Contract-Based Reasoning for Component Systems with Complex Interactions

Susanne Graf[1] and Roberto Passerone[2] and Sophie Quinton[3]
[1] VERIMAG, CNRS — [2] DISI, University of Trento — [3] IDA, TU Braunschweig

Abstract In this paper we propose a rule unifying circular and non-circular assume-guarantee reasoning and show its interest for contract-based design and verification. Our work was motivated by the need to combine, in the top-down methodology of the FP7 SPEEDS project, partial tool chains for two component frameworks derived from the HRC model and using different refinement relations. While the L0 framework is based on a simple trace-based representation of behaviors and uses set operations for defining refinement, the more elaborated L1 framework offers the possibility to build systems of components with complex interactions. Our approach in L1 is based on circular reasoning and results in a method for checking contract dominance which does not require the explicit composition of contracts. In order to formally relate results obtained in L0 and L1, we provide a definition of the minimal concepts required by a consistent contract theory and propose abstract definitions which smoothly encompass hierarchical components. Finally, using our relaxed rule for circular reasoning, we show how to use together the L0 and L1 refinement relations and as a result their respective tool chains.[1]

1 Introduction

Contract and interface frameworks are emerging as the formalism of choice for system designs that require large and dispersed teams, or where the supply chain is complex [13]. Contracts are usually expressed as pairs of *assumptions*, or properties that the environment must satisfy, and *guarantees*, the properties that must be satisfied by the component. This style of specification is typically employed for top-down design of systems of components, where the system under design is built by a sequence of decomposition and verification steps. The *refinement* relation between a specification and an implementation is at the core of component-based design. In contract-based design however, refinement takes different forms depending on whether it relates a system to a specification, two contracts or an implementation to a contract. In this paper we use a methodology which divides the design and verification process into three steps corresponding to these three forms of refinement.

Our work has its practical motivations in the component-framework HRC — standing for Heterogeneous Rich Components — defined in the SPEEDS IP project [13] and used in the COMBEST project [4]. The HRC model defines

[1] NB: for readability of the proofs, we adopt here the LNCS style. The main part of this document (i.e. before the appendix) is the same as in the submitted version.

component properties as extended transition systems and provides several composition models, ranging from low-level semantic composition to composition frameworks underlying the design tools used by system designers. We focus here on two component frameworks for HRC denoted respectively L0 and L1, and the corresponding contract frameworks. In particular, the complexity of the L1 framework leads us to focus on *circular reasoning*, which allows a component and its environment to be refined concurrently — each relying on the abstract description of its context — and entails an interesting rule for proving dominance. In order to relate L0 and L1, we define a generic contract framework that uses abstract composition operators and thus encompasses a variety of different interaction models, e.g. I/O communication and priority policies. We then show that L0 and L1 can be seen as instances of this generic contract framework. Finally, we show how to use a relaxed rule for circular reasoning to combine partial tool chains for both frameworks into a complete tool chain for our methodology.

To the best of our knowledge this is the first time that a rule combining different refinement relations is proposed and used to unify two contract frameworks. While circular reasoning has been extensively studied, e.g. in [1,9] existing work focuses on finding sufficient conditions for soundness of circular reasoning while we focus on how to use circular reasoning in a contract-based methodology. Non-circular assume-guarantee reasoning is also a topic of intense research. The difficulty there lies in finding a decomposition of the system that would satisfy the strong condition imposed on at least one of its components. It was shown in [3] that in most cases assume-guarantee reasoning is not more efficient than monolithic verification. Finally our contract frameworks are related to interface automata [5]. Since de Alfaro and Henzinger's seminal paper many contract and interface theories have been developed for numerous frameworks (see e.g. [8,14,6] to name just a few). However these theories all focus on composition of contracts and furthermore do not handle complex interactions. Preliminary versions of our contract framework appeared in [11,7] but did not address the question of combining results from different refinement relations.

This paper is structured as follows: Section 2 describes our design and verification methodology as well as generic definitions of component and contract framework. It then discusses sufficient reasoning rules for establishing dominance without composing contracts. Section 3 presents how the proposed approach is applied to the L0 and L1 frameworks. In particular it shows how their different satisfaction relations may be used together using *relaxed circular reasoning* and discusses practical consequences of this result. Section 4 concludes the paper. The proofs of all theorems presented in this paper are presented in [10].

2 Design methodology

Our methodology is based on an abstract notion of components. We characterize a component K by its *interface* defined as a set $\mathcal{P}$ of *ports* which describe what can be observed by its environment. We suppose given a global set of ports $Ports$, which all sets of ports in the following are subsets of. In addition, components are

also characterized by their *behavior*. We are not concerned with how behaviors are represented at this level, and develop our methodology independently of the particular formalism employed.

In order to separate the implementation phase of a component from its integration into the system under design, we use *contracts* [2,11,13]. A contract for a component K describes the interface $\mathcal{P}$ of K, the interaction between K and its environment E, the expected behavior of E, called the *assumption* A of the contract, and the expected behavior of K, called the *guarantee* G.

Interactions are expressed using the concept of *glue* operator [12]. A glue defines how the ports of different components are connected and the kind of synchronization and data exchange that may take place. We denote the composition of two components K_1 and K_2 through a glue gl as $gl\{K_1, K_2\}$. The glue must be defined on the union of the ports $\mathcal{P}_1$ and $\mathcal{P}_2$ of the components.

Assumptions and guarantees are in turn expressed as components, defining the interface and the behavior that are considered acceptable from the environment and from the component. Thus, formally, a contract $\mathcal{C}$ for an interface $\mathcal{P}$ is a triple (A, gl, G), where gl is a glue operator on $\mathcal{P} \cup \mathcal{P}_A$ for some $\mathcal{P}_A$ disjoint from $\mathcal{P}$; the assumption A is a component with interface $\mathcal{P}_A$; and the guarantee G is a component with interface $\mathcal{P}$. Note that the interface of the environment is implicitly defined by gl. A expresses a constraint on the environment E and G a constraint on K. Graphically, we represent contracts as in Figure 1.

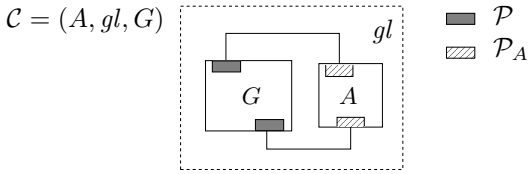

Figure1. A contract (A, gl, G) for an interface $\mathcal{P}$

From a macroscopic point of view, we adopt a top-down design and verification methodology (see Figure 2) in which global requirements are pushed progressively from the top-level system to the low-level atomic components. As usual, this is just a convenient representation; in real life, the final picture is always obtained in several iterations alternatively going up and down the hierarchy.

We assume that the system K under construction has to realize a global requirement φ together with an environment on which we may have some knowledge, expressed by a property A. Both φ and A are expressed w.r.t. the interface $\mathcal{P}$ of K. We proceed as follows: (1) define a *contract* $\mathcal{C} = (A, gl, G)$ for $\mathcal{P}$ such that $gl\{A, G\}$ *conforms* to φ; (2) decompose K as subcomponents K_i connected through a glue operator gl_I and provide a contract $\mathcal{C}_i$ for each of them; possibly iterate this step if needed; (3) prove that whenever a set of implementations K_i *satisfy* their contracts $\mathcal{C}_i$, then their composition satisfies the top-level contract $\mathcal{C}$ (*dominance*) — and thus guarantee φ; (4) provide such implementations.

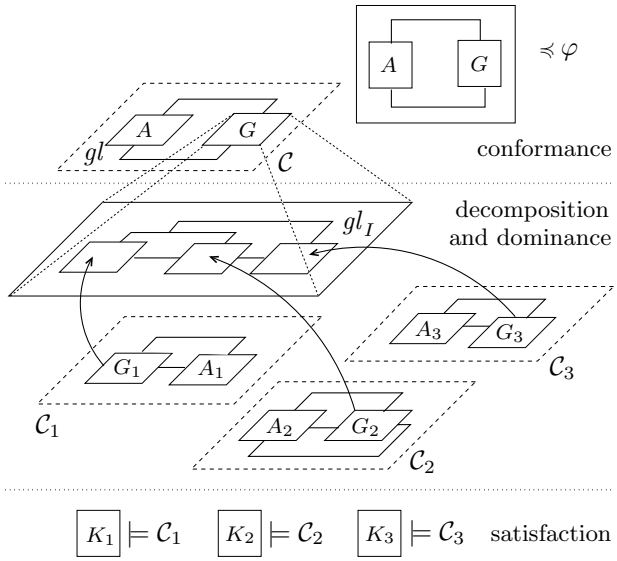

Figure2. Proof of $gl\{A, gl_I\{K_1, K_2, K_3\}\} \preccurlyeq \varphi$

The correctness proof for a particular system is therefore split into 3 phases: *conformance* (denoted $\preccurlyeq$) of the system defined by the top-level contract $\mathcal{C}$ to φ; *dominance* between the composition of the set of contracts $\{\mathcal{C}_i\}$ through gl_I and $\mathcal{C}$; and *satisfaction* (denoted $\models$) of each $\mathcal{C}_i$ by the corresponding implementation K_i. Thus, *conformance* relates closed systems, *dominance* relates contracts, while *satisfaction* relates components to contracts.

The assumption of $\mathcal{C}_1$ is represented as one component A_1 while in the actual system K_1 will be used in the context of three components, namely K_2, K_3 and A. Thus, we need to relate the actual glues gl and gl_I to the glue gl_1 of $\mathcal{C}_1$. In other words, we need a glue gl_{E_1} to compose K_2, K_3 and A as well as an operation $\circ$ on glues such that $gl \circ gl_I = gl_1 \circ gl_{E_1}$. In most cases $\circ$ cannot simply be composition of functions and has to involve some *flattening* of the system.

2.1 Contract framework

To summarize, we consider a component framework that smoothly supports complex composition operators and hierarchical components. The elements of the component framework are as follows:

Definition 1 (Component framework). *A* component framework *is a tuple* $(\mathcal{K}, GL, \circ, \cong)$ *where:*

- $\mathcal{K}$ *is a set of* components*. Each component* $K \in \mathcal{K}$ *has as* interface *a set of* ports*, denoted* $\mathcal{P}_K$ *and subset of our global set of ports* $Ports$*.*
- GL *is a set of* glues*. A glue is a partial function* $2^{\mathcal{K}} \longrightarrow \mathcal{K}$ *transforming a set of components into a new composite component. Each* $gl \in GL$ *is defined on a set of ports* S_{gl}*, called* support set*, and defines a new interface* $\mathcal{P}_{gl}$ *for the*

new component, called exported interface. $K = gl(\{K_1, \ldots, K_n\})$ *is defined if* $K_1, \ldots, K_n \in \mathcal{K}$ *have disjoint interfaces,* $S_{gl} = \bigcup_{i=1}^{n} \mathcal{P}_{K_i}$ *and* $\mathcal{P}_K = \mathcal{P}_{gl}$.
- $\circ$ *is a partial operator on GL, called* flattening*, to compose glues.* $gl \circ gl'$ *is defined if* $\mathcal{P}_{gl'} \subseteq S_{gl}$*. Then, its support set is* $S_{gl} \backslash \mathcal{P}_{gl'} \cup S_{gl'}$ *and its interface is* $\mathcal{P}_{gl}$
- $\cong \subseteq \mathcal{K} \times \mathcal{K}$ *is an equivalence relation between components.*

We simplify our notation by writing $gl\{K_1, \ldots, K_n\}$ instead of $gl(\{K_1, \ldots, K_n\})$. The equivalence relation $\cong$ is typically used for relating composite components with their semantics given as an atomic component. More importantly, $\circ$ must be coherent with $\cong$ in the sense that $gl\{gl'\{\mathcal{K}_1\}, \mathcal{K}_2\} \cong (gl \circ gl')\{\mathcal{K}_1 \cup \mathcal{K}_2\}$ for any sets of components $\mathcal{K}_i$ such that all terms are defined.

After formalizing generic properties required from a component framework, we now define the relations used in the methodology for dealing with contracts. Satisfaction is usually considered as a derived relation and chosen as the weakest relation implying conformance and preserved by composition. We loosen the coupling between satisfaction and conformance to obtain later stronger reasoning schemata for dominance. Furthermore we propose a representation of satisfaction as a set of *refinement under context* relations denoted $\sqsubseteq_{A,gl}$ and such that $K \sqsubseteq_{A,gl} G$ iff $K \models (A, gl, G)$.

Definition 2 (Contract framework). *A* contract framework *is a tuple* $(\mathcal{K}, GL, \circ, \cong, \preccurlyeq, \models)$ *where:*
- $(\mathcal{K}, GL, \circ, \cong)$ *is a component framework.*
- $\preccurlyeq \subseteq \mathcal{K} \times \mathcal{K}$ *is a preorder called* conformance *relating components having the same interface.*
- $\models$ *is a relation called* satisfaction *between components and contracts s.t.: the relations* $\sqsubseteq_{A,gl}$ *defined by* $K \sqsubseteq_{A,gl} G$ *iff* $K \models (A, gl, G)$ *are preorders; and, if* $K \models (A, gl, G)$ *then* $gl\{A, K\} \preccurlyeq gl\{A, G\}$.

Our definition of satisfaction emphasizes the fact that $\models$ can be seen as a set of refinement relations where $K \sqsubseteq_{A,gl} G$ means that K refines G in the context of A and gl. The condition which relates satisfaction and conformance ensures that the actual system $gl\{A, K\}$ will conform to the global requirement φ discussed in the methodology because $\preccurlyeq$ is transitive and $gl\{A, G\} \preccurlyeq \varphi$.

Example 1. Typical notions of conformance for LTS are *trace inclusion* and its structural counterpart *simulation*. For these, satisfaction is usually defined as the weakest relation implying conformance.

$$K \models (A, gl, G) \triangleq gl\{K, A\} \preccurlyeq gl\{G, A\}$$

Dominance is a key notion for reasoning about contracts rather than using refinement between components. Proving that a contract $\mathcal{C}$ dominates $\mathcal{C}'$ means showing that every component satisfying $\mathcal{C}$ also satisfies $\mathcal{C}'$.[2] However, a

[2] One may also need to ensure that the assumptions of the low-level contracts are indeed satisfied in the actual system. This is achieved by strengthening the definition with:

$$\forall E \text{ on } \mathcal{P}_A, \text{ if } E \models (G', gl', A') \text{ then } E \models (G, gl, A)$$

dominance check involves in general not just a pair of contracts: a typical situation would be the one depicted in Figure 2, where a set of contracts $\{\mathcal{C}_i\}_{i=1}^n$ are attached to disjoint interfaces $\{\mathcal{P}_i\}_{i=1}^n$. Besides, a glue gl_I is defined on $P = \bigcup_{i=1}^n \mathcal{P}_i$ and a contract $\mathcal{C}$ is given for P. In this context, a set of contracts $\{\mathcal{C}_i\}_{i=1}^n$ dominates a contract $\mathcal{C}$ w.r.t. a glue gl_I if any set of components satisfying contracts $\mathcal{C}_i$, when composed using gl_I, makes a component satisfying $\mathcal{C}$.

Definition 3 (Dominance). *Let $\mathcal{C}$ be a contract on $\mathcal{P}$, $\{\mathcal{C}_i\}_{i=1}^n$ a set of contracts on $\mathcal{P}_i$ and gl_I a glue such that $S_{gl_I} = \bigcup_{i=1}^n \mathcal{P}_i$ and $\mathcal{P} = \mathcal{P}_{gl_I}$. Then $\{\mathcal{C}_i\}_{i=1}^n$* dominates $\mathcal{C}$ with respect to gl_I *iff for all components $\{K_i\}_{i=1}^n$:*

$$(\forall i : K_i \models \mathcal{C}_i) \implies gl_I\{K_1, \ldots, K_n\} \models \mathcal{C}$$

Note that this formal definition of dominance does not help establishing dominance in practice because looking at all possible components satisfying a contract is not realistic. What we need is a sufficient condition that refers to assumptions and guarantees, rather than components. One such condition is when the composition of the low-level guarantees satisfies the top-level contract and each low-level assumption is discharged by its abstract environment built from the guarantees of the other components. Formally:

$$\begin{cases} gl_I\{G_1, \ldots, G_n\} \models \mathcal{C} \\ \forall i : gl_{E_i}\{A, G_1, \ldots, G_{i-1}, G_{i+1}, \ldots, G_n\} \models \mathcal{C}_i^{-1} \end{cases} \quad \textbf{(1)}$$

In the next subsection, we provide two rules which indeed make the previous condition sufficient for establishing dominance: one is similar to circular assume-guarantee reasoning and the other one deals with preservation of satisfaction by composition. This result is particularly significant because one can check dominance while avoiding composition of contracts, which is impossible in the general case and leads to state explosion in most concrete contract frameworks.

2.2 Reasoning within a contract framework

We use here the representation of satisfaction as a set of refinement under context relations $\sqsubseteq_{A,gl}$ where $K \sqsubseteq_{A,gl} G$ if and only if $K \models (A, gl, G)$. The usual non-circular assume-guarantee rule reads as follows in our context:

$$K \sqsubseteq_{A,gl} G \wedge E \sqsubseteq A \implies K \sqsubseteq_{E,gl} G$$

where $E \sqsubseteq A$ denotes that for any component G and gl such that $\sqsubseteq_{G,gl}$ is defined $E \sqsubseteq_{G,gl} A$. This rule relates the behavior of K, when composed with the abstract environment A, to the behavior of K, when composed with its actual environment E. However it is quite limited as it imposes a very strong condition on E. Hence the following rule which is commonly referred to as *circular reasoning*.

$$K \sqsubseteq_{A,gl} G \wedge E \sqsubseteq_{G,gl} A \implies K \sqsubseteq_{E,gl} G$$

Note that E and K may symmetrically rely on each other. For a given contract framework, this property can be proven by an induction based on the semantics of composition and refinement. Unfortunately, circular reasoning is not sound in general. In particular it does not hold for parallel composition with synchronizations (as in Petri Nets or process algebras) or instantaneous mutual dependencies between inputs and outputs (as in synchronous formalisms). The following example illustrates one possible reason for the non validity of circular reasoning[3].

Example 2. Consider a contract framework where components are labeled transition systems and composition is strong synchronization between corresponding labels and interleaving of others denoted $\|$. Define conformance as simulation and satisfaction as the usual relation defined in Example 1. The circular reasoning rule translates into: if $K \| A$ is simulated by $G \| A$ and $E \| G$ is simulated by $A \| G$ then $K \| E$ is simulated by $G \| E$.

In the example of Figure 3, both G and A forbid a synchronization between b_K and b_E from occurring. This allows their respective refinements according to $\sqsubseteq^{\preccurlyeq}$, namely K and E, to offer respectively b_K and b_E, since they can rely on respectively G and A to forbid their actual occurrence. But obviously, the composition $K \| E$ now allows a synchronization between b_K and b_E.

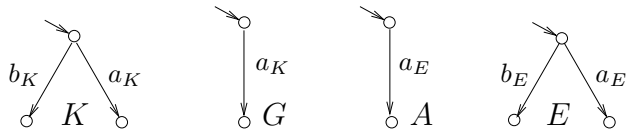

Figure3. $K \| A \preccurlyeq G \| A$ and $E \| G \preccurlyeq A \| G$ but $K \| E \not\preccurlyeq G \| E$.

A second rule which is used for compositional reasoning in most frameworks is: if $I \sqsubseteq S$, then $I \parallel E \sqsubseteq S \parallel E$. It states that if an implementation I refines its specification S then it refines it in any environment E. The equivalent of this rule for satisfaction is more complex as refinement here relates closed systems.

Definition 4. *Satisfaction $\models$ is preserved by composition iff for any component E, gl such that $S_{gl} = \mathcal{P}_E \cup \mathcal{P}$ for some $\mathcal{P}$ such that $\mathcal{P} \cap \mathcal{P}_E = \emptyset$ and gl_E, E_1, E_2 such that $E = gl_E\{E_1, E_2\}$, the following holds for any components I, S on $\mathcal{P}$:*

$$I \sqsubseteq_{E,gl} S \implies gl_1\{I, E_1\} \sqsubseteq_{E_2, gl_2} gl_1\{S, E_1\}$$

where gl_1 and gl_2 are such that $gl \circ gl_E = gl_2 \circ gl_1$.

Theorem 1. *Suppose that circular reasoning is sound and satisfaction is preserved by composition. If $\forall i \; \exists gl_{E_i} : \; gl \circ gl_I = gl_i \circ gl_{E_i}$ then to prove that $\{\mathcal{C}_i\}_{i=1}^n$ dominates $\mathcal{C}$ w.r.t. gl, it is sufficient to prove that condition* **(1)** *holds.*

This condition reduces a dominance proof to a set of satisfaction checks, one for proving refinement between the guarantees and n for discharging individual assumptions.

[3] Note that non-determinism may also be a problem.

3 Circular reasoning in practice

In this section, we show how the results presented in the previous section have been applied within the SPEEDS project: we build two contract frameworks, called L0 and L1, and show how to combine them.

3.1 The L0 framework

A component K with interface $\mathcal{P}_K$ at level L0 of HRC is defined as a set of behaviors in the form of traces, or runs, over $\mathcal{P}_K$. The behaviors correspond to the history of values seen at the ports of the component for each particular behavior. For instance, these histories could be the traces generated by a labeled transition system (LTS). Composition is defined as a composite that retains only the matching behaviors of the components. If the ports of the two components have the same names, composition at the level of trace sets boils down to a simple intersection of the sets of behaviors. Because in our framework components must have disjoint sets of ports under composition, we must introduce glues, or connectors, as explicit components that establish a synchronous relation between the histories of connected ports. The collection of these simple connectors forms the glues $gl \in GL$ of our framework at the L0 level.

We can model a glue as an extra component K_{gl}, whose set of ports includes all the ports of the components involved in the composition. This component has as set of behaviors all the identity traces. Composition can then be taken as the intersection of the sets of behaviors of the components, together with the glue. To make this work, we must also equalize the ports of all trace sets using inverse projection $\mathbf{proj}^{-1}_{\mathcal{P}_i,\mathcal{P}}$, which extends behaviors over $\mathcal{P}_1$ with the appropriate additional ports of $\mathcal{P}$. If we denote the interface of the composite as $\mathcal{P}_{gl}$, and if $\mathcal{K} = \{K_1, \ldots, K_n\}$ is a set of components such that $\mathcal{P}_1, \ldots, \mathcal{P}_n$ are pairwise disjoint, then a glue gl for $\mathcal{K}$ is a component K_{gl} defined on the ports $\mathcal{P} = \mathcal{P}_{gl} \cup (\bigcup_{i=1}^{n} \mathcal{P}_i)$, and:

$$
\begin{aligned}
&K \Leftarrow gl\{K_1, \ldots, K_n\} \\
&= \mathbf{proj}_{p_{gl},\mathcal{P}} \left(K_{gl} \cap \mathbf{proj}^{-1}_{\mathcal{P}_1,\mathcal{P}} (K_1) \cap \cdots \cap \mathbf{proj}^{-1}_{\mathcal{P}_n,\mathcal{P}} (K_n) \right)
\end{aligned}
$$

The definition of $\circ$ is straightforward: since glues are themselves components, their composition follows the same principle as component composition. Finally, the $\cong$ relation on $\mathcal{K}$ is taken as equality of sets of traces.

In the L0 model there exists a unique maximal component satisfying a contract $\mathcal{C}$, namely $M_{\mathcal{C}} = G \cup \neg A$, where $\neg$ denotes the operation of complementation on the set of all behaviors over ports $\mathcal{P}_A$. A contract $\mathcal{C} = (A, G)$ is in *canonical form* when $G = M_{\mathcal{C}}$. Every contract has an equivalent contract in canonical form, which is obtained by replacing G with $M_{\mathcal{C}}$. The operation of computing a canonical form is well defined, since the maximal implementation is unique, and it is idempotent. It is easy to show that $K \models \mathcal{C}$ if and only if $K \subseteq M_{\mathcal{C}}$.

The L0 contract framework has strong compositional properties, which derive from its simple definition and operators. The theory, however, depends on

the effectiveness of certain operators, complementation in particular, which are necessary for the computation of canonical forms. While the complete theory can be formulated without the use of canonical forms, complementation remains fundamental in the definition of contract composition, which is at the basis of system construction. Circular reasoning is not sound for contracts which are *not* in canonical form. This is a limitation of the L0 framework, since working with canonical forms could prove computationally hard.

3.2 The L1 framework

L1 composition is based on *interactions*, which involve non-empty sets of ports. An interaction is defined by the components that synchronize when it takes place and the ports through which these components synchronize. Interactions are structured into connectors which are used as a mechanism for *encapsulation*: only these connectors appear at the interface of a composite component. This enables to abstract the behavior of a component in a black-box manner, by describing which connector is triggered but not exactly which interaction takes place. Furthermore L1 is expressive enough to encompass synchronous systems.

Definition 5. *An* atomic component *on a interface $\mathcal{P}$ is defined by an LTS $K = (Q, q^0, 2^{\mathcal{P}}, \longrightarrow)$, where Q is a set of states, q^0 an initial state and $\longrightarrow \subseteq Q \times 2^{\mathcal{P}} \times Q$ is a transition relation.*

Note that atomic components are labeled by sets of ports rather than ports because we allow several ports of a component to be triggered at the same time.

Definition 6. *An* interaction *is a non-empty set of ports. A* connector γ *is defined by a set of ports S_γ called the* support set *of γ, a port p_γ called its* exported port *and a set $\mathcal{I}(\gamma)$ of interactions in S_γ.*

The intuition behind support set and exported port is illustrated in Figure 4, where connectors relate in a composition a set of inner ports (of the subcomponents) to an outer port (of the composite component). One should keep in mind that a connector γ, and thus the exported port p_γ, represents a set of interactions rather than a single interaction.

Typical connectors represent rendezvous (only one interaction, equal to the support set), broadcast (all the interactions containing a specific port called *trigger*) and also mutual exclusion (some interactions but not their union).

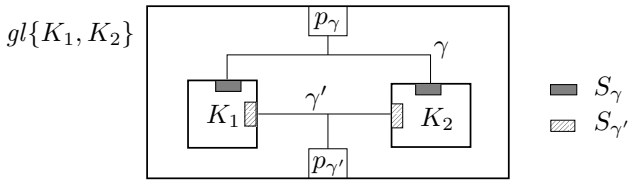

Figure4. The role of connectors in a composition

We now define glues as sets of connectors which may be used together in order to compose components.

Definition 7. *A glue gl on a support set S_{gl} is a set of connectors with distinct exported ports and with support sets included in S_{gl}.*

A glue gl defines as exported interface $\mathcal{P}_{gl}$ the set $\{p_\gamma \mid \gamma \in gl\}$. Besides, $\mathcal{I}(gl)$ denotes the set of all interactions of the connectors in gl, i.e.: $\mathcal{I}(gl) = \bigcup_{\gamma \in gl} \mathcal{I}(\gamma)$. In Figure 4, gl is composed of connectors γ and γ' and defines a composite component denoted $gl\{K_1, K_2\}$.

Definition 8. *A* component *is either an atomic component or it is inductively defined as the composition of a set of components $\{K_i\}_{i=1}^n$ with disjoint interfaces $\{\mathcal{P}_i\}_{i=1}^n$ using a glue gl on $\mathcal{P} = \bigcup_{i=1}^n \mathcal{P}_i$. Such a composition is called a* composite component *on $\mathcal{P}_{gl}$ and it is denoted $gl\{K_i\}_{i=1}^n$.*

So far we have defined components and glues. Glues can be composed so as to allow flattening of components. Such a composition requires to handle *hierarchical connectors* built by merging connectors defined at different levels of hierarchy. The definition of the operator $\circ$ used for this purpose is omitted here and can be found in [10]. Connectors whose exported ports and support sets are not related are called *disjoint* and need not be composed. The operator $\circ$ is then easily extended to glues: the composition $gl \circ gl'$ of two glues gl and gl' is obtained from $gl \cup gl'$ by inductively composing all connectors which are not disjoint.

We can now formally define the flattened form of a component. This in turn will allow us to provide an equivalence relation between components based on the semantics of their flattened form. A component is called *flat* if it is atomic or of the form $gl\{K_1, \ldots, K_n\}$, where all K_i are atomic components. A component that is not flat is called *hierarchical*. A hierarchical component K is of the form $gl\{K_1, \ldots, K_n\}$ such that at least one K_i is composite. Thus, such a K can be represented as $gl\{gl'\{\mathcal{K}^1\}, \mathcal{K}^2\}$, where $\mathcal{K}^1$ and $\mathcal{K}^2$ are sets of components.

Definition 9. *The* flattened form *of a component K is denoted $flat(K)$ and defined inductively as:*

- *if K is a flat component, then $flat(K)$ is equal to K.*
- *otherwise, K is of the form $gl\{gl'\{\mathcal{K}^1\}, \mathcal{K}^2\}$, and then $flat(K)$ is the flattened form of $(gl \circ gl')\{\mathcal{K}^1 \cup \mathcal{K}^2\}$.*

Definition 10. *The* semantics $[\![K]\!]$ *of a flat component $K = gl\{K_1, \ldots, K_n\}$ is defined as $(Q, q^0, \mathcal{I}(gl), \longrightarrow)$, where $Q = \prod_{i=1}^n Q_i$, $q^0 = (q_1^0, \ldots, q_n^0)$ and $\longrightarrow$ is such that: given two states $q^1 = (q_1^1, \ldots, q_n^1)$ and $q^2 = (q_1^2, \ldots, q_n^2)$ in Q and an interaction $\alpha \in \mathcal{I}(gl)$, $q^1 \xrightarrow{\alpha} q^2$ if and only if $\forall i, q_i^1 \xrightarrow{\alpha_i}_i q_i^2$, where $\alpha_i = \alpha \cap \mathcal{P}_i$.*

We use the convention that $\forall q : q \xrightarrow{\emptyset} q$ so components not involved in an interaction do not move. Thus the semantics of a flat component is obtained as the composition of its constituting LTS where labels are synchronized according to the interactions of $\mathcal{I}(gl)$.

We then define equivalence $\cong$ as follows: two components are equivalent if their flattened forms have the same semantics. Note that in practice one would prefer to define the semantics of a hierarchical component as a function of the semantics of its constituting components. In presence of encapsulation this requires to distinguish between closed and open systems and thus to provide two different semantics. More detail can be found in [10].

We now have the ingredients for defining the L1 component framework and we focus on its contract framework.

Definition 11. *$K_1 \preccurlyeq^{L1} K_2$ if and only if $[\![K_1]\!]$ is simulated by $[\![K_2]\!]$.*

Thus L1-conformance is identical to L0-conformance for components without non-observable non-determinism, and otherwise stronger. Note that in verification tools, in order to check trace inclusion efficiently, one will generally check simulation anyway. Satisfaction is defined as follows.

Definition 12. *A component K* satisfies *a contract $\mathcal{C} = (A, gl, G)$ for $\mathcal{P}_K$, denoted $K \models^{L1} (A, gl, G)$, if and only if:*

$$\begin{cases} gl\{K, A_{det}\} \preccurlyeq^{L1} gl\{G, A_{det}\} \\ (q_K, q_A)\,\mathcal{R}\,(q_G, q'_A) \wedge q_K \xrightarrow{\alpha}_K \implies q_G \xrightarrow{\alpha}_G \end{cases}$$

where A_{det} is the determinization of A, $\mathcal{R}$ is the relation on states proving that $gl\{K, A_{det}\} \preccurlyeq^{L1} gl\{G, A_{det}\}$ and $\alpha \in 2^{\mathcal{P}_K}$ is such that $\exists \alpha' \in \mathcal{I}(gl) : \alpha \subseteq \alpha'$.

That is, $\models^{L1}$ strengthens the satisfaction relation used in the L0 framework by: 1) determinizing A; 2) requiring every transition of K to have a counterpart in each related state of G — unless it is structurally forbidden by gl; however the target states of the transition must be related only if the environment allows this transition. As a consequence, $\models^{L1}$ allows circular reasoning.

3.3 Relaxed circular reasoning

We have presented in the previous sections two contract frameworks developed in the SPEEDS project. We show now how we use their respective tool chains together. Unifying the L0 and L1 component frameworks is quite straightforward. However we have introduced two different notions of satisfaction: $\models^{L0}$ and $\models^{L1}$ where the second one is strictly stronger than the first one. To combine results based on L0 and L1, we propose a rule called *relaxed circular reasoning* for two (possibly different) refinement relations:

$$K \sqsubseteq^1_{A,gl} G \wedge E \sqsubseteq^2_{G,gl} A \implies K \sqsubseteq^1_{E,gl} G$$

This rule generalizes circular and non-circular reasoning by not restricting $\sqsubseteq^2_{G,gl}$ to refinement under context $\sqsubseteq^1_{G,gl}$ or refinement in any context $\sqsubseteq^1$. Depending on which relation is the most restrictive it can be used in two different ways: if the first relation allows circular reasoning and is stronger than the second one (i.e. $K \sqsubseteq^1_{A,gl} G \implies K \sqsubseteq^2_{A,gl} G$) then our new rule relaxes circular reasoning by

requiring $E \sqsubseteq^2_{G,gl} A$ rather than $E \sqsubseteq^1_{G,gl} A$; symmetrically if the first relation does not allow circular reasoning and refinement in any context $\sqsubseteq^1$ is stronger than the second one then this rule relaxes non circular reasoning by requiring $E \sqsubseteq^2_{G,gl} A$ rather than $E \sqsubseteq^1 A$. Interestingly, relaxed circular reasoning can be used both ways for L0- and L1-satisfaction. First it leads to a relaxed sufficient condition for dominance in L1.

Theorem 2. *$K \sqsubseteq^{L1}_{A,gl} G \wedge E \sqsubseteq^{L0}_{G,gl} A$ implies $K \sqsubseteq^{L1}_{E,gl} G$.*

Theorem 3. *If $\forall i \, \exists gl_{E_i} : gl \circ gl_I = gl_i \circ gl_{E_i}$ the following is sufficient to prove that $\mathcal{C}$ dominates $\{\mathcal{C}_i\}_{i=1..n}$ w.r.t. gl:*

$$\begin{cases} gl_I\{G_1, \ldots, G_n\} \models^{L1} \mathcal{C} \\ \forall i : gl_{E_i}\{A, G_1, \ldots, G_{i-1}, G_{i+1}, \ldots, G_n\} \models^{L0} \mathcal{C}_i^{-1} \end{cases}$$

This is particularly interesting since checking L0-satisfaction is obviously less costly than checking L1-satisfaction. Thus, checking that contracts $\{\mathcal{C}_i\}_{i=1}^n$ L1-dominate a contract $\mathcal{C}$ requires one L1-satisfaction check and n L0-satisfaction checks which increases the scalability of dominance checks. Moreover, dominance can be established more often. Second:

Theorem 4. *$K \sqsubseteq^{L0}_{A,gl} G \wedge E \sqsubseteq^{L1}_{G,gl} A$ implies $K \sqsubseteq^{L0}_{E,gl} G$.*

This result made it possible in SPEEDS to incorporate results from tools checking L0-satisfaction with results obtained through L1-dominance (implemented by a set of L1-satisfaction checks), thus building a complete tool chain.

4 Conclusion and future work

The work presented in this paper has been motivated by necessity of combining contract-based verification tools and corresponding results for two component frameworks L0 and L1 defined in the context of the European SPEEDS project. In particular, we were interested in using dominance results established in L1 — and which cannot be obtained using the L0 refinement relation — for further reasoning in L0. To that purpose, we have presented an abstract notion of contract framework for a given component framework that defines three different notions of refinement, that is, conformance, dominance and satisfaction. We show how to derive these notions from refinement of closed systems and refinement under context and we provide a methodology for compositional and hierarchical verification of global properties.

We have studied circular reasoning as a powerful means for proving dominance. As circular reasoning does not always hold for usual notions of refinement, we provide proof rules for dominance relying on a relaxed notion of circular reasoning based on two notions of refinement. We have then shown that our abstract framework is general enough to represent both L0 and L1 as specific instances and proved that the L0 and L1 refinement relations satisfy the condition for relaxed circular reasoning.

These theoretical results have been implemented in the context of the HRC model developed in the SPEEDS project. There, analysis tools were built for both frameworks and the theory presented in this paper was used to combine them on concrete examples. A more detailed description of the design and verification methodology is provided in [13]. In particular, for every design step, a set of checks to be carried out is defined. Future work includes strategies to be deployed when some check fails, which should depend on the failed check and on whether a top-down or bottom-up approach is followed, and which should be able to exploit more detailed diagnostic information when it exists.

References

1. R. Alur and T. A. Henzinger. Reactive modules. *Formal Methods in System Design*, 15(1):7–48, 1999.
2. L. Benvenuti, A. Ferrari, L. Mangeruca, E. Mazzi, R. Passerone, and C. Sofronis. A contract-based formalism for the specification of heterogeneous systems. In *Proc. of FDL'08*, Stuttgart, Germany, September 23–25, 2008.
3. J. M. Cobleigh, G. S. Avrunin, and L. A. Clarke. Breaking up is hard to do: An evaluation of automated assume-guarantee reasoning. *ACM Trans. Softw. Eng. Methodol.*, 17(2), 2008.
4. Combest project. `http://www.combest.eu`.
5. L. de Alfaro and T. A. Henzinger. Interface automata. In *Proc. of ESEC/SIGSOFT FSE'01*, pages 109–120. ACM Press, 2001.
6. B. Delahaye, B. Caillaud, and A. Legay. Probabilistic contracts: A compositional reasoning methodology for the design of stochastic systems. In *Proc. of ACSD'10*, pages 223–232, 2010.
7. I. B. Hafaiedh, S. Graf, and S. Quinton. Reasoning about safety and progress using contracts. In *Proc. of ICFEM'10*, pages 436–451, 2010.
8. K. G. Larsen, U. Nyman, and A. Wasowski. Interface input/output automata. In *Proc. of FM'06*, volume 4085 of *LNCS*, pages 82–97, 2006.
9. P. Maier. *A Lattice-Theoretic Framework for Circular Assume-Guarantee Reasoning*. PhD thesis, Universität des Saarlandes, 2003.
10. Extended version of this paper including proofs. `http://www-verimag.imag.fr/~quinton/TIMOBD11-proofs.pdf`.
11. S. Quinton and S. Graf. Contract-based verification of hierarchical systems of components. In *Proc. of SEFM'08*, pages 377–381. IEEE Computer Society, 2008.
12. J. Sifakis. A framework for component-based construction. In *Proc. of SEFM'05*, pages 293–300. IEEE Computer Society, 2005.
13. SPEEDS methodology for speculative and exploratory design in systems engineering. `http://www.speeds.eu.com`.
14. S. Tripakis, B. Lickly, T. A. Henzinger, and E. A. Lee. On relational interfaces. In *Proc. of EMSOFT'09*, pages 67–76, 2009.

A Relaxed circular reasoning

NB: The following theorem is given for $\sqsubseteq^1$ stronger than or equal to $\sqsubseteq^2$. It implies Theorem 1 and also Theorem 3 when combined with the results of the following sections.

(**CR**) denotes the circular reasoning rule defined in Section 2 and (**CMP**) denotes the preservation of satisfaction by composition rule.

Theorem 5. *Suppose that relaxed circular reasoning is sound for $\sqsubseteq^1$ and $\sqsubseteq^2$ and satisfaction is preserved by composition. If $\forall i,\ \exists gl_{E_i},\ gl \circ gl_I = gl_i \circ gl_{E_i}$ then to prove that $\{\mathcal{C}_i\}_{i=1}^n$ dominates $\mathcal{C}$ w.r.t. gl_I, it is sufficient to prove that:*

$$\begin{cases} gl_I\{G_1, \ldots, G_n\} \models^1 \mathcal{C} \\ \forall i,\ gl_{E_i}\{A, G_1, \ldots, G_{i-1}, G_{i+1}, \ldots, G_n\} \models^2 \mathcal{C}_i^{-1} \end{cases}$$

Proof. For every $i \in [1, n]$, let K_i be a component on $\mathcal{P}_i$. Suppose the following:

1. $\forall i,\ \exists gl_{E_i},\ gl \circ gl_I = gl_i \circ gl_{E_i}$
2. $gl_I\{G_1, \ldots, G_n\} \sqsubseteq^1_{A,gl} G$
3. $\forall i,\ gl_{E_i}\{A, G_1, \ldots, G_{i-1}, G_{i+1}, \ldots, G_n\} \sqsubseteq^2_{G_i,gl_i} A_i$
4. $\forall i,\ K_i \sqsubseteq^1_{A_i,gl_i} G_i$

We aim at proving that $gl_I\{K_1, \ldots, K_n\} \models^1 \mathcal{C}$, that is: $gl_I\{K_1, \ldots, K_n\} \sqsubseteq^1_{A,gl} G$. For this, we show by induction that for any l in $[0, n]$, for any partition $\{J, K\}$ of $[1, n]$ such that $|J| = l$:

$$\begin{cases} gl_I\{\mathcal{K}^J \cup \mathcal{G}^K\} \sqsubseteq^1_{A,gl} G \\ \forall i \in K,\ gl_{E_i}\{A, \mathcal{E}_i^{J,K}\} \sqsubseteq^2_{G_i,gl_i} A_i \end{cases}$$

with $\mathcal{K}^J = \{K_j\}_{j\in J}$, $\mathcal{G}^K = \{G_k\}_{k\in K}$ and with $\mathcal{E}_i^{J,K} = \mathcal{K}^J \cup (\mathcal{G}^K \backslash \{G_i\})$.

- $l = 0$. By (2) and (3) the property holds.
- $0 \leq l < n$. We suppose that our property holds for l. Let $\{J', K'\}$ be a partition of $[1, n]$ such that $|J'| = l + 1$. Let q be an element of J'. We fix $J = J' \backslash \{q\}$ and $K = K' \cup \{q\}$.

Step 1 We first prove that $gl_I\{\mathcal{K}^{J'} \cup \mathcal{G}^{K'}\} \sqsubseteq^1_{A,gl} G$.

$$\begin{cases} K_q \sqsubseteq^1_{A_q,gl_q} G_q \text{ from (4)} \\ gl_{E_q}\{A,\ \mathcal{E}_q^{J,K}\} \sqsubseteq^2_{G_q,gl_q} A_q \end{cases}$$

The second property is our recurrence hypothesis, as $q \in K$. Thus, by circular reasoning (**CR**):

$$K_q \sqsubseteq^1_{gl_{E_q}\{A,\mathcal{E}_q^{J,K}\},gl_q} G_q$$

As refinement under context is preserved by composition, we obtain by (**CMP**):

$$gl_I\{K_q, \mathcal{E}_q^{J,K}\} \sqsubseteq^1_{A,gl} gl_I\{G_q, \mathcal{E}_q^{J,K}\}$$

This is equivalent to $gl_I\{\mathcal{K}^{J'} \cup \mathcal{G}^{K'}\} \sqsubseteq^1_{A,gl} gl_I\{\mathcal{K}^J \cup \mathcal{G}^K\}$.
Finally, by using the recurrence hypothesis: $gl_I\{\mathcal{K}^{J'} \cup \mathcal{G}^{K'}\} \sqsubseteq^1_{A,gl} G$.

Step 2 We now have to prove that:

$$\forall i \in K',\ gl_{E_i}\{A, \mathcal{E}_i^{J',K'}\} \sqsubseteq^2_{G_i,gl_i} A_i$$

We fix $i \in K'$. We have proved in step 1 that:

$$K_q \sqsubseteq^1_{gl_{E_q}\{A,\mathcal{E}_q^{J,K}\},gl_q} G_q$$

As $\sqsubseteq^1$ is stronger than $\sqsubseteq^2$, we also have:

$$K_q \sqsubseteq^2_{gl_{E_q}\{A,\mathcal{E}_q^{J,K}\},gl_q} G_q$$

$K = K' \cup \{q\}$, so $i \in K$. Thus, by compositionality (**CMP**), we obtain:

$$gl_{E_i}\{K_q, A, \mathcal{E}_q^{J,K\setminus\{i\}}\} \sqsubseteq^2_{G_i,gl_i} gl_{E_i}\{G_q, A, \mathcal{E}_q^{J,K\setminus\{i\}}\}$$

This boils down to $gl_{E_i}\{A, \mathcal{E}_i^{J',K'}\} \sqsubseteq^2_{G_i,gl_i} gl_{E_i}\{A, \mathcal{E}_i^{J,K}\}$.
Hence, using the recurrence hypothesis: $gl_{E_i}\{A, \mathcal{E}_i^{J',K'}\} \sqsubseteq^2_{G_i,gl_i} A_i$.

Conclusion By applying our property to $l = n$, we get:

$$gl_I\{K_1, \ldots, K_n\} \sqsubseteq^1_{A,gl} G$$

B Composition of glues in L1

Glues can be composed so as to allow flattening of components. Such a composition requires to handle *hierarchical connectors* built by merging connectors defined at different levels of hierarchy. More precisely, if $p_{\gamma'} \in S_\gamma$ then γ and γ' can be composed to form a hierarchical connector denoted $\gamma \circ \gamma'$ with support set $S_\gamma \cup S_{\gamma'} \setminus \{p_{\gamma'}\}$, with exported port p_γ and whose interaction set is computed from $\mathcal{I}(\gamma)$ as follows: each interaction α in which $p_{\gamma'}$ occurs is replaced by a set of interactions identical to α except that the occurrence of $p_{\gamma'}$ is replaced by an interaction of $\mathcal{I}(\gamma')$.

Definition 13. *Let γ and γ' be two connectors such that $p_{\gamma'} \in S_\gamma$. The* hierarchical *connector $\gamma \circ \gamma'$ resulting from their composition is defined as follows:*

- $S_{\gamma\circ\gamma'} \triangleq S_\gamma \cup S_{\gamma'} \setminus \{p_{\gamma'}\}$
- $p_{\gamma\circ\gamma'} \triangleq p_\gamma$
- $\mathcal{I}(\gamma \circ \gamma') \triangleq \{\alpha \in \mathcal{I}(\gamma) \mid p_{\gamma'} \notin \alpha\} \cup \{\alpha.\alpha' \mid \alpha.p_{\gamma'} \in \mathcal{I}(\gamma) \wedge \alpha' \in \mathcal{I}(\gamma')\}$

Example 3. Let γ be such that $S_\gamma = \{p_{\gamma'}, c\}$ and $\mathcal{I}(\gamma) = \{\{p_{\gamma'}, c\}\}$; consider also γ' with $S_{\gamma'} = \{a, b\}$ and $\mathcal{I}(\gamma') = \{\{a\}, \{b\}, \{a, b\}\}$. Then $\gamma \circ \gamma'$ has as support set $\{a, b, c\}$, exported port p_γ and interaction set $\{\{a, c\}, \{b, c\}, \{a, b, c\}\}$.

Note that if $p_\gamma \in S_{\gamma'}$ and $p_{\gamma'} \in S_\gamma$, then composing γ and γ' would result in a circular connector (a connector γ'' with $p_{\gamma''} \in S_{\gamma''}$), hence this situation must be prevented. Connectors whose exported ports and support sets are not related are called *disjoint* and need not be composed. The operator $\circ$ is then easily extended to glues: the composition $gl \circ gl'$ of two glues gl and gl' is obtained from $gl \cup gl'$ by inductively composing all connectors which are not disjoint.

C Consistency between L0 and L1

In this section we prove the following theorems:

1. L1-satisfaction implies L0-satisfaction
2. Soundness of circular reasoning for $\sqsubseteq^{L1}$
3. Soundness of relaxed circular reasoning for $\sqsubseteq^{L1}$ and $\sqsubseteq^{L0}$
4. Soundness of relaxed circular reasoning for $\sqsubseteq^{L0}$ and $\sqsubseteq^{L1}$

Before this we recall some basic definitions and properties of LTS and provide characterizations of $\sqsubseteq^{L1}$ and $\sqsubseteq^{L0}$ which are used in the proofs. Throughout this section we deal only with atomic components and we discuss in the next section how to handle composite components. Furthermore, K is implicitely defined as $(Q_K, q_K^0, 2^{\mathcal{P}_K}, \longrightarrow_K)$ and q_K denotes a state of Q_K. A, G, E are defined similarly. Finally, we use everywhere the convention that $\forall q \xrightarrow{\emptyset} q$.

C.1 Basic definitions and properties of LTS

Definition 14 (Trace). *A (finite)* trace *of an LTS S is a sequence of labels $a_1.a_2 \dots a_n$ for which there exists a sequence of states $q_0.q_1 \dots q_n$ such that $q_0 = q^0$ and for every $0 \leq i < n$ there is a transition $q_i \xrightarrow{a_{i+1}} q_{i+1}$. The* length *of the trace is n. The set of traces of S is denoted $Tr(S)$.*

Traces are in general denoted σ. The sequence $q_0.q_1 \dots q_n$ is called the sequence of states *corresponding to* $a_1.a_2 \dots a_n$. In a deterministic LTS, every trace has only one corresponding sequence of states.

Definition 15 (Determinization). *The* determinization *of an LTS S, denoted S^{det}, is the LTS $(2^Q, \{q^0\}, \Sigma, \longrightarrow_{det})$ where $\longrightarrow_{det}$ consists of triples in $2^Q \times \Sigma \times 2^Q$ and is the smallest relation such that for $\mathcal{Q}, \mathcal{Q}' \subseteq Q$, $\mathcal{Q} \xrightarrow{a}_{det} \mathcal{Q}'$ if and only if $\mathcal{Q}' \neq \emptyset$ and $\mathcal{Q}' = \{q' \mid \exists q \in \mathcal{Q} \ s.t. \ q \xrightarrow{a} q'\}$.*

The condition $\mathcal{Q}' \neq \emptyset$ ensures that there exists a transition in the determinized LTS only if there exists at least one corresponding transition in the original LTS. Note that for a given S, S^{det} is unique and deterministic. Uniqueness and determinism come from the fact that in every state $\mathcal{Q}'$ is uniquely defined.

Lemma 1. *For any LTS S, it holds that $Tr(S) = Tr(S^{det})$.*

Proof. We prove by induction the following lemma: for any $l \geq 0$, the set of traces of length l of S is equal to the set of traces of length l of S^{det}, and for any σ of length l in these sets, if $\mathcal{Q}_0.\mathcal{Q}_1 \dots \mathcal{Q}_{l-1}$ is the (unique) sequence of states corresponding to σ in S^{det}, then we have $\mathcal{Q}_{l-1} = \{q_{l-1} \in Q \mid q_0.q_1 \dots q_{l-1}$ is a sequence of states in S corresponding to $\sigma\}$.

Definition 16 (Simulation). *Let S_1 and S_2 be two LTS. A relation $\mathcal{R} \subseteq Q_1 \times Q_2$ is a* simulation *relation of S_2 by S_1 iff $q_1^0 \,\mathcal{R}\, q_2^0$ and for any pair $(q_1, q_2) \in Q_1 \times Q_2$ and any $q_1' \in Q_1$:*

$$q_1 \,\mathcal{R}\, q_2 \text{ and } q_1 \xrightarrow{a}_1 q_1' \text{ implies } \exists q_2' \in Q_2 \text{ such that } q_2 \xrightarrow{a}_2 q_2' \text{ and } q_1' \,\mathcal{R}\, q_2'$$

S_1 is simulated by S_2 if and only if there exists such a relation.

Intuitively, an LTS S_1 is simulated by S_2 if any reachable state q_1 of S_1 can be mapped to a state q_2 in S_2 such that all labels enabled in q_1 (w.r.t S_1) are also enabled in q_2 (w.r.t. S_2).

Lemma 2. *For any LTS S, S is simulated by S^{det}.*

Proof. Let $\mathcal{R} \subseteq Q \times 2^Q$ be a relation defined as follows: $q \,\mathcal{R}\, \mathcal{Q}$ if and only if $q \in \mathcal{Q}$, for any $q \in Q$ and $\mathcal{Q} \subseteq Q$. Relation $\mathcal{R}$ is a simulation of S^{det} by S.

Lemma 3. *If S_1 is simulated by S_2 then $Tr(S_1) \subseteq Tr(S_2)$.*

Proof. Again, the proof is a simple induction on the length of the traces.

Simulation is strictly stronger than inclusion of traces. Furthermore, inclusion of traces and simulation can be related by the following equivalence.

Lemma 4. *$Tr(S_1) \subseteq Tr(S_2)$ if and only if S_1^{det} is simulated by S_2^{det}.*

Proof. We decompose this proof into two implications.
$\Longrightarrow$: Suppose every trace of S_1 is a trace of S_2. We define $\mathcal{R} \subseteq 2^{Q_1} \times 2^{Q_2}$ as follows: $\{q_1^0\} \,\mathcal{R}\, \{q_2^0\}$, and if $\mathcal{Q}_1 \,\mathcal{R}\, \mathcal{Q}_2$ and $\mathcal{Q}_1 \xrightarrow{a} \mathcal{Q}_1'$, then $\mathcal{Q}_1' \,\mathcal{R}\, \mathcal{Q}_2'$, where $\mathcal{Q}_2' = \{q_2' \mid \exists q_2 \in \mathcal{Q}_2 \; s.t. \; q_2 \xrightarrow{a} q_2'\}$. Relation $\mathcal{R}$ is a simulation of S_2^{det} by S_1^{det}.
$\Longleftarrow$: is a direct consequence of Lemma 3 and Lemma 1.

C.2 Characterization of L1-satisfaction

Lemma 5. *$[\![gl\{K, A\}]\!]$ is simulated by $[\![gl\{G, A\}]\!]$ if and only if there exists a relation $\mathcal{R} \subseteq (Q_K \times Q_A) \times Q_G$ such that:*

1. $(q_K^0, q_A^0) \,\mathcal{R}\, q_G^0$
2. *if $(q_K, q_A) \,\mathcal{R}\, q_G$ and $(q_K, q_A) \xrightarrow{\alpha} (q_K', q_A')$ for $\alpha \in \mathcal{I}(gl)$, then there exists q_G' such that $q_G \xrightarrow{\alpha_K} q_G'$ and $(q_K', q_A') \,\mathcal{R}\, q_G'$ where $\alpha_K = \alpha \cap \mathcal{P}_K$.*

Proof. We consider first the right-to-left implication. Define $\mathcal{R}'$ such that $(q_K, q_A) \,\mathcal{R}'\, (q_G, q_A')$ if and only if $q_A = q_A'$ and $(q_K, q_A) \,\mathcal{R}\, q_G$.

– By definition of $\mathcal{R}'$ and using (1): $(q_K^0, q_A^0) \,\mathcal{R}'\, (q_G^0, q_A^0)$

– Suppose that $(q_K, q_A)\,\mathcal{R}'\,(q_G, q_A)$ and $(q_K, q_A) \xrightarrow{\alpha} (q'_K, q'_A)$ for $\alpha \in \mathcal{I}(gl)$.
By definition of $\mathcal{R}'$ we have $(q_K, q_A)\,\mathcal{R}\,q_G$. Then from (2) we know that there exists q'_G such that $q_G \xrightarrow{\alpha_K} q'_G$ and $(q'_K, q'_A)\,\mathcal{R}\,q'_G$ where $\alpha_K = \alpha \cap \mathcal{P}_K$.
Thus we obtain $(q'_K, q'_A)\,\mathcal{R}\,(q'_G, q'_A)$.
Besides $(q_K, q_A) \xrightarrow{\alpha} (q'_K, q'_A)$ implies that $q_A \xrightarrow{\alpha_A} q'_A$ where $\alpha_A = \alpha \cap \mathcal{P}_A$.
Hence $(q_G, q_A) \xrightarrow{\alpha} (q'_G, q'_A)$ because $\alpha = \alpha_K \cup \alpha_A$.

Thus $\mathcal{R}'$ is a simulation.

Let us consider now the left-to-right implication. Suppose that there exists a simulation $\mathcal{R}'$. We define $\mathcal{R}$ such that $(q_K, q_A)\,\mathcal{R}\,q_G$ if and only if $(q_K, q_A)\,\mathcal{R}'\,(q_G, q'_A)$ for some q'_A.

1. By definition of $\mathcal{R}$ and simulation: $(q^0_K, q^0_A)\,\mathcal{R}\,q^0_G$
2. Suppose that $(q_K, q_A)\,\mathcal{R}\,q_G$ and $(q_K, q_A) \xrightarrow{\alpha} (q'_K, q'_A)$ for $\alpha \in \mathcal{I}(gl)$.
By definition of $\mathcal{R}$ we have $(q_K, q_A)\,\mathcal{R}'\,(q_G, q''_A)$ for some q''_A. It follows, because $\mathcal{R}'$ is a simulation, that $(q_G, q''_A) \xrightarrow{\alpha} (q'_G, q'''_A)$ and $(q'_K, q'_A)\,\mathcal{R}'\,(q'_G, q'''_A)$. We conclude from $(q_G, q''_A) \xrightarrow{\alpha} (q'_G, q'''_A)$ that $q_G \xrightarrow{\alpha_K} q'_G$ for $\alpha_K = \alpha \cap \mathcal{P}_K$ and from $(q'_K, q'_A)\,\mathcal{R}'\,(q'_G, q'''_A)$ that $(q'_K, q'_A)\,\mathcal{R}\,q'_G$.

Hence the result.

Lemma 6 (Characterization of L1). *$K \models^{L1} (A, gl, G)$ if and only if there exists a relation $\mathcal{R} \subseteq (Q_K \times 2^{Q_A}) \times Q_G$ such that:*

– $(q^0_K, \{q^0_A\})\,\mathcal{R}\,q^0_G$
– *if $(q_K, \mathcal{Q}_A)\,\mathcal{R}\,q_G$ and $q_K \xrightarrow{\alpha_K}_K q'_K$ with $\alpha_K \subseteq \alpha$ for some $\alpha \in \mathcal{I}(gl)$, then there exists q'_G such that:*
  1. *$q_G \xrightarrow{\alpha_K}_G q'_G$*
  2. *if there exists $q_A \in \mathcal{Q}_A$, $q'_A \in Q_A$ and α_A s.t. $q_A \xrightarrow{\alpha_A}_A q'_A$ and $\alpha_K \cup \alpha_A \in \mathcal{I}(gl)$, then $(q'_K, \mathcal{Q}'_A)\,\mathcal{R}\,q'_G$ where $\mathcal{Q}'_A$ is $\{q'_A \mid \exists q_A \in \mathcal{Q}_A$ s.t. $q_A \xrightarrow{\alpha_A}_A q'_A\}$.*

Note that it would be possible in the definition of L1-satisfaction to strengthen the condition on α_K by considering only interactions α_K such that $\alpha_K = \alpha \cap 2^{\mathcal{P}_K}$ for some $\alpha \in \mathcal{I}(gl)$.

Proof. This lemma is a direct consequence of Lemma 5 and of the definitions of L1-satisfaction and determinization. Remember that:

$$K \models^{L1} (A, gl, G) \triangleq \begin{cases} [\![gl\{K, A_{det}\}]\!] \text{ is simulated by } [\![gl\{G, A_{det}\}]\!] \\ (q_K, \mathcal{Q}_A)\,\mathcal{R}'\,(q_G, \mathcal{Q}'_A) \wedge q_K \xrightarrow{\alpha_K}_K \implies q_G \xrightarrow{\alpha_K}_G \end{cases}$$

where A_{det} is the determinization of A, $\mathcal{R}$ is the relation on states proving that $gl\{K, A_{det}\} \preccurlyeq^{L1} gl\{G, A_{det}\}$ and $\alpha_K \in 2^{\mathcal{P}_K}$ is such that $\exists \alpha \in \mathcal{I}(gl) : \alpha_K \subseteq \alpha$.

Let us show the left-to-right implication. Suppose that $K \models^{L1} (A, gl, G)$. According to the definition of L1-satisfaction, to Lemma 5 and to the definition of determinization, there exists a relation $\mathcal{R} \subseteq (Q_K \times 2^{Q_A}) \times Q_G$ such that:

a) $(q_K^0, \{q_A^0\})\,\mathcal{R}\,q_G^0$
b) if $(q_K, \mathcal{Q}_A)\,\mathcal{R}\,q_G$ and $(q_K, \mathcal{Q}_A) \xrightarrow{\alpha} (q'_K, \mathcal{Q}'_A)$ for $\alpha \in \mathcal{I}(gl)$, then there exists q'_G such that $q_G \xrightarrow{\alpha_K} q'_G$ and $(q'_K, \mathcal{Q}'_A)\,\mathcal{R}\,q'_G$ where $\alpha_K = \alpha \cap \mathcal{P}_K$
c) $(q_K, \mathcal{Q}_A)\,\mathcal{R}\,q_G \wedge q_K \xrightarrow{\alpha_K}_K \Longrightarrow q_G \xrightarrow{\alpha_K}_G$ with $\alpha_K \subseteq \alpha$ for some $\alpha \in \mathcal{I}(gl)$
d) $\mathcal{Q}_A \xrightarrow{\alpha_A}_{A_{det}} \mathcal{Q}'_A$ iff $\mathcal{Q}'_A \neq \emptyset$ and $\mathcal{Q}'_A = \{q'_A \mid \exists q_A \in \mathcal{Q}_A \ s.t.\ q_A \xrightarrow{\alpha_A} q'_A\}$.

Thus the condition on the initial state is trivially satisfied. Let us show conditions 1 and 2. Suppose that $(q_K, \mathcal{Q}_A)\,\mathcal{R}\,q_G$ and $q_K \xrightarrow{\alpha_K}_K q'_K$ with $\alpha_K \subseteq \alpha$ for some $\alpha \in \mathcal{I}(gl)$. Item (c) above implies condition 1. To show condition 2, let us suppose that there exists $q_A \in \mathcal{Q}_A$, $q'_A \in Q_A$ and α_A such that $q_A \xrightarrow{\alpha_A}_A q'_A$ and $\alpha_K \cup \alpha_A \in \mathcal{I}(gl)$. Item (d) implies that $\mathcal{Q}_A \xrightarrow{\alpha_A}_{A_{det}} \mathcal{Q}'_A$. Then item (b) implies that $(q'_K, \mathcal{Q}'_A)\,\mathcal{R}\,q'_G$.

Let us show now the right-to-left implication. Let $\mathcal{R}$ be as above and let us show that $K \models^{L1} (A, gl, G)$. As $\mathcal{R}'$ satisfies the conditions of Lemma 5 (where A is replaced by A_{det}), it holds that $[\![gl\{K, A_{det}\}]\!]$ is simulated by $[\![gl\{G, A_{det}\}]\!]$, where the simulation is defined by $\mathcal{R}'$ as $(q_K, \mathcal{Q}_A)\,\mathcal{R}'\,(q_G, \mathcal{Q}'_A)$ iff $Q_A = \mathcal{Q}'_A$ and $(q_K, \mathcal{Q}_A)\,\mathcal{R}\,q_G$. The second condition on $\mathcal{R}'$ required by the definition of $L1$-satisfaction is ensured by item 1 in the definition of $\mathcal{R}$.

C.3 Characterization of L0-satisfaction

Lemma 7. *$K \sqsubseteq^{L0}_{A,gl} G$ iff there exists a relation $\mathcal{R} \subseteq (2^{Q_K} \times Q_A) \times 2^{Q_G}$ with:*

- $(\{q_K^0\}, q_A^0)\,\mathcal{R}\,\{q_G^0\}$
- *if $(\mathcal{Q}_K, q_A)\,\mathcal{R}\,\mathcal{Q}_G$ and $(q_K, q_A) \xrightarrow{\alpha} (q'_K, q'_A)$ with $q_K \in \mathcal{Q}_K$ and $\alpha \in \mathcal{I}(gl)$, there exists $q_G \in \mathcal{Q}_G$ and $q'_G \in Q_G$ such that $q_G \xrightarrow{\alpha_K} q'_G$ and $(\mathcal{Q}'_K, q'_A)\,\mathcal{R}\,\mathcal{Q}'_G$ where $\alpha_K = \alpha \cap \mathcal{P}_K$, $\mathcal{Q}'_K = \{q'_K \mid \exists q_K \in \mathcal{Q}_K \ s.t.\ q_K \xrightarrow{\alpha_K}_K q'_K\}$ and $\mathcal{Q}'_G$ is defined as $\{q'_G \mid \exists q_G \in \mathcal{Q}_G \ s.t.\ q_G \xrightarrow{\alpha_K}_G q'_G\}$.*

Proof. By definition $K \sqsubseteq^{L0}_{A,gl} G$ iff $Tr(gl\{K, A\}) \subseteq Tr(gl\{G, A\})$. According to Lemma 4, this is equivalent with $gl\{K, A\}^{det}$ is simulated by $gl\{G, A\}^{det}$. Besides, using Lemma 5 and the definition of determinization, we know that there exists $\mathcal{R}$ as above if and only if $gl\{K^{det}, A\}$ is simulated by $gl\{G^{det}, A\}$. Thus we have to show that determinizing A is not necessary when A appears on both sides of the simulation relation.

Let us focus first on the left-to-right implication. We suppose that $gl\{K, A\}^{det}$ is simulated by $gl\{G, A\}^{det}$ and denote $\mathcal{R}'$ the corresponding simulation relation. We define $\mathcal{R}$ by: $(\mathcal{Q}_K, q_A)\,\mathcal{R}\,\mathcal{Q}_G$ if and only if $(\mathcal{Q}_K, \mathcal{Q}_A)\,\mathcal{R}'\,(\mathcal{Q}_G, \mathcal{Q}'_A)$ for some $\mathcal{Q}_A, \mathcal{Q}'_A$ such that $q_A \in \mathcal{Q}_A$. The condition on the initial states is obviously satisfied so let us focus on the second condition required from $\mathcal{R}$: suppose that $(\mathcal{Q}_K, q_A)\,\mathcal{R}\,\mathcal{Q}_G$ and $(q_K, q_A) \xrightarrow{\alpha} (q'_K, q'_A)$ with $q_K \in \mathcal{Q}_K$ and $\alpha \in \mathcal{I}(gl)$. By definition of $\mathcal{R}$ this implies that $(\mathcal{Q}_K, \mathcal{Q}_A)\,\mathcal{R}'\,(\mathcal{Q}_G, \mathcal{Q}''_A)$ with $q_A \in \mathcal{Q}_A$. Besides $(q_K, q_A) \xrightarrow{\alpha} (q'_K, q'_A)$ implies that $(\mathcal{Q}_K, \mathcal{Q}_A) \xrightarrow{\alpha}_{det} (\mathcal{Q}'_K, \mathcal{Q}'_A)$ where $\mathcal{Q}'_K = \{q'_K \mid \exists q_K \in \mathcal{Q}_K \ s.t.\ q_K \xrightarrow{\alpha_K}_K q'_K\}$ for $\alpha_K = \alpha \cap \mathcal{P}_K$. Thus,

because $\mathcal{R}'$ is a simulation there exists $\mathcal{Q}'_G, \mathcal{Q}'''_A$ such that $\mathcal{Q}_G \xrightarrow{\alpha_K} \mathcal{Q}'_G$ and $(\mathcal{Q}'_K, \mathcal{Q}'_A)\,\mathcal{R}'\,(\mathcal{Q}'_G, \mathcal{Q}'''_A)$. This implies condition 2 for $\mathcal{R}$.

Now we consider the right-to-left implication. We suppose that we have $\mathcal{R}$ as above and show that this implies that $gl\{K, A\}^{det}$ is simulated by $gl\{G, A\}^{det}$. Define $\mathcal{R}'$ as $(\mathcal{Q}_K, \mathcal{Q}_A)\,\mathcal{R}'\,(\mathcal{Q}_G, \mathcal{Q}'_A)$ if and only if $\mathcal{Q}_A = \mathcal{Q}'_A$ and $(\mathcal{Q}_K, q_A)\,\mathcal{R}'\,\mathcal{Q}_G$ for some $q_A \in \mathcal{Q}_A$. We show that $\mathcal{R}'$ is a simulation using an argument similar to that of the left-to-right implication.

C.4 L1-satisfaction implies L0-satisfaction

Proof. We suppose that $K \sqsubseteq^{L1}_{A,gl} G$, and then we prove that $K \sqsubseteq^{L0}_{A,gl} G$ using the characterizations provided in Lemma 6 and 7.

As $K \sqsubseteq^{L1}_{A,gl} G$, there exists a relation $\mathcal{R} \subseteq (Q_K \times 2^{Q_A}) \times Q_G$ as in Lemma 6. We define $\mathcal{R}' \subseteq (2^{Q_K} \times Q_A) \times 2^{Q_G}$ as follows: $(\mathcal{Q}_K, q_A)\,\mathcal{R}'\,\mathcal{Q}_G$ iff there exists $\mathcal{Q}_A \subseteq Q_A$, $q_K \in \mathcal{Q}_K$ and $q_G \in \mathcal{Q}_G$ such that $q_A \in \mathcal{Q}_A$ and $(q_K, \mathcal{Q}_A)\,\mathcal{R}\,q_G$. Obviously $(\{q^0_K\}, q^0_A)\,\mathcal{R}'\,\{q^0_G\}$.

Now suppose $(\mathcal{Q}_K, q_A)\,\mathcal{R}'\,\mathcal{Q}_G$ and $(q_K, q_A) \xrightarrow{\alpha} (q'_K, q'_A)$ with $q_K \in \mathcal{Q}_K$ and $\alpha \in \mathcal{I}(gl)$. Denote $\alpha_K = \alpha \cap \mathcal{P}_K$ and $\alpha_A = \alpha \cap \mathcal{P}_A$ so that $\alpha = \alpha_K \cup \alpha_A$. We must show that there exists $q_G \in \mathcal{Q}_G$ and $q'_G \in Q_G$ such that $q_G \xrightarrow{\alpha_K} q'_G$ and $(\mathcal{Q}'_K, q'_A)\,\mathcal{R}\,\mathcal{Q}'_G$ where $\mathcal{Q}'_K = \{q'_K \mid \exists q_K \in \mathcal{Q}_K \ s.t.\ q_K \xrightarrow{\alpha_K}_K q'_K\}$ and $\mathcal{Q}'_G$ is defined as $\{q'_G \mid \exists q_G \in \mathcal{Q}_G \ s.t.\ q_G \xrightarrow{\alpha_K}_G q'_G\}$.

By definition we know that $(q_K, \mathcal{Q}_A)\,\mathcal{R}\,q_G$ for some $\mathcal{Q}_A \supseteq q_A$, $q_K \in \mathcal{Q}_K$ and $q_G \in \mathcal{Q}_G$. Besides $(q_K, q_A) \xrightarrow{\alpha} (q'_K, q'_A)$ implies that $q_K \xrightarrow{\alpha_K}_K q'_K$. Thus from L1-satisfaction we obtain that there exists q'_G such that $q_G \xrightarrow{\alpha_K}_G q'_G$. Furthermore, $(q_K, q_A) \xrightarrow{\alpha} (q'_K, q'_A)$ also implies that $q_A \xrightarrow{\alpha_A}_A q'_A$. Hence again from L1-satisfaction: $(q'_K, \mathcal{Q}'_A)\,\mathcal{R}\,q'_G$ where $\mathcal{Q}'_A$ is $\{q'_A \mid \exists q_A \in \mathcal{Q}_A \ s.t.\ q_A \xrightarrow{\alpha_A}_A q'_A\}$.

By definition of $\mathcal{R}'$, we may now conclude that $(\mathcal{Q}'_K, q'_A)\,\mathcal{R}'\,\mathcal{Q}'_G$.

C.5 Soundness of circular reasoning for $\sqsubseteq^{L1}$

Proof. Let K be a component on $\mathcal{P}$, $\mathcal{C} = (A, gl, G)$ a contract for $\mathcal{P}$ and E a component such that $\mathcal{P}_E = \mathcal{P}_A$. Suppose that $K \sqsubseteq^{L1}_{A,gl} G$ and $E \sqsubseteq^{L1}_{G,gl} A$. We have to prove that $K \sqsubseteq^{L1}_{E,gl} G$.

As $K \sqsubseteq^{L1}_{A,gl} G$ and $E \sqsubseteq^{L1}_{G,gl} A$, there exist two relations $\mathcal{R}_1$ and $\mathcal{R}_2$ on respectively $(Q_K \times 2^{Q_A}) \times Q_G$ and $(Q_E \times 2^{Q_G}) \times Q_A$ as in Lemma 6. We define $\mathcal{R} \subseteq (Q_K \times 2^{Q_E}) \times Q_G$ as follows: for any $q_K \in Q_K$, $\mathcal{Q}_E \subseteq Q_E$ and $q_G \in Q_G$, $(q_K, \mathcal{Q}_E)\,\mathcal{R}\,q_G$ iff there exists $\mathcal{Q}_A \subseteq Q_A$, $\mathcal{Q}_G \subseteq Q_G$, $q_E \in \mathcal{Q}_E$ and $q_A \in \mathcal{Q}_A$ such that $q_G \in \mathcal{Q}_G$, $(q_K, \mathcal{Q}_A)\,\mathcal{R}_1\,q_G$ and $(q_E, \mathcal{Q}_G)\,\mathcal{R}_2\,q_A$. We now have to prove that $\mathcal{R}$ ensures the conditions of Lemma 6. Obviously, $(q^0_K, \{q^0_E\})\,\mathcal{R}\,q^0_G$.

Suppose that $(q_K, \mathcal{Q}_E)\,\mathcal{R}\,q_G$. Then by definition $(q_K, \mathcal{Q}_A)\,\mathcal{R}_1\,q_G$ and $(q_E, \mathcal{Q}_G)\,\mathcal{R}_2\,q_A$ for some $\mathcal{Q}_A$, $\mathcal{Q}_G$ with $q_G \in \mathcal{Q}_G$, $q_E \in \mathcal{Q}_E$ and $q_A \in \mathcal{Q}_A$. Now suppose $q_K \xrightarrow{\alpha_K}_K q'_K$. We have to prove that there exists q'_G such that:

1. $q_G \xrightarrow{\alpha_K}_G q'_G$
2. if $\exists \alpha_E$, $q_E \in \mathcal{Q}_E$ and $q'_E \in Q_E$ s.t. $q_E \xrightarrow{\alpha_E}_E q'_E$ and $\alpha_K \cup \alpha_E \in \mathcal{I}(gl)$, then $(q'_K, \mathcal{Q}'_E) \mathcal{R} q'_G$ where $\mathcal{Q}'_E$ is $\{q'_E \mid \exists q_E \in \mathcal{Q}_E \; s.t. \; q_E \xrightarrow{\alpha_E}_E q'_E\}$.

As $(q_K, \mathcal{Q}_A) \mathcal{R}_1 q_G$ and $q_K \xrightarrow{\alpha_K}_K q'_K$, we know that there exists q'_G such that:

- $q_G \xrightarrow{\alpha_K}_G q'_G$
- if $\exists \alpha_A$, $q_A \in \mathcal{Q}_A$ and $q'_A \in Q_A$ s.t. $q_A \xrightarrow{\alpha_A}_A q'_A$ and $\alpha_K \cup \alpha_A \in \mathcal{I}(gl)$, then $(q'_K, \mathcal{Q}'_A) \mathcal{R} q'_G$ with $\mathcal{Q}'_A$ defined as $\{q'_A \mid \exists q_A \in \mathcal{Q}_A \; s.t. \; q_A \xrightarrow{\alpha_A}_A q'_A\}$.

We now show that this q'_G satisfies the two conditions required above from $\mathcal{R}$. Condition 1 is exactly the same as for $\mathcal{R}_1$, so let focus on the second condition.

Suppose that there exists α_E, $q_E \in \mathcal{Q}_E$ and $q'_E \in Q_E$ such that $q_E \xrightarrow{\alpha_E}_E q'_E$ and $\alpha_K \cup \alpha_E \in \mathcal{I}(gl)$. Let $\mathcal{Q}'_E = \{q'_E \mid \exists q_E \in \mathcal{Q}_E \; s.t. \; q_E \xrightarrow{\alpha_E}_E q'_E\}$ as defined above. We have to show that $(q'_K, \mathcal{Q}'_E) \mathcal{R} q'_G$. As $(q_E, \mathcal{Q}_G) \mathcal{R}_2 q_A$ and $q_E \xrightarrow{\alpha_E}_E q'_E$, we know that there exists q'_A such that:

- $q_A \xrightarrow{\alpha_E}_A q'_A$
- if $\exists \alpha_G$, $q_G \in \mathcal{Q}_G$ and $q'_G \in Q_G$ s.t. $q_G \xrightarrow{\alpha_G}_G q'_G$ and $\alpha_G \cup \alpha_E \in \mathcal{I}(gl)$, then $(q'_E, \mathcal{Q}'_G) \mathcal{R} q'_A$ with $\mathcal{Q}'_G$ defined as $\{q'_G \mid \exists q_G \in \mathcal{Q}_G \; s.t. \; q_G \xrightarrow{\alpha_G}_G q'_G\}$.

Thus, applying the second property offered by $\mathcal{R}_1$ to this α_E and q'_A, we obtain that $(q'_K, \mathcal{Q}'_A) \mathcal{R}_1 q'_G$ where $\mathcal{Q}'_A$ is defined as $\{q'_A \mid \exists q_A \in \mathcal{Q}_A \; s.t. \; q_A \xrightarrow{\alpha_E}_A q'_A\}$. Besides, as there exist indeed $q_G \in \mathcal{Q}_G$ and $q'_G \in Q_G$ such that $q_G \xrightarrow{\alpha_K}_G q'_G$, then applying the second property offered by $\mathcal{R}_2$, we obtain $(q'_E, \mathcal{Q}'_G) \mathcal{R}_2 q'_A$ for $\mathcal{Q}'_G$ defined as $\{q'_G \mid \exists q_G \in \mathcal{Q}_G \; s.t. \; q_G \xrightarrow{\alpha_K}_G q'_G\}$. Finally, according to the definition of $\mathcal{R}$, we can conclude that $(q'_K, \mathcal{Q}'_E) \mathcal{R} q'_G$.

C.6 Relaxed circular reasoning for $\sqsubseteq^{L1}$ and $\sqsubseteq^{L0}$

Proof. Let K be a component on an interface $\mathcal{P}$, $\mathcal{C} = (A, gl, G)$ a contract for $\mathcal{P}$ and E a component such that $\mathcal{P}_E = \mathcal{P}_A$. We suppose that $K \sqsubseteq^{L1}_{A,gl} G$ and $E \sqsubseteq^{L0}_{G,gl} A$, and then we prove that $K \sqsubseteq^{L1}_{E,gl} G$.

As $K \sqsubseteq^{L1}_{A,gl} G$, there exists a relation $\mathcal{R}_1$ on $(Q_K \times 2^{Q_A}) \times Q_G$ as in Lemma 6. As $E \sqsubseteq^{L0}_{G,gl} A$, there exists a relation $\mathcal{R}_2$ on $(2^{Q_E} \times Q_G) \times 2^{Q_A}$ as in Lemma 7.

We define $\mathcal{R} \subseteq (Q_K \times 2^{Q_E}) \times Q_G$ as follows: for any $q_K \in Q_K$, $\mathcal{Q}_E \subseteq Q_E$ and $q_G \in Q_G$, we define $(q_K, \mathcal{Q}_E) \mathcal{R} q_G$ iff there exists $\mathcal{Q}_A \subseteq Q_A$ such that $(q_K, \mathcal{Q}_A) \mathcal{R}_1 q_G$ and $(\mathcal{Q}_E, q_G) \mathcal{R}_2 \mathcal{Q}_A$. We have to prove that $\mathcal{R}$ ensures the conditions of Lemma 6. Obviously, $(q^0_K, \{q^0_E\}) \mathcal{R} q^0_G$.

Let $q_K \in Q_K, \mathcal{Q}_E \subseteq Q_E, q_G \in Q_G$ be such that $(q_K, \mathcal{Q}_E) \mathcal{R} q_G$. Let $\mathcal{Q}_A$ be such that $(q_K, \mathcal{Q}_A) \mathcal{R}_1 q_G$ and $(\mathcal{Q}_E, q_G) \mathcal{R}_2 \mathcal{Q}_A$. Now suppose $q_K \xrightarrow{\alpha_K}_K q'_K$. We have to prove that there exists p'_G such that:

1. $q_G \xrightarrow{\alpha_K}_G q'_G$

2. if $\exists \alpha_E$, $q_E \in \mathcal{Q}_E$ and $q'_E \in Q_E$ s.t. $q_E \xrightarrow{\alpha_E}_E q'_E$ and $\alpha_K \cup \alpha_E \in \mathcal{I}(gl)$, then $(q'_K, \mathcal{Q}'_E)\,\mathcal{R}\,q'_G$ where $\mathcal{Q}'_E$ is $\{q'_E \mid \exists q_E \in \mathcal{Q}_E \; s.t. \; q_E \xrightarrow{\alpha_E}_E q'_E\}$.

As $(q_K, \mathcal{Q}_A)\,\mathcal{R}_1\,q_G$ and $q_K \xrightarrow{\alpha_K}_K q'_K$, we know that there exists q'_G such that:

- $q_G \xrightarrow{\alpha_K}_G q'_G$
- if $\exists \alpha_A$, $q_A \in \mathcal{Q}_A$ and $q'_A \in Q_A$ s.t. $q_A \xrightarrow{\alpha_A}_A q'_A$ and $\alpha_K \cup \alpha_A \in \mathcal{I}(gl)$, then $(q'_K, \mathcal{Q}'_A)\,\mathcal{R}\,q'_G$ with $\mathcal{Q}'_A$ defined as $\{q'_A \mid \exists q_A \in \mathcal{Q}_A \; s.t. \; q_A \xrightarrow{\alpha_A}_A q'_A\}$.

We show that this q'_G satisfies the two conditions required from $\mathcal{R}$. Condition 1. is exactly the same as for $\mathcal{R}_1$. Let us show that the second condition holds.

Suppose that there exist α_E, $q_E \in \mathcal{Q}_E$ and $q'_E \in Q_E$ such that $q_E \xrightarrow{\alpha_E}_E q'_E$ and $\alpha_K \cup \alpha_E \in \mathcal{I}(gl)$. Let $\mathcal{Q}'_E = \{q'_E \mid \exists q_E \in \mathcal{Q}_E \; s.t. \; q_E \xrightarrow{\alpha_E}_E q'_E\}$ as above. We have to show that $(q'_K, \mathcal{Q}'_E)\,\mathcal{R}\,q'_G$. As $q_E \xrightarrow{\alpha_E}_E q'_E$ and $q_G \xrightarrow{\alpha_K}_G q'_G$, we know that $(q_E, q_G) \xrightarrow{\alpha} (q'_E, q'_G)$. Thus, from $(\mathcal{Q}_E, q_G)\,\mathcal{R}_2\,Q_A$ and because $q_E \in \mathcal{Q}_E$, we can conclude that there exist $q_A \in \mathcal{Q}_A$ and $q'_A \in Q_A$ such that $q_A \xrightarrow{\alpha_E} q'_A$ and $(\mathcal{Q}'_E, q'_G)\,\mathcal{R}_2\,\mathcal{Q}'_A$ for $\mathcal{Q}'_E$ as above and $\mathcal{Q}'_A = \{q'_A \mid \exists q_A \in \mathcal{Q}_A \; s.t. \; q_A \xrightarrow{\alpha_E}_A q'_A\}$.

Now, applying the second property offered by $\mathcal{R}_1$ to this α_E, q_A and q'_A, we obtain that $(q'_K, \mathcal{Q}'_A)\,\mathcal{R}_1\,q'_G$ with $\mathcal{Q}_A$ as defined above. Finally, according to the definition of $\mathcal{R}$, we can conclude that $(q'_K, \mathcal{Q}'_E)\,\mathcal{R}\,q'_G$.

C.7 Relaxed circular reasoning for $\sqsubseteq^{L0}$ and $\sqsubseteq^{L1}$

Proof. Let K be a component on an interface $\mathcal{P}$, $\mathcal{C} = (A, gl, G)$ a contract for $\mathcal{P}$ and E a component such that $\mathcal{P}_E = \mathcal{P}_A$. We suppose that $K \sqsubseteq^{L0}_{A,gl} G$ and $E \sqsubseteq^{L1}_{G,gl} A$, and then we prove that $K \sqsubseteq^{L0}_{E,gl} G$.

As $K \sqsubseteq^{L0}_{A,gl} G$, there exists a relation $\mathcal{R}_1$ on $(2^{Q_K} \times Q_A) \times 2^{Q_G}$ as in Lemma 7. As $E \sqsubseteq^{L1}_{G,gl} A$, there exists a relation $\mathcal{R}_2$ on $(Q_E \times 2^{Q_G}) \times Q_A$ as in Lemma 6.

We define $\mathcal{R} \subseteq (2^{Q_K} \times Q_E) \times 2^{Q_G}$ as follows: for any $\mathcal{Q}_K \subseteq Q_K$, $q_E \in Q_E$ and $\mathcal{Q}_G \subseteq Q_G$, we define $(\mathcal{Q}_K, q_E)\,\mathcal{R}\,\mathcal{Q}_G$ iff there exists $q_A \in Q_A$ such that $(\mathcal{Q}_K, q_A)\,\mathcal{R}_1\,\mathcal{Q}_G$ and $(q_E, \mathcal{Q}_G)\,\mathcal{R}_2\,q_A$. We have to prove that $\mathcal{R}$ ensures the conditions of Lemma 7. Obviously, $(\{q^0_K\}, q^0_E)\,\mathcal{R}\,\{q^0_G\}$.

Let $\mathcal{Q}_K \subseteq Q_K$, $q_E \in Q_E$ and $\mathcal{Q}_G \subseteq Q_G$ be such that $(\mathcal{Q}_K, q_E)\,\mathcal{R}\,\mathcal{Q}_G$. Let q_A be such that $(\mathcal{Q}_K, q_A)\,\mathcal{R}_1\,\mathcal{Q}_G$ and $(q_E, \mathcal{Q}_G)\,\mathcal{R}_2\,q_A$. Suppose $(q_K, q_E) \xrightarrow{\alpha} (q'_K, q'_E)$ for $q_K \in \mathcal{Q}_K$ and $\alpha = \alpha_K \cup \alpha_E \in \mathcal{I}(gl)$. Define $\mathcal{Q}'_K$ and $\mathcal{Q}'_G$ as follows: $\mathcal{Q}'_K = \{q'_K \mid \exists q_K \in \mathcal{Q}_K \, s.t. \, q_K \xrightarrow{\alpha_K}_K q'_K\}$ and $\mathcal{Q}'_G$ as $\{q'_G \mid \exists q_G \in \mathcal{Q}_G \; s.t. \; q_G \xrightarrow{\alpha_K}_G q'_G\}$. We have to prove that there exists $q_G \in \mathcal{Q}_G$ and $q'_G \in Q_G$ such that $q_G \xrightarrow{\alpha_K} q'_G$ and $(\mathcal{Q}'_K, q'_E)\,\mathcal{R}\,\mathcal{Q}'_G$.

As $(q_K, q_E) \xrightarrow{\alpha} (q'_K, q'_E)$, we know that $q_E \xrightarrow{\alpha_E}_E q'_E$. Then, because $(q_E, \mathcal{Q}_G)\,\mathcal{R}_2\,q_A$ and $q_E \xrightarrow{\alpha_E}_E q'_E$, there exists q'_A such that:

- $q_A \xrightarrow{\alpha_E}_A q'_A$

– if $\exists \alpha_G$, $q_G \in \mathcal{Q}_G$ and $q'_G \in Q_G$ s.t. $q_G \xrightarrow{\alpha_G}_G q'_G$ and $\alpha_G \cup \alpha_E \in \mathcal{I}(gl)$, then $(q'_E, \mathcal{Q}'_G)\,\mathcal{R}\,q'_A$ with $\mathcal{Q}'_G$ defined as above.

This implies in particular that $(q_K, q_A) \xrightarrow{\alpha} (q'_K, q'_A)$. Thus, from $(q_K, \mathcal{Q}_A)\,\mathcal{R}_1\,q_G$ and because $q_A \in \mathcal{Q}_A$, we can conclude that there exists $q_G \in \mathcal{Q}_G$ and $q'_G \in Q_G$ such that $q_G \xrightarrow{\alpha_K} q'_G$ and $(\mathcal{Q}'_K, q'_A)\,\mathcal{R}_1\,\mathcal{Q}'_G$ for $\mathcal{Q}'_K$ and $\mathcal{Q}'_G$ defined as above. This gives us the q_G and q'_G we were looking for. There only remains to prove that $(\mathcal{Q}'_K, q'_E)\,\mathcal{R}\,\mathcal{Q}'_G$. Now, applying the second property offered by $\mathcal{R}_2$ to this α_K, q_G and q'_G, we obtain that $(q'_E, \mathcal{Q}'_G)\,\mathcal{R}_2\,q'_A$. Hence, according to the definition of $\mathcal{R}$, the conclusion that $(q'_K, \mathcal{Q}'_E)\,\mathcal{R}\,q'_G$.

D Dealing with hierarchical components in L1

D.1 A compositional semantics for L1 without encapsulation

So far, we have supposed that all components were atomic in order to simplify the presentation of the proofs. However, Section 3 defines the semantics of a component $gl\{K, A\}$ as the semantics of its flattened form. Therefore we provide now a theorem that allows reusing all proofs of the previous section in the context of hierarchical components. It uses the fact that glues are defined on sets of ports rather than sets of interfaces, which allows us to replace in a composition a set of components $\mathcal{K}^1$ with a single component.

Theorem 6. $[\![gl\{gl'\{\mathcal{K}^1\}, \mathcal{K}^2\}]\!] = [\![(gl \circ gl')\{[\![gl'\{\mathcal{K}^1\}]\!], \mathcal{K}^2\}]\!]$.

Proof. By definition we have $[\![gl\{gl'\{\mathcal{K}^1\}, \mathcal{K}^2\}]\!] = [\![(gl \circ gl')\{\mathcal{K}^1 \cup \mathcal{K}^2\}]\!]$. Denote $gl_f = gl \circ gl'$ the glue used for flattening the system. For readability and w.l.o.g. we suppose that $\mathcal{K}^1 = \{K_1, \ldots, K_m\}$ and $\mathcal{K}^2 = \{K_{m+1}, \ldots, K_n\}$ for $0 < m < n$.

We consider two states $q^1 = (q^1_1, \ldots, q^1_n)$ and $q^2 = (q^2_1, \ldots, q^2_n)$ of $[\![gl_f\{\mathcal{K}^1 \cup \mathcal{K}^2\}]\!]$. Again by definition, $q^1 \xrightarrow{\alpha} q^2$ for $\alpha \in \mathcal{I}(gl_f)$ if and only if $\forall i : q^1_i \xrightarrow{\alpha_i}_i q^2_i$, where $\alpha_i = \alpha \cap \mathcal{P}_i$.

Consider now two states $q^1 = (\mathcal{Q}^1_h, q^1_{m+1}, \ldots, q^1_n)$ and $q^2 = (\mathcal{Q}^2_h, q^2_{m+1}, \ldots, q^2_n)$ of $[\![gl_f\{[\![gl'\{\mathcal{K}^1\}]\!], \mathcal{K}^2\}]\!]$ where $\mathcal{Q}^1_h$ and $\mathcal{Q}^2_h$ are states of the hierarchical component $[\![gl'\{\mathcal{K}^1\}]\!]$, i.e. $\mathcal{Q}^1_h$ is of the form $(q^1_1, \ldots, q^1_m)$ and $\mathcal{Q}^2_h = (q^2_1, \ldots, q^2_m)$. Here we have $q^1 \xrightarrow{\alpha} q^2$ for $\alpha \in \mathcal{I}(gl_f)$ if and only if $\mathcal{Q}^1_h \xrightarrow{\alpha_h}_h \mathcal{Q}^2_h$ and $\forall i > m : q^1_i \xrightarrow{\alpha_i}_i q^2_i$, with $\alpha_i = \alpha \cap \mathcal{P}_i$, $\alpha_h = \alpha \cap \mathcal{I}(gl')$ and $\longrightarrow_h$ the transition relation of $[\![gl'\{\mathcal{K}^1\}]\!]$. Now we also have that $\mathcal{Q}^1_h \xrightarrow{\alpha_h}_h \mathcal{Q}^2_h$ if and only if $\forall i \leq m : q^1_i \xrightarrow{\alpha_i}_i q^2_i$, where $\alpha_i = \alpha_h \cap \mathcal{P}_i$. Thus $\alpha_i = \alpha \cap \mathcal{I}(gl') \cap \mathcal{P}_i$. As $\alpha \in \mathcal{I}(gl \circ gl')$ it holds according to the definition of $\circ$ that $\alpha \cap \mathcal{P}_i \subseteq \mathcal{I}(gl')$. Hence the result. □

Based on this theorem we obtain that if K is a flat component $K = gl'\{\mathcal{K}^1\}$ then $K \models^{L1} (A, gl, G)$ if and only if $[\![\mathcal{K}^1]\!] \models^{L1} (A, gl \circ gl', [\![G]\!])$. By using this inductively, we can extend all results from the previous section to (possibly hierarchical) composite components.

D.2 A compositional semantics for L1 with encapsulation

As already mentioned, the closed semantics that was proposed for L1 is not very satisfying because it does not take encapsulation into account. Indeed, K may not refine G in the context of A and gl even though K and G show the same behavior at their interface, because K and G may not have the same internal actions. To tackle this issue, we propose here another (looser) definition of closed semantics that does not rely on flattening of components. It is based on a *compositional* semantics which provides for each (possibly hierarchical) component an atomic component that behaves in the same way when part of a larger system. In other words, we define a semantics that is consistent with satisfaction, meaning that a component and its compositional semantics satisfy exactly the same contracts. This allows us here too to reuse the proofs of the previous section, as composite components can be replaced by their compositional semantics. Note that the compositional semantics is *black box*, i.e., it refers to exported ports.

In contrast, the semantics that was presented in Section 3 as well as the new one defined in this section are used for expressing the behavior of a closed system, that is, a system that has no interaction with an environment. These two closed semantics are white-box in the sense that they reflect the inner interactions taking place and not only the corresponding exported ports at the interface of the system. The difference between the closed and the compositional semantics that we define below are illustrated in Figure 5, where a simple composite component is given with its two corresponding semantics, compositional (top-right) and closed (down-right).

Note also that the equivalence induced by the compositional semantics relates *open* systems, thus being a refinement in any context, while the equivalence required by our definition of component framework relates *closed* systems.

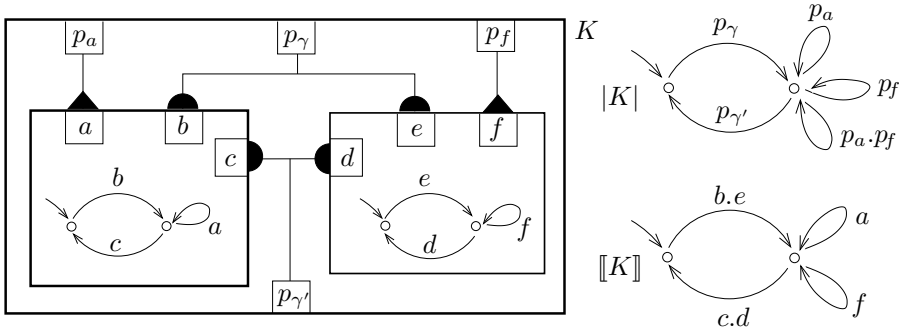

Figure5. A composite component (left) and its semantics (right)

Consider now a composite component K defined by a set of components $\{K_i\}_{i=1}^n$ and a glue gl with support set $S_{gl} = \bigcup_{i=1}^n \mathcal{P}_i$. For defining our compositional semantics, we need to define what a multi-shot interaction is, and in

fact we need two such notions, one for representing black-box interactions and one for white-box interactions.

Definition 17. *Given a glue gl, a* black-box multi-shot *interaction is of the form $\{p_{\gamma_1}, \ldots, p_{\gamma_k}\}$, where all connectors γ_i have pairwise disjoint support sets. Each such interaction $m = \{p_{\gamma_1}, \ldots, p_{\gamma_k}\}$ is associated with a set of* white-box multi-shot *interactions denoted $wb(m)$ and defined as:*

$$wb(m) = \{\alpha_1 \cup \ldots \cup \alpha_k \mid \forall i \in [1,k],\, \alpha_i \in \mathcal{I}(\gamma_i)\}$$

The set of black-box, respectively white-box, multi-shot interactions of gl is denoted $\mathcal{M}_{bb}(gl)$, respectively $\mathcal{M}_{wb}(gl)$. Multi-shot interactions allow concurrency, as interactions from non-conflicting connectors may be fired simultaneously (unless stated otherwise by the components' behaviors).

Definition 18 (Compositional semantics). *The* compositional semantics *of K is denoted $|K|$ and is defined as $(Q, q^0, \mathcal{M}_{bb}(gl), \rightsquigarrow)$, where $Q = \prod_{i=1}^{n} Q_i$, $q^0 = (q_1^0, \ldots, q_n^0)$ and given two states $q^1 = (q_1^1, \ldots, q_n^1)$ and $q^2 = (q_1^2, \ldots, q_n^2)$ in Q and a multi-shot interaction $m \in \mathcal{M}_{bb}(gl)$, $q^1 \overset{m}{\rightsquigarrow} q^2$ iff there exists $\alpha \in wb(m)$ such that $\forall i, q_i^1 \overset{\alpha_i}{\rightsquigarrow}_i q_i^2$, where $\alpha_i = \alpha \cap \mathcal{P}_i$.*

Here too we use the convention that $\forall q,\, q \xrightarrow{\emptyset} q$ so that components not involved in the interaction do not move. The closed semantics only differs from the compositional semantics in that it is white-box, so labels are interactions in $\mathcal{I}(gl)$ rather than ports in $\mathcal{P}_{gl}$. Furthermore, it differs from the one defined in Section 3 in that it is based on the compositional semantics provided here rather than on a flattening of the system.

Definition 19 (Closed semantics). *The* closed semantics *of K is denoted $[\![K]\!]$ and is defined as $(Q, q^0, \mathcal{I}(gl), \longrightarrow)$, where $Q = \prod_{i=1}^{n} Q_i$, $q^0 = (q_1^0, \ldots, q_n^0)$ and $\longrightarrow$ is defined as follows. Given two states $q^1 = (q_1^1, \ldots, q_n^1)$ and $q^2 = (q_1^2, \ldots, q_n^2)$ in Q and an interaction $\alpha \in \mathcal{I}(gl)$, $q^1 \xrightarrow{\alpha} q^2$ if and only if $\forall i, q_i^1 \overset{\alpha_i}{\rightsquigarrow}_i q_i^2$, where $\alpha_i = \alpha \cap \mathcal{P}_i$.*

Let us now state a theorem that relates the semantics of a composite component and its flattened form. We need this theorem for redefining the L1 component framework.

Theorem 7. *$[\![gl\{gl'\{\mathcal{K}^1\}, \mathcal{K}^2\}]\!]$ and $[\![(gl \circ gl')\{\mathcal{K}^1 \cup \mathcal{K}^2\}]\!]$ are equivalent in the following sense: there exists a renaming of labels in $(gl \circ gl')\{\mathcal{K}^1 \cup \mathcal{K}^2\}$ as below such that they are bisimilar.*

For simplifying notation, we suppose that $\mathcal{K}^1 = \{K_1, K_2\}$ and $\mathcal{K}^2 = \{K_3\}$ and we denote B_h the behavior of the hierarchical component, i.e., $[\![gl\{gl'\{K_1, K_2\}, K_3\}]\!]$ and B_f the behavior of its flattened form $[\![(gl \circ gl')\{K_1, K_2, K_3\}]\!]$. The transition relation of $|gl'\{K_1, K_2\}|$ is denoted $\rightsquigarrow_{1,2}$.

Formally, the renaming of B_f is an LTS $B_{f'} = (Q_f, q_f^0, \mathcal{I}(gl), \longrightarrow_{f'})$, that is, only the set of labels and the transition relation are modified: the renaming consists in replacing every interaction in $\mathcal{I}(gl')$ by its corresponding exported port.

However, due to the fact that an interaction may be part of several connectors, possibly through hierarchical connectors, some problems may arise in this renaming. Figure 6 illustrates this: depending on the context, $\alpha_f = \{a, c\}$ of the flattened component may be renamed into either $\alpha_h = \{p_a, p_c\}$ or $\alpha'_h = \{p_r\}$. The new transition relation is defined as follows: $(q_1, q_2, q_3) \xrightarrow{\alpha_h}_{f'} (q_1, q_2, q_3)$ iff:

1. $(q_1, q_2, q_3) \xrightarrow{\alpha_f}_{f} (q_1, q_2, q_3)$; decompose α_f into $\alpha_f = \alpha_1 \cup \alpha_2 \cup \alpha_3$ with $\alpha_i \in \mathcal{P}_i$
2. α_h can be decomposed into $m \cup \alpha_3$ s.t. $m \in \mathcal{M}_{bb}(gl')$ and $\alpha_1 \cup \alpha_2 \in wb(m)$

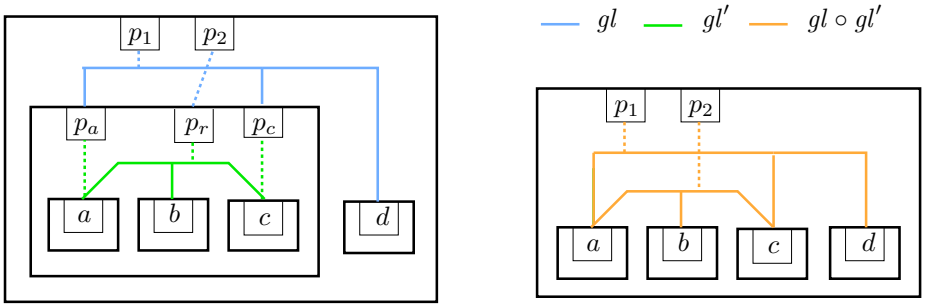

Figure6. Interaction $\{a, c\}$ may be part of several hierarchical connectors

Proof. We define $\mathcal{R} \subseteq ((Q_1 \times Q_2) \times Q_3) \times (Q_1 \times Q_2 \times Q_3)$ as:

$$((q_1, q_2), q_3)\, \mathcal{R}\, (q'_1, q'_2, q'_3) \triangleq q_1 = q'_1 \wedge q_2 = q'_2 \wedge q_3 = q'_3$$

Let us show that $\mathcal{R}$ is a bisimulation. The initial states are trivially related.

Suppose that $((q_1, q_2), q_3) \xrightarrow{\alpha_h}_{h} ((q'_1, q'_2), q'_3)$. Let us show that $(q_1, q_2, q_3) \xrightarrow{\alpha_h}_{f'} (q'_1, q'_2, q'_3)$. According to the definition of closed semantics, α_h can be decomposed as $\alpha_h = m \cup \alpha_3$ where $m \in \mathcal{M}_{bb}(gl')$, $\alpha_3 \in \mathcal{P}_3$, and furthermore $(q_1, q_2) \overset{m}{\rightsquigarrow}_{1,2} (q'_1, q'_2)$ and $q_3 \overset{\alpha_3}{\rightsquigarrow}_3 q'_3$. This in turn implies, according to the definition of compositional semantics, that there exists $\alpha_1 \in \mathcal{P}_1$ and $\alpha_2 \in \mathcal{P}_2$ such that $\alpha_1 \cup \alpha_2 \in wb(m)$, $q_1 \overset{\alpha_1}{\rightsquigarrow}_1 q'_1$ and $q_2 \overset{\alpha_2}{\rightsquigarrow}_2 q'_2$.

As $q_i \overset{\alpha_i}{\rightsquigarrow}_i q'_i$ for $i \in [1, 3]$, we have $(q_1, q_2, q_3) \xrightarrow{\alpha_f}_{f} (q'_1, q'_2, q'_3)$ for $\alpha_f = \alpha_1 \cup \alpha_2 \cup \alpha_3$. Thus, according to the definition of renaming and of α_f, this implies that $(q_1, q_2, q_3) \xrightarrow{\alpha_h}_{f'} (q'_1, q'_2, q'_3)$.

Symmetrically, suppose that $(q_1, q_2, q_3) \xrightarrow{\alpha_h}_{f'} (q'_1, q'_2, q'_3)$. By definition of renaming, this implies that α_h can be decomposed into $m \cup \alpha_3$ with $m \in \mathcal{M}_{bb}(gl')$ and $\alpha_3 \in \mathcal{P}_3$ and furthermore that there exists $\alpha_1 \in \mathcal{P}_1$ and $\alpha_2 \in \mathcal{P}_2$ such that $\alpha_1 \cup \alpha_2 \in wb(m)$ and $(q_1, q_2, q_3) \xrightarrow{\alpha_f}_{f} (q'_1, q'_2, q'_3)$ for $\alpha_f = \alpha_1 \cup \alpha_2 \cup \alpha_3$, which implies that $q_i \overset{\alpha_i}{\rightsquigarrow}_i q'_i$ for $i \in [1, 3]$. We obtain from this that $(q_1, q_2) \overset{m}{\rightsquigarrow}_{1,2} (q'_1, q'_2)$

and then $((q_1, q_2), q_3) \xrightarrow{\alpha_h}_h ((q'_1, q'_2), q'_3)$. □

Finally, let us show that a component and its compositional semantics satisfy exactly the same contracts.

Theorem 8. *$K \models^{L1} (A, gl, G)$ if and only if $|K| \models^{L1} (A, gl, G)$.*

Proof. L1-satisfaction is based on $[\![gl\{K, A\}]\!]$ which is defined based on $|K|$ and $|A|$. Thus it does not matter whether K or $|K|$ is considered. □

E Preservation of satisfaction by composition

Preservation of satisfaction by composition still has to be proven for L0 and L1. We have to prove that for any E, gl such that $S_{gl} = \mathcal{P}_E \cup \mathcal{P}$ for some $\mathcal{P}$ such that $\mathcal{P} \cap \mathcal{P}_E = \emptyset$ and gl_E, E_1, E_2 such that $E = gl_E\{E_1, E_2\}$, the following holds for any I, S on $\mathcal{P}$:

$$I \sqsubseteq_{E,gl} S \implies gl_1\{I, E_1\} \sqsubseteq_{E_2,gl_2} gl_1\{S, E_1\}$$

where gl_1 and gl_2 are such that $gl \circ gl_E = gl_2 \circ gl_1$.

E.1 Preservation of satisfaction by composition for L0

In the case of L0, this property is trivial as:

$$I \sqsubseteq^{L0}_{E,gl} S \triangleq Tr(gl\{I, E\}) \subseteq Tr(gl\{S, E\})$$

$$\begin{aligned} gl_1\{I, E_1\} \sqsubseteq^{L0}_{E_2,gl_2} gl_1\{S, E_1\} &\triangleq Tr(gl_2\{gl_1\{I, E_1\}, E_2\}) \subseteq Tr(gl_2\{gl_1\{S, E_1\}, E_2\}) \\ &= Tr((gl \circ gl_E)\{I, E_1, E_2\}) \subseteq Tr((gl \circ gl_E)\{S, E_1, E_2\}) \\ &= Tr(gl\{I, E\}) \subseteq Tr(gl\{S, E\}) \end{aligned}$$

So there is even an equivalence between the two parts of the implication.

E.2 Preservation of satisfaction by composition for L1

Proof. Let us suppose that $I \sqsubseteq^{L1}_{E,gl} S$ with all terms defined as above. Let gl_1 and gl_2 be such that $gl \circ gl_E = gl_2 \circ gl_1$. What we have to show now is that $gl_1\{I, E_1\} \sqsubseteq^{L1}_{E_2,gl_2} gl_1\{S, E_1\}$.

There exists a relation $\mathcal{R} \subseteq (Q_I \times 2^{Q_E}) \times Q_S$ as in Lemma 6. In the following we denote $I1 = gl_1\{I, E_1\}$ and $S1 = gl_1\{S, E_1\}$. We define $\mathcal{R}' \subseteq (Q_{I1} \times 2^{Q_2}) \times Q_{S1}$ by: $(q_{I1}, \mathcal{Q}_2)\, \mathcal{R}'\, q_{S1}$ iff $(q_I, \mathcal{Q}_E)\, \mathcal{R}\, q_S$ where $q_{I1} = (q_I, q_1)$, $q_{S1} = (q_S, q_1)$ and $\mathcal{Q}_E$ contains $q_E = (q_1, q_2)$ such that $q_2 \in \mathcal{Q}_2$.

We have to show that $\mathcal{R}'$ also satisfies the conditions of Lemma 6, i.e.:

- $(q^0_{I1}, \{q^0_2\})\, \mathcal{R}'\, q^0_{S1}$
- if $(q_{I1}, \mathcal{Q}_2)\, \mathcal{R}'\, q_{S1}$ and $q_{I1} \xrightarrow{\alpha_{I1}}_{I1} q'_{I1}$, then there exists q'_{S1} such that:
  1. $q_{S1} \xrightarrow{\alpha_{I1}}_{S1} q'_{S1}$

2. if there exists $q_2 \in \mathcal{Q}_2$, $q'_2 \in Q_2$ and α_2 s.t. $q_2 \xrightarrow{\alpha_2}_2 q'_2$, then $(q'_{I1}, \mathcal{Q}'_2)\, \mathcal{R}'\, q'_{S1}$ where $\mathcal{Q}'_2$ is $\{q'_2 \mid \exists q_2 \in \mathcal{Q}_2 \; s.t. \; q_2 \xrightarrow{\alpha_2}_2 q'_2\}$.

The condition on the initial states is once more quite obvious. Item 1 of the second condition is also easily discharged: $q_{I1} \xrightarrow{\alpha_{I1}}_{I1} q'_{I1}$ implies that $q_I \xrightarrow{\alpha_I}_I q'_I$ and $q_1 \xrightarrow{\alpha_1}_1 q'_1$ where $\alpha_{I1} = \alpha_I \cup \alpha_1$. This, according to the definition of $\mathcal{R}$, implies that $q_S \xrightarrow{\alpha_I}_S q'_S$. Hence the result.

Finally let us suppose that there exists $q_2 \in \mathcal{Q}_2$, $q'_2 \in Q_2$ and α_2 s.t. $q_2 \xrightarrow{\alpha_2}_2 q'_2$. Then $(q_1, q_2) \xrightarrow{\alpha_E}_E (q'_1, q'_2)$ where $\alpha_E = \alpha_1 \cup \alpha_2$. Based on this, $\mathcal{R}$ ensures that $(q'_I, \mathcal{Q}'_E)\, \mathcal{R}\, q'_S$ where $\mathcal{Q}'_E$ is $\{q'_E \mid \exists q_E \in \mathcal{Q}_E \; s.t. \; q_E \xrightarrow{\alpha_E}_E q'_E\}$. This implies condition 2 by direct application of the definition of $\mathcal{R}'$. □