Science of Computer Programming 91 (2014) 115–137

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Compositional assume–guarantee reasoning for input/output component theories

Chris Chilton[a], Bengt Jonsson[b], Marta Kwiatkowska[a,*]

[a] Department of Computer Science, University of Oxford, UK
[b] Department of Information Technology, Uppsala University, Sweden

H I G H L I G H T S

- Contract-based assume–guarantee framework for asynchronous I/O component models.
- Operations of parallel, conjunction and quotient defined on contracts.
- Sound and complete compositional assume–guarantee rules for operators.
- Rules preserve safety and progress properties.

A R T I C L E   I N F O

Article history:
Received 2 February 2013
Received in revised form 23 November 2013
Accepted 17 December 2013
Available online 8 January 2014

Keywords:
Assume–guarantee
Contracts
Interface automata
Substitutivity
Quotient

A B S T R A C T

We formulate a sound and complete assume–guarantee framework for reasoning compositionally about components modelled as a variant of interface automata. The specification of a component, which expresses both safety and progress properties of input and output interactions with the environment, is characterised by finite traces. The framework supports dynamic reasoning about components and specifications, and includes rules for parallel composition, logical conjunction and disjunction corresponding to independent development, and quotient for incremental synthesis. Practical applicability of the framework is demonstrated through a link layer protocol case study.

1. Introduction

Component-based methodologies enable both design- and runtime assembly of software systems from heterogeneous components, facilitating component reuse, incremental development and independent implementability. To improve the reliability and predictability of such systems, specification theories have been proposed that permit the mixing of specifications and implementations, and allow for the construction of new components from existing ones by means of compositional operators [6,21,15,25]. A specification should make explicit the *assumptions* that a component can make about the environment, and the corresponding *guarantees* that it will provide about its behaviour. This allows for the use of compositional assume–guarantee (AG) reasoning, which enables the decomposition of the system into smaller components, each of which may be reasoned about in isolation during system development and verification.

In earlier work [8], we introduced a component-based specification theory, in which components communicate by synchronisation of input/output (I/O) actions, where inputs are controlled by the environment, while outputs (which are non-blocking) are controlled by the component. The component model is conceptually similar to the interface automata of de Alfaro and Henzinger [4], except that we use a different underlying semantic model, which is based on classical

[*] Corresponding author.

0167-6423/$ – see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.scico.2013.12.010

sets of traces, rather than alternating simulation. This approach allows us to define the weakest refinement preorder that preserves substitutivity of components, which implies a full abstraction result. Distinguishing features of our theory, an extension of which is contained in [9,10], are the inclusion of conjunction and quotient operators (which are more general than those of Doyen et al. [15], and Bhaduri and Ramesh [7]), as well as logical disjunction and hiding, in addition to a progress-sensitive variant of refinement based on quiescence, whereby a refining component must make progress whenever the original can. The theory enjoys strong algebraic properties, with all the operators being compositional under refinement.

In [9,4], the assumptions and guarantees of components are merged into one behavioural representation. In many cases, this avoids duplication of common information, although it can be desirable to manipulate the assumptions and guarantees separately. For instance, we may want to express a simple guarantee (such as “no failure will occur”) without having to weave it into a complex assumption. Separation of assumptions from guarantees also supports specification reuse, in that the same guarantees (or assumptions) can be used for several related interfaces, each representing different versions of a component.

Contributions In this article, we present a specification theory for reasoning about AG specifications (or contracts) of components as modelled in [9,10]. The formalism is well suited to modelling components of distributed systems, such as communication protocols and mediators, to name but a few. A contract consists of an assumption, guarantee and liveness property, all of which are represented by sets of finite traces. This facilitates reasoning about safety and progress properties, and differs from (arguably) more complex approaches based on modal specifications and alternating simulation. Treating contracts as first-class citizens, we define the operators of parallel, conjunction, disjunction and quotient on contracts, and prove compositionality. This is the first work to present such an extensive collection of operators directly on contracts (to our knowledge, quotient has not previously been defined), which supports flexible development and verification of component-based systems using AG principles. In relating implementations (components) with contracts by means of satisfaction, a notion of refinement corresponding to implementation containment is defined on contracts. Based on this, we formulate a collection of sound and complete AG reasoning rules for the preservation of safety and progress properties under the operations and refinement preorder of the specification theory. The AG rule for parallel is inspired by the Compositionality Principle of Abadi and Lamport [1], and Abadi and Plotkin [3], while the others admit novel treatment. The rules allow us to infer properties of compositions for both contracts and components, thus enabling designers to deduce whether it is safe to substitute a component, for example one synthesised at runtime by means of the quotient operator, with another. A preliminary version of this paper appeared as [11].

Related work Compositional AG reasoning has been extensively studied in the literature. Traditionally, the work was concerned with compositional reasoning for processes, components and properties expressed in temporal logics [24,12,17]. A variety of rule formats have been proposed, although Maier [22] demonstrates through a set-theoretic setting that compositional circular AG rules (where compositionality is defined in a precise way) for parallel composition (corresponding to intersection) cannot both be sound and complete. In [23], a sound and complete circular rule is presented, which is non-compositional. We obtain soundness and completeness of our compositional rule by relying on the fact that the outputs of components to be composed are disjoint, which breaks circularity.

Abadi and Lamport [1] consider compositional reasoning for contracts in the generic setting of state-based processes. They formulate a Compositionality Principle for parallel, which is sound for safety properties. A logical formulation of specifications is discussed by Abadi and Plotkin [3], where intuitionistic and linear logic approaches are adopted. In contrast, our work considers an action-based component model and has a richer set of composition operators, including conjunction and quotient. Furthermore, we prove completeness, as remarked in the previous paragraph.

More recent proposals focus on compositional verification for component theories such as interface and I/O automata. Emmi et al. [16] extend a learning-based compositional AG method to interface automata. Sound and complete rules are presented for the original operators defined by de Alfaro and Henzinger [4], namely compatibility, parallel and refinement based on alternating simulation, but conjunction, disjunction and quotient are absent. Moreover, the rules are limited to being asymmetric in nature. Larsen et al. [20] define an AG framework for I/O automata, where assumptions and guarantees are themselves specified as I/O automata. A parallel operator is defined on contracts, yielding the weakest specification respecting independent implementability, for which a sound and complete rule is presented. Our work differs by not requiring input-enabledness of components or guarantees, and allowing for specifications to have non-identical interfaces to their implementations. We also define conjunction, disjunction and quotient, and support progress properties, thus providing a significantly richer reasoning framework.

Raclet et al. [25] have developed a compositional theory based on modal specifications, which includes the operations we consider in this article, but for systems without I/O distinction. Larsen et al. [21] consider a cross between modal specifications and interface automata, where refinement is given in terms of alternating simulation/modal refinement (stronger than our trace containment), but conjunction and quotient are not defined. Both of these works use single models to encode assumptions and guarantees, whereas we adopt a contract-based approach.

Benveniste et al. [6] present an abstract mathematical framework for contract-based design, based on set-theoretic operations on sets of behaviours. The framework does not give consideration to the specifics of the execution model, hence it is unclear whether the rules can be instantiated for any particular communication model.

Bauer et al. [5] provide a generic construction for obtaining a contract framework from a component-based specification theory. The abstract ideas share similarity with our framework, and it is interesting to note how parallel composition of