is that when the offers of the participants mutually depend on their requests, either there is a participant willing to perform the first offer, and so giving up protection, or each participant wants someone else to move first, so preventing an agreement to be reached.

The proofs of all our statements are provided either in the main text, or in Section A.

## 2. Event structures

Event structures (ES) are a model for concurrency introduced in [10]; they describe a process as performing events as time goes on. An *event* is a particular occurrence of an *action*, and different events may be occurrences of the same action. Each event $e$ is labelled with the action $\ell(e)$ it is associated with. For instance, pressing $n$ times a certain button