ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Offer Semantics: Achieving Compositionality, Flattening and Full Expressiveness for the Glue Operators in BIP

EPFL IC IIF RiSD Technical Report
№ EPFL-REPORT-203507
http://infoscience.epfl.ch/record/203507

Eduard Baranov and Simon Bliudze

November 21, 2014

Abstract: Based on a concise but comprehensive overview of some fundamental properties required from component-based frameworks, namely *compositionality*, *incrementality*, *flattening*, *modularity* and *expressiveness*, we review three modifications of the semantics of glue operators in the Behaviour-Interaction-Priority (BIP) framework. We provide theoretical results and examples illustrating the degree, to which the three semantics meet these requirements. In particular, we show that the latest semantics, based on the offer predicate is the only one that satisfies all of them.

The classical and offer semantics are not comparable: there are systems that can be assembled in the classical semantics, but not in the offer one. We present a strict characterisation of the behaviour hierarchy determining the conditions, under which systems in the classical semantics can be transposed into the offer semantics directly, with minor modifications, by introducing a new type of synchronisation or not at all.

The offer semantics allows us to extend the algebras, which are used to model glue operators in BIP, to encompass priorities. This extension uses the Algebra of Causal Interaction Trees, $\mathcal{T}(P)$, as a pivot: existing transformations automatically provide the extensions for the Algebra of Connectors. We then extend the axiomatisation of $\mathcal{T}(P)$, since the equivalence induced by the new operational semantics is weaker than that induced by the interaction semantics. This extension leads to canonical normal forms for all structures and to a simplification of the algorithm for the synthesis of connectors from Boolean coordination constraints.

```
@TechReport{BarBliu14-Offer-TR,
  author      = {Baranov, Eduard and
                 Bliudze, Simon},
  title       = {Offer Semantics: Achieving Compositionality, Flattening
                       and Full Expressiveness for the Glue Operators in {BIP}},
  institution = {EPFL IC IIF RiSD},
  month       = nov,
  year        = 2014,
  number      = {EPFL-REPORT-203507},
  note        = {Available at: \texttt{http://infoscience.epfl.ch/record/203507}}
}
```

Offer Semantics: Achieving Compositionality, Flattening and Full Expressiveness for the Glue Operators in BIP

Eduard Baranov * Simon Bliudze *

Abstract

Based on a concise but comprehensive overview of some fundamental properties required from component-based frameworks, namely *compositionality*, *incrementality*, *flattening*, *modularity* and *expressiveness*, we review three modifications of the semantics of glue operators in the Behaviour-Interaction-Priority (BIP) framework. We provide theoretical results and examples illustrating the degree, to which the three semantics meet these requirements. In particular, we show that the latest semantics, based on the offer predicate is the only one that satisfies all of them.

The classical and offer semantics are not comparable: there are systems that can be assembled in the classical semantics, but not in the offer one. We present a strict characterisation of the behaviour hierarchy determining the conditions, under which systems in the classical semantics can be transposed into the offer semantics directly, with minor modifications, by introducing a new type of synchronisation or not at all.

The offer semantics allows us to extend the algebras, which are used to model glue operators in BIP, to encompass priorities. This extension uses the Algebra of Causal Interaction Trees, $\mathcal{T}(P)$, as a pivot: existing transformations automatically provide the extensions for the Algebra of Connectors. We then extend the axiomatisation of $\mathcal{T}(P)$, since the equivalence induced by the new operational semantics is weaker than that induced by the interaction semantics. This extension leads to canonical normal forms for all structures and to a simplification of the algorithm for the synthesis of connectors from Boolean coordination constraints.

1 Introduction

Fundamentally, each component-based design framework can be viewed as a triple $(\mathcal{A}, \sigma, \simeq)$. Here, $\mathcal{A}$ is an algebraic structure generated by a *behaviour type* $\mathcal{B}$ [6] and a set $\mathcal{G}$ of *glue operators*:

$$\mathcal{A} ::= B \mid f(C_1, \ldots, C_n)\,, \qquad \text{with } B \in \mathcal{B},\ C_1, \ldots, C_n \in \mathcal{A} \text{ and } f \in \mathcal{G}. \tag{1}$$

We call the elements of $\mathcal{A}$ *systems* and the elements of $\mathcal{B}$ *behaviours*. The structure $\mathcal{A}$ represents the set of all systems constructible within the framework. Behaviour type $\mathcal{B}$ defines the *nature* of the components manipulated by the framework.

The notion "behaviour type" can cover a very large spectrum, ranging from programs and labelled transition systems, through OSGi bundles and browser plug-ins, to systems of differential equations etc. Behaviour types can be organised in type systems and studied separately, as, for example, in the co-algebra theory [35]. However, this notion should be distinguished, for instance, from classes in object-oriented programming or session types for communication protocols [26]. Although, in principle, component-based frameworks can be heterogeneous, e.g. Ptolemy II [23], that is rely on several distinct behaviour types for the design process, those aimed at the design

*École Polytechnique Fédérale de Lausanne, Station 14, 1015 Lausanne, Switzerland; firstname.lastname@epfl.ch

of executable systems must have an underlying unifying behaviour type allowing to study and manipulate the system as a whole.

The second element of the triple defining a component-based framework is the *semantic mapping* $\sigma : \mathcal{A} \to \mathcal{B}$, which assigns to each system its meaning in terms of the behaviour type $\mathcal{B}$. The semantic mapping must be consistent in the following sense:

$$\textit{for all } B \in \mathcal{B}\,, \qquad\qquad \sigma(B) = B\,. \tag{2}$$

A trivial consequence of (2) is that application of σ is idempotent, i.e. $\sigma\big(\sigma(C)\big) = \sigma(C)$, for all $C \in \mathcal{A}$. The semantic mapping is called *structural*, if it is defined by associating to each n-ary glue operator $f : \mathcal{A}^n \to \mathcal{A}$ a corresponding operator $\tilde{f} : \mathcal{B}^n \to \mathcal{B}$ and putting

$$\sigma\big(f(C_1, \ldots, C_n)\big) \stackrel{def}{=} \tilde{f}\big(\sigma(C_1), \ldots, \sigma(C_n)\big)\,, \qquad \text{for all } C_1, \ldots, C_n \in \mathcal{A} \text{ and } f \in \mathcal{G}. \tag{3}$$

Finally, $\simeq \subseteq \mathcal{A} \times \mathcal{A}$ is an *equivalence relation*, which allows comparing systems in terms, for example, of their functionality, observable behaviour or capability of interaction with the environment.[1] The equivalence relation must respect the semantics:

$$\textit{for all } C_1, C_2 \in \mathcal{A}\,, \qquad\qquad \sigma(C_1) = \sigma(C_2) \implies C_1 \simeq C_2\,. \tag{4}$$

Again, a trivial consequence of (2) and (4) is that a system is always equivalent to its semantics: $C \simeq \sigma(C)$, for all $C \in \mathcal{A}$. In the remainder of this section, we assume that (2) and (4) do hold.

Glue operators used to compose systems in a component-based design framework must possess the following properties [36].

Incrementality This property represents a generalised form of associativity. It requires that it be possible to view the sub-systems of a system in separation:

$$\textit{for all } i \in [1,n],\ C_1, C_2, \ldots, C_n \in \mathcal{A} \textit{ and } f \in \mathcal{G}\textit{, there exist } g, h \in \mathcal{G}\textit{, such that}$$
$$f(C_1, C_2, \ldots, C_n) \simeq g\big(C_i, h(C_1, \ldots, C_{i-1}, C_{i+1}, \ldots, C_n)\big)\,. \tag{5}$$

Flattening This property is complementary to incrementality. It requires that, for any system obtained by hierarchically applying two glue operators to a finite set of sub-systems, there must exist an equivalent system obtained by applying a single glue operator to the *same* sub-systems:

$$\textit{for all } i, j \in [1,n]\ (i \leq j),\ C_1, C_2, \ldots, C_n \in \mathcal{A} \textit{ and } f, g \in \mathcal{G}\textit{, there exists } h \in \mathcal{G}\textit{, such that}$$
$$f\big(C_1, \ldots, C_{i-1}, g(C_i, \ldots, C_j), C_{j+1}, \ldots, C_n\big) \simeq h(C_1, \ldots, C_n)\,. \tag{6}$$

In other words, $\mathcal{G}$ must be closed under composition. Flattening enables model transformations, e.g. for optimising code generation or component placement on multicore platforms [13, 16].

Compositionality This property requires that glue operators preserve the equivalence of their operands:

$$\textit{for all } i \in [1,n],\ C_1, \ldots, C_n, C_i' \in \mathcal{A} \textit{ and } f \in \mathcal{G}\textit{,}$$
$$C_i \simeq C_i' \implies f(C_1, \ldots, C_i, \ldots C_n) \simeq f(C_1, \ldots, C_i', \ldots C_n)\,. \tag{7}$$

[1] A finer approach, adopted in [6], relies on a semantic preorder, rather than an equivalence relation, in order to encompass the notion of refinement. However, this distinction is not necessary in the context of the present paper.

Another version of this property, which we will call *relaxed compositionality*, only requires that individual glue operators respect behaviour equivalence:

$$\textit{for all } i \in [1,n],\ B_1,\ldots,B_n,B_i' \in \mathcal{B} \textit{ and } f \in \mathcal{G},\\ B_i \simeq B_i' \implies f(B_1,\ldots,B_i,\ldots B_n) \simeq f(B_1,\ldots,B_i',\ldots B_n)\,. \quad (8)$$

Notice that, combined with flattening, this relaxed notion of compositionality is already quite strong: essentially, compositionality allows replacing sub-systems, whereas relaxed compositionality with flattening allow replacing "atomic" behaviours.[2]

Modularity By combining the requirement that the equivalence relation $\simeq$ must respect the semantics of the framework (4) with compositionality (7), we obtain a special case that is important enough to be considered a separate property:

$$\textit{for all } i \in [1,n],\ C_1,\ldots,C_n \in \mathcal{A} \textit{ and } f \in \mathcal{G},\quad f(C_1,\ldots,C_i,\ldots,C_n) \simeq f(C_1,\ldots,\sigma(C_i),\ldots,C_n)\,. \quad (9)$$

Compositionality and modularity are related to the concepts of encapsulation and information hiding from object-oriented programming. Component-based frameworks provide a disciplined mechanism for restricting access to component's data, exposing only those elements that are explicitly used for communication, e.g. shared memory and buffers used for receiving messages from the environment. However, in order to provide full modularity, designers must have the possibility to bundle several components together with the connecting glue operators into a new component in order to *hide from the user the details of the component implementation*. This achieves two main goals: 1) the use of the component cannot rely on the specifics of its implementation, allowing the component to be replaced with an alternative solution; 2) components can be delivered to the user without disclosing the details of complex solutions constituting intellectual property of the designer.

The central subject of this paper is the semantics of the BIP component-based framework [5], which we introduce below. Glue operators in BIP are n-ary. Hence, we will focus our attention on compositionality, modularity and flattening, disregarding incrementality. Indeed, as mentioned above, incrementality can be viewed as generalised associativity, which is mainly useful, in our context, to be able to reason about binary operators and generalise the results to n-ary ones.

One can make the following observations about the relations between compositionality and modularity:

1. Compositionality implies modularity and relaxed compositionality.
2. Modularity and relaxed compositionality together imply compositionality:

   Proof. Without loss of generality, let $i = 1$; for all $C_1,\ldots,C_n,C_1' \in \mathcal{A}$ and $f \in \mathcal{G}$, we have $\sigma(C_1) \simeq C_1 \simeq C_1' \simeq \sigma(C_1')$ and, consequently,

   $$f(C_1,\ldots,C_n) \simeq f\big(\sigma(C_1),\ldots,\sigma(C_n)\big) \simeq f\big(\sigma(C_1'),\ldots,\sigma(C_n)\big) \simeq f(C_1',\ldots,C_n)\,.$$

   □
3. If the semantic mapping is structural and its defining operators $\tilde{f}$, for all $f \in \mathcal{G}$, have relaxed compositionality, then the framework has compositionality:

[2] A formal definition of the notion of behaviour atomicity is given in Section 2.3. Here, we use it intuitively to designate the behaviours at the lowest level the structural hierarchy of a system in the sense of (1).

Proof. Without loss of generality, let $i = 1$; for all $C_1, \ldots, C_n, C'_1 \in \mathcal{A}$ and $f \in \mathcal{G}$, we have

$$f(C_1, \ldots, C_n) \simeq \sigma\big(f(C_1, \ldots, C_n)\big) = \tilde{f}\big(\sigma(C_1), \ldots, \sigma(C_n)\big)$$
$$\simeq \tilde{f}\big(\sigma(C'_1), \ldots, \sigma(C_n)\big) = \sigma\big(f(C'_1, \ldots, C_n)\big) \simeq f(C'_1, \ldots, C_n)\,.$$

□

In this paper, we study the semantics and algebraic representations of glue operators in BIP, a component framework for constructing concurrent systems by superposing three layers: Behaviour, Interaction and Priorities. BIP is based on the separation of concerns between coordination and computation, which is essential for component-based design of concurrent systems. This separation allows systems to be built from units processing sequential code insulated from concurrent execution issues. The isolation of coordination mechanisms allows global treatment and analysis.

In BIP, component behaviour is defined in terms of *Labelled Transition Systems* (LTS). Glue operators are separated in two categories: *interaction models* define the sets of allowed *interactions*, that is synchronisations between the transitions of their operand components; *priority models* define the scheduling—or conflict resolution—policies, reducing non-determinism when several synchronisations allowed by the interaction model are enabled simultaneously. The semantics of glue operators is given in terms of Structural Operational Semantics (SOS) rules [33] following a certain restricted sub-format of GSOS [12]. The semantics of interaction models is given by rules involving only positive premises, whereas that of priorities introduces additional negative premises. The intuition behind is clear: an enabled interaction can be fired only if all higher-priority interactions are disabled. The use of SOS rules to define the semantics of glue operators has lead us to consider yet another property—that of *full expressiveness*. Indeed, it is desirable, in the above setting, that all combinations of SOS rules in the sub-format that we are considering, be expressible as glue operators in BIP.

In the previous work [7, 9, 10, 11], we have conducted an extensive study of the semantics and algebraic representations of the BIP glue operators. While the semantics of interaction models is very straightforward and did not change throughout those papers, that of the priority models has proven to be much subtler. In Section 2, we provide a concise overview of the historical evolution of the semantics of glue operators in BIP: from the semantics that, here, we call *classical* [7], through a very slight modification introduced in [9], to the latest semantics, proposed in [11], which we call *offer semantics*, since it relies on a different kind of negative premises—the *offer predicate*—than those used in the classical semantics of priority models.

The new contributions of this paper :

- We show how these minor, apparently innocuous modifications have impacted the flattening, modularity and expressiveness of BIP glue operators:
  - the classical BIP semantics [7] is structural, it has compositionality (hence also modularity), but does not have flattening and full expressiveness;
  - a very slightly modified version [9] has flattening, full expressiveness and relaxed compositionality, but it does not have modularity (hence also compositionality), since it is not structural;
  - the offer semantics [11] is structural, it has compositionality, flattening and full expressiveness.
- In [11], we have shown that, in general, the classical and the offer semantics are incomparable. In this paper, we provide a characterisation of a hierarchy of behaviours that allows to establish the correspondence between the classical and the proposed semantics and to evaluate the impact of changing the semantics on the existing systems.

In addition, this paper integrates our results published in the Proceedings of the 6$^{\text{th}}$ Interaction and Concurrency Experience workshop [4], where we transposed the theory of algebraic representations of BIP interaction models to the new semantics. In particular, these results allow the synthesis of connectors, encompassing both interaction and priority models, from Boolean formulas representing the state properties to be imposed by the glue operators. Thus, this paper provides a summary of all currently available results on the new semantics for BIP glue operators, defined using the offer predicate.

The paper is structured as follows. In Section 2, we provide a historical perspective on the evolution of the BIP glue operator semantics and exhibit the impact of this evolution on their flattening, modularity and expressiveness. In Section 3, we provide a characterisation of a hierarchy of behaviours, establishing the correspondence between the classical and the proposed semantics. In Section 4, we briefly recall the algebras, their syntax and semantics, introduced in our previous work to represent the interaction models. In Section 5, we present the extension of these algebras encompassing the priority models in the semantics based on the offer predicate. In Section 6, we illustrate the use of these extended algebras with a connector synthesis example. Finally, in Section 7, we provide a brief overview of related work. Section 8 concludes the paper.

2 A historical perspective on the evolution of the BIP glue semantics

2.1 Classical semantics

In the classical BIP semantics [24, 7], component behaviour is modelled by Labelled Transition Systems.

Definition 2.1. A *labelled transition system* (LTS) is a triple $(Q, P, \rightarrow)$, where Q is a set of *states*, P is a set of *ports*, and $\rightarrow \subseteq Q \times 2^P \times Q$ is a set of *transitions* labelled by sets of ports, such that only self-loops can be labelled by the empty set of ports, i.e. $(q, \emptyset, q') \in \rightarrow$ implies $q = q'$.

For $q, q' \in Q$ and $a \in 2^P$, we write $q \xrightarrow{a} q'$ iff $(q, a, q') \in \rightarrow$. A label $a \in 2^P$ is *active* in a state $q \in Q$ (denoted $q \xrightarrow{a}$), iff there exists $q' \in Q$ such that $q \xrightarrow{a} q'$. We abbreviate $q \not\xrightarrow{a} \stackrel{def}{=} \neg(q \xrightarrow{a})$.

Intuitively, transitions labelled by $\emptyset$ represent idling: a component that remains idle should not change state, hence the restriction to self-loops. Notice that we distinguish idling from unobservable internal transitions, which we do not model explicitly. To model unobservable transitions, one can use a reserved label, e.g. τ or ε, and restrict the ways it can be syncrhonised with other transitions. This is the approach traditionally taken in the litterature [30, 25].

Note 2.2. In the rest of the paper, whenever we speak of a set of LTS $B_i = (Q_i, P_i, \rightarrow_i)$, for $i \in [1, n]$, we assume that all P_i and Q_i are pairwise disjoint, i.e. $i \neq j$ implies $P_i \cap P_j = Q_i \cap Q_j = \emptyset$. We denote $P \stackrel{def}{=} \bigcup_{i=1}^{n} P_i$. We will drop the indices on transition relations and denote them by $\rightarrow$, whenever the indices are clear from the context.

Glue operators are defined using interaction and priority models.

Interaction models We call an *interaction* a subset of ports $a \subseteq P$. An *interaction model* is a set of interactions $\gamma \subseteq 2^P$. The component $\gamma(B_1, \ldots, B_n)$ is defined by the behaviour $(Q, P, \rightarrow_\gamma)$, with $Q = \prod_{i=1}^{n} Q_i$ and the transition relation $\rightarrow_\gamma$ inductively defined by the rule

$$\frac{a \in \gamma \quad \left\{ q_i \xrightarrow{a \cap P_i} q_i' \;\middle|\; i \in I \right\} \quad \left\{ q_i = q_i' \;\middle|\; i \notin I \right\}}{q_1 \ldots q_n \xrightarrow{a}_\gamma q_1' \ldots q_n'} \tag{10}$$

where, in the second rule, $I = \{i \in [1,n] \mid a \cap P_i \neq \emptyset\}$.

Intuitively, this means that an interaction a allowed by the interaction model γ can be fired when all the components involved in a are ready to fire the corresponding transitions. All the components that are not involved in a remain in their current state. Notice that, when the interaction model allows idling, i.e. $\emptyset \in \gamma$, the composed component has a self-loop labelled by $\emptyset$ in every state. The fact that components can have idling self-loops does not introduce any ambiguity in the interpretation of (10), since, by Definition 2.1, $q \xrightarrow{\emptyset} q'$ implies $q = q'$.

Priority models For a behaviour $B = (Q, P, \to)$, a *priority model* is a strict partial order $\pi \subseteq 2^P \times (2^P \setminus \{\emptyset\})$ (we write $a \prec b$ as a shorthand for $(a,b) \in \pi$). We put $\pi(B) \stackrel{def}{=} (Q, P, \to_\pi)$, with the transition relation $\to_\pi$ inductively defined by the rule

$$\frac{q \xrightarrow{a} q' \quad \left\{ q \not\xrightarrow{b} \;\middle|\; a \prec b \right\}}{q \xrightarrow{a}_\pi q'}. \tag{11}$$

Intuitively, this means that an interaction can be fired only if no higher-priority interaction is enabled. Notice that we exclude the priority $a \prec \emptyset$. Indeed, if idling is allowed by the interaction model, it will always be possible, effectively suppressing interaction a in all states. If this is the desired outcome, then a should rather be removed from the interaction model. Furthermore, such a priority could induce a kind of "disguised deadlock", when an interaction is suppressed in favour of doing nothing (cf. also Lemma 2.8).

Note 2.3. The rules (10) and (11) defining the semantics of BIP operators require that a partition $\bigcup_{i=1}^n P_i = P$ be defined, but not on the specific behaviours $B_1, \ldots, B_n$.

We are now in position to introduce the BIP glue operators.

Definition 2.4. A *BIP glue operator* is a quadruple $\big(P, (P_i)_{i=1}^n, \gamma, \pi\big)$, where P is a set of ports, $\bigcup_{i=1}^n P_i = P$ is a partition of P, $\gamma \subseteq 2^P$ and $\pi \subseteq 2^P \times (2^P \setminus \{\emptyset\})$ are, respectively interaction and priority models on P.

To avoid excessive notation, in the rest of the paper, we will only mention explicitly the interaction and priority models defining a BIP glue operator. The set of ports and the partition $P = \bigcup_{i=1}^n P_i$ will be implicitly assumed known.

Notice that both interaction and priority models can be neutral. Indeed, a neutral interaction model over the set of ports P is the set 2^P of all possible interactions. A neutral priority model is empty with none of the interactions having higher priority than any other. Thus, both interaction and priority models are also considered as BIP glue operators on their own.

We define the behaviour equivalence as follows.

Definition 2.5. Two behaviours $B_i = (Q_i, P_i, \to)$, for $i = 1, 2$ are *equivalent* if $P_1 = P_2$, and the two LTS are bisimilar, i.e. there exists a bisimulation [32] relation $R \subseteq Q_1 \times Q_2$ total on both Q_1 and Q_2.

Example 2.6. Consider the two components B_1 and B_2 shown in Figures 1a and 1b, with $P_1 = \{p, q\}$ and $P_2 = \{r\}$, and put $\gamma = \{p, q, r, qr\}$ and $\pi = \{q \prec r\}$.[3] The glue operator defined

[3]To simplify the notation we use the juxtaposition $\gamma = \{p, q, r, qr\}$ instead of the set notation $\gamma = \{\{p\}, \{q\}, \{r\}, \{q, r\}\}$ for interactions. Similarly, we directly write $\pi = \{q \prec r\}$ instead of $\pi = \{(q, r)\}$

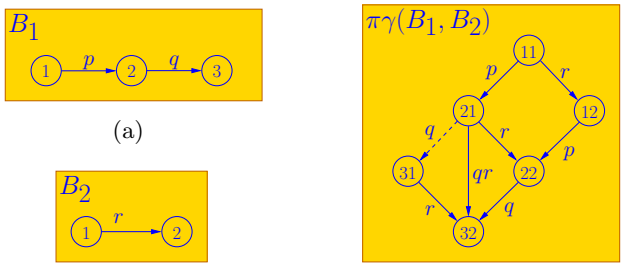


Figure 1: Component behaviours for Example 2.6

by the combination of the interaction model γ and the priority model π is given by the following four rules:

$$\frac{q_1 \xrightarrow{p} q_1'}{q_1q_2 \xrightarrow{p} q_1'q_2}, \qquad \frac{q_2 \xrightarrow{r} q_2'}{q_1q_2 \xrightarrow{r} q_1q_2'}, \qquad \frac{q_1 \xrightarrow{q} q_1' \quad q_2 \xrightarrow{r} q_2'}{q_1q_2 \xrightarrow{qr} q_1'q_2'}, \qquad \frac{q_1 \xrightarrow{q} q_1' \quad q_2 \not\xrightarrow{r}}{q_1q_2 \xrightarrow{q} q_1'q_2}. \tag{12}$$

The composed component $\pi\gamma(B_1, B_2)$ is shown in Figure 1c. The dashed arrow $21 \xrightarrow{q} 31$ shows the transition present only in $\gamma(B_1, B_2)$, but not in $\pi\gamma(B_1, B_2)$. Solid arrows show the transitions of $\pi\gamma(B_1, B_2)$.

Among the transitions labeled by q, only the transition $22 \xrightarrow{q} 32$ is enabled and not $21 \xrightarrow{q} 31$ (Figure 1c). Indeed, the negative premise in the fourth rule of (12), generated by the priority $q \prec r$, suppresses the interaction q when a transition labeled r is possible in the second component. □

It is important to observe that the rules in (12) are obtained by composing rules of forms (10) and (11). In particular, the fourth rule is obtained by the following derivation:

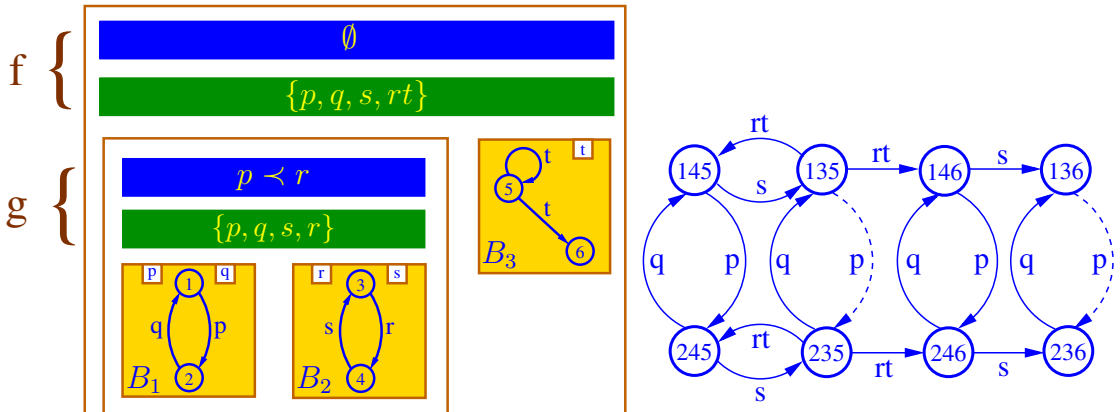
$$\frac{\dfrac{q \in \gamma \quad q_1 \xrightarrow{q} q_1' \quad q_2 = q_2'}{q_1q_2 \xrightarrow{q}_\gamma q_1'q_2'} \qquad \dfrac{r \notin \gamma \quad \vee \quad q_2 \not\xrightarrow{r}}{q_1q_2 \not\xrightarrow{r}_\gamma}\,(*)}{q_1q_2 \xrightarrow{q}_\pi q_1'q_2'}. \tag{13}$$

The sub-derivation (*) in (13) is obtained by negating the premises of the instance of the second rule in (10) with $a = r$. This is possible because the transition relation in $\gamma(B_1, B_2)$ is defined by (10) inductively, i.e. it is the minimal transition relation satisfying (10).

In (12), we have simplified (13) by removing premises, whereof satisfaction does not depend on the state of the operand components: $q \in \gamma$ (satisfied in all states) and $r \notin \gamma$ (dissatisfied in all states), and by replacing q_2' with q_2. Notice that the priority $q \prec r$ affects the behaviour of the composed system only because $r \in \gamma$. Indeed, if r did not belong to γ, the premise $r \notin \gamma$ would always be satisfied independently of the state of the system.

Notice that, after the simplification by removing the constant premises all rules used to define the semantics of BIP glue operators follow the following format (a restriction of GSOS [12]):

$$\frac{\left\{q_i \xrightarrow{a \cap P_i} q_i' \,\middle|\, i \in I\right\} \quad \left\{q_i = q_i' \,\middle|\, i \notin I\right\} \quad \left\{q_j \not\xrightarrow{b_j^k} \,\middle|\, j \in J, k \in K_j\right\}}{q_1 \dots q_n \xrightarrow{a} q_1' \dots q_n'}, \tag{14}$$



(a) Composed system

(b) Composed LTS (in dashed, the transitions suppressed by the priority model)

Figure 2: BIP component that cannot be flattened (Example 2.9).

where $I = \{i \in [1,n] \,|\, a \cap P_i \neq \emptyset\}$, $J, K_j \subseteq [1,n]$ and, for each $j \in J$ and $k \in K_j$, $b_j^k \subseteq P_j$.

The classical BIP semantics presented defined by (10) and (11) is structural. Furthermore, since both rule schemata follow the GSOS format, they preserve bisimilarity [12], i.e. they have relaxed compositionality and, consequently, the classical semantics has compositionality (see the discussion in Section 1).

Example 2.9, below, shows that BIP glue operators with the classical semantics presented above do not possess neither flattening, nor full expressiveness: in general, when combined hierarchically BIP glue operators with the classical semantics above cannot be flattened w.r.t. any bisimilarity-compatible equivalence; there exist operators defined by rules in format (14) that cannot be expressed as a combination of an interaction and a priority model.

First, we recall an important property of the BIP glue operators with the above semantics, which was originally shown in [24], is that application of a priority model does not introduce deadlocks.

Definition 2.7. Let $B = (Q, P, \to)$ be a behaviour. A state $q \in Q$ is a *deadlock* iff holds $\forall a \subseteq P$, $q \not\xrightarrow{a}$.

Lemma 2.8 ([24])**.** *Let $B_i = (Q_i, P_i, \to)$, for $i \in [1,n]$, be a set of behaviours, γ and π be respectively interaction and priority models on $P = \bigcup_{i=1}^{n} P_i$. A state $q \in \prod_{i=1}^{n} Q_i$ is a deadlock in $\pi\gamma(B_1, \ldots, B_n)$ if and only if it is a deadlock in $\gamma(B_1, \ldots, B_n)$.*

Proof. The "if" implication is trivial. To prove the "only if" implication, assume that, for some $a \in \gamma$, we have $q \xrightarrow{a}_{\gamma}$. Let $b \subseteq P$ be an interaction, maximal w.r.t. π, such that $b \in \gamma$, $a \prec b$ and $q \xrightarrow{b}_{\gamma}$. If such b exists, holds $q \xrightarrow{b}_{\pi}$. Otherwise holds $q \xrightarrow{a}_{\pi}$. In both cases, q is not a deadlock in $\pi\gamma(B_1, \ldots, B_n)$. □

Although Definition 2.7 is strict in the sense that it does not consider idling as a deadlock, the result of Lemma 2.8 can be straightforwardly strengthened to include idling states: if q is a purely idling state in $\pi\gamma(B_1, \ldots, B_n)$, i.e. $q \not\xrightarrow{a}_{\pi}$, for all $a \neq \emptyset$, then q is a purely idling state in $\gamma(B_1, \ldots, B_n)$.

Example 2.9. Consider the composed behaviour $f(g(B_1, B_2), B_3)$ (Figure 2a), with the glue operator g defined by the interaction model $\gamma_1 = \{p, q, r, s\}$ and priority model $\pi_1 = \{p \prec r\}$;

f defined by the interaction model $\gamma_2 = \{p, q, s, rt\}$ and the empty priority model. The LTS of the composed behaviour is shown in Figure 2b with the transitions, suppressed as the result of applying priority in g, shown as dashed arrows. Composing the rules corresponding to these operators as shown in (13), we obtain the four rules

$$\frac{p \in \gamma_1 \cap \gamma_2 \quad q_1 \xrightarrow{p} q_1' \quad (q_2 \not\xrightarrow{r} \ \vee \ r \notin \gamma_1)}{q_1q_2q_3 \xrightarrow{p} q_1'q_2q_3}, \qquad \frac{q \in \gamma_1 \cap \gamma_2 \quad q_1 \xrightarrow{q} q_1'}{q_1q_2q_3 \xrightarrow{q} q_1'q_2q_3},$$

$$\frac{s \in \gamma_1 \cap \gamma_2 \quad q_2 \xrightarrow{s} q_2'}{q_1q_2q_3 \xrightarrow{s} q_1q_2'q_3}, \qquad \frac{r \in \gamma_1 \quad rt \in \gamma_2 \quad q_2 \xrightarrow{r} q_2' \quad q_3 \xrightarrow{t} q_3'}{q_1q_2q_3 \xrightarrow{rt} q_1q_2'q_3'}. \tag{15}$$

Assume that an interaction model γ and a priority model π are such that $\pi\gamma(B_1, B_2, B_3)$ is equivalent to $f(g(B_1, B_2), B_3)$. By the first rule in (15), the transition $14x \xrightarrow{p} 24x$ is possible in $(f \circ g)(B_1, B_2, B_3)$, for any $x \in \{5, 6\}$. Hence, $p \in \gamma$. Clearly, 136 is a deadlock state in $(f \circ g)(B_1, B_2, B_3)$. Hence, 136 must be a deadlock state in $\pi\gamma(B_1, B_2, B_3)$ and, by Lemma 2.8, also in $\gamma(B_1, B_2, B_3)$, which is not possible, since all the premises of the rule

$$\frac{p \in \gamma \quad q_1 \xrightarrow{p} q_1'}{q_1q_2q_3 \xrightarrow{p} q_1'q_2q_3},$$

corresponding to p in the semantics (10) of γ, are satisfied for $q_1 = 1$ and $q_1' = 2$. □

In the Example 2.9, flattening is not possible due to the fact that the information used by the priority model refers only to interactions authorised by the underlying interaction model. All the information about transitions enabled in sub-components is lost (cf. $r \notin \gamma_1$ in the last premise of the first rule in (15)).

Simplifying (15) by removing the constant premises, we obtain a set of rules in the format (14)

$$\frac{q_1 \xrightarrow{p} q_1' \quad q_2 \not\xrightarrow{r}}{q_1q_2q_3 \xrightarrow{p} q_1'q_2q_3}, \qquad \frac{q_1 \xrightarrow{q} q_1'}{q_1q_2q_3 \xrightarrow{q} q_1'q_2q_3}, \qquad \frac{q_2 \xrightarrow{s} q_2'}{q_1q_2q_3 \xrightarrow{s} q_1q_2'q_3}, \qquad q\frac{q_2 \xrightarrow{r} q_2' \quad q_3 \xrightarrow{t} q_3'}{q_1q_2q_3 \xrightarrow{rt} q_1q_2'q_3'}, \tag{16}$$

defining an operator that cannot be expressed as a BIP glue operator in the classical semantics, which shows that this semantics does not have full expressiveness.

2.2 Achieving flattening and full expressiveness

Whereas combining the application of two interaction models in the classical semantics is straightforward—intuitively such a combination applies the synchronisation constraints imposed by both—, combining priority models is less intuitive. Consider, for instance, the following example.

Example 2.10. Let B_1, B_2 and B_3 be three behaviours with one state and one self-loop transition each, labelled by the ports p, q and r, respectively. Thus, for instance, $B_1 = \big(\{*\}, \{p\}, \{* \xrightarrow{p} *\}\big)$. Consider a unary glue operator f and a ternary glue operator g, obtained by combining the interaction model $\gamma = \{p, q, r\}$ with priority models $\pi_f = \{p \prec q\}$ and $\pi_g = \{q \prec r\}$, respectively.

Hence, the rules defining the semantics of the two operators are[4]

$$f: \quad \frac{q \xrightarrow{p} q' \quad q \not\xrightarrow{q}}{q \xrightarrow{p}_f q'}, \qquad \frac{q \xrightarrow{q} q'}{q \xrightarrow{q}_f q'}, \qquad \frac{q \xrightarrow{r} q'}{q \xrightarrow{r}_f q'}, \tag{17}$$

$$g: \quad \frac{q_1 \xrightarrow{p} q_1'}{q_1q_2q_3 \xrightarrow{p}_g q_1'q_2q_3}, \qquad \frac{q_2 \xrightarrow{q} q_2' \quad q_3 \not\xrightarrow{r}}{q_1q_2q_3 \xrightarrow{q}_g q_1q_2'q_3}, \qquad \frac{q_3 \xrightarrow{r} q_3'}{q_1q_2q_3 \xrightarrow{r}_g q_1q_2q_3'}. \tag{18}$$

Consider now the operator $f \circ g$. The priority model π_g inhibits port q when port r is active. Hence, $g(B_1, B_2, B_3) = \big(\{*\}, \{p, q, r\}, \{* \xrightarrow{p} *, * \xrightarrow{r} *\}\big)$. Similarly, the priority model π_f inhibits port p when port q is active, however the port q becomes inactive after the application of g. Thus, we also have $(f \circ g)(B_1, B_2, B_3) = \big(\{*\}, \{p, q, r\}, \{* \xrightarrow{p} *, * \xrightarrow{r} *\}\big)$, i.e. the priority $p \prec q$ does not affect the system, even though transition labelled by q is actually enabled in B_2.

Assume that firing the transition labelled by p leads the system to a critical state, whereas firing the transition labelled by q takes the system out of such state. If q is active, p should not be fired even if q is inhibited by another transition, here the one labelled by r. □

Another manifestation of the same problem can be observed by composing the rules corresponding to the two operations. Composing the first rule in (17) with the first and second rules in (18), we obtain

$$\frac{\dfrac{q_1 \xrightarrow{p} q_1'}{q_1q_2q_3 \xrightarrow{p}_g q_1'q_2q_3} \quad \dfrac{q_2 \not\xrightarrow{q} \ \vee \ q_3 \xrightarrow{r}}{q_1q_2q_3 \not\xrightarrow{q}_g}}{q_1q_2q_3 \xrightarrow{p}_{f\circ g} q_1'q_2q_3},$$

which can be expanded to the first two rules of (19) below. The other two rules are obtained in a more straightforward manner, so we omit their derivations:

$$\frac{q_1 \xrightarrow{p} q_1' \quad q_2 \not\xrightarrow{q}}{q_1q_2q_3 \xrightarrow{p}_{f\circ g} q_1'q_2q_3}, \qquad \frac{q_1 \xrightarrow{p} q_1' \quad q_3 \xrightarrow{r}}{q_1q_2q_3 \xrightarrow{p}_{f\circ g} q_1'q_2q_3}, \qquad \frac{q_2 \xrightarrow{q} q_2' \quad q_3 \not\xrightarrow{r}}{q_1q_2q_3 \xrightarrow{q}_{f\circ g} q_1q_2'q_3}, \qquad \frac{q_3 \xrightarrow{r} q_3'}{q_1q_2q_3 \xrightarrow{r}_{f\circ g} q_1q_2q_3'}. \tag{19}$$

Notice, however, that the second rule does not respect the format (14), in particular, since the label of the conclusion, p, is not the union of the labels of the premises.

In [24], this problem is addressed by defining the combination of two priority models as a single one by taking the transitive closure of their unions. For instance, the combination of π_f with π_g is the priority model $\pi_{fg} = \{p \prec q \prec r\}$ (notice that since priority models in the classical semantics are strict partial orders, we also have $p \prec r$). In combination with $\gamma = \{p, q, r\}$ this gives the following set of rules:

$$\frac{q_1 \xrightarrow{p} q_1' \quad q_2 \not\xrightarrow{q} \quad q_3 \not\xrightarrow{r}}{q_1q_2q_3 \xrightarrow{p}_{f\circ g} q_1'q_2q_3}, \qquad \frac{q_2 \xrightarrow{q} q_2' \quad q_3 \not\xrightarrow{r}}{q_1q_2q_3 \xrightarrow{q}_{f\circ g} q_1q_2'q_3}, \qquad \frac{q_3 \xrightarrow{r} q_3'}{q_1q_2q_3 \xrightarrow{r}_{f\circ g} q_1q_2q_3'}. \tag{20}$$

The rules in (20) can be obtained from the rules (17) and (18) by combining, for each of the considered rule conclusions, the relevant negative premises from both sets of rules and adding the negative premise $q_3 \not\xrightarrow{r}$ in the first rule, which corresponds to the priority $p \prec r$, induced by the

[4] Although, for the rules defining the operator f, we overload letter q, the meaning is clear from the context: in $q \xrightarrow{q} q'$, q on the arrow is the port, whereas q and q' at the ends of the arrow are states.

transitive closure on the union of priority models. Combining the negative premises from both sets essentially corresponds to discarding the requirement that a priority can only refer to interactions belonging to the interaction model.

In [9], we have adopted this approach—relaxing the transitive closure requirement—to define a slightly modified version of the semantics of the BIP glue operators. Instead of defining the semantics of a BIP system structurally, by applying the rules (10) and (11) starting from the atomic behaviours and proceeding up in the hierarchy of the system, we proceed in two steps. 1) We combine the rules in the top-down in the hierarchy, by deriving the positive premises in the usual way and keeping all negative premises along the way; 2) The previous step produces a set of rules defining a single operator, which we apply to the set of atomic behaviours to obtain the overall behaviour of the system.

In particular, the semantics of a combination of an interaction model γ with a priority model π, is defined directly by the following set of rules:

$$\frac{a \in \gamma \quad \left\{ q_i \xrightarrow{a \cap P_i} q_i' \right\}_{i \in I} \quad \left\{ q_i = q_i' \right\}_{i \notin I} \quad \left\{ q_{j_b} \not\xrightarrow{b \cap P_{j_b}} \;\middle|\; a \prec b \right\}}{q_1 \dots q_n \xrightarrow{a}_{\pi\gamma} q_1' \dots q_n'}, \tag{21}$$

where $I = \{i \in [1,n] \,|\, a \cap P_i \neq \emptyset\}$ and, for each b in the last premise, $j_b \in [1,n]$ is such that $b \cap P_{j_b} \neq \emptyset$. We take all rules (21), for all $a \in \gamma$ and all possible choices of j_b. Intuitively, for each interaction b having higher priority than a, it is required that at least one of the components involved in b be unable to take the corresponding transition.

We give the formal definition of the composition $f \circ g$ of a unary operator f and an n-ary operator g defined respectively by the sets of rules R_f and R_g in the format (14). Notice that, since f is unary, the rules in R_f have the following format

$$r: \qquad \frac{q \xrightarrow{a_r} q' \quad \left\{ q \not\xrightarrow{b} \;\middle|\; b \in N_r \right\}}{q \xrightarrow{a_r}_f q'},$$

where we denote by a_r and N_r the label of the conclusion (and, since f is unary, the only positive premise) and the set of the labels of the negative premises of a rule $r \in R_f$. Similarly, for a rule $r \in R_g$, we denote a_r the label of the conclusion of r and N_r^i the set of labels of the negative premises $q_i \not\xrightarrow{}$ in r, for $i \in [1,n]$. The composed operator $f \circ g$ is then defined by the following set of rules

$$\left\{ \frac{\left\{ q_i \xrightarrow{a \cap P_i} q_i' \;\middle|\; i \in I \right\} \quad \left\{ q_i = q_i' \;\middle|\; i \notin I \right\} \quad \left\{ q_i \not\xrightarrow{b} \;\middle|\; i \in [1,n], b \in N_{r_g}^i \right\} \quad \left\{ q_{j_b} \not\xrightarrow{b \cap P_{j_b}} \;\middle|\; b \in N_{r_f} \right\}}{q_1 \dots q_n \xrightarrow{a}_{f \circ g} q_1' \dots q_n'} \;\middle|\; \begin{array}{c} r_f \in R_f, r_g \in R_g : \text{ s.t. } a_{r_f} = a_{r_g} = a \\ I = \{i \in [1,n] \,|\, a \cap P_i \neq \emptyset\} \\ \forall b \in N_{r_f}, (j_b \in [1,n] \wedge b \cap P_{j_b} \neq \emptyset) \end{array} \right\}. \tag{22}$$

We omit here the generalisation of this definition to the composition $f \circ (g_1, \dots, g_m)$ of an m-ary operator with m operators, such that the sum of their arities is equal to n. Such generalisation is straightforward, but cumbersome.

Example 2.11. Applying this definition to the composition $f \circ g$ from Example 2.10, we obtain the following rules.

$$\frac{q_1 \xrightarrow{p} q_1' \qquad q_2 \not\xrightarrow{q}}{q_1 q_2 q_3 \xrightarrow{p}_{f \circ g} q_1' q_2 q_3}, \qquad \frac{q_2 \xrightarrow{q} q_2' \qquad q_3 \not\xrightarrow{r}}{q_1 q_2 q_3 \xrightarrow{q}_{f \circ g} q_1 q_2' q_3}, \qquad \frac{q_3 \xrightarrow{r} q_3'}{q_1 q_2 q_3 \xrightarrow{r}_{f \circ g} q_1 q_2 q_3'}.$$

□

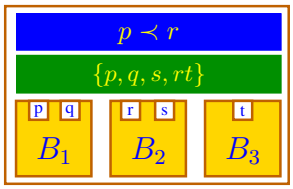


Figure 3: Flat system in the modified semantics equivalent to that in Figure 2a

Notice that, when all three interactions p, q, and r are enabled, both p and q are inhibited by q and r respectively, since the semantics of priority models is defined in terms of transitions of individual components, rather than their synchronisations obtained after the application of an interaction model. This allows us to relax the transitivity restriction in the definition of priority models.

Definition 2.12. Let P be a set of ports. A *relaxed priority model* on P is a relation $\pi \subseteq 2^P \times (2^P \setminus \{\emptyset\})$.

Notice that we do not require, in Definition 2.12, the relation π to be acyclic. If all interactions involved in a cyclic dependency in π are enabled simultaneously, they block each other, potentially introducing a deadlock. However, as it was shown in Section 2.1, preservation of deadlock-freedom by priorities is incompatible with full expressiveness. In the modified semantics, full expressiveness is a consequence of the results obtained in [9].[5]

Proposition 2.13. *Any operator defined by rules in format (14) can be represented by a combination of an interaction model and a relaxed priority model with the modified semantics presented in this section.*

Example 2.14. Recall Example 2.9. The composed operator defined by the rules (16), which cannot be flattened in the classical semantics, can be obtained, with the modified semantics by combining the interaction model $\gamma = \{p, q, s, rt\}$ with the priority model $\pi = \{p \prec r\}$ (Figure 3). □

Corollary 2.15. *Lemma 2.8 does not hold in the modified semantics.*

Proof. In the modified semantics, the state 136 is a deadlock in $\pi\gamma(B_1, B_2, B_3)$, with $\pi = \{p \prec r\}$, $\gamma = \{p, q, s, rt\}$, and B_1, B_2, B_3 from Figure 2a. However, 136 is not a deadlock in $\gamma(B_1, B_2, B_3)$. □

Proposition 2.16 (Flattening)**.** *Any hierarchical BIP glue operator can be flattened, when considered with the modified semantics.*

Sketch of the proof. The modified semantics associates to any BIP glue operator, i.e. a combination of an interaction and a priority model, a set of rules in the format (14). Composition (22) of operators, defined by such rules is an operator defined by rules in format (14). By Proposition 2.13, any such operator can again be expressed as a combination of an interaction and a priority model. □

[5] In [9, Proposition 4], we have mistakenly claimed that any such operator can be expressed as a combination of an interaction model and a classical priority model, i.e. a strict partial order on interactions. Replacing this claim by Proposition 2.13 does not fundamentally affect any of the other results in [9].

The semantic modification, introduced in this sub-section, provides the flattening of glue operators by definition, but at the price of loosing compositionality and even modularity. Indeed, consider again the hierarchical system from Example 2.10. Replacing the sub-system $g(B_1, B_2, B_3)$ by its corresponding semantic behaviour, would eliminate the transition labelled by q. Hence, the priority $p \prec q$ will not affect the behaviour of the system, exactly as illustrated by Example 2.10 in the classical semantics. Thus denoting, as in Section 1, $\sigma\big(g(B_1, B_2, B_3)\big)$, we have $f\big(g(B_1, B_2, B_3)\big) \not\simeq f\big(\sigma\big(g(B_1, B_2, B_3)\big)\big)$, which proves that modularity does not hold in the modified semantics.

2.3 Reconciling compositionality, flattening and full expressiveness

In order to recover compositionality, we have to redefine the semantics structurally. On the other hand, in order to maintain flattening, we have to provide the information about the active ports of atomic (see Definition 2.17 below) components throughout the composition process. In order to simultaneously achieve these two goals, we must enrich the notion of behaviour.

Definition 2.17 (Extended behaviour [11])**.** An *extended behaviour* is a quadruple $B = (Q, P, \rightarrow, \uparrow)$, where $(Q, P, \rightarrow)$ is an LTS and $\uparrow$ is an *offer* predicate on $Q \times P$, such that $q \uparrow p$ holds (a port $p \in P$ is *offered* in a state $q \in Q$) whenever there is a transition from q containing p, that is $(\exists a \in 2^P : p \in a \wedge q \xrightarrow{a}) \Rightarrow q \uparrow p$. If the converse implication also holds, i.e. $(\exists a \subseteq P : p \in a \wedge q \xrightarrow{a}) \iff q \uparrow p$, we call the extended behaviour *atomic*.

The offer predicate extends to sets of ports: for $a \in 2^P$, $q \uparrow a \stackrel{def}{=} \bigwedge_{p \in a} q \uparrow p$. Notice that $q \uparrow \emptyset \equiv \mathtt{tt}$. We denote $q \not\uparrow a \stackrel{def}{=} \neg(q \uparrow a) = \bigvee_{p \in a} q \not\uparrow p$.

Notice that, for any behaviour, an offer predicate can be defined that makes it atomic [11]. Thus, our notion of atomicity is weaker than the intuitive one. For instance, if a composed component is obtained by putting in parallel two atomic components without any coordination constraints, we consider it as one atomic component. In other words, we use the offer predicate to make explicit part of the information about the transitions of the atomic behaviours that is lost when these are composed by a restrictive operator. This notion of atomicity of behaviours is, however, more formal than that used in the introduction. We use it in Section 3 to characterize BIP systems for which classical glue operators can be systematically transformed into offer-based operators.

Definition 2.18. Two extended behaviours $B_i = (Q_i, P_i, \rightarrow_i, \uparrow_i)$, with $i = 1, 2$, are *equivalent* if $P_1 = P_2$ and there exists a bisimulation relation $R \subseteq Q_1 \times Q_2$, total on both Q_1 and Q_2, such that the offer predicates coincide on bisimilar states, i.e. for all $(q_1, q_2) \in R$ and $p \in P_1$, holds $q_1 \uparrow_1 p \Leftrightarrow q_2 \uparrow_2 p$.

BIP composition operators, consisting of an interaction and a relaxed priority model, can be given new operational semantics in terms of the offer predicate as follows.

For a set of behaviours $B_i = (Q_i, P_i, \rightarrow, \uparrow)$[6] and an interaction model $\gamma \subseteq 2^P$, the transition relation $\rightarrow_\gamma$ of the behaviour $\gamma(B_1, \ldots, B_n) = (Q, P, \rightarrow_\gamma, \uparrow_\gamma)$ is inductively defined by (10) and the offer predicate is defined by putting $q_1 \ldots q_n \uparrow_\gamma p \stackrel{def}{\iff} \exists i \in [1, n] : q_i \uparrow p$, for all $p \in P$ and $q_1 \ldots q_n \in Q$.

For a behaviour $B = (Q, P, \rightarrow, \uparrow)$ and a relaxed priority model $\pi \subseteq 2^P \times (2^P \setminus \{\emptyset\})$ (see Definition 2.12), we define $\pi(B) \stackrel{def}{=} (Q, P, \rightarrow_\pi, \uparrow)$, with the same sets of states and ports, and the

[6] As in Note 2.2, we omit the indices on $\uparrow$, whenever they are clear from the context.

same offer predicate as those of B and the transition relation π inductively defined by the rule

$$\frac{q \xrightarrow{a} q' \quad \left\{ q \not\nearrow b \;\middle|\; a \prec b \right\}}{q \xrightarrow{a}_{\pi} q'}. \tag{23}$$

In [11], we have considered a more general set of operators, defined by the rules in the following format:

$$\frac{\left\{ q_i \xrightarrow{a \cap P_i} q_i' \;\middle|\; i \in I \right\} \quad \left\{ q_i = q_i' \;\middle|\; i \notin I \right\} \quad \left\{ q_k \not\nearrow b_k^l \;\middle|\; k \in K, l \in L_k \right\} \quad \left\{ q_j \uparrow c_j \;\middle|\; j \in J \right\}}{q_1 \dots q_n \xrightarrow{a} q_1' \dots q_n'}, \tag{24}$$

where $I = \{i \in [1,n] \,|\, a \cap P_i \neq \emptyset\}$, $J, K, L_k \subseteq [1,n]$ and $c_j \subseteq P_j$, $b_k^l \subseteq P_k$, for all $j \in J$, $k \in K$ and $l \in L_k$. In (24), we have three types of premises respectively called *firing*, *witness*, and *negative* premises. Firing and witness premises are collectively called *positive*. Notice that $q\uparrow c_1 \wedge q\uparrow c_2 = q\uparrow c_1 c_2$. Hence one witness premise per component behaviour is sufficient to define any inference rule.

For a set of ports P, we denote $\dot{P} \stackrel{def}{=} \{\dot{p} \,|\, p \in P\}$. We call the elements of P and $\dot{P}$ respectively *activation* and *firing port typings*. The above generalisation can be translated into BIP terms by generalising interaction models to include witness port typings.

Definition 2.19. Let P be a set of ports. An *interaction with witnesses* is a subset $a \subseteq P \cup \dot{P}$. An *interaction model with witnesses* over P is a set $\gamma \subseteq 2^{P \cup \dot{P}}$ of interactions with witnesses.

The component $\gamma(B_1, \dots, B_n)$ is defined by the behaviour $(Q, P, \rightarrow_\gamma, \uparrow_\gamma)$, with $Q = \prod_{i=1}^n Q_i$ and the transition relation $\rightarrow_\gamma$ inductively defined by the following rule in format (24)

$$\frac{a \in \gamma \quad \left\{ q_i \xrightarrow{\{p \in P_i \,|\, \dot{p} \in a\}} q_i' \;\middle|\; i \in I \right\} \quad \left\{ q_i = q_i' \;\middle|\; i \notin I \right\} \quad \left\{ q_j \uparrow (a \cap P_j) \;\middle|\; j \in J \right\}}{q_1 \dots q_n \xrightarrow{a}_\gamma q_1' \dots q_n'}. \tag{25}$$

where $I = \{i \in [1,n] \,|\, a \cap \dot{P}_i \neq \emptyset\}$ and $J = \{j \in [1,n] \,|\, a \cap P_j \neq \emptyset\}$. The offer predicate $\uparrow_\gamma$ is defined, as above, by putting $q_1 \dots q_n \uparrow_\gamma p \stackrel{def}{\iff} \exists i \in [1,n] : q_i \uparrow p$, for all $p \in P$.

Boolean characterisation of operators defined by rules in format (24) In [11], we have considered an algebra $\mathbb{B}[P, \dot{P}]$ of Boolean formulas over *activation* variables P and the *firing* variables $\dot{P}$, with the additional axiom:

$$\dot{p} \Rightarrow p, \text{ for all } p \in P. \tag{26}$$

Note 2.20. A valuation of an activation variable $p \in P$ indicates whether the port p is active, i.e. the corresponding component has an enabled transition containing p in its label, whereas a valuation of a firing variable $\dot{p} \in \dot{P}$ indicates whether the corresponding port p will participate in the next interaction. A formula in $\mathbb{B}[P, \dot{P}]$ defines the constraints on the firing of ports, based on their activation: in a given global state of the system, the valuations of the activation variables are determined by the enabled transitions of the components; a valuation of the firing variables that complements the valuation of the activation ones in such a manner, that the overall valuation satisfies the formula, defines an admissible interaction (for formal presentation, see [11]). Obviously, a port cannot participate in an interaction if it is not active, justifying axiom (26).

In [11], we have established a correspondence between $\mathbb{B}[P, \dot{P}]$ and the glue operators defined by the rules in the format (24). For a rule r, denote $A = \bigcup_{i \in I} a_i$ and $C = \bigcup_{j \in J} c_j$. We associate to such a rule the formula $\varphi_r \in \mathbb{B}[P, \dot{P}]$ defined by putting

$$\varphi_r \stackrel{def}{=} \bigwedge_{p \in A} \dot{p} \ \wedge \bigwedge_{p \in P \setminus A} \overline{\dot{p}} \ \wedge \bigwedge_{p \in C} p \ \wedge \bigwedge_{k \in K} \bigwedge_{l \in L_k} \overline{b_k^l}\,, \tag{27}$$

where $\overline{b_k^l} = \bigvee_{p \in b_k^l} \overline{p}$. A formula associated to a glue operator is then the disjunction of formulas associated to the rules defining the operator.

Notice that the formulas that we obtain in this manner are in *firing-full* Disjunctive Normal Form (DNF), i.e. each firing variable appears in a positive or negative form in each monomial. The firing variables that appear in the negative form are precisely those, for which the respective ports do not appear in the firing premises of the corresponding rule.

In the opposite direction, given a formula $\varphi \in \mathbb{B}[P, \dot{P}]$, we consider its firing-full DNF. By grouping the monomials with the same positive variables, we have

$$\varphi = \bigvee_{a \in \gamma} \left(\bigwedge_{p \in a} p \ \wedge \bigwedge_{\dot{p} \in \dot{P} \setminus a} \overline{\dot{p}} \ \wedge \ \varphi_a \right), \tag{28}$$

with some $\gamma \subseteq 2^{P \cup \dot{P}}$ and, for each $a \in \gamma$, φ_a is a purely negative formula on activation variables, i.e. a DNF formula, where all variables are negative. (Notice that $p \in a$, in the sub-script of the first conjunct, can be both an activation and a firing variable.) To each monomial b in the DNF of φ_a, we associate the following rule

$$\frac{\left\{ q_i \xrightarrow{\{p \in P_i \,|\, \dot{p} \in a\}} q_i' \,\middle|\, i \in I \right\} \quad \left\{ q_i = q_i' \,\middle|\, i \notin I \right\} \quad \left\{ q_j \uparrow (a \cap P_j) \,\middle|\, j \in J \right\} \quad \left\{ q_i \not\uparrow p \,\middle|\, i \in [1,n], p \in b \cap P_i \right\}}{q_1 \dots q_n \xrightarrow{a} q_1' \dots q_n'}\,, \tag{29}$$

where $I = \{i \in [1,n] \,|\, a \cap \dot{P}_i \neq \emptyset\}$ and $J = \{j \in [1,n] \,|\, a \cap P_j \neq \emptyset\}$.

In particular, this correspondence gives the $\mathbb{B}[P, \dot{P}]$ formulas a semantics in terms of operators defined by rules in the format (24). The following proposition implies that the axioms of Boolean algebra are sound w.r.t. the operator semantics (29) of $\mathbb{B}[P, \dot{P}]$, which means that we can apply the full power of Boolean calculus for the manipulation of glue operators.

Proposition 2.21. *Consider two formulas $\varphi, \psi \in \mathbb{B}[P, \dot{P}]$, such that $\varphi \equiv \psi$. Let f_φ and f_ψ be the corresponding glue operators defined by the rules (29). Then, for any set of behaviours $B_1, \dots, B_n$, the behaviours $f_\varphi(B_1, \dots, B_n)$ and $f_\psi(B_1, \dots, B_n)$ are equivalent.*

Sketch of the proof. The proof is straightforward, by considering the impact of each axiom on the corresponding rules. Consider, for instance, the axiom of the excluded middle. In $\mathbb{B}[P, \dot{P}]$, this axiom can be instantiated by either $\dot{p} \vee \overline{\dot{p}} = \mathtt{tt}$, or $p \vee \overline{p} = \mathtt{tt}$. Clearly, if φ is obtained from ψ by the conjunction of one of its sub-formulas with $\dot{p} \vee \overline{\dot{p}}$, their firing-full DNF are the same, hence the corresponding operators are the same. Assume that φ is obtained from ψ by the conjunction of one of its sub-formulas with $p \vee \overline{p}$. Then, for some of the rules corresponding to ψ, the set of rules corresponding to φ would have two rules differing only by one premise: one rule with a positive premise $q \uparrow p$ and another with the negative premise $q \not\uparrow p$, all the other premises being the same as in the rule for ψ. It is sufficient, now, to notice that exactly in the same states, where the rule for ψ is applicable, one of the two rules would be applicable independently of the activation status of the port p. □

Properties of extended BIP glue operators in the offer semantics The above Boolean characterisation allows us to associate an interaction model with witnesses $\gamma \subseteq 2^{P \cup \dot{P}}$ and a relaxed priority model $\pi \subseteq 2^{P \cup \dot{P}} \times (2^P \setminus \{\emptyset\})$[7] to any operator defined by the set of rules in the format (24). We proceed in three steps: 1) we associate to the set of rules a $\mathbb{B}[P, \dot{P}]$ formula φ as in (27); 2) we rewrite φ as in (28); 3) for each $a \in \gamma$, we rewrite the positive formula $\overline{\varphi_a}$ in DNF. The set γ in (28), is the extended interaction model, whereas the relaxed priority model is $\pi = \{a \prec b \,|\, b \text{ is a monomial of the DNF of } \overline{\varphi_a}\}$.

Thus, we conclude that, in the offer semantics, extended BIP glue operators consisting of an extended interaction model $\gamma \subseteq 2^{P \cup \dot{P}}$ and a relaxed priority model $\pi \subseteq 2^{P \cup \dot{P}} \times (2^P \setminus \{\emptyset\})$ have full expressiveness w.r.t. the rule format (24), as well as w.r.t. the algebra $\mathbb{B}[P, \dot{P}]$ of Boolean formula over the variables P and $\dot{P}$.

Proposition 2.22. *Operators defined by sets of rules in the format (24) have relaxed compositionality w.r.t. the equivalence in Definition 2.18.*

Sketch of the proof. The proof is straightforward by definition of bisimulation. Indeed, if two behaviours are equivalent in the sense of Definition 2.18, the same rules are applicable to both of them in the bisimilar states. □

Since the semantics of the extended BIP glue operators is structural, Proposition 2.22 implies that it has full compositionality (hence also modularity).

Finally, observe that, as a consequence of the discussion in Note 2.20, in the offer semantics, an interaction $a \subseteq P$ is enabled in a global state $q_1 \dots q_n$ of a system $f(B_1, \dots, B_n)$ if and only if the transition $a \cap P_i$ is enabled in B_i, for each $i \in [1, n]$, such that $a \cap P_i \neq \emptyset$, and the valuations of variables from P and $\dot{P}$ defined, respectively, by the state $q_1 \dots q_n$ and by a satisfy the $\mathbb{B}[P, \dot{P}]$ formula corresponding to f.

Proposition 2.23. *Extended BIP glue operators in the offer semantics have the flattening property.*

Sketch of the proof. Consider a system $C = f(C_1, \dots, C_k, g(C_{k+1}, \dots, C_n))$. Since extended BIP glue operators in the offer semantics have compositionality, $C \simeq f\big(\sigma(C_1), \dots, \sigma(C_k), \sigma\big(g(C_{k+1}, \dots, C_n)\big)\big)$, where σ is the semantic mapping (see Section 1), induced by the offer semantics. Hence, an interaction a is possible in a global state of C if and only if the transitions labelled by the projections of a onto the sets of ports of $\sigma(C_i)$, for $i \in [1, k]$, and onto the set of ports of $\sigma\big(g(C_{k+1}, \dots, C_n)\big)$ are enabled and the valuations of variables from P and $\dot{P}$ defined, respectively, by the global state of C and by a satisfy the $\mathbb{B}[P, \dot{P}]$ formula φ_f corresponding to f.

Similarly, the transition labelled by the projection of a onto the set of ports of $\sigma\big(g(C_{k+1}, \dots, C_n)\big)$ is enabled if and only if the transitions labelled by the projections of a onto the sets of ports of C_i, for $i \in [k+1, n]$, are enabled and the valuations of variables from P and $\dot{P}$ defined, respectively, by the global state of $g(C_{k+1}, \dots, C_n)$ and by a satisfy the $\mathbb{B}[P, \dot{P}]$ formula φ_g corresponding to g. Thus, we conclude that a transition labelled by a is enabled in C if and only if the transitions labelled by the corresponding projections of a are enabled in $\sigma(C_i)$, for $i \in [1, n]$, and the valuations of variables from P and $\dot{P}$ defined, respectively, by the global state of C and by a satisfy $\varphi_f \wedge \varphi_g$. Let h be the operator corresponding to $\varphi_f \wedge \varphi_g$. We have $C \simeq h(C_1, \dots, C_n)$. □

To conclude this sub-section, observe that, if we restricted ourselves to 1) classical interaction models, i.e. subsets of 2^P, 2) rules in format (24) without the witness premises and 3) $\mathbb{B}[P, \dot{P}]$ formulas without positive occurrences of activation variables in the firing-full DNF form, all three properties—compositionality, full expressiveness and flattening—would still hold. However, we

[7] Notice that $2^P \subset 2^{P \cup \dot{P}}$.

would loose the soundness of some Boolean axioms. For instance, the excluded middle would not hold for activation variables, since the positive disjunct would introduce witness premises in the corresponding rules.

2.4 Further extension of interaction models to encompass priority

In this section, we further extend the notions of interaction and interaction model to include the negative port typings. This allows us to incorporate priorities into interaction models and, therefore, also extend the theory of algebraic representations of interaction models to encompass priorities.

In addition to the activation and firing port typings introduced in the previous sub-section, we consider the *negative port typing* $\overline{P} \stackrel{def}{=} \{\overline{p} \,|\, p \in P\}$.

Definition 2.24. An *extended interaction* is a subset $a \subseteq P \cup \dot{P} \cup \overline{P}$. An *extended interaction model* is a set $\gamma \subseteq 2^{P \cup \dot{P} \cup \overline{P}}$.

For a given extended interaction a, we define the following sets of ports:

- $\mathbf{act}(a) \stackrel{def}{=} a \cap P$, the *activation support* of a,
- $\mathbf{fire}(a) \stackrel{def}{=} \{p \in P \,|\, \dot{p} \in a\}$, the *firing support* of a,
- $\mathbf{neg}(a) \stackrel{def}{=} \{p \in P \,|\, \overline{p} \in a\}$, the *negative support* of a.

Definition 2.25. Let $B_i = (Q_i, P_i, \rightarrow, \uparrow)$, with $i \in [1, n]$ and $P = \bigcup_{i=1}^{n} P_i$, be a set of component behaviours. Let $\gamma \subseteq 2^{P \cup \dot{P} \cup \overline{P}}$ be a set of extended interactions. The composition of $\{B_i\}_{i=1}^{n}$ with γ is a behaviour $\gamma(B_1, \ldots, B_n) \stackrel{def}{=} (Q, P, \rightarrow_\gamma, \uparrow_\gamma)$ with $Q = \prod_{i=1}^{n} Q_i$, the offer predicate $\uparrow$ defined by putting, as above, $q_1 \ldots q_n \uparrow_\gamma p \stackrel{def}{\iff} \exists i \in [1, n] : q_i \uparrow p$, for all $p \in P$ and the transition relation $\rightarrow_\gamma$ inductively defined by the rule

$$\frac{a \in \gamma \quad \left\{q_i \xrightarrow{\mathbf{fire}(a) \cap P_i} q_i'\right\}_{i \in I} \quad \left\{q_i = q_i'\right\}_{i \notin I} \quad \left\{q_i \uparrow \left(\mathbf{act}(a) \cap P_i\right)\right\}_{i=1}^{n} \quad \left\{q_i \not\uparrow p \;\middle|\; p \in \mathbf{neg}(a) \cap P_i\right\}_{i=1}^{n}}{q_1 \ldots q_n \xrightarrow{\mathbf{fire}(a)}_\gamma q_1' \ldots q_n'}, \tag{30}$$

where $I = \{i \in [1, n] \,|\, \mathbf{fire}(a) \cap P_i \neq \emptyset\}$.

Taking on the Example 2.9, a flat composition of B_1, B_2 and B_3 equivalent to that of Figure 2a in the semantics of Definition 2.25 is obtained by taking $\gamma = \{\dot{p}\,\overline{r}, \dot{q}, \dot{s}, \dot{r}\,\dot{t}\} \subseteq 2^{P \cup \dot{P} \cup \overline{P}}$.

It is important to observe that, as stated by Lemma 2.26 below, sets of interactions can have redundancies.

Lemma 2.26. *Let $\gamma_1 \subseteq 2^{P \cup \dot{P} \cup \overline{P}}$ be a set of interactions, $\gamma_2 = \gamma_1 \cup \{a\}$, with $a \subseteq P \cup \dot{P} \cup \overline{P}$, such that there is an interaction $b \in \gamma_1$, $b \subseteq a$ and $\mathbf{fire}(b) = \mathbf{fire}(a)$. Then $\gamma_1(B_1, \ldots, B_n) = \gamma_2(B_1, \ldots, B_n)$.*

Proof. According to rule (30) any transition generated by the interaction a can also be generated by the interaction b. Thus, interaction a does not impact the behaviour of the composed system, and $\gamma_1(B_1, \ldots, B_n) = \gamma_2(B_1, \ldots, B_n)$. □

Intuitively, this lemma states that if any interaction that allows the same transition in the composed behaviour as another interaction, but under more restrictive conditions, cannot impact the composed system and, therefore, can be removed from the extended interaction model.

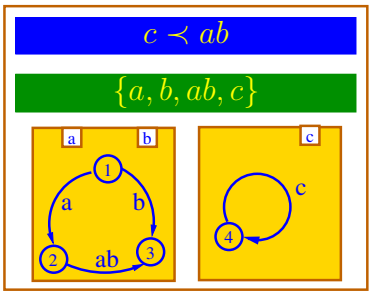


Figure 4: Behaviours of a system inexpressible in offer semantics.

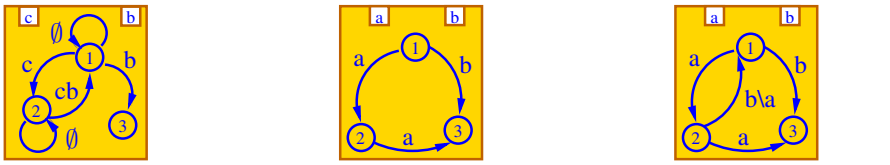
3 Transformation of systems in classical semantics into offer semantics

In [11] it was shown that the expressiveness of BIP glue in the classical (Section 2.1) and in the offer (Section 2.3) semantics are incomparable. Not all BIP systems in the classical semantics can be expressed in the offer semantics. For the sake of simplicity and in order to better distinguish the BIP glues considered in the classical and in the offer semantics, we will refer to the former through pairs consisting of an interaction model $\gamma \subseteq 2^P$ and a priority model $\pi \subseteq 2^P \times (2^P \setminus \{\emptyset\})$; the latter will be given by extended interaction models $\gamma \subseteq 2^{P \cup \dot{P} \cup \overline{P}}$ (Section 2.4). Recall that, in the classical semantics, interactions that do not appear in the interaction model have no effect, when used in the priority model. Therefore, in this section, we will assume that all interactions appearing in a priority model also belong to the corresponding interaction model.

Example 3.1. Consider a system of two behaviours, in the classical semantics, shown in Figure 4. The interaction model is $\{a, b, ab, c\}$ and priority model is $\{c \prec ab\}$. Since, classical priority semantics refers to the activation of an interaction, in the composed system the interaction c is available at the state 14, and not available at the state 24. In the offer semantics, all three ports are offered in both states 14 and 24 of this system. Therefore, these states are indistinguishable and c is inhibited in both. □

Note 3.2. In the remainder of this section, we will compare composed systems in the classical and offer semantics obtained by applying glue operators to the same set of behaviours. To simplify the presentation, we will assume that the predicate $\uparrow \subseteq Q \times P$ is also defined, in the classical semantics, on atomic behaviours as in Definition 2.17 and, for composed systems as in Definition 2.25. Notice that this unambiguously defines the offer predicate in both cases. Hence, we will not explicitly provide it in the examples of this section. Furthermore, sets of states and ports of composed systems, as well as the corresponding offer predicates do not depend on the glue operator used to obtain them. Therefore, to prove that two composed behaviours coincide, we will only have to check that their respective transition relations are equal. (Indeed, in this context, bisimilarity and equality coincide.) The following lemma shows that it is not necessary to consider target states of transitions and it is sufficient to compare labels of outgoing transitions for each state of composed systems.

Lemma 3.3. *Let $B_i = (Q_i, P_i, \rightarrow, \uparrow)$ for $i \in [1, n]$ be a set of behaviours and let $P = \bigcup_{i=1}^{n} P_i$. Let $\pi\gamma$ and γ' be glue operators on P in the classical and offer semantics respectively. Then for composed systems $(Q, P, \rightarrow_c, \uparrow) = \pi\gamma(B_1, \ldots, B_n)$ and $(Q, P, \rightarrow_o, \uparrow) = \gamma'(B_1, \ldots, B_n)$ the following holds: for any state q and for any transition label a, if $q \xrightarrow{a}_c \Leftrightarrow q \xrightarrow{a}_o$ then $\{q' | (q, a, q') \in \rightarrow_c\} = \{q' | (q, a, q') \in \rightarrow_o\}$.*



(a) $\emptyset \prec b$ (b) $a \prec b$, $a \neq \emptyset$ and $b \subseteq a$ (c) $a \prec b$, $a \neq \emptyset$ and $b \not\subseteq a$

Figure 5: Behaviours for Theorem 3.4.

Proof. If $q \not\xrightarrow{a}_c$ then $q \not\xrightarrow{a}_o$ and both sets are empty.

If $q \xrightarrow{a}_c$ then there is an interaction $a \in \gamma$. By (10) $(q, a, q') \in \rightarrow_c$ iff for all $i \in [1, n]$, $q_i \xrightarrow{a \cap P_i} q'_i$, if $a \cap P_i \neq \emptyset$, and $q_i = q'_i$ otherwise. At the same time $q \xrightarrow{a}_o$ and there is an interaction $a' \in \gamma'$, such that $\mathbf{fire}(a') = a$. By (30) $(q, a, q') \in \rightarrow_o$ iff for all $i \in [1, n]$, $q_i \xrightarrow{\mathbf{fire}(a') \cap P_i} q'_i$, $\mathbf{fire}(a') \cap P_i \neq \emptyset$, and $q_i = q'_i$ otherwise. Since $a = \mathbf{fire}(a')$, $\{q' | (q, a, q') \in \rightarrow_c\} = \{q' | (q, a, q') \in \rightarrow_o\}$. □

If a priority model of a glue operator is empty, then such glue operator can be easily transformed into an operator in the offer semantics. However, for any non-empty priority model there exists a set of behaviours, such that the transformation of this glue operator into the offer semantics is not possible.

Theorem 3.4. *Let $\pi\gamma$ be a glue operator on a set of ports P in the classical semantics, such that γ contains at least two non-empty interactions and π has at least one priority $a \prec b$ with $a \neq b$. There exists a set of atomic*[8] *behaviours $B_i = (Q_i, P_i, \rightarrow, \uparrow)$ for $i \in [1, n]$, where $\bigcup_{i=1}^{n} P_i = P$, such that, for any extended interaction model γ' in the offer semantics, the composed systems would not be equivalent, i.e. for $(Q, P, \rightarrow_c) = \pi\gamma(B_1, \ldots, B_n)$ and $(Q, P, \rightarrow_o, \uparrow) = \gamma'(B_1, \ldots, B_n)$ holds $\rightarrow_c \neq \rightarrow_o$.*

Proof. Let $a \prec b$, with $a \neq b$, be a priority in π and assume that the rule $a \prec ab$ is not in π (otherwise consider, instead, the rule $a \prec ab$). There are three cases. If $a = \emptyset$, let B_1 be the behaviour in Figure 5a, and assume $c \in \gamma$, $c \neq b$ and $c \neq \emptyset$ (such c exists by the assumption of the theorem). Recall that $a \prec \emptyset$ is not an valid priority. Therefore, we only have to consider two other cases, where neither a nor b are empty interactions: if $b \subseteq a$, let B_1 be the behaviour in Figure 5b; otherwise let B_1 be the behaviour in Figure 5c. The proof below applies identically to all three cases.

States 1 and 2 offer the same sets of ports. There is a transition a from both states. However, transition b is available only in the state 1. Let P_1 be a set of ports in B_1. Consider a second atomic behaviour $B_2 = \big(\{*\}, P \setminus P_1, \{* \xrightarrow{p} * \,|\, p \in P \setminus P_1\}, \uparrow\big)$, such that the union of sets of ports of B_1 and B_2 is equal to P. Both $\pi\gamma$ and γ' can be applied to the pair of behaviours (B_1, B_2). In the composed behaviour, transition a is available in the state $2*$, but not available in the state $1*$. Since these states offer the same ports, for any glue operator in the offer semantics transition a is either available in both states or in none of them. □

We are now in position to define three classes of behaviours, for which it is possible to generate an equivalent system in the offer semantics. The first class of behaviours is characterised by Property 3.5. For any glue operator in the classical semantics there exists a glue operator in the offer semantics, such that their applications to any set of behaviours satisfying Property 3.5 result in equivalent composed systems. The two remaining classes of behaviours comprise sets of components, such that the transformation for any glue operator exists, but depends on the

[8] Here and in the rest of the paper, the term *atomic* has the meaning provided in Definition 2.17.

Input: A glue operator in the classical semantics: an interaction model γ and a priority model π.
Output: A glue operator in the offer semantics: an extended interaction model γ'.

1. $I := \{\dot{a} \,|\, a \in \gamma\}$;
2. for each $(a \prec b) \in \pi$
3. $\quad m := b \setminus a$;
4. $\quad$ for each $c \in I$, such that $\mathbf{fire}(c) = a$;
5. $\quad\quad C := \{c\,\overline{p} \,|\, p \in m\}$;
6. $\quad\quad I := (I \setminus \{c\}) \cup C$;
7. $\gamma' := I$;

Figure 6: Algorithm transforming a glue operator in the classical semantics into a glue operator in the offer semantics.

set of behaviours. Behaviours from the first of these two classes, characterised by Property 3.10, allow a transformation without activation port typings. Finally, behaviours, characterised by Property 3.14, allow a transformation using activation port typings.

Below, we use the following notations: for $a \in 2^P$, we denote $\dot{a} \stackrel{def}{=} \{\dot{p} \,|\, p \in a\}$ and $\overline{a} \stackrel{def}{=} \{\overline{p} \,|\, p \in a\}$.

3.1 Behaviours allowing transformation of arbitrary glue

Consider a class of behaviours, which satisfy the following property:

Property 3.5. *Let L be a set of transition labels of a behaviour B. For any state q of the behaviour B, and any $a \in L \setminus \{\emptyset\}$, holds $q{\uparrow}a \Rightarrow q \stackrel{a}{\rightarrow}$.*

For any system, such that the behaviours of all its sub-systems belong to this class, any glue operator in the classical semantics can be transformed into a glue operator in the offer semantics. Applying the initial and the generated glue operators to any set of behaviours from this class would result in two equal composed systems.

Given a classical (non-extended) interaction model and a classical (non-relaxed) priority model, the algorithm in Figure 6 computes an extended interaction model, corresponding to *the same interaction and priority models considered in the offer semantics*. In particular, all interactions, generated starting from an interaction a, have the firing support $\mathbf{fire}(a') = a$, since no firing ports are added after initial generation of the set I.

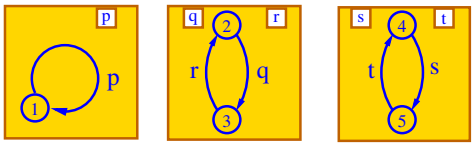
Lemma 3.6. *Let $B_i = (Q_i, P_i, \rightarrow, \uparrow)$ for $i \in [1, n]$ be a set of behaviours, all satisfying Property 3.5. Let $(Q, P, \rightarrow, \uparrow) = \pi\gamma(B_1, \ldots, B_n)$ be a composed system, with γ an interaction model and π a priority model. Then, in any state $q \in Q$, any offered interaction a in γ is active in $(Q, P, \rightarrow, \uparrow)$ if and only if it is not inhibited by a priority:*

$$\left(\nexists b \in \gamma : a \prec b \wedge q \stackrel{b}{\rightarrow}_{\gamma}\right) \Leftrightarrow q \stackrel{a}{\rightarrow}_{\pi} \ . \tag{31}$$

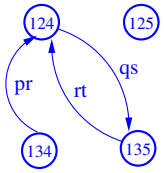
Proof. $\Leftarrow$ is straightforward by (11).

$\Rightarrow$: Since, for all $i \in [1, n]$, behaviours B_i satisfy Property 3.5, $q_i{\uparrow}(a \cap P_i) \Rightarrow q_i \stackrel{a \cap P_i}{\longrightarrow}$. Hence, by (10), for any $a \in \gamma$, $q{\uparrow}a \Rightarrow q \stackrel{a}{\rightarrow}_{\gamma}$. By (11), $\nexists b \in \gamma : a \prec b \wedge q \stackrel{b}{\rightarrow}_{\gamma}$ implies $q \stackrel{a}{\rightarrow}_{\pi}$. □

Theorem 3.7. *Let $\pi\gamma$ be a glue operator on a set of ports P in the classical semantics and let γ' be a glue operator obtained by applying the algorithm in Figure 6. Let $B_i = (Q_i, P_i, \rightarrow, \uparrow)$, for $i \in [1, n]$, be a set of behaviours, all satisfying Property 3.5, and $\bigcup_{i=1}^{n} P_i = P$. Then for $(Q, P, \rightarrow_c, \uparrow) = \pi\gamma(B_1, \ldots, B_n)$ and $(Q, P, \rightarrow_o, \uparrow) = \gamma'(B_1, \ldots, B_n)$ holds $\rightarrow_c \ = \rightarrow_o$.*

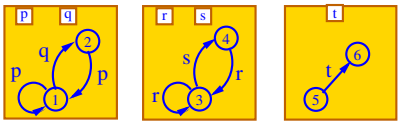


(a) First set of behaviours for Example 3.8

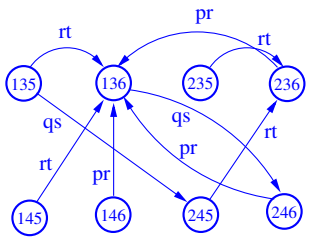


(b) Composed system from the first set of behaviours for Example 3.8

Figure 7: Second set of behaviours and composed behaviour for Example 3.8.



(a) Second set of behaviours for Example 3.8



(b) Composed system from the second set of behaviours for Example 3.8

Figure 8: Second set of behaviours and composed behaviour for Example 3.8.

Proof. 1) $q \xrightarrow{a}_o \Longrightarrow q \xrightarrow{a}_c$: Since $q \xrightarrow{a}_o$, there is an interaction $a' \in \gamma'$, having $\mathbf{fire}(a') = a$. By construction, the generation of a' started from the interaction $a \in \gamma$. Since $q \xrightarrow{a}_o$, $q \uparrow \mathbf{fire}(a')$ and $q \not\uparrow \mathbf{neg}(a')$. For any b, such that $a \prec b$, we have $b \cap \mathbf{neg}(a') \neq \emptyset$ and, consequently, $q \not\xrightarrow{b}_c$, since at least one port of b is not available. By Lemma 3.6 we have $q \xrightarrow{a}_c$.

2) $q \xrightarrow{a}_c \Longrightarrow q \xrightarrow{a}_o$: Since $q \xrightarrow{a}_c$, for all $b \in \gamma$, such that $a \prec b$, holds $q \not\xrightarrow{b}_c$. By Lemma 3.6, if all ports of b are offered at the state q, then either b is enabled or some $b' : b \prec b'$ is enabled. The priority model is a partial order and, in particular, is transitive. Hence, a has to be suppressed due to availability of b or b'. Thus, at least one port of b is not offered at the state q. Consider a subset $c \subseteq \bigcup_{a \prec b}(b \setminus a)$, such that, for all ports $p \in c$, we have $q \not\uparrow p$ and, for all $a \prec b$, the set $c \cap (b \setminus a)$ contains exactly one port, i.e. $|c \cap (b \setminus a)| = 1$. By construction of γ', we have $a' = \dot{a} \cup \overline{c} \in \gamma'$. Thus, $\mathbf{fire}(a') = a$ and, for all $p \in \mathbf{neg}(a')$, $q \not\uparrow p$. Hence by Definition 2.17 $q \xrightarrow{a}_o$. □

Example 3.8. Consider $\gamma = \{pr, qs, rt\}$ and $\pi = \{pr \prec qs, pr \prec rt\}$ in the classical semantics. The algorithm in Figure 6 generates an extended interaction model in the offer semantics. In the first step the set $I = \{\dot{pr}, \dot{qt}, \dot{rt}\}$. Considering the first priority rule $pr \prec qs$ we have $m = qs \setminus pr = qs$. For each interaction in I with firing support pr we generate a set of new interactions, thus from the interaction $\dot{pr}$ we obtain a pair of interactions $\dot{pr}\overline{q}$ and $\dot{pr}\overline{s}$. The new set $I = \{\dot{pr}\overline{q}, \dot{pr}\overline{s}, \dot{qt}, \dot{rt}\}$. For the second priority rule $pr \prec rt$ we have $m = rt \setminus pr = t$. There are two interactions in I with firing support pr: $\dot{pr}\overline{q}$ and $\dot{pr}\overline{s}$. The algorithm adds $\overline{t}$ to both of them and the final glue in the offer semantics is $\gamma' = \{\dot{pr}\overline{q}\,\overline{t}, \dot{pr}\overline{s}\,\overline{t}, \dot{qt}, \dot{rt}\}$.

Consider behaviours in Figure 7a. All of them satisfy Property 3.5. Applying glues in the classical and in the offer semantics we obtain equal composed behaviours shown in Figure 7b. Transitions qs and rt are available in the states 124 and 135 respectively in both composed behaviours. Ports p and r are offered in the states 134 and 135. The priority rule $pr \prec rt$ forbids the transition pr in the state 135, thus in the system in the classical semantics this transition is

available only in the state 134. In the system in the offer semantics in the state 134 ports q and t are not offered, thus the interaction $\dot{p}\dot{r}\overline{q}\,\overline{t}$ allows a transition from this state, whereas in the state 135 t is offered and none of the interactions allows a transition pr from this state.

Consider behaviours in Figure 8a and the same glue. All of them also satisfy Property 3.5. Applying glues in the classical and in the offer semantics we obtain equal composed behaviours shown in Figure 8b. Transitions qs and rt are available simultaneously in both composed behaviours, since they depend only on availability of the corresponding ports. There are transitions qs from states 135 and 136, transitions rt from states 135, 145, 235 and 245. Ports p and r are offered in all states. The priority rules forbids transition pr from all states where qs or rt are available, thus there are transitions pr from states 146, 236 and 246. In the system in the offer semantics the interaction $\dot{p}\dot{r}\overline{q}\,\overline{t}$ allows transitions pr from states 236 and 246, the interaction $\dot{p}\dot{r}\overline{s}\,\overline{t}$ allows transitions pr from the state 146. □

Theorem 3.9. *Let $B_i = (Q_i, P_i, \rightarrow, \uparrow)$ for $i \in [1, n]$ and $n \geq 2$ be a set of behaviours, such that at least one of them violates Property 3.5, and let $P = \bigcup_{i=1}^{n} P_i$. There exists $\pi\gamma$, a glue operator on P in the classical semantics, such that for the glue γ' computed by the algorithm in Figure 6, and for $(Q, P, \rightarrow_c, \uparrow) = \pi\gamma(B_1, \ldots, B_n)$ and $(Q, P, \rightarrow_o, \uparrow) = \gamma'(B_1, \ldots, B_n)$ holds $\rightarrow_c \; \neq \rightarrow_o$.*

Proof. Without loss of generality, we assume that B_1 violates Property 3.5. Thus there is a state q_1 and a transition a, such that $q_1 \uparrow a$ and $q \stackrel{a}{\not\rightarrow}$. Let b be a transition from a state q_2 in B_2. Let $\gamma = \{a, b\}$ and $\pi = \{b \prec a\}$. The algorithm in Figure 6 computes $\gamma' = \{\dot{a}\} \cup \{\dot{b}\,\overline{p} \mid p \in a \setminus b\}$. The transition labelled b is available in the state $q_1 \ldots q_n$ of $(Q, P, \rightarrow_c)$, but it is not available in the state $q_1 \ldots q_n$ of $(Q, P, \rightarrow_o, \uparrow)$, since all ports of a are offered in this state. □

3.2 Behaviours allowing glue transformation without using activation port typings

Let us now consider the class of behaviours, satisfying the following property:

Property 3.10. *Let L be a set of transition labels of B. For any state q, such that $S_q \stackrel{def}{=} \{a \in L \setminus \{\emptyset\} \mid q \uparrow a \wedge q \stackrel{a}{\not\longrightarrow}\} \neq \emptyset$, the following holds: for any state $q' \neq q$, such that $\exists\, a \in S_q : q' \stackrel{a}{\rightarrow}$, there exists a port p, such that $q' \uparrow p$ and $q \not\uparrow p$.*

Intuitively, this property means the following. Assume there is a state that offers an interaction a, which does not correspond to an enabled transition, e.g. a is a proper subset of a label of an enabled transition. Assume, furthermore, that there is also a state that actually has an enabled transition labelled by a. Then Property 3.10 requires that these two states be distinguishable by considering whether some other port p is offered or not.

If the Property 3.10 holds for a set of behaviours, we can build a composed system in the offer semantics without using activation port typings. Let $B_i = (Q_i, P_i, \rightarrow, \uparrow)$ for $i \in [1, n]$ be a set of behaviours, let $P = \bigcup_{i=1}^{n} P_i$ and let $\pi\gamma$ be a glue operator on P in the classical semantics. We start the transformation by applying the algorithm in Figure 6. This algorithm generates an extended interaction model γ'' in the offer semantics. However, this property is weaker than Property 3.5 and composed behaviours $\pi\gamma(B_1, \ldots, B_n)$ and $\gamma''(B_1, \ldots, B_n)$ can be not equal. A transition relation of the former composed behaviour can contain transitions, which are not present in the latter one. For each such transition a from the state q we add the interaction $\dot{a}\overline{b}$ to the interaction model, where $b = \{p \mid q \not\uparrow p\}$. The application of the final extended interaction model γ' results in an equivalent composed system.

Theorem 3.11. *Let $B_i = (Q_i, P_i, \rightarrow, \uparrow)$ for $i \in [1, n]$ be a set of behaviours, such that all of them satisfy Property 3.10 and let $P = \bigcup_{i=1}^{n} P_i$. Let $\pi\gamma$ be a glue operator on P in classical*

(a) Set of behaviours for Example 3.12

(b) Composed system for Example 3.12

Figure 9: Behaviours and composed behaviour for Example 3.12

semantics. Let γ' be the extended interaction model generated in the way it was shown above. Then for $(Q, P, \rightarrow_c, \uparrow) = \pi\gamma(B_1, \ldots, B_n)$ and $(Q, P, \rightarrow_o, \uparrow) = \gamma'(B_1, \ldots, B_n)$ holds $\rightarrow_c = \rightarrow_o$.

Proof. 1) $q \xrightarrow{a}_o \implies q \xrightarrow{a}_c$: By construction if $q \xrightarrow{a}_o$ then there exists an interaction $a' \in \gamma'$, such that $\mathbf{fire}(a') = a$ and $\mathbf{neg}(a') = \{p \mid q \not\uparrow p\}$. There are two possibilities for the moment, when a' was added to γ'. Let γ'' be the extended interaction model computed by the algorithm in Figure 6.

If $a' \in \gamma''$, the proof is similar to the one in Theorem 3.7. By construction the generation of a' started from interaction $a \in \gamma$. Since $q \xrightarrow{a}_o$, we have $q \uparrow a$ and $q \not\uparrow \mathbf{neg}(a')$. For any b, such that $a \prec b$ we have $b \cap \mathbf{neg}(a') \neq \emptyset$ and, consequently, $q \not\xrightarrow{b}_c$, as at least one port of b is not available. Thus $q \xrightarrow{a}_c$.

If a' was added during the second step of the computation above, a' could have been added to γ' only if there is a state $q' = q'_1 \ldots q'_n$ in the composed system $(Q, P, \rightarrow_c, \uparrow)$, such that $q' \xrightarrow{a}_c$ and $\mathbf{neg}(a') = \{p \mid q' \not\uparrow p\}$. Assume $q \xrightarrow{a}_c$. Since $q \xrightarrow{a}_o$, we have $q \uparrow a$. Thus, a was forbidden by the application of some priority rule $a \prec b$ and $q \xrightarrow{c} b$. Since $q' \xrightarrow{a}_c$, we also have $q' \not\xrightarrow{b}_c$. If $q' \not\uparrow b$ then there exists $p \in b$, such that $q' \not\uparrow p$, so $\overline{p} \in a'$ and $q \not\xrightarrow{a}_o$. If $q' \uparrow b$ then there exists a behaviour B_i, such that $q'_i \uparrow (b \cap P_i)$ and $q'_i \not\xrightarrow{b \cap P_i}$. Let $q = q_1 \ldots q_n$ then $q_i \uparrow (b \cap P_i)$ and, by Property 3.10, there exists a port p, such that $q'_i \not\uparrow p$ and $q_i \uparrow p$. Consequently, $q' \not\uparrow p$ and $q \uparrow p$, so $\overline{p} \in a'$ and $q \not\xrightarrow{a}_o$, which contradicts the assumption $q \not\xrightarrow{a}_c$.

2) $q \xrightarrow{a}_c \implies q \xrightarrow{a}_o$: By construction of γ', either the extended interaction model γ'', obtained after the application of the algorithm in Figure 6 contains an interaction, which generates this transition in the composed system, or an additional interaction is added to γ', such that this transition is present in the composed system. □

Example 3.12. Consider behaviours in Figure 9a and a glue operator in the classical semantics with $\gamma = \{p, pq, q, rt, s\}$ and $\pi = \{s \prec p\}$. Both behaviours satisfy Property 3.10. The composed system in the classical semantics is shown in Figure 9b. The algorithm in Figure 6 generates the extended interaction model $\gamma'' = \{\dot{p}, \dot{p}\dot{q}, \dot{q}, \dot{r}\dot{t}, \dot{s}\overline{p}\}$. If we apply γ'' to the behaviours in Figure 9a the composed behaviour would not contain a transition s from the state 14 (dashed in Figure 9b), as port p is offered at this state. Thus we need to add an interaction to the extended interaction model, such that this transition becomes available, but no other transitions are added. The interaction $\dot{s}\overline{r}$ adds the transition s from the state 14 in the composed behaviour and it does not add a transition from the state 24, since r is offered at the state 24. Thus, the final extended interaction model is $\gamma' = \{\dot{p}, \dot{p}\dot{q}, \dot{q}, \dot{r}\dot{t}, \dot{s}\overline{p}, \dot{s}\overline{r}\}$. □

Theorem 3.13. *For any set of behaviours $B_i = (Q_i, P_i, \rightarrow, \uparrow)$ for $i \in [1, n]$ and $n \geq 2$, such that at least one of B_i violates Property 3.10, there exists a glue operator $\pi\gamma$ on $P = \bigcup_{i=1}^{n} P_i$ in the*

classical semantics, such that the composed system $\pi\gamma(B_1, \ldots, B_n)$ cannot be expressed through the offer semantics without using the activation port typings.

Proof. Without loss of generality assume that B_1 violates Property 3.10. Thus, there exists a state q_1 and a transition a, such that $q_1 \uparrow a$ and $q_1 \overset{a}{\not\rightarrow}$ and there exists a state q_1', such that $q_1' \overset{a}{\rightarrow}$ and all ports which are not offered in q_1 are also not offered in q_1'.

Let b be some transition label of the behaviour B_2, and let q_2 be a state, such that $q_2 \overset{b}{\rightarrow}$. Let $\gamma = \{a, b\}$ and $\pi = \{b \prec a\}$. In a composed system $\pi\gamma(B_1, \ldots, B_n)$, a transition labeled by b is available from the state $q_1 q_2 \ldots q_n$, but not available from the state $q_1' q_2 \ldots q_n$.

Assume there exists an extended interaction model γ' without activation port typings, such that $\gamma'(B_1, \ldots, B_n) = \pi\gamma(B_1, \ldots, B_n)$. In order to have a transition labelled by b from the state $q_1 \ldots q_n$, γ' has to contain the interaction $\dot{b}\bar{c}$, where $c \subseteq \{p \mid q_1 \not\uparrow p\}$. However this interaction allows a transition b from the state $q_1' q_2 \ldots q_n$, which contradicts the assumption of the existence of γ'. □

3.3 Behaviours allowing glue transformation using witness ports

We now consider the third class of behaviours, characterised by the following property:

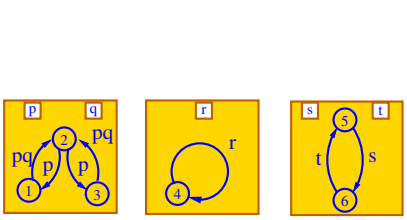
Property 3.14. *There are no two states q_1, q_2 in the behaviour, such that $\{p \mid q_1 \uparrow p\} = \{p \mid q_2 \uparrow p\}$ and $\{a \mid q_1 \overset{a}{\rightarrow}\} \setminus \{\emptyset\} \neq \{a \mid q_2 \overset{a}{\rightarrow}\} \setminus \{\emptyset\}$.*

A composed system in the offer semantics can be built with the following idea. Let $B_i = (Q_i, P_i, \rightarrow, \uparrow)$ for $i \in [1, n]$ be a set of behaviours. To each state q of each behaviour, we associate a set $\chi(q) = \{p \mid q \uparrow p\} \cup \{\overline{p} \mid q \not\uparrow p\}$. Notice, that since sets of ports of behaviours are pairwise disjoint, $\chi(q)$ and $\chi(q')$ are disjoint if q and q' are states of different behaviours. Let $\pi\gamma$ be a glue operator on P in the classical semantics. Let $(Q, P, \rightarrow, \uparrow) = \pi\gamma(B_1, \ldots, B_n)$ be the corresponding composed system. To each state $q_1 \ldots q_n$ of the composed system, we associate a set $\chi(q_1 \ldots q_n) = \bigcup_{i=1}^{n} \chi(q_i)$. For each transition $q_1 \ldots q_n \overset{a}{\rightarrow} q_1' \ldots q_n'$ of the composed system, we generate the interaction $\{\dot{p} \mid p \in a\} \cup \{p \mid p \in \chi(q_1 \ldots q_n), p \notin a\}$. Since $q_1 \ldots q_n \overset{a}{\rightarrow}$, all ports of a are offered in $q_1 \ldots q_n$. The set of all these interactions forms an extended interaction model γ', such that if all $B_1, \ldots, B_n$ satisfy Property 3.14, then $\gamma'(B_1, \ldots, B_n)$ is equivalent to $(Q, P, \rightarrow, \uparrow)$. Notice, that for any interaction $a \in \gamma'$, $\mathbf{fire}(a) \cup \mathbf{act}(a) \cup \mathbf{neg}(a) = P$.

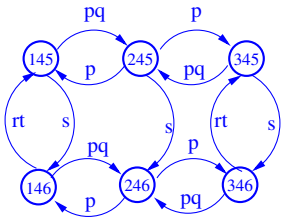
Theorem 3.15. *Let $B_i = (Q_i, P_i, \rightarrow, \uparrow)$, for $i \in [1, n]$, be a set of behaviours, such that all of them satisfy Property 3.14 and let $P = \bigcup_{i=1}^{n} P_i$. Let $\pi\gamma$ be a glue operator on P in the classical semantics. Let γ' be the extended interaction model obtained by the application of the algorithm described above. Then, for composed behaviours $(Q, P, \rightarrow_c, \uparrow) = \pi\gamma(B_1, \ldots, B_n)$ and $(Q, P, \rightarrow_o, \uparrow) = \gamma'(B_1, \ldots, B_n)$, holds $\rightarrow_c = \rightarrow_o$.*

Proof. 1) $\underline{q \overset{a}{\rightarrow}_o \Longrightarrow q \overset{a}{\rightarrow}_c}$: By construction if $q \overset{a}{\rightarrow}_o$ then there is an interaction $a' \in \gamma'$, such that $\mathbf{fire}(a') = a$, $\mathbf{fire}(a') \cup \mathbf{act}(a') = \{p \mid q \uparrow p\}$ and $\mathbf{neg}(a') = \{p | q \not\uparrow p\}$. Assume $q \overset{a}{\not\rightarrow}_c$. Since $a' \in \gamma'$, there is a state q', such that $q' \overset{a}{\rightarrow}_c$, $\mathbf{fire}(a') = a$, $\mathbf{fire}(a') \cup \mathbf{act}(a') = \{p | q' \uparrow p\}$ and $\mathbf{neg}(a') = \{p | q' \not\uparrow p\}$. Thus $a \in \gamma$. Let $q = q_1 \ldots q_n$ and $q' = q_1' \ldots q_n'$. Since $\{p | q \uparrow p\} = \{p | q' \uparrow p\}$, we have $\{p | q_i \uparrow p\} = \{p | q_i' \uparrow p\}$, for all $i \in [1, n]$. By Property 3.14, for any $i \in [1, n]$, holds $\{b | q_i \overset{b}{\rightarrow}\} = \{b | q_i' \overset{b}{\rightarrow}\}$. Thus $\{b | q \overset{b}{\rightarrow}\} = \{b | q' \overset{b}{\rightarrow}\}$ in $\gamma(B_1, \ldots, B_n)$ and consequently $\{b | q \overset{b}{\rightarrow}\} = \{b | q' \overset{b}{\rightarrow}\}$ in $(Q, P, \rightarrow_c, \uparrow)$. Since $q' \overset{a}{\rightarrow}_c$, we also have $q \overset{a}{\rightarrow}_c$.

2) $\underline{q \overset{a}{\rightarrow}_c \Longrightarrow q \overset{a}{\rightarrow}_o}$: By construction, if $q \overset{a}{\rightarrow}_c$ then the interaction $a' = \{\dot{p} | p \in a\} \cup \{p | p \in \chi(q), p \notin a\} \in \gamma'$. Since all ports from $\mathbf{fire}(a')$ are available at state q, all ports from $\mathbf{act}(a')$ are offered and all ports from $\mathbf{neg}(a')$ are inhibited, thus $q \overset{\mathbf{fire}(a')}{\longrightarrow}_o$. □



(a) Set of behaviours for Example 3.16



(b) Composed system for Example 3.16

Figure 10: Behaviours and composed behaviour for Example 3.16

Example 3.16. Consider the behaviours in Figure 10a and the glue operator in the classical semantics defined by $\gamma = \{p, pq, rt, s\}$ and $\pi = \{rt \prec p\}$. Both behaviours satisfy Property 3.14. The composed system in the classical semantics is shown in Figure 10b. This system cannot be expressed in the offer semantics without activation port typings. There should be a transition rt from the state 146, but any interaction allowing it also allows interaction rt from the state 246, as all ports, which are not offered in the state 146, are also not offered in the state 246.

In order to transform the system into one in the offer semantics we associate a set χ to each state as follows:

$$\chi(145) = \{p, q, r, s, \overline{t}\}, \quad \chi(245) = \{p, \overline{q}, r, s, \overline{t}\}, \quad \chi(345) = \{p, q, r, s, \overline{t}\},$$
$$\chi(146) = \{p, q, r, \overline{s}, t\}, \quad \chi(246) = \{p, \overline{q}, r, \overline{s}, t\}, \quad \chi(346) = \{p, q, r, \overline{s}, t\}.$$

Now, we start generating γ', considering all transition labels in the composed system. From the state 145 there are transitions pq and s, thus we take interactions $\dot{p}\,\dot{q}\,r\,s\,\overline{t}$ and $\dot{s}\,p\,q\,r\,\overline{t}$. From the state 146 there are transitions pq and rt, hence we add interactions $\dot{p}\,\dot{q}\,r\,\overline{s}\,t$ and $\dot{r}\,\dot{t}\,p\,q\,\overline{s}$. Proceeding similarly for the remaining states, we obtain γ':

$$\{\dot{p}\,\dot{q}\,r\,s\,\overline{t}, \quad \dot{s}\,p\,q\,r\,\overline{t}, \quad \dot{p}\,\dot{q}\,r\,\overline{s}\,t, \quad \dot{r}\,\dot{t}\,p\,q\,\overline{s}, \quad \dot{p}\,r\,s\,\overline{q}\,\overline{t}, \quad \dot{p}\,r\,t\,\overline{q}\,\overline{s}, \quad \dot{s}\,p\,r\,\overline{q}\,\overline{t}\}.$$

Noticing, that s and t are mutually exclusive and p, r are offered in all states, this extended interaction model can be simplified to $\gamma'' = \{\dot{p}\,\dot{q}, \dot{p}\,\overline{q}, \dot{s}, \dot{r}\,\dot{t}\,q\}$. □

Theorem 3.17. *Let $B_i = (Q_i, P_i, \to, \uparrow)$, for $i \in [1, n]$ and $n \geq 2$, be a set of behaviours, such that at least one of them violates Property 3.14, and let $P = \bigcup_{i=1}^{n} P_i$. There exists a glue operator in the classical semantics with an interaction model γ and a priority model π, such that the system $\pi\gamma(B_1, \ldots, B_n)$ cannot be expressed in the offer semantics.*

Proof. Without loss of generality assume that B_1 violates Property 3.14. Thus there is a pair of states q_1, q_1', such that $\{p | q_1 \uparrow p\} = \{p | q_1' \uparrow p\}$ and $q_1 \xrightarrow{a}$, while $q_1' \not\xrightarrow{a}$. Let b be a transition from a state q_2 in B_2. Let $\gamma = \{a, b\}$ and $\pi = \{b \prec a\}$. A composed system $\pi\gamma(B_1, \ldots, B_n)$ cannot be expressed in the offer semantics.

Consider two states $q_1 q_2 \ldots q_n$ and $q_1' q_2 \ldots q_n$. In the system $\pi\gamma(B_1, \ldots, B_n)$, transition b is forbidden from the first state by the priority rule, but b is allowed from the second state. However, sets of offered ports from both states are equal. Thus, for any interaction b', such that $\mathbf{fire}(b') = b$, either b' can be allowed from both states or it is forbidden from both states. These states cannot be distinguished in the offer semantics, which implies that the system $\pi\gamma(B_1, \ldots, B_n)$ cannot be expressed in the offer semantics. □

3.4 Hierarchical systems

In hierarchical systems, glue operators can be applied not only to atomic behaviours, but also to composed ones. If we consider behaviours that satisfy Property 3.10 or Property 3.14, application

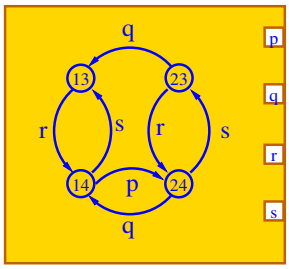


Figure 11: Composed behaviour of a hierarchical system from Example 3.20.

of any glue operator results in a composed behaviour of the same class.

Proposition 3.18. *Let $B_i = (Q_i, P_i, \to, \uparrow)$, for $i \in [1, n]$, be a set of behaviours, such that all of them satisfy Property 3.10 and let $P = \bigcup_{i=1}^{n} P_i$. Then, for any interaction model γ on P, and for any priority model π on P, the composed behaviour $\pi\gamma(B_1, \ldots, B_n)$ satisfies Property 3.10.*

Proof. Assume $\pi\gamma(B_1, \ldots, B_n)$ violates this property. There are two states $q = q_1 \ldots q_n$, $q' = q'_1 \ldots q'_n$ and a transition a, such that $q{\uparrow}a$, $q \not\xrightarrow{a}$, $q' \xrightarrow{a}$ and there exists no port p, such that $q'{\uparrow}p$ and $q \not\uparrow p$. Since $q \not\xrightarrow{a}$, for some $i \in [1, n]$, $q_i{\uparrow}a \cap P_i$ and $q_i \not\xrightarrow{a \cap P_i}$. Since B_i satisfies Property 3.10 and $q'_i \xrightarrow{a \cap P_i}$, there exists p, such that $q_i \not\uparrow p$ and $q'_i{\uparrow}p$. By Definition 2.17, $q \not\uparrow p$ and $q'{\uparrow}p$. Thus, states q and q' cannot violate Property 3.10. □

Proposition 3.19. *Let $B_i = (Q_i, P_i, \to, \uparrow)$, for $i \in [1, n]$, be a set of behaviours, such that all of them satisfy Property 3.14, and let $P = \bigcup_{i=1}^{n} P_i$. Then for any interaction model γ on P and for any priority model π on P the composed behaviour $\pi\gamma(B_1, \ldots, B_n)$ satisfies Property 3.14.*

Proof. Assume $\pi\gamma(B_1, \ldots, B_n)$ violates this property. Thus, there are two states $q = q_1 \ldots q_n$ and $q' = q'_1 \ldots q'_n$, such that $\{p|q{\uparrow}p\} = \{p|q'{\uparrow}p\}$ and $\{a|q \xrightarrow{a}\} \neq \{a|q' \xrightarrow{a}\}$. By Definition 2.17, we can deduce that, for $i \in [1, n]$, holds $\{p|q_i{\uparrow}p\} = \{p|q'_i{\uparrow}p\}$. Since all $B_1, \ldots, B_n$ satisfy Property 3.14, we have $\{a|q_i \xrightarrow{a}\} = \{a|q'_i \xrightarrow{a}\}$, for $i \in [1, n]$. Consequently, if only the interaction model is applied, $\{a|q \xrightarrow{a}\} = \{a|q' \xrightarrow{a}\}$ by (10). The priority model can only remove transitions from these states simultaneously, as the sets of outgoing transitions are equal. Thus, states q and q' cannot violate Property 3.14. □

Propositions 3.18 and 3.19 show that composition of behaviours with glue operators in the classical semantics preserves, respectively, Properties 3.10 and 3.14. Therefore, for an application of a hierarchical glue operator in the classical semantics to a set of atomic components, all satisfying the same property (recall that Property 3.10 implies Property 3.14), we can iteratively construct the corresponding operator in the offer semantics. We start at the lowest level of operator hierarchy, i.e. from the atomic behaviours. Since atomic behaviours satisfy one of the properties, the corresponding glue operator in the offer semantics can be generated and the composed behaviour also satisfies the property. Repeating this reasoning for the operators on higher levels, we obtain the corresponding hierarchy of glues in the offer semantics.

In general, glue operators do not preserve Property 3.5.

Example 3.20. Recall Example 2.9. All behaviours in Figure 2a satisfy Property 3.5. Applying the glue operator defined, in the classical semantics, by the interaction model $\gamma = \{p, q, r, s\}$ and the priority model $\pi = \{p \prec r\}$ to B_1 and B_2, results in the composed behaviour shown in Figure 11. In the state 13, p is offered, since it is offered by B_1, however it is forbidden by the

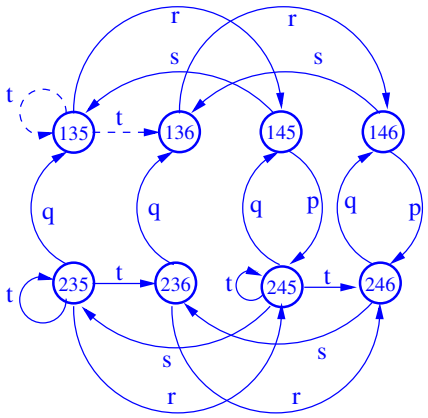


Figure 12: Composed behaviour of a hierarchical system from Example 3.20.

priority model. If we consider a glue operator on the next level of hierarchy with $\gamma = \{p, q, r, s, t\}$ and $\pi = \{t \prec p\}$ (notice that this is a different glue operator from the one used in the top level of Example 2.9) the application of the algorithm in Figure 6 will generate an incorrect extended interaction model (see the next example). □

Note 3.21. Notice that Property 3.5 is preserved by glue operators in hierarchical systems where all priority models are applied after all interaction models.

For any behaviour, Property 3.5 implies Property 3.10. Thus, hierarchical systems with atomic behaviours that satisfy Property 3.5 can always be transformed into offer semantics, but the algorithm in Figure 6 can only be applied to the lowest level of hierarchy.

Example 3.22. Consider behaviours and glue operators from the previous example. For the first level of hierarchy the algorithm in Figure 6 generates the extended interaction model $\gamma_1' = \{\dot{p}\overline{r}, \dot{q}, \dot{r}, \dot{s}\}$. The composed behaviour B (Figure 11) does not satisfy Property 3.5, but it satisfies Property 3.10. Thus, the extended interaction model on the second level of hierarchy can be generated. If we consider behaviours B (Figure 11) and B_3 (Figure 2a) and a glue operator on the next level of hierarchy defined by $\gamma = \{p, q, r, s, t\}$ and $\pi = \{t \prec p\}$ the application of the algorithm in Figure 6 will generate an incorrect extended interaction model. The final composed system in the classical semantics is shown in Figure 12. The algorithm in Figure 6 generates the extended interaction model $\gamma_2'' = \{\dot{p}, \dot{q}, \dot{r}, \dot{s}, \dot{t}\,\overline{p}\}$. Two transitions t from the state 135 (dashed in Figure 12) are present in the system in the classical semantics, as interaction p is not available in this state, however p is offered at this state and interaction $\dot{t}\,\overline{p}$ does not allow transitions from this state. The extended interaction model has to be enlarged with the interaction $\dot{t}\,\overline{q}\,\overline{s}$, which allows transitions t from the state 135 and does not add transitions from the state 145, as s is offered at that state. Thus, the hierarchy of extended interaction models $\gamma_1' = \{\dot{p}\,\overline{r}, \dot{q}, \dot{r}, \dot{s}\}$ and $\gamma_2' = \{\dot{p}, \dot{q}, \dot{r}, \dot{s}, \dot{t}\,\overline{p}, \dot{t}\,\overline{q}\,\overline{s}\}$ generates the equivalent composed system. □

If we consider a hierarchical system in the classical semantics, such that all atomic behaviours satisfy Property 3.14, it is possible to build an equivalent flat system in the offer semantics directly. Since extended interaction model generation depends only on the composed system, the extended interaction model generated for the highest level of hierarchy can be applied directly to the set of atomic behaviours, from which we start building the hierarchical system. Thus, instead of considering each level of hierarchy separately, we can start from the final composed system and, based on the knowledge of its states, generate a flat glue operator in the offer semantics.

4 Representations of the interaction model

In the remainder of the paper, we adapt the existing algebraic theory of representations of interaction models to the semantics based on the offer predicate. Indeed, as it was shown in Section 2.4, this semantics allows us to encode priorities as an extension to the interaction models, by using additional port typings. Hence, the existing algebraic theory can also be adapted to encompass priorities in the offer-based semantics. In the following sections, we rely on these algebra extensions to define transformations from glue operators into Boolean formulas and vice-versa, which, in particular, allow the synthesis of connectors from Boolean state properties.

In this section, we briefly recall the syntax and semantics of the algebras used to represent BIP interaction models. All the algebras are parameterised by a set P of all the ports in a given system. The semantics of the Algebra of Interactions is given in terms of sets of interactions by a function $\|\cdot\| : \mathcal{AI}(P) \to 2^{2^P}$. The corresponding equivalence relation on $\mathcal{AI}(P)$ is defined as follows: two terms $x, y \in \mathcal{AI}(P)$ are *equivalent* $x \simeq y$ iff $\|x\| = \|y\|$. For any other algebra, $\mathcal{A}(P)$, among those appearing in the paper, we define its semantics by the function $|\cdot| : \mathcal{A}(P) \to \mathcal{AI}(P)$. A function $\|\cdot\| : \mathcal{A}(P) \to 2^{2^P}$ is obtained by composing $|\cdot| : \mathcal{A}(P) \to \mathcal{AI}(P)$ and $\|\cdot\| : \mathcal{AI}(P) \to 2^{2^P}$. The axiomatisation of $\mathcal{AI}(P)$ given in [7] is sound and complete with respect to $\simeq$. Hence, for other algebras, the equivalences induced by $\|\cdot\|$ and $|\cdot|$ coincide.

Below, we assume that a set of ports P is given, such that $0, 1 \notin P$.

For any set X of propositional variables, we denote by $\mathbb{B}[X]$ the corresponding Boolean algebra generated by X. For presentation clarity, we will often omit the conjunction operator and write $a \vee bc$ instead of $a \vee (b \wedge c)$.

4.1 Algebra of Interactions

The Algebra of Interactions is used to define the interaction semantics of other algebras. The elements of this algebra can be bijectively mapped to interaction models, i.e. subsets of 2^P.

Syntax. The syntax of the *Algebra of Interactions*, $\mathcal{AI}(P)$, is defined by the following grammar

$$x \quad ::= \quad 0 \mid 1 \mid p \in P \mid x \cdot x \mid x + x\,, \tag{32}$$

where '$+$' and '$\cdot$' are binary operators, respectively called *union* and *synchronisation*. Synchronisation binds stronger than union.

As follows from the interaction semantics given below, the additive identity element 0 represents blocking, since it does not authorise any interaction. The multiplicative identity element 1 corresponds to the empty interaction, which represents idling (see the discussion after Definition 2.1).

Semantics. The semantics of $\mathcal{AI}(P)$ is given by the function $\|\cdot\| : \mathcal{AI}(P) \to 2^{2^P}$, defined by

$$\begin{array}{lcl} \|0\| \;=\; \emptyset, \quad \|1\| \;=\; \{\emptyset\}, \quad \|p\| \;=\; \Big\{\{p\}\Big\}, \\ \|x_1 + x_2\| & = & \|x_1\| \cup \|x_2\|, \\ \|x_1 \cdot x_2\| & = & \Big\{a_1 \cup a_2 \;\Big|\; a_1 \in \|x_1\|, a_2 \in \|x_2\|\Big\}, \end{array} \tag{33}$$

for $p \in P$, $x_1, x_2 \in \mathcal{AI}(P)$. Terms of $\mathcal{AI}(P)$ represent sets of interactions between the ports P.

Sound and complete axiomatisation of $\mathcal{AI}(P)$ with respect to the semantic equivalence is provided in [7]. In a nutshell, $(\mathcal{AI}(P), +, \cdot, 0, 1)$ is a commutative semi-ring idempotent in both $+$ and $\cdot$.

4.2 Algebra of Connectors

The Algebra of Connectors provides an algebraic formalisation for structuring the interaction models. It underlies the graphical notation (e.g. Figure 13) and the syntax for connectors used in the BIP language.

Syntax. The syntax of the *Algebra of Connectors*, $\mathcal{AC}(P)$, is defined by the following grammar

$$\begin{array}{rcll} s & ::= & [0] \mid [1] \mid [p] \mid [x] & (\mathit{synchrons}) \\ t & ::= & [0]' \mid [1]' \mid [p]' \mid [x]' & (\mathit{triggers}) \\ x & ::= & s \mid t \mid x \cdot x \mid x + x\,, & \end{array} \tag{34}$$

for $p \in P$, and where '$+$' is a binary operator called *union*, '$\cdot$' is a binary operator called *fusion*, and brackets '$[\cdot]$' and '$[\cdot]'$' are unary *typing* operators. Fusion binds stronger than union.

Union has the same meaning as union in $\mathcal{AI}(P)$. Fusion is a generalisation of the synchronisation in $\mathcal{AI}(P)$. Typing is used to form typed connectors: '$[\cdot]$' defines *synchrons* (need synchronisation with other ports in order to interact) and '$[\cdot]'$' defines *triggers* (can initiate an interaction).

In order to simplify notation, we will omit brackets on 0, 1, and ports $p \in P$, as well as '$\cdot$' for the fusion operation.

Definition 4.1. In a system with a set of ports P, connectors are elements of $\mathcal{AC}(P)$.

The operations of the Algebra of Connectors satisfy the following axioms.

- Union '$+$' is associative, commutative, idempotent and has the identity element 0.
- Fusion '$\cdot$' is associative, commutative and has the identity element 1. It is idempotent on monomial connectors, i.e., for any $x \in \mathcal{AC}(P)$, not involving the union operation, we have $x \cdot x = x$.
- Typing '$[\cdot]^*$' satisfies the following axioms, for $x, y, z \in \mathcal{AC}(P)$ and $[\cdot]^\alpha, [\cdot]^\beta \in \{[\cdot]', [\cdot]\}$ arbitrary typings (trigger or synchron):
  1. $[0] = [0]'$,
  2. $[[x]^\alpha]^\beta = [x]^\beta$,
  3. $[x + y]^\alpha = [x]^\alpha + [y]^\alpha$,
  4. $[x]'[y]' = [x]'[y] + [x][y]'$.

Complete axiomatisation of $\mathcal{AC}(P)$ with respect to the semantic equivalence is provided in [8].

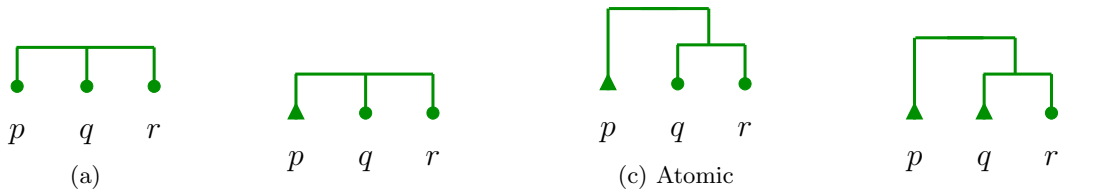
Semantics. The semantics of $\mathcal{AC}(P)$ is given by the function $|\cdot| : \mathcal{AC}(P) \to \mathcal{AI}(P)$ (we use the $\sum$ and $\prod$ notation for, respectively, the union and fusion of multiple terms of $\mathcal{AC}(P)$):

$$|[p]| = p\,, \qquad |x_1 + x_2| = |x_1| + |x_2|\,, \qquad \left|\prod_{i=1}^{n} [x_i]\right| = \prod_{i=1}^{n} |x_k|\,, \tag{35}$$

$$\left|\prod_{i=1}^{n} [x_i]' \prod_{j=1}^{m} [y_j]\right| = \sum_{i=1}^{n} |x_i| \left(\prod_{k \neq i} \Big(1 + |x_k|\Big) \prod_{j=1}^{m} \Big(1 + |y_j|\Big)\right) \tag{36}$$

for $n > 0$, $m \geq 0$, $x_1, \ldots, x_n, y_1, \ldots, y_m \in \mathcal{AC}(P)$ and $p \in P \cup \{0, 1\}$.

Figure 13 shows four basic examples of the graphical representation of connectors. Triggers are denoted by triangles, whereas synchrons are denoted by bullets. The interaction semantics of the four connectors is given in the sub-figure captions.



(a) Rendezvous pqr $\|pqr\| = \{pqr\}$

(b) Broadcast $p'qr$ $\|p'qr\| = \{p, pq, pr, pqr\}$

(c) Atomic broadcast $p'[qr]$ $\|p'[qr]\| = \{p, pqr\}$

(d) Causal chain $p'[q'r]$ $\|p'[q'r]\| = \{p, pq, pqr\}$

Figure 13: Basic connector examples

4.3 Algebra of Causal Interaction Trees

The Algebra of Causal Interaction Trees serves as pivot for transformations between all other algebraic representations. It makes explicit the causal dependencies between ports contributing to the interactions defined by a connector. In particular, this allows efficient computation of the Boolean representation for connectors and, conversely, the synthesis of connectors from Boolean formulas.

Syntax. The syntax of the *Algebra of Causal Interaction Trees*, $\mathcal{T}(P)$, is given by

$$t ::= a \,|\, a \to t \,|\, t \oplus t\,, \tag{37}$$

where $a \in \mathcal{AI}(P)$ is an interaction, i.e. 0, 1 or a synchronisation of ports (without the use of the union operator), and '$\to$' and '$\oplus$' are respectively the *causality* and the *parallel composition* operators. Causality binds stronger than parallel composition. Notice that a causal interaction tree can have several roots.

"Atomic" strongly synchronised interactions in the nodes of a causal interaction tree are the building blocks for the interactions provided by the connector. The causality operator defines a dependency between two interactions: $a \to b$ means that for b to participate in the overall interaction, a must also participate. The parallel composition allows to combine interactions without introducing dependencies: any combination of a and b can participate in $a \oplus b$.

The causality operator is right- (but not left-) associative, for interactions $a_1, \ldots, a_n$, we have $a_1 \to (a_2 \to (\cdots \to a_n)\ldots)) = a_1 \to a_2 \to \cdots \to a_n$. We call this construction a *causal chain*.

Semantics. The semantics of $\mathcal{T}(P)$ is given by the function $|\cdot| : \mathcal{T}(P) \to \mathcal{AI}(P)$

$$|a| = a\,, \qquad |a \to t| = a\Big(1 + |t|\Big)\,, \qquad |t_1 \oplus t_2| = |t_1| + |t_2| + |t_1|\,|t_2|\,, \tag{38}$$

where $a \in 2^P \cup \{0, 1\}$ is an interaction and $t, t_1, t_2 \in \mathcal{T}(P)$.

A sound axiomatisation of $\mathcal{T}(P)$ is provided in [10]. Rather than reproduce it here, we directly provide the extended version in Section 5.1.

4.4 Systems of Causal Rules

Systems of causal rules represent an intermediate structure between causal interaction trees and arbitrary Boolean formulas. They directly encode the causality information explicit in the causal interaction trees, by transforming causality relations into dual Horn clauses. Apart from supporting connector synthesis, they provide a convenient way for expressing properties to be enforced by the glue operators (see Section 6 for some examples). Causal rules have also served as basis for the macro-notation used to specify the glue in Dy-BIP—a dynamic flavour of BIP [15].

Definition 4.2. A *causal rule* is a Boolean formula in $\mathbb{B}[P]$ of the form $E \Rightarrow C$. The *effect* E is either the constant $\mathtt{tt}$ or a port variable $p \in P$. The *cause* C is either a constant, $\mathtt{tt}$ or $\mathtt{ff}$, or a disjunction of interactions, i.e. $\bigvee_{i=1}^{n} a_i$ where, for all $i \in [1,n]$, a_i are conjunctions of positive port variables.

Note 4.3. Notice that $a_1 \vee a_1\, a_2 = a_1$, and therefore causal rules can be simplified by replacing $p \Rightarrow a_1 \vee a_1\, a_2$ with $p \Rightarrow a_1$). We assume that all the causal rules are simplified by this absorption rule.

Definition 4.4. A *system of causal rules* is a set $R = \{p \Rightarrow x_p\}_{p \in P \cup \{\mathtt{tt}\}}$. An interaction $a \in 2^P$ satisfies the system R (denoted $a \models R$), iff the characteristic valuation of a on P satisfies the formula $\bigwedge_{p \in P \cup \{\mathtt{tt}\}} (p \Rightarrow x_p)$. We denote by $|R| \stackrel{def}{=} \sum_{a \models R} a$ the union (in terms of the Algebra of Interactions) of the interactions satisfying R. Thus we have $|\cdot| : \mathcal{CR}(P) \to \mathcal{AI}(P)$, where $\mathcal{CR}(P)$ is the set of all systems of causal rules over the set of port variables P.

4.5 Transformations between algebraic representations of interaction models

Transformations $\mathcal{AC}(P) \underset{\sigma}{\overset{\tau}{\rightleftarrows}} \mathcal{T}(P)$, $\mathcal{T}(P) \overset{R}{\rightleftarrows} \mathcal{CR}(P)$ and $\mathcal{CR}(P) \rightleftarrows \mathbb{B}[P]$ were defined in [10] and have been shown to respect $\simeq$. Below, we recall the transformations, which will be used later in the paper.

$\sigma : \mathcal{T}(P) \to \mathcal{AC}(P)$ is defined recursively by putting

$$\sigma(a) = [a]\,, \qquad \sigma(a \to t) = [a]'\,[\sigma(t)]\,, \qquad \sigma(t_1 \oplus t_2) = [\sigma(t_1)]'\,[\sigma(t_2)]'\,. \tag{39}$$

We define $R : \mathcal{T}(P) \to \mathcal{CR}(P)$ by putting

$$R(t) \;=\; \{p \Rightarrow c_p(t)\}_{p \in P \cup \{\mathtt{tt}\}}\,, \tag{40}$$

where the function $c_p : \mathcal{T}(P) \to \mathbb{B}[P]$ is defined recursively as follows. For $a \in 2^P$ (with $p \notin a$) and $t, t_1, t_2 \in \mathcal{T}(P)$, we put

$$\begin{aligned}
c_p(0) &= \mathtt{ff}\,, & c_{\mathtt{tt}}(0) &= \mathtt{ff}\,,\\
c_p(p \to t) &= \mathtt{tt}\,, & c_{\mathtt{tt}}(1 \to t) &= \mathtt{tt}\,,\\
c_p(pa \to t) &= a\,, & c_{\mathtt{tt}}(a \to t) &= a\,,\\
c_p(a \to t) &= a \wedge c_p(t)\,, & &\\
c_p(t_1 \oplus t_2) &= c_p(t_1) \vee c_p(t_2)\,, & c_{\mathtt{tt}}(t_1 \oplus t_2) &= c_{\mathtt{tt}}(t_1) \vee c_{\mathtt{tt}}(t_2)\,.
\end{aligned}$$

Observe that this transformation associates to each port $p \in P$ a causal rule $p \Rightarrow C$, where C is the disjunction of all prefixes leading from roots of t to some node containing p, including the ports of this node other than p.

The transformation $\mathcal{CR}(P) \to \mathcal{T}(P)$ consists of two steps. First, one saturates the system of causal rules. Definition 4.5 below, defines the notion of a saturated system fo causal rules formally. Intuitively, saturation consists in making all causal rules self-contained in terms of information about the constraints imposed by the system.

Definition 4.5. A system of causal rules $\{p_i \Rightarrow x_i\}_{i=1}^{n}$ is *saturated* iff, for all $i \in [1,n]$, $x_i = x_i[x_j/p_j]$, where $x_i[x_j/p_j]$ is obtained by substituting x_j for p_j in x_i, for all $j \neq i$.

For a given system of causal rules $R = \{p_i \Rightarrow x_i\}_{i=1}^{n}$, we denote by R_{sat} the corresponding saturated system.

In [10] it was shown that R and the corresponding R_{sat} are equivalent.

Given a saturated system of causal rules $R_{sat} = \{p \Rightarrow x_p\}_{p \in P \cup \{\mathtt{tt}\}}$ with $x_p = \bigvee_{i=1}^{m_p} a_i^p$ we build an auxiliary set

$$Y = \{p a_i^p \mid p \in P, i \in [1, m_p]\} \cup \{a_i^{\mathtt{tt}} \mid i \in [1, m_{\mathtt{tt}}]\} \tag{41}$$

by taking all monomials from the causes of the rules conjuncted with the corresponding effects. As shown in [10], this set comprises all "atomic" interactions allowed by the system of causal rules, that is sets of ports that can only appear together in a valid interaction. An inclusion tree, built from the elements of the set Y, is a corresponding causal tree for the system of causal rules.

5 Algebra extensions

In Section 2.3, we have replaced the classical BIP combination of interaction and priority models with an extended interaction model, where each occurrence of a port in an interaction is annotated with one of the three types: firing, activation and negative.

Recall (Section 4) that, for any algebra $\mathcal{A}(P)$ in our context, we define, the equivalence on $\mathcal{A}(P)$ by putting, for $x, y \in \mathcal{A}(P)$, $x \simeq y$ iff $\|x\| = \|y\|$, where $\|\cdot\| : \mathcal{A}(P) \to 2^{2^P}$ is the interaction semantics of the algebra. As a simple corollary of the results in [9], $\|x\| = \|y\|$ is equivalent to $\|x\|(\mathbf{B}) = \|y\|(\mathbf{B})$, for any finite family $\mathbf{B}$ of behaviours (where $\|x\|(\mathbf{B})$ denotes the application of an interaction model $\|x\|$ to the set of behaviours $\mathbf{B}$). However, this is not the case for extended interaction models, where $\|x\| = \|y\|$ implies $\|x\|(\mathbf{B}) = \|y\|(\mathbf{B})$, for any finite family $\mathbf{B}$ of behaviours, but the converse implication does not hold (cf. Lemma 2.26).

Definition 5.1. Let $\mathcal{A}(P)$ be an algebra, $\|\cdot\| : \mathcal{A}(P) \to 2^{2^P}$. Two terms $x, y \in \mathcal{A}(P)$ are *equivalent* $x \sim y$ iff, for any finite family $\mathbf{B}$ of behaviours, $\|x\|(\mathbf{B}) = \|y\|(\mathbf{B})$.

The Algebra of Interactions, $\mathcal{AI}(P)$, extends in a straightforward manner. Indeed, it is sufficient to consider $\mathcal{AI}(P \cup \dot{P} \cup \overline{P})$ with the equivalence $\sim$.

We can now similarly extend the other algebras [4]. The interaction semantics of the causal interaction trees $|\cdot| : \mathcal{T}(P) \to \mathcal{AI}(P)$ is transposed without any change to $|\cdot| : \mathcal{T}(P \cup \dot{P} \cup \overline{P}) \to \mathcal{AI}(P \cup \dot{P} \cup \overline{P})$. Similarly, the functions $\tau : \mathcal{AC}(P) \to \mathcal{T}(P)$ and $\sigma : \mathcal{T}(P) \to \mathcal{AC}(P)$ are transposed identically to $\mathcal{AC}(P \cup \dot{P} \cup \overline{P})$ and $\mathcal{T}(P \cup \dot{P} \cup \overline{P})$. The same goes for the mapping $R(t)$ associating to a causal interaction tree $t \in \mathcal{T}(P)$ the corresponding system of causal rules [10]. The only difference is that, in $\mathcal{CR}(P \cup \dot{P} \cup \overline{P})$ we introduce the following additional axiom: $\dot{p} \Rightarrow p$, for all $p \in P$ (cf. the discussion leading up to (26)).

The first consequence of this extension is that, rather than extending the existing graphical representation of connectors, it can be used as is to express priorities and activation conditions (the use of the offer predicate in the positive premises of the rule (30)) by adding a trivalued attribute to ports: *firing*, *activation* and *negative*. It is important to observe the difference between, on one hand, adding an attribute to ports and, on the other hand, modifying the typing operator (*synchron* vs. *trigger* typing), since the latter is applied at each level of the connector hierarchy, whereas the former is applied to ports, that is only at the leaves of the connector.

Example 5.2. Let $P = \{p, q, r, s\}$ and consider the (non-extended) interaction model $\gamma = \{p, q, pr, ps, prs\}$ and the priority model $\pi = \{pr \prec q, ps \prec q, prs \prec q\}$. The glue operator $\pi\gamma$ can be equivalently representated in the extended algebas as follows. The corresponding extended interaction model is $\{\dot{p},\ \dot{q},\ \dot{p}\,\overline{q}\,\dot{r}\,,\ \dot{p}\,\overline{q}\,\dot{s}\,,\ \dot{p}\,\overline{q}\,\dot{r}\,\dot{s}\}$, which can be represented by the union of two extended connectors: $\dot{q} + \dot{p}'[\overline{q}'\,\dot{r}\,\dot{s}]$ or, equivalently, $\dot{q} + \dot{p}'[\dot{r}\,\overline{q}][\dot{s}\,\overline{q}]$. The causal interaction trees corresponding to the second summands in these connectors are shown, respectively, in Figure 14a and Figure 14b.

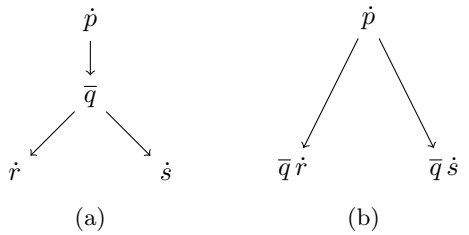


Figure 14: A pair of equivalent causal interaction trees.

5.1 Refinement of the extension

When we apply $x, y \in \mathcal{AI}(P \cup \dot{P} \cup \overline{P})$ to compose behaviour with operational semantics of Definition 2.25, $\|x\|(\mathbf{B}) = \|y\|(\mathbf{B})$ does not imply $x = y$. $\mathcal{AI}$ axioms are not complete (although still sound) with respect to $\sim$, since this equivalence is weaker than $\simeq$. Consequently, on $\mathcal{T}(P \cup \dot{P} \cup \overline{P})$, $\sim$ is also weaker than $\simeq$.

Example 5.3. Let $P = \{p, q, r, s\}$ and consider the $\mathcal{T}(P \cup \dot{P} \cup \overline{P})$ trees shown in Figure 14.

The causal interaction tree in Figure 14a defines a redundant interaction. Indeed,

$$\|\dot{p} \to \overline{q} \to (\dot{r} \oplus \dot{s})\| \;=\; \{\dot{p},\, \dot{p}\,\overline{q}\,,\, \dot{p}\,\overline{q}\,\dot{r}\,,\, \dot{p}\,\overline{q}\,\dot{s}\,,\, \dot{p}\,\overline{q}\,\dot{r}\,\dot{s}\,\}\ .$$

Although the interaction $\dot{p}\,\overline{q}$ does contain a firing port $\dot{p}$, it is redundant (Lemma 2.26). We conclude, therefore, that the causal interaction trees in Figure 14a and Figure 14b are equivalent, since

$$\|\dot{p} \to (\overline{q}\,\dot{r} \oplus \overline{q}\,\dot{s})\| \;=\; \{\dot{p},\, \dot{p}\,\overline{q}\,\dot{r}\,,\, \dot{p}\,\overline{q}\,\dot{s}\,,\, \dot{p}\,\overline{q}\,\dot{r}\,\dot{s}\,\}\ .$$

□

The above example illustrates the idea that the nodes of causal interaction trees, which do not contain firing ports, can be "pushed" down the tree.

Another notable case leading to redundant interactions corresponds to trees containing *contradictory port typings*. For example, either of the two equivalent trees $\overline{p} \to \dot{p}$ and $\overline{p}\,\dot{p}$ authorises the interaction $\overline{p}\,\dot{p}$. However, when considered in the context of the rule (30), this interaction generates two conflicting premises $q_i \xrightarrow{p} q_i'$ and $q_i \not\uparrow p$. Thus, this instance of the rule (30) does not authorise any transitions and the interaction $\overline{p}\,\dot{p}$ can be safely discarded. This example corresponds to the additional axiom $\dot{p} \Rightarrow p$ imposed in [11] on the Boolean formulas in $\mathbb{B}[P, \dot{P}]$. Similarly, redundant interactions appear when a tree contains other distinct port typings of the same port: p and $\overline{p}$, generating conflicting premises $q_i \uparrow p$ and $q_i \not\uparrow p$; p and $\dot{p}$, whereof the former generates the premise $q_i \uparrow p$ redundant alongside the premise $q_i \xrightarrow{p} q_i'$ generated by the latter.

Below, we provide a set of axioms reducing interaction redundancy. We enrich axioms for $\mathcal{T}(P \cup \dot{P} \cup \overline{P})$ from [10] by adding some new ones, specific for the trivalued port attribute.

Axioms. 1. For all $p \in P$ and $a \subseteq P \cup \dot{P} \cup \overline{P}$ such that $a \neq \emptyset$,

   (a) $a \cdot 0 = 0$,

   (b) $a \cdot 1 = a$, for $a \neq 0$,

   (c) $\dot{p} \cdot p = \dot{p}$ (cf. the additional axiom $\dot{p} \Rightarrow p$ in $\mathcal{CR}(P \cup \dot{P} \cup \overline{P})$),

   (d) $\dot{p} \cdot \overline{p} = p \cdot \overline{p} = 0$.

2. Parallel composition, '$\oplus$', is associative, commutative, idempotent, and its identity element is 0.

3. $a \to 0 = a$, for all $a \subseteq P \cup \dot{P} \cup \overline{P}$.

4. $0 \to t = 0$, for all $t \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$.

5. $ap \to b = ap \to bp$ for all $a, b \subseteq P \cup \dot{P} \cup \overline{P}$, $p \in P \cup \dot{P} \cup \overline{P}$.

6. $c \to a \to b \to t = c \to ab \to t$ for all $a, b, c \subseteq P \cup \dot{P} \cup \overline{P}$, such that $\mathbf{fire}(a) = \emptyset$, and $t \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$.

7. $a \to (t_1 \oplus t_2) = a \to t_1 \ \oplus \ a \to t_2$, for all $a \subseteq P \cup \dot{P} \cup \overline{P}$, $t_1, t_2 \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$.

Axioms 1 equalise redundant interactions due to contradictory port typings, whereas Axiom 5 propagates ports down in order to find contradictory port typings. Axiom 6 eliminates the nodes with empty firing support. Axioms 2, 3, 4 and 7 are the same as in [10]. The two remaining axioms from [10] are replaced by Lemmata 5.6 and 5.7 in this paper.

Proposition 5.4. *The equivalence relation $\sim$ on $\mathcal{T}(P \cup \dot{P} \cup \overline{P})$ is a congruence.*

Sketch of the proof. The proof is the same as for $\mathcal{T}(P)$ [10]. For any two trees $t_1, t_2 \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$ and for any context $C(z) \in \mathcal{T}(P \cup \dot{P} \cup \overline{P} \cup \{z\})$, we have to show that the equivalence $t_1 \sim t_2$ implies $C(t_1/z) \sim C(t_2/z)$, where $C(t_i/z)$ is the tree obtained, by replacing in $C(z)$ all occurrences of z by t_i. Since the semantics $\mathcal{T}$ is compositional [10, Proposition 4.6], structural induction on the context $C(z)$ proves the proposition. □

Proposition 5.5. *The above axiomatisation is sound with respect to $\sim$.*

Proof. Since, by Proposition 5.4, the equivalence relation $\sim$ is a congruence, it is sufficient to show that all the axioms respect $\sim$. This is proved by verifying that the semantics for left and right sides coincide.

Axioms 2, 3, 4 and 7 are the same as in [10]. Hence, their respective left- and right-hand sides are related by $\simeq$, which is stronger than $\sim$. Axiom 1(a) and Axiom 1(b) are trivial. Axiom 1(c) is a consequence of Lemma 2.26. In the Axiom 1(d), both pairs p and $\overline{p}$, and $\dot{p}$ and $\overline{p}$ produce conflicting premises in the rule (30) and, therefore, do not generate any transitions. For the Axiom 6, we have

$$\|c \to a \to b \to t\| = \{c,\, a\,c,\, a\,b\,c\} \cup \{a\,b\,c\,a_2 \,|\, a_2 \in \|t\|\}\,, \tag{42}$$

$$\|c \to ab \to t\| = \{c,\, a\,b\,c\} \cup \{a\,b\,c\,a_2 \,|\, a_2 \in \|t\|\}\,. \tag{43}$$

Hence, $\|c \to a \to b \to t\| = \|c \to ab \to t\| \cup \{ac\}$. Since $c \subseteq ac$ and $\mathbf{fire}(a) = \emptyset$, we conclude, by Lemma 2.26, that the two causal interaction trees are equivalent: $c \to a \to b \to t \sim c \to ab \to t$.

For the Axiom 5, we have $\|ap \to b\| = \{ap,\, abp\} = \|ap \to bp\|$. Thus $ap \to b \simeq ap \to bp$, which implies $ap \to b \sim ap \to bp$. □

Notice that, for the soundness of Axiom 6, it is essential that a is not a root node, since application of Lemma 2.26 is made possible by the presence of the interaction $c \in \|c \to a \to b \to t\|$. For a counter-example, consider two interaction trees $\overline{p} \to \dot{q}$ and $\overline{p}\dot{q}$. The former allows self-loops in states of a composed system, where p is not offered, whereas the latter does not.

Lemma 5.6. *For all $a, b \subseteq P \cup \dot{P} \cup \overline{P}$, such that $\mathbf{fire}(b) = \emptyset$, holds the equality $a \to b = a$.*

Proof. $a \to b = a \to b \to 0 \to 0 = a \to b \cdot 0 \to 0 = a \to 0 \to 0 = a$ (Axioms 3, 6) □

Lemma 5.7. *For all $a \subseteq P \cup \dot{P} \cup \overline{P}$ and $t \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$, holds the equality $a \to 1 \to t = a \to t$.*

Proof. If $t = 0$ the statement of this lemma is a special case of Lemma 5.6 with $b = 1$. If $t \neq 0$ it can be represented as a parallel composition of non-zero trees $t = \bigoplus_{i=1}^{n} r_i \to t_i$, with $r_i \subseteq P \cup \dot{P} \cup \overline{P}$. By Axioms 6 and 7, we have

$$a \to 1 \to t \;=\; \bigoplus_{i=1}^{n}(a \to 1 \to r_i \to t_i) \;=\; \bigoplus_{i=1}^{n}(a \to r_i \to t_i) \;=\; a \to \bigoplus_{i=1}^{n}(r_i \to t_i) \;=\; a \to t\,.$$

□

Lemma 5.8. *For all $a, b_i, c \subseteq P \cup \dot{P} \cup \overline{P}$, such that $\mathbf{fire}(a) = \emptyset$ and $t_i \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$, holds the equality*

$$c \to a \to \bigoplus_{i=1}^{n}(b_i \to t_i) = c \to \bigoplus_{i=1}^{n}(ab_i \to t_i)\,.$$

Proof. As above, applying Axioms 6 and 7, we have

$$c \to a \to \bigoplus_{i=1}^{n}(b_i \to t_i) \;=\; \bigoplus_{i=1}^{n}(c \to a \to b_i \to t_i) \;=\; \bigoplus_{i=1}^{n}(c \to ab_i \to t_i) \;=\; c \to \bigoplus_{i=1}^{n}(ab_i \to t_i)\,.$$

□

5.2 Normalisation of extended algebras

As it was shown in Example 5.3, causal interaction trees can contain nodes generating redundant interactions. These nodes can be removed by consecutively applying semantics-preserving transformations based on the axioms of the Algebra of Causal Interaction Trees.

Definition 5.9. A causal interaction tree $t \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$ is in *normal form* if it satisfies the following properties:

1. All nodes of t except roots have non-empty firing support.
2. In any causal chain of t a port p can appear more than once only in the form $ap \to \cdots \to b\dot{p}$, where $a, b \subseteq P \cup \dot{P} \cup \overline{P}$ and $p \in P$.

In the proof of Proposition 5.10 below, we provide a constructive procedure for normalising causal interaction trees.

Proposition 5.10 (Normal form for causal interaction trees)**.** *Every causal interaction tree $t \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$ has a normal form $t = \tilde{t} \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$.*

Proof. Consider $t \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$. We start by computing $t_1 = t$ with all nodes, except potentially the roots, having non-empty firing support.

Let a be a non-root node of t with $\mathbf{fire}(a) = \emptyset$, such that the tree s rooted in a does not have any nodes with empty firing support. If s is empty, that is a is a leaf then remove a from the tree (Lemma 5.6). Otherwise, let c be the parent of a, which exists since a is not a root and move the parallel composition operator up using Axiom 7:

$$c \to \left((a \to s) \oplus \bigoplus_{i=1}^{n} t_i\right) = (c \to a \to s) \oplus \left(\bigoplus_{i=1}^{n} c \to t_i\right)\,. \tag{44}$$

The sub-tree s can be further decomposed as $s = \bigoplus_{i=1}^{n}(b_i \to s_i)$, so, by Lemma 5.8, we have

$$c \to a \to s \;=\; c \to a \to \bigoplus_{i=1}^{n}(b_i \to s_i) \;=\; c \to \bigoplus_{i=1}^{n}(ab_i \to s_i)\,. \tag{45}$$

Each of nodes ab_i has non-empty firing support, since $\mathbf{fire}(b_i) \neq \emptyset$ by the choice of a. Substituting (45) into (44) and applying Axiom 7, we obtain

$$\left(c \to \bigoplus_{i=1}^{n}(ab_i \to s_i)\right) \oplus \left(\bigoplus_{i=1}^{n} c \to t_i\right) = c \to \left(\left(\bigoplus_{i=1}^{n} ab_i \to s_i\right) \oplus \bigoplus_{i=1}^{n} t_i\right) .$$

In the resulting tree, there is one node with empty firing support less than in t. Hence, repeating this procedure as long as there are such nodes, we will compute a tree t_1, where all nodes except roots have non-empty firing support. This computation is confluent, since the order is irrelevant among causally independent nodes, whereas among causally dependent ones it is fixed by the algorithm.

Consider a causal chain $a\tilde{p} \to \cdots \to b\hat{p}$ within t_1, with $\tilde{p}$ and $\hat{p}$ being two typings of the same port. If $\tilde{p} = p$ and $\hat{p} = \dot{p}$, there is nothing to do, since such dependencies are allowed by Definition 5.9. Otherwise, we propagate $\tilde{p}$ down by applying Axiom 5:

$$a\tilde{p} \to c_1 \to \cdots \to c_k \to b\hat{p} \;=\; a\tilde{p} \to c_1\tilde{p} \to \cdots \to c_k \to b\hat{p} = \;\ldots\; = \; a\tilde{p} \to c_1\tilde{p} \to \cdots \to c_k\tilde{p} \to b\hat{p}\tilde{p} .$$

Case 1: $\tilde{p} = \hat{p}$ or both $\tilde{p}, \hat{p} \neq \overline{p}$. We apply Axioms 1(c) and 5:

$$a\tilde{p} \to c_1\tilde{p} \to \cdots \to c_k\tilde{p} \to b\hat{p}\tilde{p} \;=\; a\tilde{p} \to c_1\tilde{p} \to \cdots \to c_k\tilde{p} \to b\tilde{p} \;= a\tilde{p} \to c_1 \to \cdots \to c_k \to b\,.$$

Case 2: $\tilde{p} \neq \hat{p}$ and either $\tilde{p} = \overline{p}$ or $\hat{p} = \overline{p}$. We apply Axioms 1(d), 3 and 5:

$$\begin{aligned} a\tilde{p} \to c_1\tilde{p} \to \cdots \to c_k\tilde{p} \to b\hat{p}\tilde{p} \;&=\; a\tilde{p} \to c_1\tilde{p} \to \cdots \to c_k\tilde{p} \to 0 \;= \\ &= \; a\tilde{p} \to c_1 \to \cdots \to c_k \to 0 \;=\; a\tilde{p} \to c_1 \to \cdots \to c_k\,. \end{aligned}$$

To compute $\tilde{t}$, we apply this transformation to all relevant causal chains within t_1. □

When synthesising connectors from causal interaction trees, their complexity can be reduced by tree normalization. Furthermore, since semantics-preserving transformations can be applied in both directions, a normal form on causal interaction trees induces a normal form on connectors.

Definition 5.11. A connector $x \in \mathcal{AC}(P \cup \dot{P} \cup \overline{P})$ is in *normal form* iff $x = \sigma(t)$, where t is a causal interaction tree in normal form and σ is the function defined in (39).

The following proposition is a direct consequence of the definitions of the normal form of causal interaction trees and function σ.

Proposition 5.12 (Normal form for connectors)**.** *A connector $x \in \mathcal{AC}(P \cup \dot{P} \cup \overline{P})$ in normal form has the following properties:*

1. *Nodes at every hierarchical level of the connector, except the bottom one, have at least one trigger.*
2. *Each node at the bottom hierarchical level, is a strong synchronisation of pairwise distinct ports.*
3. *Every node at the bottom hierarchical level, without firing ports, has only triggers as ancestors.*

5.3 Simplification of systems of causal rules

The port typings in the algebraic representations of extended interaction models, increase the complexity of systems of causal rules: without additional simplifications, the number of rules in the system is essentially tripled. The goal of this sub-section is to prove that we can consider only rules with firing port typings or $\mathtt{tt}$ as effects, and other rules can be removed as redundant.

The generation of systems of causal rules from Boolean formula starts with $\phi \in \mathbb{B}[P \cup \dot{P}]$ with additional axiom $\dot{p} \Rightarrow p$. This formula is transformed into conjunctive normal form (CNF). At this point we change the domain to consider ϕ as a formula from $\mathbb{B}[P \cup \dot{P} \cup \overline{P}]$ with two additional axioms: $\dot{p} \Rightarrow p$ and $\overline{p}$ $\mathtt{XOR}$ p. Essentially, this boils down to considering negative occurrences of variables from P as positive occurrences of variables from $\overline{P}$. Thus, seen as a formula from $\mathbb{B}[P \cup \dot{P} \cup \overline{P}]$, ϕ only has firing port variables in negative form. All other variables appear only in positive form.

We then proceed exactly as in [10]: We have $\phi' = C_1 \wedge C_2 \wedge \cdots \wedge C_n$ with, for $k \in [1, n]$, $C_k = \bigvee_{i \in I_k} p_i \vee \bigvee_{j \in J_k} \overline{p_j}$, where $I_k \cap J_k = \emptyset$, $p_i \in P \cup \dot{P} \cup \overline{P}$ and $p_j \in \dot{P}$ for all $i \in I_k$ and $j \in J_k$. We can now rewrite every clause C_k, with $J_k \neq \emptyset$, as a disjunction of dual Horn clauses $C_k = \bigvee_{j \in J_k} \left(\overline{p_j} \vee \bigvee_{i \in I_k} p_i\right)$. By distributivity, we obtain a representation of ϕ' as a disjunction of dual Horn formulas and, after combining the clauses with the same negative variable, we obtain $\phi' = R_1 \vee R_2 \vee \cdots \vee R_m$ with, for $k \in [1, m]$,

$$R_k = \bigwedge_{i \in \widetilde{I}_k} \left(\overline{p_i} \vee \bigvee_{j \in \widetilde{J}_{k,i}} a_j \right) = \bigwedge_{i \in \widetilde{I}_k} \left(p_i \Rightarrow \bigvee_{j \in \widetilde{J}_{k,i}} a_j \right) ,$$

where, for all $i \in \widetilde{I}_k$, $p_i \in \dot{P} \cup \mathtt{tt}$ and, for all $j \in \widetilde{J}_{k,i}$, a_j is $\mathtt{ff}$, $\mathtt{tt}$, or a conjunction of positive variables. Thus, each R_k is a system of causal rules, with only firing variables in the effects.

The algorithm synthesising causal interaction trees from systems of causal rules (see Section 4.4) expects that the input system is complete in the sense that it should have one rule for each port variable. Thus, for each $p \in P \cup \overline{P}$ the rule $p \Rightarrow \mathtt{tt}$ has to be added to the system. However, the rules with non-firing effects do not impose additional constraints on the system. Theorem 5.13 shows that such rules do not affect the causal interaction tree generated from the system of causal rules. Therefore, the synthesis algorithm remains correct, even when simplified by excluding all causal rules with effect $p \in P \cup \overline{P}$.

Theorem 5.13. *Let R be a system of causal rules over $P \cup \dot{P} \cup \overline{P}$, where all rules with the effect $p \in P \cup \overline{P}$ have the form $p \Rightarrow \mathtt{tt}$, and let R' be a set of causal rules containing only rules from R with effects $p \in \dot{P} \cup \mathtt{tt}$. Then, the causal interaction trees, generated for R and R' with the procedure described in Section 4.5, are equivalent with respect to $\sim$.*

Proof. The construction of causal interaction trees consists of two steps: saturation of the system of causal rules and building the tree. Clearly, rules of the form $p \Rightarrow \mathtt{tt}$ do not affect the saturation of other rules. On the other hand, such rules are saturated to $p \Rightarrow C$, where C is the saturated cause of the rule with the effect $\mathtt{tt}$.

Let Y and Y' be the auxiliary sets (41) containing monomials of the causes composed with the effects of the corresponding rules, for R and R' respectively. Clearly, $Y' \subseteq Y$ and $Y \setminus Y' \subseteq \{pc \mid p \in P \cup \overline{P}, c \in Y' \cup \{\emptyset\}\}$.[9] Hence, in the inclusion tree corresponding to R, elements of $Y \setminus Y'$ can generate additional nodes compared to the inclusion tree corresponding to R'. Every such node necessarily appears in a context of the form $c \to pc \to \bigoplus(pcq_i \to t_i)$ for some port variables q_i and sub-trees t_i. However, by Axioms 1 and 6, $c \to pc \to \bigoplus(pcq_i \to t_i) = c \to \bigoplus(pcq_i \to t_i)$, which is a fragment of the tree corresponding to R'. □

[9] It is possible that $c = \emptyset$ if the rule for the effect $\mathtt{tt}$ is $\mathtt{tt} \Rightarrow \mathtt{tt}$. Recall that the empty interaction corresponds to 1 in the algebra of Causal Interaction Trees.

The complexity of causal interaction tree synthesis algorithm [10] greatly depends on the number of rules in the system. Indeed, the saturation phase consists in substituting each port in the cause part of each rule with the cause of the corresponding rule, where this port is the effect. This is repeated until a fixpoint is reached. Theorem 5.13 removes two thirds of the rules, thus greatly reducing the synthesis complexity.

We have shown above that, while synthesising causal interaction trees from Boolean formulas, we can discard rules with non-firing effects in the intermediate systems of causal rules. Theorem 5.14 below shows that we can also discard rules with non-firing effects, when generating systems of causal rules from causal interaction trees, thus considerably reducing the obtained Boolean formulas.

Theorem 5.14. *Consider a causal interaction tree $t \in \mathcal{T}(P \cup \dot{P} \cup \overline{P})$ and a system of causal rules $R(t) = \{p \Rightarrow C_p(t)\}_{p \in P \cup \dot{P} \cup \overline{P} \cup \{\mathtt{tt}\}}$ obtained by the transformation $R : \mathcal{T}(P \cup \dot{P} \cup \overline{P}) \to \mathcal{CR}(P \cup \dot{P} \cup \overline{P})$ defined in Section 4.5. Let $\tilde{R}(t) = \{p \Rightarrow C_p(t)\}_{p \in \dot{P} \cup \{\mathtt{tt}\}}$ be a system of causal rules, obtained from $R(t)$ by omitting rules for port variables in $P \cup \overline{P}$. Then holds the equivalence $R(t) \sim \tilde{R}(t)$.*

Proof. Recall that applying the transformation $R : \mathcal{T}(P) \to \mathcal{CR}(P)$ defined in Section 4.5 to a tree $t \in \mathcal{T}(P)$, gives a system of causal rules of the form $p \Rightarrow C$, where C is a DNF Boolean formula and each monomial is a conjunction of the nodes on the way from a root of t to p (some prefix in t leading to p, excluding p).

$\tilde{R}(t)$ has less constraints than $R(t)$. Hence, it allows more interactions, i.e. $\|R(t)\| \subseteq \|\tilde{R}(t)\|$. Let $a \in \|\tilde{R}(t)\| \setminus \|R(t)\|$, i.e. there exists $p \in P \cup \overline{P}$, such that $p \in a$ and the rule $p \Rightarrow C_1$ is violated by a. First of all, notice that this implies immediately that $a \neq \emptyset$. Furthermore, we have $\emptyset \in \|R(t)\| \Leftrightarrow \emptyset \in \|\tilde{R}(t)\|$. Let $\tilde{a} = a \setminus p$.

Assume $\tilde{a} \notin \|\tilde{R}(t)\|$, i.e. there exists $\dot{q} \in \dot{P}$ and a rule $(\dot{q} \Rightarrow C_2) \in \tilde{R}(t)$, such that $\dot{q} \in \tilde{a}$ and the rule $\dot{q} \Rightarrow C_2$ is violated by $\tilde{a}$. This rule is not violated by a. Hence $C_2 = pC_2'$ and, consequently, p lies on all prefixes in t, leading to $\dot{q}$. $a \in \|\tilde{R}(t)\|$, $\dot{q} \in \tilde{a} \subseteq a$, thus there is at least one prefix in t, leading to $\dot{q}$ and contained in a. As p lies on this prefix, the rule $(p \Rightarrow C_1)$ is satisfied by a, contradicting the conclusion above. Therefore our assumption is wrong and $\tilde{a} \in \|\tilde{R}(t)\|$.

Since $\|\tilde{R}(t)\|$ is finite and $\tilde{a}$ is a proper subset of a (i.e. $\tilde{a} \subset a$ and $\tilde{a} \neq a$), by applying this reasoning iteratively, we conclude that, for all $a \in \|\tilde{R}(t)\| \setminus \|R(t)\|$, there exists $\tilde{a} \subseteq a$, such that $\mathbf{fire}(\tilde{a}) = \mathbf{fire}(a)$ and $\tilde{a} \in \|R(t)\|$. Hence, by Lemma 2.26, we have $\|\tilde{R}(t)\|(\mathbf{B}) = \|R(t)\|(\mathbf{B})$, for any finite family of behaviours $\mathbf{B}$, i.e. $R(t) \sim \tilde{R}(t)$. □

6 Connector synthesis (example)

In order to synthesise $\mathcal{AC}(P \cup \dot{P} \cup \overline{P})$ connectors from $\mathbb{B}[P \cup \dot{P}]$ constraints, one must perform the following steps.

1. Take the conjunction of all the constraints;
2. By adding the axioms $\dot{p} \Rightarrow p$ and p `XOR` $\overline{p}$, transform the obtained formula into a set of systems of causal rules over $P \cup \dot{P} \cup \overline{P}$, as described in the previous section;
3. Saturate the obtained systems of causal rules;
4. Convert each saturated system of causal rules into a causal interaction tree;
5. Normalize all trees;
6. Generate corresponding connectors from causal interaction trees.

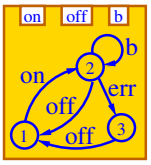


Figure 15: Main module.

This procedure is illustrated by the following example.

Consider a system providing some given functionality in two modes: *normal* and *backup*. The system consists of four modules: the Backup module A can only perform one action a; the Main module B (Figure 15) can perform an action b corresponding to the normal mode activity, it can also be switched on and off, as well as perform an internal (unobservable) error transition err; the Monitor module M is a black box, which performs some internal logging by observing the two actions a and b through the corresponding ports a_l and b_l; finally, the black box Controller module C takes the decisions to switch on or off the main module through the corresponding ports on_c and off_c, furthermore, it can perform a $test$ to check whether the main module can be restarted.

We want to synthesise connectors ensuring the properties below (encoded by Boolean constraints).

- The main and backup actions must be logged: $\dot{a} \Leftrightarrow \dot{a_l}$ and $\dot{b} \Leftrightarrow \dot{b_l}$;
- Only Controller can turn on the Main module: $\dot{on} \Leftrightarrow \dot{on_c}$;
- When Controller switches off, the Main module must stop operation: $\dot{off_c} \Rightarrow \dot{off}$ and $\dot{b} \Rightarrow \overline{\dot{off_c}}$;
- Controller can only test the execution of Backup: $\dot{test} \Rightarrow \dot{a}$;
- Backup can only execute when Main is not possible: $\dot{a} \Rightarrow \overline{b} \vee \dot{off}$;
- Main can only switch off when ordered to do so or after a failure: $\dot{off} \Rightarrow \overline{b} \vee \dot{off_c}$;

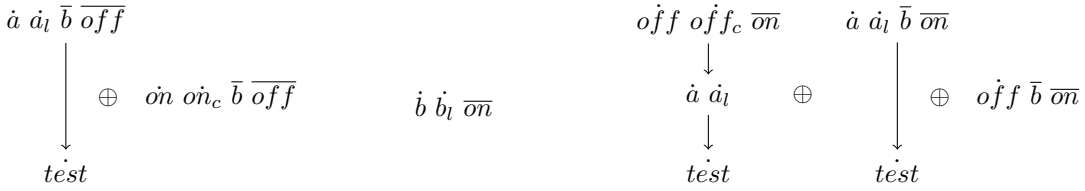
In order to compute the required glue, we take the conjunction of the above constraints together with the *progress* constraint $\dot{a} \vee \dot{b} \vee \dot{on} \vee \dot{off} \vee \dot{test} \vee \dot{a_l} \vee \dot{b_l} \vee \dot{off_c} \vee \dot{on_c}$ stating that at every round some action must be taken. Notice that, combined with the above constraints, the progress constraint can be immediately simplified by absorption to $\dot{a} \vee \dot{b} \vee \dot{on} \vee \dot{off}$. In order to simplify the resulting connectors, we also use part of the information about the behaviour of the Main module, namely the fact that on, on one hand, and b or off, on the other, are mutually exclusive: $on \Rightarrow \overline{b} \wedge \overline{off}$. We obtain the following global constraint (omitting the conjunction symbol):

$$(\dot{a} \Rightarrow \dot{a_l}\ \overline{b}\ \vee\ \dot{a_l}\ \dot{off})(\dot{a_l} \Rightarrow \dot{a})(\dot{b} \Rightarrow \dot{b_l}\ \overline{\dot{off_c}})(\dot{b_l} \Rightarrow \dot{b})(\dot{off} \Rightarrow \overline{b}\ \vee\ \dot{off_c})(\dot{off_c} \Rightarrow \dot{off})(\dot{test} \Rightarrow \dot{a})$$
$$\wedge\ (\dot{on} \Rightarrow \dot{on_c})(\dot{on_c} \Rightarrow \dot{on})(on \Rightarrow \overline{b}\ \overline{off})(\dot{a}\ \vee\ \dot{b}\ \vee\ \dot{on}\ \vee\ \dot{off})\,.$$

Recall now that causal rules must have the form $p \Rightarrow C$, where $p \in \dot{P} \cup \{\mathtt{tt}\}$ and C is a DNF Boolean formula on $P \cup \dot{P} \cup \overline{P}$ without negative firing variables or a logical constant. A system of causal rules is a conjunction of such clauses. Among the constraints above, there are two that do not have this form: $on \Rightarrow \overline{b}\ \overline{off}$ and $\dot{b} \Rightarrow \dot{b_l}\ \overline{\dot{off_c}}$. We rewrite them as $\overline{on} \vee \overline{b}\ \overline{off}$ and $\overline{\dot{b}} \vee \dot{b_l}\ \overline{\dot{off_c}}$, and distribute over the conjunction of the rest of the constraints. Finally, we implicitly apply the additional axioms $\dot{p} \Rightarrow p$ and p $\mathtt{XOR}$ $\overline{p}$ and, making some straightforward simplifications, obtain

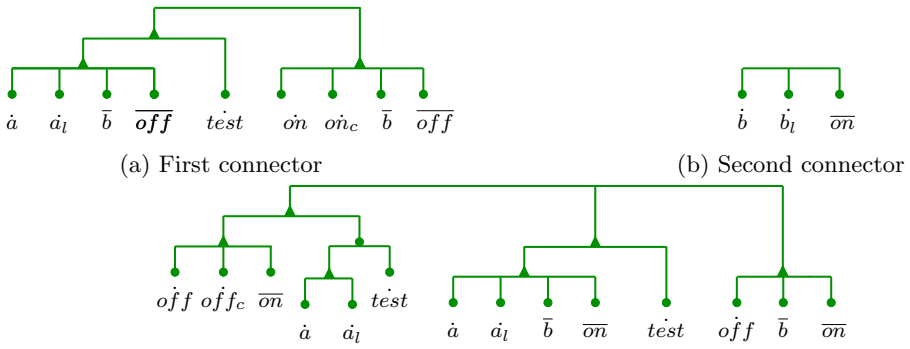
Table 1: Systems of causal rules for the example of Section 6

$\mathtt{tt}$	$\Rightarrow \dot{a}\,\overline{\dot{b}}\,\overline{o\dot{f}f} \vee o\dot{n}\,\overline{\dot{b}}\,\overline{o\dot{f}f}$			$\mathtt{tt}$	$\Rightarrow \dot{a}\,\dot{a}_l\,\overline{\dot{b}}\,\overline{o\dot{f}f} \vee o\dot{n}\,o\dot{n}_c\,\overline{\dot{b}}\,\overline{o\dot{f}f}$		
$\dot{a}$	$\Rightarrow \dot{a}_l\,\overline{\dot{b}} \vee \dot{a}_l\,o\dot{f}f$			$\dot{a}$	$\Rightarrow \dot{a}_l\,\overline{\dot{b}}\,\overline{o\dot{f}f}$		
$\dot{a}_l$	$\Rightarrow \dot{a}$	$\dot{b}$	$\Rightarrow \mathtt{ff}$	$\dot{a}_l$	$\Rightarrow \dot{a}\,\overline{\dot{b}}\,\overline{o\dot{f}f}$	$\dot{b}$	$\Rightarrow \mathtt{ff}$
$o\dot{n}$	$\Rightarrow o\dot{n}_c$	$\dot{b}_l$	$\Rightarrow \dot{b}$	$o\dot{n}$	$\Rightarrow o\dot{n}_c\,\overline{\dot{b}}\,\overline{o\dot{f}f}$	$\dot{b}_l$	$\Rightarrow \mathtt{ff}$
$o\dot{n}_c$	$\Rightarrow o\dot{n}$	$o\dot{f}f$	$\Rightarrow \mathtt{ff}$	$o\dot{n}_c$	$\Rightarrow o\dot{n}\,\overline{\dot{b}}\,\overline{o\dot{f}f}$	$o\dot{f}f$	$\Rightarrow \mathtt{ff}$
$t\dot{e}st$	$\Rightarrow \dot{a}$	$o\dot{f}f_c$	$\Rightarrow o\dot{f}f$	$t\dot{e}st$	$\Rightarrow \dot{a}\,\dot{a}_l\,\overline{\dot{b}}\,\overline{o\dot{f}f}$	$o\dot{f}f_c$	$\Rightarrow \mathtt{ff}$
$\mathtt{tt}$	$\Rightarrow \dot{b}\,\dot{b}_l\,\overline{o\dot{n}}$			$\mathtt{tt}$	$\Rightarrow \dot{b}\,\dot{b}_l\,\overline{o\dot{n}}$		
$\dot{a}$	$\Rightarrow \dot{a}_l\,\overline{\dot{b}} \vee \dot{a}_l\,o\dot{f}f$			$\dot{a}$	$\Rightarrow \mathtt{ff}$		
$\dot{a}_l$	$\Rightarrow \dot{a}$	$\dot{b}$	$\Rightarrow \mathtt{tt}$	$\dot{a}_l$	$\Rightarrow \mathtt{ff}$	$\dot{b}$	$\Rightarrow \dot{b}_l\,\overline{o\dot{n}}$
$o\dot{n}$	$\Rightarrow \mathtt{ff}$	$\dot{b}_l$	$\Rightarrow \dot{b}$	$o\dot{n}$	$\Rightarrow \mathtt{ff}$	$\dot{b}_l$	$\Rightarrow \dot{b}\,\overline{o\dot{n}}$
$o\dot{n}_c$	$\Rightarrow o\dot{n}$	$o\dot{f}f$	$\Rightarrow \overline{\dot{b}} \vee o\dot{f}f_c$	$o\dot{n}_c$	$\Rightarrow \mathtt{ff}$	$o\dot{f}f$	$\Rightarrow \mathtt{ff}$
$t\dot{e}st$	$\Rightarrow \dot{a}$	$o\dot{f}f_c$	$\Rightarrow \mathtt{ff}$	$t\dot{e}st$	$\Rightarrow \mathtt{ff}$	$o\dot{f}f_c$	$\Rightarrow \mathtt{ff}$
$\mathtt{tt}$	$\Rightarrow \dot{a}\,\overline{o\dot{n}} \vee o\dot{f}f\,\overline{o\dot{n}}$			$\mathtt{tt}$	$\Rightarrow \dot{a}\,\dot{a}_l\,\overline{\dot{b}}\,\overline{o\dot{n}} \vee o\dot{f}f\,o\dot{f}f_c\,\overline{o\dot{n}} \vee o\dot{f}f\,\overline{\dot{b}}\,\overline{o\dot{n}}$		
$\dot{a}$	$\Rightarrow \dot{a}_l\,\overline{\dot{b}} \vee \dot{a}_l\,o\dot{f}f$			$\dot{a}$	$\Rightarrow \dot{a}_l\,\overline{\dot{b}}\,\overline{o\dot{n}} \vee \dot{a}_l\,o\dot{f}f\,o\dot{f}f_c\,\overline{o\dot{n}}$	$\dot{b}$	$\Rightarrow \mathtt{ff}$
$\dot{a}_l$	$\Rightarrow \dot{a}$	$\dot{b}$	$\Rightarrow \mathtt{ff}$	$\dot{a}_l$	$\Rightarrow \dot{a}\,\overline{\dot{b}}\,\overline{o\dot{n}} \vee \dot{a}\,o\dot{f}f\,o\dot{f}f_c\,\overline{o\dot{n}}$	$\dot{b}_l$	$\Rightarrow \mathtt{ff}$
$o\dot{n}$	$\Rightarrow \mathtt{ff}$	$\dot{b}_l$	$\Rightarrow \dot{b}$	$o\dot{f}f$	$\Rightarrow \overline{\dot{b}}\,\overline{o\dot{n}} \vee o\dot{f}f_c\,\overline{o\dot{n}}$	$o\dot{n}$	$\Rightarrow \mathtt{ff}$
$o\dot{n}_c$	$\Rightarrow o\dot{n}$	$o\dot{f}f$	$\Rightarrow \overline{\dot{b}} \vee o\dot{f}f_c$	$o\dot{f}f_c$	$\Rightarrow o\dot{f}f\,\overline{o\dot{n}}$	$o\dot{n}_c$	$\Rightarrow \mathtt{ff}$
$t\dot{e}st$	$\Rightarrow \dot{a}$	$o\dot{f}f_c$	$\Rightarrow o\dot{f}f$	$t\dot{e}st$	$\Rightarrow \dot{a}\,\dot{a}_l\,\overline{\dot{b}}\,\overline{o\dot{n}} \vee \dot{a}\,\dot{a}_l\,o\dot{f}f\,o\dot{f}f_c\,\overline{o\dot{n}}$		



(a) First tree (b) Second tree (c) Third tree

Figure 16: Three causal interaction trees.



(a) First connector (b) Second connector

(c) Third connector

Figure 17: Connectors corresponding to trees from Figure 16.

a disjunction of three systems of causal rules. In Table 1, these systems are shown in the first column and their corresponding saturated equivalents are shown in the second column.

The corresponding auxiliary sets (41) obtained by combining the effects with the causes are then:

$$\left\{\dot{a}\ \dot{a}_l\ \overline{b}\ \overline{off},\ \dot{on}\ \dot{on}_c\ \overline{b}\ \overline{off}\ \dot{a}\ \dot{a}_l\ \overline{b}\ \overline{off}\ \dot{test}\right\},\quad \left\{\dot{b}\ \dot{b}_l\ \overline{on}\right\},$$
$$\left\{\dot{a}\ \dot{a}_l\ \overline{b}\ \overline{on},\ \dot{a}\ \dot{a}_l\ \overline{b}\ \overline{on}\ \dot{test},\ \dot{off}\ \overline{b}\ \overline{on},\ \dot{off}\ \dot{off}_c\ \overline{on},\ \dot{a}\ \dot{a}_l\ \dot{off}\ \dot{off}_c\ \overline{on},\ \dot{a}\ \dot{a}_l\ \dot{off}\ \dot{off}_c\ \overline{on}\ \dot{test}\right\}$$

$\mathcal{T}(P \cup \dot{P} \cup \overline{P})$ trees, shown on Figure 16, are obtained by normalising the inclusion trees corresponding to these sets. Applying (39) we obtain $\mathcal{AC}(P \cup \dot{P} \cup \overline{P})$ connectors in Figure 17.

In terms of classical BIP, one can, for example, easily distinguish here two priorities: $x\ a\ a_l \prec b\ b_l$ and $x\ off \prec b\ b_l$ for all x not containing $off\ off_c$. In general, priorities are replaced by local inhibitors. In this example, these appear to characterise states of the Main module. For instance, $\dot{a}\ \dot{a}_l\ \overline{b}\ \overline{off}$ defines possible interactions involving $a\ a_l$ when neither b nor off are possible, i.e. in state 1 (see Figure 15).

7 Related work

The results in this paper build mainly on our previous work. However, the following related work should also be mentioned. The comprehensive presentation of the key properties, required from a component-based framework for the design of concurrent software and systems, could be of relevance to other hierarchical frameworks, such as, for instance, Sofa [20], Koala [38] or rCos [29], and for languages implementing the Fractal specification [17].

Table 2: Correspondence between valuations of port variables in BIP and colours in the 3-colouring model of Reo

p	$\dot{p}$	BIP	Reo
$\mathtt{tt}$	$\mathtt{tt}$	active and firing	data flows
$\mathtt{tt}$	$\mathtt{ff}$	active, but not firing	no flow due to absence of take requests
$\mathtt{ff}$	$\mathtt{ff}$	not active	no flow due to absence of write requests

The approach we used in [10] for the Boolean encoding of connectors is close to that used for computing flows in Reo connectors in [22], where it is further extended to data flow. In [28], the authors discuss the extension of the coloring semantics of Reo [21] from the 2-colouring to the 3-colouring model. This extension is necessary to account for context-dependencies. For example, as suggested by its name, a `LossySync` channel can loose data provided on its source end. However, the semantics of Reo channels requires that this happens only if there is no take requests on the sink end of the channel. In the BIP context, we call this requirement *maximal progress*: if two distinct interactions $a \subseteq b$ are enabled, then, under maximal progress, we implicitly assume $a \prec b$. Thus, in BIP, maximal progress is a special case of priority. In the Reo context, this is also supported by the results in [14]. The authors of [28] encode context-dependency by duplicating all connector nodes: to each *base node* they associate a dual *context node* with complementary flow constraints. This is very similar to our use of firing and negative port typings to encode priority. Furthermore, in [28, page 5], one reads: "Whereas 2-coloring models can express synchronization, they cannot express context-dependency: to model context-sensitive connectors, three colors seem necessary." This observation reflects very closely our use of the additional axiom $\dot{p} \implies p$ in $\mathbb{B}[P, \dot{P}]$ (see, in particular, Section 2.3). Indeed, this axiom excludes the valuation $\dot{p} = \mathtt{tt}, p = \mathtt{ff}$, leaving only three possible valuations of the two variables. It seems that the correspondence between BIP notions of activation and firing and colours in the Reo 3-colouring model can be established as summarised in Table 2. This suggests that our notion of witness port typings could also be used in Reo to define nodes that allow the flow of data only if data is available (but will not be consumed) on an additional "control" channel.

Several methodologies for synthesis of component coordination have been proposed in the literature, e.g. connector synthesis in [1, 2, 27]. Both approaches are very different from ours. In [1], Reo circuits are generated from constraint automata [3]. This approach is limited, in the first place, by the complexity of building the automaton specification of interactions. An attempt to overcome this limitation is made in [2] by generating constraint automata from UML sequence diagrams. In [27], connectors are synthesised in order to ensure deadlock freedom of systems that follow a very specific architectural style imposing both the interconnection topology and communication primitives (notification and request messages). Our approach, focuses on the properties (expressed as glue constraints) that do not bear computation, which allows us to reduce a very hard and, in general, undecidable problem of synthesising controllers [34] to a tractable one.

Recently a comparative study [19] of three connector frameworks—tile model [18], wire calculus [37] and BIP[5]—has been performed. From the operational semantics perspective, this comparison only takes in account operators with positive premises. In particular, priority in BIP is not considered. It would be interesting to see whether using "local" offer predicate instead of "global" priorities of the classical BIP could help generalising this work.

Finally, we should mention that the offer predicate used in our formalism has, indeed, some similarity with the concept of barbs [31]. Although, in [31], the barbs do not appear to be used in the premises of the SOS rules defining the semantics of the processes, it would be interesting to further explore this relation.

8 Conclusion

This paper summarises and extends our work on the semantics of the BIP component-based framework and, in particular, on the composition operators defined by interaction and priority models, which we collectively call *glue operators*. Over the years, the semantics of the BIP glue operators was subject to some minor, apparently innocuous modifications, which happened to have considerable impact on the properties of the BIP framework. In this paper, we have provided a concise but comprehensive overview of the main properties: *compositionality*, *incrementality*, *flattening*, *modularity* and *expressiveness*, on a very fundamental level, which applies to any component-based framework.

In the light of these fundamental properties, we have reviewed the three modifications of the BIP semantics: the classical semantics introduced in [7]; a very slight modification introduced in [9]; and the offer semantics, proposed in [11], which relies on the *offer predicate* in the negative premises defining the semantics of priority models. We provided theoretical results and examples showing that the classical BIP semantics has compositionality, but does not have neither flattening, nor full expressiveness; the slightly modified version, introduced in [9], has flattening and full expressiveness, but it does not have neither compositionality, nor modularity, since it is not structural; the offer semantics [11] is structural, it has all the desired properties above. Furthermore, this semantics generalises to a larger class of operators—using the so-called *witness* ports in addition to the usual interaction models and priorities—in correspondence with formulas of $\mathbb{B}[P, \dot{P}]$, the Boolean algebra over the sets of *activation and firing port variables*, P and $\dot{P}$, respectively, with the additional axiom $\dot{p} \implies p$, for each port $p \in P$.

The classical and offer semantics are not comparable: there are systems that can be assembled in the classical semantics, but not in the offer one. We have presented a characterisation of the behaviour hierarchy, consisting of three properties. The first property, when satisfied by all operand behaviours, guarantees that the same glue operators (interaction and priority models) that are used with the classical semantics, can be used with the offer semantics to obtain an equivalent system. In general, the first property is not preserved by composition. However, it is preserved in hierarchical systems, where all priority models are applied after all interaction models. This allows us to conclude that the behaviour of any such system, where atomic behaviours satisfy the first property, is not affected by switching from the classical to the offer semantics.

The second and third properties are more technical. If satisfied by all atomic behaviours in the system, the second property, which is weaker than the first one, guarantees that the combination of glue operators (interaction and priority models) in the classical semantics can be adapted to the offer semantics by some additional modifications to the priority model.

The third property—the weakest of the three—guarantees only that, for any system built in the classical semantics, if the property is satisfied by all atomic behaviours, a glue operator can be found in the offer semantics that would construct an equivalent system with the same atomic behaviours.

This characterisation is strict in the sense that, if a given property is violated by at least one atomic behaviour in a set, a glue operator in the classical semantics can be exhibited, for which the result guaranteed by the violated property does not hold when the glue operator is applied to this set of behaviours.

The correspondence between the offer semantics of the BIP glue operators and the formulas of $\mathbb{B}[P, \dot{P}]$ allows us to use the full power of the Boolean algebra for the manipulation of glue operators and their representations. In particular, it allows the synthesis of connectors from Boolean formulas encoding coordination properties to be imposed by the glue operators.

The equivalence induced by the new operational semantics on the algebras representing the glue operators is weaker than the standard equivalence induced by the interaction semantics. Extending accordingly the axioms of the Algebra of Causal Interaction Trees, $\mathcal{T}(P)$, we define normal forms

for connectors and causal interaction trees. Finally, we show that the causal rule representation can also be simplified by considering only rules with firing effects. This has direct impact on the complexity of the synthesis algorithm. Algebra extensions are illustrated on a connector synthesis example.

In this paper, we have only extended the axiomatisation of $\mathcal{T}(P \cup \dot{P} \cup \overline{P})$. Studying corresponding extensions for the axiomatisations of other algebras as well as their completeness could be part of the future work. Another question that we leave for future work, is how to map, for example, the $\mathcal{AC}(P)$ connectors with separately specified priorities into $\mathcal{AC}(P \cup \dot{P} \cup \overline{P})$ and back. Indeed, in this paper we focused only on comparing the semantics and on transferring the algebraic representation theory to the offer semantics.

References

[1] Farhad Arbab, Christel Baier, Frank de Boer, Jan Rutten, and Marjan Sirjani. Synthesis of Reo circuits for implementation of component-connector automata specifications. In *Coordination Models and Languages*, volume 3454 of *LNCS*, pages 236–251, Berlin / Heidelberg, 2005. Springer.

[2] Farhad Arbab and Sun Meng. Synthesis of connectors from scenario-based interaction specifications. In *CBSE'08*, volume 5282 of *LNCS*, pages 114–129. Springer Berlin/Heidelberg, 2008.

[3] Christel Baier, Marjan Sirjani, Farhad Arbab, and Jan J. M. M. Rutten. Modeling component connectors in Reo by constraint automata. *Sci. Comput. Program.*, 61(2):75–113, 2006.

[4] Eduard Baranov and Simon Bliudze. Extended connectors: Structuring glue operators in BIP. In *ICE 2013*, volume 131 of *EPTCS*, pages 20–35, 2013.

[5] Ananda Basu, Marius Bozga, and Joseph Sifakis. Modeling heterogeneous real-time components in BIP. In 4^{th} *IEEE Int. Conf. on Software Engineering and Formal Methods (SEFM06)*, pages 3–12, September 2006. Invited talk.

[6] Simon Bliudze. Towards a theory of glue. In *ICE 2012: Distributed coordination, execution models, and resilient interaction*, volume 104 of *EPTCS*, pages 48–66, December 2012.

[7] Simon Bliudze and Joseph Sifakis. The algebra of connectors — Structuring interaction in BIP. In *Proc. of the EMSOFT'07*, pages 11–20. ACM SigBED, October 2007.

[8] Simon Bliudze and Joseph Sifakis. The algebra of connectors—structuring interaction in BIP. *IEEE Transactions on Computers*, 57(10):1315–1330, 2008.

[9] Simon Bliudze and Joseph Sifakis. A notion of glue expressiveness for component-based systems. In Franck van Breugel and Marsha Chechik, editors, *CONCUR 2008*, volume 5201 of *LNCS*, pages 508–522. Springer, 2008.

[10] Simon Bliudze and Joseph Sifakis. Causal semantics for the algebra of connectors. *Formal Methods in System Design*, 36(2):167–194, June 2010.

[11] Simon Bliudze and Joseph Sifakis. Synthesizing glue operators from glue constraints for the construction of component-based systems. In Sven Apel and Ethan Jackson, editors, 10^{th} *International Conference on Software Composition*, volume 6708 of *LNCS*, pages 51–67. Springer, 2011.

[12] Bard Bloom. *Ready Simulation, Bisimulation, and the Semantics of CCS-Like Languages.* PhD thesis, Massachusetts Institute of Technology, 1989.

[13] Borzoo Bonakdarpour, Marius Bozga, Mohamad Jaber, Jean Quilbeuf, and Joseph Sifakis. From high-level component-based models to distributed implementations. In *Proceedings of the tenth ACM international conference on Embedded software*, EMSOFT '10, pages 209–218, New York, NY, USA, 2010. ACM.

[14] Marcello Bonsangue, Dave Clarke, and Alexandra Silva. A model of context-dependent component connectors. *Science of Computer Programming*, 77(6):685–706, 2012. (1) Coordination 2009 (2) {WCRE} 2009.

[15] Marius Bozga, Mohamad Jaber, Nikolaos Maris, and Joseph Sifakis. Modeling dynamic architectures using Dy-BIP. In Thomas Gschwind, Flavio Paoli, Volker Gruhn, and Matthias Book, editors, *Software Composition*, volume 7306 of *Lecture Notes in Computer Science*, pages 1–16. Springer Berlin Heidelberg, 2012.

[16] Marius Bozga, Mohamad Jaber, and Joseph Sifakis. Source-to-source architecture transformation for performance optimization in BIP. In *Industrial Embedded Systems, 2009. SIES '09. IEEE International Symposium on*, pages 152–160, 2009.

[17] Eric Bruneton, Thierry Coupaye, Matthieu Leclercq, Vivien Quéma, and Jean-Bernard Stefani. The fractal component model and its support in java. *Software: Practice and Experience*, 36(11-12):1257–1284, 2006.

[18] Roberto Bruni, Ivan Lanese, and Ugo Montanari. A basic algebra of stateless connectors. *Theor. Comput. Sci.*, 366(1):98–120, 2006.

[19] Roberto Bruni, Hernn Melgratti, and Ugo Montanari. Connector algebras, petri nets, and bip. In Edmund Clarke, Irina Virbitskaite, and Andrei Voronkov, editors, *Perspectives of Systems Informatics*, volume 7162 of *Lecture Notes in Computer Science*, pages 19–38. Springer Berlin Heidelberg, 2012.

[20] Tomas Bures, Petr Hnetynka, and Frantisek Plasil. Sofa 2.0: Balancing advanced features in a hierarchical component model. In *Software Engineering Research, Management and Applications, 2006. Fourth International Conference on*, pages 40–48. IEEE, 2006.

[21] Dave Clarke, David Costa, and Farhad Arbab. Connector colouring I: Synchronisation and context dependency. *Electr. Notes Theor. Comput. Sci.*, 154(1):101–119, 2006.

[22] Dave Clarke, José Proença, Alexander Lazovik, and Farhad Arbab. Deconstructing Reo. *ENTCS*, 229(2):43–58, 2009.

[23] J. Eker, J.W. Janneck, E.A. Lee, J. Liu, X. Liu, J. Ludvig, S. Neuendorffer, S. Sachs, and Y. Xiong. Taming heterogeneity: The Ptolemy approach. *Proceedings of the IEEE*, 91(1):127–144, 2003.

[24] Gregor Gößler and Joseph Sifakis. Priority systems. In Frank S. de Boer, Marcello M. Bonsangue, Susanne Graf, and Willem P. de Roever, editors, *FMCO*, volume 3188 of *Lecture Notes in Computer Science*, pages 314–329. Springer, 2003.

[25] C. A. R. Hoare. *Communicating Sequential Processes.* Prentice Hall International Series in Computer Science. Prentice Hall, April 1985.

[26] Kohei Honda, VascoT. Vasconcelos, and Makoto Kubo. Language primitives and type discipline for structured communication-based programming. In Chris Hankin, editor, *Programming Languages and Systems*, volume 1381 of *LNCS*, pages 122–138. Springer Berlin Heidelberg, 1998.

[27] Paola Inverardi and Simone Scriboni. Connectors synthesis for deadlock-free component-based architectures. In *ASE '01*, pages 174–181, Washington, DC, USA, 2001. IEEE Computer Society.

[28] Sung-ShikT.Q. Jongmans, Christian Krause, and Farhad Arbab. Encoding context-sensitivity in reo into non-context-sensitive semantic models. In Wolfgang De Meuter and Gruia-Catalin Roman, editors, *Coordination Models and Languages*, volume 6721 of *LNCS*, pages 31–48. Springer Berlin Heidelberg, 2011.

[29] Zhiming Liu, Charles Morisset, and Volker Stolz. rcos: Theory and tool for component-based model driven development. In Farhad Arbab and Marjan Sirjani, editors, *Fundamentals of Software Engineering*, volume 5961 of *Lecture Notes in Computer Science*, pages 62–80. Springer Berlin Heidelberg, 2010.

[30] Robin Milner. *Communication and Concurrency*. Prentice Hall International Series in Computer Science. Prentice Hall, 1989.

[31] Robin Milner and Davide Sangiorgi. Barbed bisimulation. In W. Kuich, editor, *Automata, Languages and Programming*, volume 623 of *Lecture Notes in Computer Science*, pages 685–695. Springer Berlin Heidelberg, 1992.

[32] David M. R. Park. Concurrency and Automata on Infinite Sequences. *Proceedings of the 5th GI-Conference on Theoretical Computer Science*, pages 167–183, 1981.

[33] Gordon D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, University of Aarhus, 1981.

[34] Amir Pnueli and Roni Rosner. Distributed reactive systems are hard to synthesize. *Annual IEEE Symposium on Foundations of Computer Science*, 2:746–757, 1990.

[35] Jan J. M. M. Rutten. Universal coalgebra: a theory of systems. *Theor. Comput. Sci.*, 249(1):3–80, 2000.

[36] Joseph Sifakis. A framework for component-based construction. In 3^{rd} *IEEE Int. Conf. on Software Engineering and Formal Methods (SEFM05)*, pages 293–300, September 2005. Keynote talk.

[37] Pawel Sobocinski. A non-interleaving process calculus for multi-party synchronisation. In Filippo Bonchi, Davide Grohmann, Paola Spoletini, and Emilio Tuosto, editors, *ICE*, volume 12 of *EPTCS*, pages 87–98, 2009.

[38] Rob Van Ommering, Frank Van Der Linden, Jeff Kramer, and Jeff Magee. The Koala component model for consumer electronics software. *Computer*, 33(3):78–85, 2000.