arXiv:2007.10770v2 [cs.LO] 15 Sep 2020

# COMPOSITIONAL EQUIVALENCES BASED ON OPEN PNETS

RABÉA AMEUR-BOULIFA, LUDOVIC HENRIO, AND ERIC MADELAINE

LTCI, Télécom Paris, Institut Polytechnique de Paris, France
*e-mail address*: rabea.ameur-boulifa@telecom-paris.fr

Univ Lyon, EnsL, UCBL, CNRS, Inria, LIP, F-69342, LYON Cedex 07, France.
*e-mail address*: ludovic.henrio@cnrs.fr

INRIA Sophia Antipolis Méditérannée, UCA, BP 93, 06902 Sophia Antipolis, France
*e-mail address*: eric.madelaine@inria.fr

Abstract. Establishing equivalences between programs or system is crucial both for verifying correctness of programs, by establishing that two implementations are equivalent, and for justifying optimisations and program transformations, by establishing that a modified program is equivalent to the source one. There exist several equivalence relations for programs, and bisimulations are among the most versatile of these equivalences. Among bisimulation relations one distinguishes strong bisimulation, that requires that each action of a program is simulated by a single action of the equivalent program, a weak bisimulation that is a coarser relations, allowing some of the actions to be invisible or internal moves, and thus not simulated by the equivalent program.

pNets is a generalisation of automata that includes parameters, and hierarchical composition. Open pNets are pNets with holes, i.e. placeholders inside the hierarchical structure that can be filled by sub-systems. Reasoning on open pNets allows us to reason on open systems. Open pNets have a notion of *synchronised actions* generalizing the usual internal actions (e.g. $\tau$ of CCS, or $i$ in Lotos).

This article defines bisimulation relations for the comparison of systems specified as pNets. We first define a strong bisimulation for open pNets. In practice, as happens in process algebras, strong bisimulation is too strong, and we need to define some coarser relations, taking into account invisible or internal moves. We then define an equivalence relation similar to the classical *weak bisimulation*, and study its properties. Among these properties we are interested in compositionality: If two systems are proven equivalent they will be undistinguishable by their context, and they will also be undistinguishable when their holes are filled with equivalent systems. The article is illustrated with a transport protocol running example; it shows the characteristics of our formalism and our bisimulation relations.

## 1. Introduction

In the nineties, several works extended the basic behavioural models based on labelled transition systems to address value-passing or parameterised systems, using various symbolic encodings of the transitions [1, 2]. In [3, 4], Lin, Ingolfsdottir and Hennessy developed a full hierarchy of bisimulation equivalences, together with a proof system, for value passing CCS, including notions of symbolic behavioural semantics and various symbolic bisimulations

Preprint submitted to
Logical Methods in Computer Science

© Rabéa Ameur-Boulifa, Ludovic Henrio, and Eric Madelaine
Creative Commons

(early and late, strong and weak, and their congruent versions). They also extended this work to models with explicit assignments [5]. Separately J. Rathke [6] defined another symbolic semantics for a parameterised broadcast calculus, together with strong and weak bisimulation equivalences, and developed a symbolic model-checker based on a tableau method for these processes. 30 years later, no practical verification approach and no verification platform are using this kind of approaches to provide proof methods for value-passing processes or open process expressions.

This article provides a theoretical background that allows us to implement such a verification platform. We build upon the concept of pNets that allowed us to give a behavioural semantics of distributed components and verify the correctness of distributed applications in the past 15 years. pNets is a low level semantic framework for expressing the behaviour of various classes of distributed languages, and as a common internal format for our tools. pNets allow the specification of parameterised hierarchical labelled transition systems: labelled transition systems with parameters can be combined hierarchically.

We develop here a semantics for a model of interacting processes with parameters and holes. The main interest of our symbolic approach is to define a method to prove properties directly on open structures; these properties will then be preserved by any correct instantiation of the holes. As a consequence, our model allows us to reason on composition operators as well as on full-size distributed systems. The parametric nature of the model and the properties of compositionability of the equivalence relations are thus the main strengths of our approach.

*Previous Works and Contribution.* While most of our previous works relied on closed, fully-instantiated semantics [7, 8, 9], it is only recently that we could design a first version of a parameterised semantics for pNets with a strong bisimulation equivalence [10]. This article builds upon this previous parameterised semantics and provides a cleaner version of the semantics with a slightly simplified formalism. It also adds a notion of global state to automata. The slight simplification of the formalism allowed us to adopt a much cleaner presentation where the theory for open automata. The algebra used to define the semantics of pNets with holes, can be studied independently from its application to pNets. Also, in [10] the study of compositionability was only partial, and in particular the proof that bisimulation is an equivalence is one new contribution of this article and provides a particularly interesting insight on the semantic model we use. The new formalism allowed us to extend the work and define weak bisimulation for open automata, which is entirely new. This allows us to define a weak bisimulation equivalence for open pNets with valuable properties of compositionality. To summarise, the contribution of this paper are the following:

- The definition of open automata: an algebra of parameterised automata with holes, and a strong bisimulation equivalence. This is an adaptation of [10] with an additional property stating that strong bisimulation equivalence is indeed an equivalence relation.
- A semantics for open pNets expressed as translation to open automata. This is an adaptation of [10] with a complete proof that strong bisimulation is compositional.
- A theory of weak bisimulation for open automata, and its properties. It relies on the definition of weak open transitions that are derived from transitions of the open automaton by concatenating invisible action transitions with one (visible or not) action transition. The precise and sound definition of the concatenation is also a major contribution of this article.

- A resulting weak bisimulation equivalence for open pNets and a simple static condition on synchronisation vectors inside pNets that is sufficient to ensure that weak bisimulation is compositional.
- An illustrative example based on a simple transport protocol, showing the construction of the weak open transitions, and the proof of weak bisimulation.

*Structure.* This article is organised as follows. Section 2 provides the definition of pNets and introduces the notations used in this paper, including the definition of open pNets. Section 3 defines open automata, i.e. automata with parameters and transitions conditioned by the behaviour of "holes"; a strong bisimulation equivalence for open automata is also presented in this section. Section 4 gives the semantics of open pNets expressed as open automata, and states compositional properties on the strong bisimulation for open pNets. Section 5 defines a weak bisimulation equivalence on open automata and derives weak bisimilarity for pNets, together with properties on compositionality of weak bisimulaiton for open pNets. Finally, Section 6 discusses related works and Section 7 concludes the paper.

## 2. Background and notations

This section introduces the notations we will use in this article, and recalls the definition of pNets [10] with an informal semantics of the pNet constructs. The only significant difference compared to our previous definitions is that we remove here the restriction that was stating that variables should be local to a state of a labelled transition system.

### 2.1. **Notations.**

*Term algebra.* Our models rely on a notion of parameterised actions, that are symbolic expressions using data types and variables. As our model aims at encoding the low-level behaviour of possibly very different programming languages, we do not want to impose one specific algebra for denoting actions, nor any specific communication mechanism. So we leave unspecified the constructors of the algebra that will allow building expressions and actions. Moreover, we use a generic *action interaction* mechanism, based on (some sort of) unification between two or more action expressions, to express various kinds of communication or synchronisation mechanisms.

Formally, we assume the existence of a term algebra $\mathbb{T}$, where $\Sigma$ is the signature of the data and action constructors. Within $\mathbb{T}$, we distinguish a set of expressions $\mathbb{E}$, including a set of boolean expressions $\mathbb{E}$ ($\mathbb{B} \subseteq \mathbb{E}$). On top of $\mathbb{E}$ we build the action algebra $\mathbb{A}$, with $\mathbb{A} \subseteq \mathbb{T}, \mathbb{E} \cap \mathbb{A} = \emptyset$; naturally action terms will use data expressions as sub-terms. The function $vars(t)$ identifies the set of variables in a term $t \in \mathbb{T}$.

We let $e_i$ range over expressions ($e_i \in \mathbb{E}$), $a$ range over action labels, $op$ be operators, and $x_i$ and $y_i$ range over variable names.

We distinguish two kinds of parameterised actions: one that distinguishes input variables and one that does not. We first define the set of actions that distinguish input variables, they will be used in the definition of *pLTS* below:

$$\begin{array}{rcll} \alpha \in \mathbb{A} & ::= & a(p_1, \ldots, p_n) & \text{action terms} \\ p_i & ::= & ?x \mid e_i & \text{parameters (input variable or expression)} \\ e_i & ::= & Value \mid x \mid op(e_1, .., e_n) & \text{Expressions} \end{array}$$

The *input variables* in an action term are those marked with a *?*. We additionally suppose that each input variable does not appear somewhere else in the same action term: $p_i = ?x \Rightarrow \forall j \neq i.\, x \notin vars(p_j)$. We define $iv(t)$ as the set of input variables of a term $t$ (without the '?' marker). Action algebras can encode naturally usual point-to-point message passing calculi (using $a(?x_1, ..., ?x_n)$ for inputs, $a(v_1, .., v_n)$ for outputs), but it also allows for more general synchronisation mechanisms, like gate negociation in Lotos, or broadcast communications.

The set of actions that do not distinguish input variables is denoted $\mathbb{A}_S$, it will be used in synchronisation vectors of pNets:

$$\alpha \in \mathbb{A}_S \quad ::= \quad a(e_1, \ldots, e_n)$$

*Indexed sets.* In this article, we extensively use indexed structures (maps) over some countable indexed sets. The indices can typically be integers, bounded or not. We use indexed sets in pNets because we want to consider a set of processes, and specify separately how to synchronise them. Roughly this could also be realised using tuples, however indexed sets are more general, can be infinite, and give a compact representation than using the position in a possibly long tuple.

An indexed family is denoted as follows: $a_i^{i \in I}$ is a family of elements $a_i$ indexed over the set $I$. Such a family is equivalent to the mapping $(i \mapsto a_i)^{i \in I}$, and we will also use mapping notations to manipulate indexed sets. To specify the set over which the structure is indexed, indexed structures are always denoted with an exponent of the form $i \in I$.

Consequently, $a_i^{i \in I}$ defines first $I$ the set over which the family is indexed, and then $a_i$ the elements of the family. For example $a^{i \in \{3\}}$ is the mapping with a single entry $a$ at index 3; exceptionally, such mappings with only a few entries will also be denoted $(3 \mapsto a)$.

When this is not ambiguous, we shall use abusive vocabulary and notations for sets, and typically write "indexed set over I" when formally we should speak of multisets, and "$x \in A_i^{i \in I}$" to mean $\exists i \in I.\, x = A_i$. To simplify equations, an indexed set can be denoted $\overline{a}$ instead of $a_i^{i \in I}$ when $I$ is irrelevant.

$\uplus$ is the disjoint union on sets. We extend it to disjoint union of indexed sets defined by the merge of the two sets provided they are indexed on disjoint families. The elements of the union of two indexed sets are then accessed by using an index of one of the two joined families. $\setminus$ is the standard subtraction operation on indexed sets: $\mathrm{dom}(A \setminus B) = \mathrm{dom}(A) \setminus B$.

*Substitutions.* We denote $y \leftarrow e$ a substitution. The application of the substitution is denoted $\{\!\{y \leftarrow e\}\!\}$, the operation replaces in a term all occurrences of the variable $y$ by the expression $e$. *Post* ranges over (indexed) sets of substitutions; $\{\!\{Post\}\!\}$ is the substitution that applies all the substitutions defined by *Post* in a parallel manner. $\otimes$ is the composition operator on substitutions, such that for any term $t$ we have: $t\{\!\{Post \otimes Post'\}\!\} = (t\{\!\{Post'\}\!\})\{\!\{Post\}\!\}$.

For this property to be valid even if the substitution does not operate on all variables, we define the composition operation as follows:

$$(x_k \leftarrow e_k)^{k \in K} \otimes (x'_{k'} \leftarrow e'_{k'})^{k' \in K'} = (x_k \leftarrow e_k\{\!\{(x'_{k'} \leftarrow e'_{k'})^{k' \in K'}\}\!\})^{k \in K} \cup (x'_{k'} \leftarrow e'_{k'})^{k' \in K''}$$

where $K'' = \{k' \in K' | x'_{k'} \notin \{x_k\}^{k \in K}\}$

2.2. **Parameterised Networks (pNets).** pNets are tree-like structures, where the leaves are either *parameterised labelled transition systems (pLTSs)*, expressing the behaviour of basic processes, or *holes*, used as placeholders for unknown processes. Nodes of the tree (pNet nodes) are synchronising artefacts, using a set of *synchronisation vectors* that express the possible synchronisation between the parameterised actions of a subset of the sub-trees.

A pLTS is a labelled transition system with variables; variables can be used inside states, actions, guards, and assignments. Note that we make no assumption on finiteness of the set of states nor on finite branching of the transition relation. Compared to our previous works [10, 8] we extend the expressiveness of the model by making variables global.

**Definition 1** pLTS. *A pLTS is a tuple* $pLTS \triangleq \langle\langle S, s_0, V, \rightarrow \rangle\rangle$ *where:*

- $S$ *is a set of states.*
- $s_0 \in S$ *is the initial state.*
- $V$ *is a set of global variables for the pLTS,*
- $\rightarrow \subseteq S \times L \times S$ *is the transition relation and* $L$ *is the set of labels of the form:* $\langle \alpha,\ e_b,\ (x_j := e_j)^{j\in J} \rangle$, *where* $\alpha \in \mathbb{A}$ *is a parameterised action,* $e_b \in \mathbb{B}$ *is a guard, and the variables* $x_j$ *are assigned the expressions* $e_j \in \mathbb{E}$. *If* $s \xrightarrow{\langle \alpha,\ e_b,\ (x_j := e_j)^{j\in J} \rangle} s' \in \rightarrow$ *then* $vars(\alpha) \backslash iv(\alpha) \subseteq V$, $vars(e_b) \subseteq vars(s) \cup vars(\alpha)$, *and* $\forall j \in J.\ (vars(e_j) \subseteq V \cup iv(\alpha) \wedge x_j \in V)$.

The semantics of the assignments is that a set of assignments between two states is performed in parallel so that their order do not matter and they all use the values of variables before the transition (or the values received as action parameters).

Now we define pNet nodes as constructors for hierarchical behavioural structures. A pNet has a set of sub-pNets that can be either pNets or pLTSs, and a set of holes, playing the role of process parameters. A pNet is thus a composition operator that can receive processes as parameters; it expresses how the actions of the sub-processes synchronise.

Each sub-pNet exposes a set of actions, called *internal actions*. The synchronisation between global actions exposed by the pNet and internal actions of its sub-pNets is given by *synchronisation vectors*: a synchronisation vector synchronises one or several internal actions, and exposes a single resulting global action.

We now define the structure of pNets, the following definition relies on the definition of holes, leaves and sorts formalised below in Definition 3. Informally, holes are process parameters, leaves provide the set of pLTSs at the leaves of the hierarchical structure of a pNet, and sorts give the signature of a pNet, i.e. the actions it exposes.

**Definition 2** pNets. *A pNet* $P$ *is a hierarchical structure where leaves are pLTSs and holes:* $P \triangleq pLTS \mid \langle\langle P_i^{i\in I}, Sort_j^{j\in J}, SV_k^{k\in K} \rangle\rangle$ *where*

- $P_i^{i\in I}$ *is the family of sub-pNets indexed over* $I$. $vars(P_i)$ *and* $vars(P_j)$ *must be disjoint for* $i \neq j$.
- $J$ *is a set of indexes, called* holes. $I$ *and* $J$ *are* disjoint*:* $I \cap J = \emptyset$, $I \cup J \neq \emptyset$
- $Sort_j \subseteq \mathbb{A}_S$ *is a set of action terms, denoting the* sort *of hole* $j$.
- $SV_k^{k\in K}$ *is a set of synchronisation vectors.* $\forall k \in K.\ SV_k = \alpha_l^{l\in I_k \uplus J_k} \rightarrow \alpha'_k[e_k]$ *where* $\alpha'_k \in \mathbb{A}_S$, $I_k \subseteq I$, $J_k \subseteq J$, $\forall i \in I_k.\ \alpha_i \in Sort(P_i)$, $\forall j \in J_k.\ \alpha_j \in Sort_j$, *and* $vars(\alpha'_k) \subseteq \bigcup_{l\in I_k \uplus J_k} vars(\alpha_l)$. *The global action of a vector* $SV_k$ *is* $\alpha'_k$. $e_k \in \mathbb{B}$ *is a guard associated to the vector such that* $vars(e_k) \subseteq \bigcup_{l\in I_k \uplus J_k} vars(\alpha_l)$.

*Synchronisation vectors are identified modulo renaming of variables that appear in their action terms.*

The preceding definition relies on the auxiliary functions defined below:

**Definition 3** Sorts, Holes, Leaves, Variables of pNets**.**

- *The sort of a pNet is its signature, i.e. the set of actions in $\mathbb{A}_S$ it can perform, where each action signature is an action label plus the arity of the action. In the definition of sorts, we do not need to distinguish input variables and we remove the* input marker *(?) of variables.*

$$\mathrm{Sort}(\langle\!\langle S, s_0, V, \rightarrow\rangle\!\rangle) = \{\alpha\{\!\{?x \leftarrow x | x \in iv(\alpha)\}\!\} | s \xrightarrow{\langle \alpha,\ e_b,\ (x_j := e_j)^{j\in J}\rangle} s' \in \rightarrow\}$$
$$\mathrm{Sort}(\langle\!\langle \overline{P}, \overline{Sort}, \overline{SV}\rangle\!\rangle) = \{\alpha' | \, \overline{\alpha} \rightarrow \alpha'[e_b] \in \overline{SV}\}$$

- *The set of variables of a pNet $P$, denoted $vars(P)$ is disjoint union the set of variables of all pLTSs that compose $P$.*
- *The set of holes* $\mathrm{Holes}(P)$ *of a pNet is the indexes of the holes of the pNet itself plus the indexes of all the holes of its sub-pNets. It is defined inductively (we suppose those indexes disjoints):*

$$\mathrm{Holes}(\langle\!\langle S, s_0, V, \rightarrow\rangle\!\rangle) = \emptyset$$
$$\mathrm{Holes}(\langle\!\langle P_i^{i\in I}, \overline{Sort}, \overline{SV}\rangle\!\rangle) = J \uplus \bigcup_{i\in I} \mathrm{Holes}(P_i)$$
$$\forall i \in I.\ \mathrm{Holes}(P_i) \cap J = \emptyset$$
$$\forall i_1, i_2 \in I.\, i_1 \neq i_2 \Rightarrow \mathrm{Holes}(P_{i_1}) \cap \mathrm{Holes}(P_{i_2}) = \emptyset$$

- *The set of leaves of a pNet is the set of all pLTSs occurring in the structure, as an indexed family of the form* $\mathrm{Leaves}(P) = \langle\!\langle P_i\rangle\!\rangle^{i\in L}$.

$$\mathrm{Leaves}(\langle\!\langle S, s_0, V, \rightarrow\rangle\!\rangle) = \emptyset$$
$$\mathrm{Leaves}(\langle\!\langle P_i^{i\in I}, \overline{Sort}, \overline{SV}\rangle\!\rangle) = \biguplus_{i\in I} \mathrm{Leaves}(P_i) \uplus \{i \mapsto P_i | P_i \textit{ is a pLTS}\}$$

*A pNet $Q$ is* closed *if it has no hole:* $\mathrm{Holes}(Q) = \emptyset$*; else it is said to be* open.

The informal semantics of pNets is as follows. pLTSs behave more or less like classical automata with conditional branching and variables. The actions on the *pLTS*s can send or receive values, potentially modifying the value of variables. pNets are synchronisation entities: a pNet composes several sub-pNets and synchronisation vectors define how the sub-pNets interact, where a sub-pNet is either a pNet or a pLTS. Synchronisation between sub-pNets is defined by synchronisation vectors that express how an action of a sub-pNet can be synchronised with actions of other sub-pNet, and how the resulting synchronised action is visible to the outside of the pNet. Synchronisation vectors can also express the exportation of an action of a sub-pNet to the next level, or to hide an interaction and make it non-observable. Finally, a pNet can leave sub-pNets undefined and instead declare holes with a well-defined signature. Holes can then be filled with a sub-pNet. This is defined as follows.

**Definition 4** pNet composition**.** *An open pNet: $P = \langle\!\langle P_i^{i\in I}, Sort_j^{j\in J}, \overline{SV}\rangle\!\rangle$ can be (partially) filled by providing a pNet $Q$ of the right sort to fill one of its holes. Suppose $j_0 \in J$:*

$$P\,[Q]_{j_0} = \langle\!\langle P_i^{i\in I} \uplus \{j_0 \mapsto Q\}, Sort_j^{j\in J\setminus\{j_0\}}, \overline{SV}\rangle\!\rangle$$

pNets are composition entities equipped with a rich synchronisation mechanism: synchronisation vectors allow the expression of synchronisation between any number of entities and at the same time the passing of data between processes. Their strongest feature is that the data emitted by processes can be used can be used inside the synchronisation vector to do addressing: it is easy to synchronise a process indexed by $n$ with the action $a(v,n)$ of another process. This is very convenient to model systems and encode futures or message routing. pNets have been used to model GCM distributed component systems, illustrating the expressiveness of the model [8]. These works show that pNets are convenient to express the behaviour of the system in a compositional way, which is crucial for the definition of the semantics, especially when dealing with a hierarchical component system like GCM. Unfortunately, the semantics of pNets and the existing tools at this point were only able to deal with a closed system completely instantiated: pNets could be used as composition operator in the definition of the semantics, which was sufficient to perform finite-state model checking on a closed system, but there was no theory for the use of pNets as operators and no tool for proving properties on open system. The semantics of closed pNets [8], defined as an instantiation of labelled transition systems, is not necessary to understand this article but can be useful to understand the semantics of pNets in a simpler and more operational setting. The theory of pNets as operators able to fully take into account open systems is given in the following sections.

2.3. **Running Example.** To illustrate this work, we use a simple communication protocol, that provides safe transport of data between two processes, over unsafe media.

Figure 1 (left) shows the example principle, which corresponds to the hierarchical structure of a pNet: two unspecified processes $P$ and $Q$ (holes) communicate messages, with a data value argument, through the two protocol entities. Process $P$ sends an `p-send(m)` message to the *Sender*; this communication is denoted as `in(m)`. At the other end, process $Q$ receives the message from the *Receiver*. The holes $P$ and $Q$ can also have other interactions with their environment, represented here by actions `p-a` and `q-b`. The underlying network is modelled by a medium entity transporting messages from the sender to the receiver, and that is able to detect transport errors and signal them to the sender. The return *ack* message from *Receiver* to *Sender* is supposed to be safe. The final transmission of the message to the recipient (the hole $Q$) includes the value of the "error counter" *ec*.

Figure 1 (right) shows a graphical view of the pNet that specifies the system: the full system should be equivalent (e.g. weakly bisimilar) to the two processes connected simply through a perfect medium. The pNet has a tree-like structure. The root node of the tree *SimpleSystemSpec* is the top level of the pNet structure. It acts as the parallel operator. It consists of three nodes: two holes $P$ and $Q$ and one sub-pNet, denoted *PrefectBuffer*. Nodes of the tree are synchronized using four synchronization vectors, that express the possible synchronizations between the parameterised actions of a subset of the nodes. For instance, in the vector "$<$ `p-send(m)`, `in(m)`, _ $>\rightarrow$ `in(m)`" only $P$ and *PrefectBuffer* nodes are involved in the synchronization. The synchronization between these processes occurs when process $P$ performs `p-send(m)` action sending a message, and the *PrefectBuffer* accepts the message through an `in(m)` action at the same time; the result that will be returned at upper level is the action `in(m)`.

Figure 2 shows the pNet model of the protocol implementation. When the *Medium* detects an error (modelled by a local $\tau$ action), it sends back a `m-error` message to the *Sender*. The *Sender* increments its local error counter *ec*, and resends the message (including *ec*) to the *Medium*, that will, eventually, transmit $m, ec$ to the *Receiver*.

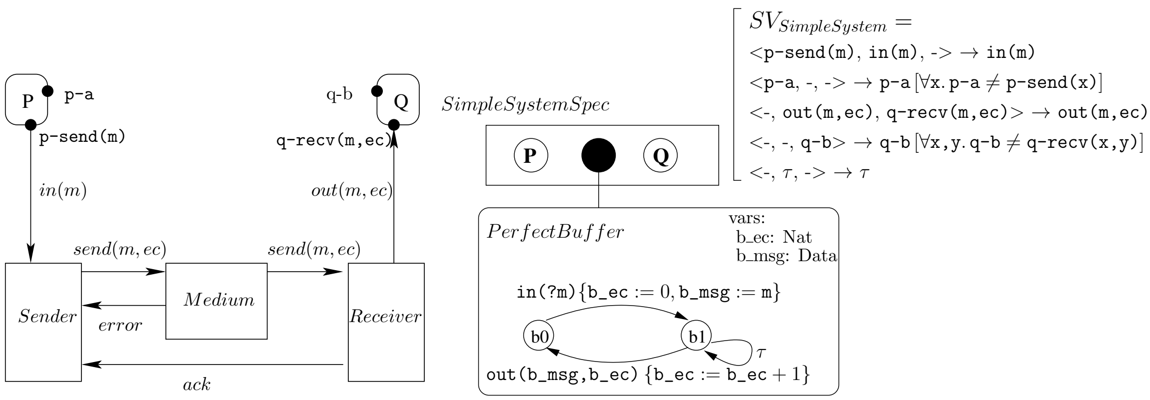



Figure 1: pNet structure of the example and specification expressed as a pNet

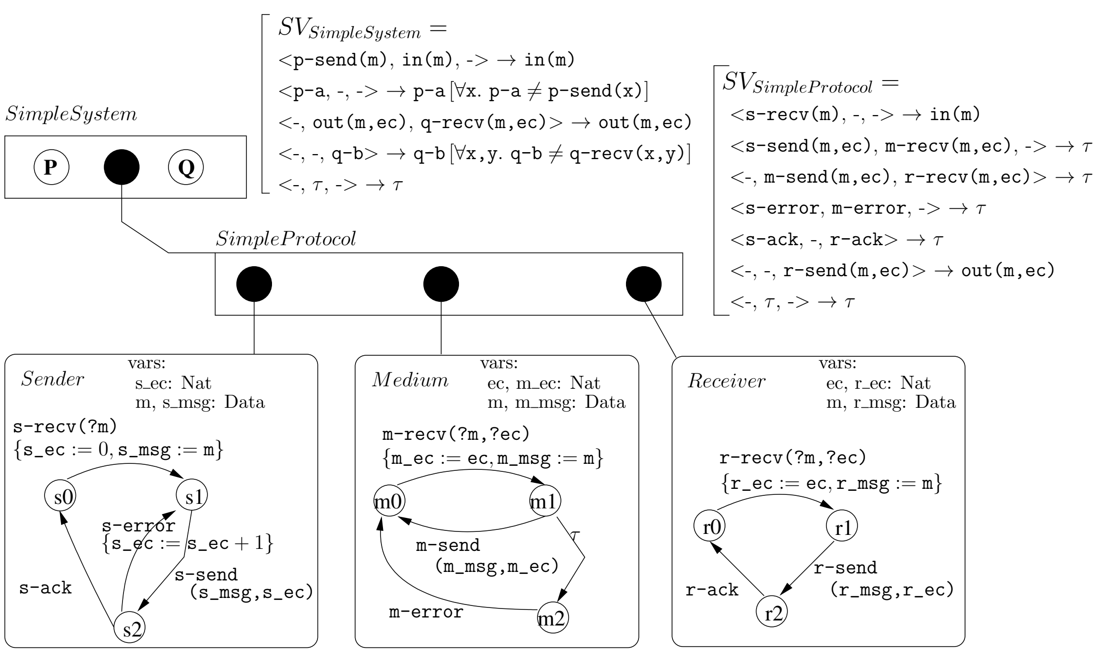



Figure 2: Composed pNet with the Simple Protocol Implementation

## 3. A model of process composition

The semantics of open pNets will be defined as an open automaton. An open automaton is an automaton where each transition composes transitions of several LTSs with action of some holes, the transition occurs if some predicates hold, and can involve a set of state modifications. This section defines open automata and a bisimulation theory for them. This section can be viewed as an improved version of the formalism described in [10], extending the automata with a notion of global variable, which makes the state of the automaton more explicit. We also adopt a semantics and logical interpretation of the automata that intuitively can be stated as follows: "if a transition belongs to an open automaton, any refinement of this transition also belongs to the automaton".

3.1. **Open Automata.** Open automata (OA) are not composition structures but they are made of transitions that are dependent of the actions of the holes, and they can reason on a set of variables (potentially with only symbolic values).

**Definition 5** Open transitions**.** *An* open transition *(OT) over a set $J$ of holes with sorts $Sort_j^{j\in J}$, a set $V$ of variables, and a set of states $\mathcal{S}$ is a structure of the form:*

$$\frac{\beta_j^{j\in J'}, Pred, Post}{s \xrightarrow{\alpha} s'}$$

*Where $J' \subseteq J$, $s, s' \in \mathcal{S}$ and $\beta_j$ is a transition of the hole $j$, with $\beta_j \in Sort_j$. $\alpha$ is an action label denoting the resulting action of this open transition. Pred is a predicate over the variables in $V$ and all variables in the different terms $\beta_j$ and $\alpha$. Post is a set of assignments that are effective after the open transition, they are represented as a substitution of the form $(x_k \leftarrow e_k)^{k\in K}$ where $\forall k. x_k \in V$, and $e_k$ are expressions over the variables $V$ and all variables in the different terms $\beta_j$ and $\alpha$. The assignments are always applied simultaneously because the variables in $V$ can be in both sides. Open transitions are identified modulo logical equivalence on their predicate.*

It is important to understand the difference between the red dotted rule and a normal inference rule. They correspond to two different logical levels. On one side, classical (black) inference rules use an expressive logic and are paper rules. On the other side, open transition rules (with dotted lines) are logical implications, but using a simple logic (this logic includes the boolean expressions $\mathbb{B}$, boolean operators, and term equality). open transitions could typically be handled in a mechanised way.

An open automaton is an automaton where each transition is an open transition.

**Definition 6** Open automaton**.** *An* open automaton *is a structure*
*$A = \langle\langle J, \mathcal{S}, s_0, V, \mathcal{T}\rangle\rangle$ where:*
- *$J$ is a set of indices.*
- *$\mathcal{S}$ is a set of states and $s_0$ an initial state among $\mathcal{S}$.*
- *$V$ is a set of variables of the automaton and each $v \in V$ may have an initial value $init(v)$.*
- *$\mathcal{T}$ is a set of open transitions and for each $t \in \mathcal{T}$ there exist $J'$ with $J' \subseteq J$, such that $t$ is an open transition over $J'$ and $\mathcal{S}$.*

*We take in this article a semantics and logical understanding of these automata. Open automata are closed by a simple form of refinement that allows us to refine the predicate, or substitute any free variable by an expression. More formally, let Pred be any predicate and Post any substitution such that $V \cap \mathrm{dom}(Post) = \emptyset$. Then we have the following implication:*

$$\frac{\overline{\beta}, Pred', Post'}{t \xrightarrow{\alpha} t'} \in \mathcal{T} \implies \frac{\overline{\beta}\{\!\{Post\}\!\}, Pred'\{\!\{Post\}\!\} \wedge Pred, Post \otimes Post'}{t \xrightarrow{\alpha\{\!\{Post\}\!\}} t'} \in \mathcal{T}$$

Because of the semantic interpretation of open automata, the set of open transition of an open automaton is infinite (for example because every free variable can be renamed). However an open automaton is characterized by a subset of these open transition which is sufficient to generate, by substitution the other ones. In the following, we will abusively write that we define an “open automaton” when we provide only the set of open transitions that is sufficient to generate a proper open automaton by saturating each open transition by all possible substitutions and refinements.

Another consequence of the semantics and logical interpretation of the formulas is that we make no distinction between the equality and the equivalence on boolean formulas, i.e. equivalence of two predicates $Pred$ and $Pred'$ can be denoted $Pred = Pred'$.

Though the definition is simple, the fact that transitions are complex structures relating events must not be underestimated in order to understand the rest of the article. The first element of theory for open automata, i.e. the definition of a strong bisimulation, is given below.

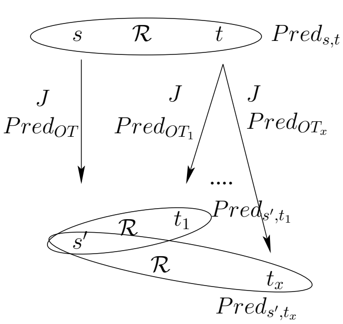
### 3.2. **Bisimulation for open Automata.**

The equivalence we need is a strong bisimulation between open automata having exactly the same holes (same indexes and same sorts), but using a flexible matching between open transitions, this will allow us to compare pNets with different architectures.

We define now a bisimulation relation adapted to open automata and their parametric nature. The relation relates states of the open automaton and states equivalence between the open transitions between the states. Its key characteristics are 1) the introduction of predicates in the bisimulation relation: as states may contain variables, relation between states may depend on the value of the variables; 2) the bisimulation property relates elements of the open transitions and take into account predicates over variables, actions of the holes, and state modifications. We name it FH-bisimulation, as a short cut for the "Formal Hypotheses" over the holes behaviour manipulated in the transitions, but also as a reference to the work of De Simone [1], that pioneered this idea.

A relation over the states of two open automata $\langle\langle J, \mathcal{S}_1, s_0, V_1, \mathcal{T}_1\rangle\rangle$ and $\langle\langle !J, \mathcal{S}_2, t_0, V_2, \mathcal{T}_2\rangle\rangle$ has the form $\mathcal{R} = \{(s,t|Pred_{s,t})\}$, it relates states of $\mathcal{S}_1$ and $\mathcal{S}_2$ constrained by a predicate. More precisely, for any pair $(s,t) \in \mathcal{S}_1 \times \mathcal{S}_2$, there is a single $(s,t|Pred_{s,t}) \in \mathcal{R}$ stating that $s$ and $t$ are related if $Pred_{s,t}$ is True, i.e. the states are related when the variables in $V_1$ and $V_2$ verify the predicate $Pred_{s,t}$. By nature, this is well-defined if the variables of the two open automata are disjoint, i.e. $V_1 \cap V_2 = \emptyset$. FH-bisimulation is defined formally:

**Definition 7** Strong FH-bisimulation**.**




*Suppose $A_1 = \langle\langle J, \mathcal{S}_1, s_0, V_1, \mathcal{T}_1\rangle\rangle$ and $A_2 = \langle\langle J, \mathcal{S}_2, t_0, V_2, \mathcal{T}_2\rangle\rangle$ are open automata with identical holes of the same sort, with disjoint sets of variables.*
*Then $\mathcal{R}$ is an FH-bisimulation if and only if for any states $s \in \mathcal{S}_1$ and $t \in \mathcal{S}_2$, $(s,t|Pred_{s,t}) \in \mathcal{R}$, we have the following:*

- *For any open transition $OT$ in $\mathcal{T}_1$:*

$$\frac{\beta_j^{j\in J'}, Pred_{OT}, Post_{OT}}{s \xrightarrow{\alpha} s'}$$

*there exist open transitions* $OT_x^{x\in X} \subseteq \mathcal{T}_2$*:*

$$\frac{\beta_{jx}^{j\in J_x}, Pred_{OT_x}, Post_{OT_x}}{t \xrightarrow{\alpha_x} t_x}$$

*such that* $\forall x.\ J' = J_x$ *and there exists* $Pred_{s',t_x}$ *such that* $(s', t_x | Pred_{s',t_x}) \in \mathcal{R}$ *and* $Pred_{s,t} \wedge Pred_{OT} \implies$

$$\bigvee_{x\in X} (\forall j.\beta_j = \beta_{jx} \wedge Pred_{OT_x} \wedge \alpha = \alpha_x \wedge Pred_{s',t_x}\{\!\{Post_{OT} \uplus Post_{OT_x}\}\!\})$$

- *and symmetrically any open transition from* $t$ *in* $\mathcal{T}_2$ *can be covered by a set of transitions from* $s$ *in* $\mathcal{T}_1$*.*

Classically, $Pred_{s',t_x}\{\!\{Post_{OT} \uplus Post_{OT_x}\}\!\}$ applies in parallel the substitutions $Post_{OT}$ and $Post_{OT_x}$ (parallelism is crucial inside each *Post* set but not between $Post_{OT}$ and $Post_{OT_x}$ that are independent), applying the assignments of the involved rules. We can prove that such a bisimulation is an equivalence relation.

**Theorem 1** FH-Bisimulation is an equivalence**.** *Suppose* $\mathcal{R}$ *is an FH-bisimualtion. Then* $\mathcal{R}$ *is an equivalence, that is,* $\mathcal{R}$ *is reflexive, symmetric and transitive.*

The proof of this theorem can be found in Annex A.1. The only non-trivial part of the proof is the proof of transitivity. It relies on the following elements. First, the transitive composition of two relations with predicate is defined; this is not exactly standard as it requires to define the right predicate for the transitive composition and producing a single predicate to relate any two states. Then the fact that one open transition is simulated by a family of open transitions leads to a doubly indexed family of simulating open transition; this needs particular care, also because of the use of renaming (*Post*) when proving that the predicates satisfy the definition (property on $Pred_{s,t} \wedge Pred_{OT}$ in the definition).

*Finite versus infinite open automata, and decidability:* As mentioned in Definition 9, we adopt here a semantic view on open automata. More precisely, in [11], we formerly define *semantic open automata* (infinite as in Definition 6), and *structural open automata* (finite) that can be generated as the semantics of pNets (see Definition 9), and used in the implementation. Then we define an alternative version of our bisimulation, called structural-FH-Bisimulation, based on structural open automata, and prove that the *semantic* and *structural* FH-Bisimulations coincide. In the sequel, all mentions of finite automata, and algorithms for bisimulations, implicitly refer to their *structural* versions.

If we assume that everything is finite (states and transitions in the open automata, and the predicates in $\mathcal{R}$), then it is easy to prove that it is decidable whether a relation is a FH-bisimulation, provided the logic of the predicates is decidable (proof can be found in [10]). Formally:

**Theorem 2** Decidability of FH-bisimulation**.** *Let* $A_1$ *and* $A_2$ *be finite open automata and* $\mathcal{R}$ *a relation over their states* $\mathcal{S}_1$ *and* $\mathcal{S}_2$ *constrained by a finite set of predicates. Assume that the predicates inclusion is decidable over the action algebra* $\mathbb{A}$*. Then it is decidable whether the relation* $\mathcal{R}$ *is a FH-bisimulation.*

## 4. Semantics of Open pNets

This section defines the semantics of an open pNet as a translation into an open automaton. In this translation, the states of the open automata are obtained from the states of the pLTSs at the leaves of the composition. The predicates on the transitions are obtained both from the predicates on the pLTSs transitions and from the synchronisation vectors involved in the transition.

The definition of bisimulation for open automata allows us to derive a bisimulation theory for open pNets. As pNets are composition structures, it then makes sense to prove composition lemmas: we prove that the composition of strongly bisimilar pNets are themselves bisimilar.

### 4.1. Deriving an open automaton from an open pNet.

To derive an open automaton from a pNet, we need to describe the set of states of the automaton, and then we will detail the construction rule for transitions of the automaton, this will rely on the derivation of predicate unifying synchronisation vectors and the actions of the pNets involved in a given synchronisation.

We first define states of open pNets as tuples of states. We denote them as $\triangleleft \dots \triangleright$ for distinguishing tuple states from other tuples.

**Definition 8** States of open pNets**.** *A state of an open pNet is a tuple (not necessarily finite) of the states of its leaves.*

*For any pNet $P$, let* $\mathrm{Leaves}(P) = \langle\!\langle S_i, s_{i0}, V, \to_i \rangle\!\rangle^{i \in L}$ *be the set of pLTS at its leaves, then* $States(P) = \{\triangleleft s_i^{i\in L} \triangleright | \forall i \in L. s_i \in S_i\}$*. A pLTS being its own single leave:* $States(\langle\!\langle S, s_0, V, \to \rangle\!\rangle) = \{\triangleleft s \triangleright | s \in S\}$.

*The initial state is defined as:* $InitState(P) = \triangleleft {s_{i0}}^{i\in L} \triangleright$.

To be precise, the state of each pLTS is entirely characterised by both the state of the automaton, and the value of its variables $V$. Consequently, the complete characterization of the state of a pNet should take into account the value of its variables $vars(P)$.

*Predicates.* Consider a synchronisation vector $(\alpha'_i)^{i\in I}, (\beta'_j)^{j\in J} \to \alpha'[e_b]$. We define a predicate $Pred_{sv}$ relating the actions of the involved sub-pNets and the resulting actions. This predicate verifies:

$$Pred_{sv}\Big((\alpha'_i)^{i\in I}, (\beta'_j)^{j\in J} \to \alpha'[e_b], \alpha_i^{i\in I}, \beta_j^{j\in J}, \alpha\Big) \Leftrightarrow \\ \forall i \in I.\, \alpha_i = \alpha'_i \wedge \forall j \in J.\, \beta_j = \beta'_j \wedge \alpha = \alpha' \wedge e_b$$

Somehow, this predicate entails a verification of satisfiability in the sense that if the predicate $Pred_{sv}$ is not satisfiable, then the transition associated with the synchronisation will not occur in the considered state, or will occur with a *False* precondition which is equivalent. If the action families do not match or if there is no valuation of variables such that the above formula can be ensured then the predicate is undefined.

The definition of this predicate is not constructive but it is easy to build the predicate constructively by brute-force unification of the sub-pNets actions with the corresponding vector actions, possibly followed by a simplification step.

**Example 1** An open-transition**.** *At the upper level, the SimpleSystem pNet of Figure 2 has 2 holes and SimpleProtocol as a sub-pNet, itself containing 3 pLTSs. One of its possible open transitions (synchronizing the hole $P$ with the Sender within the* SimpleProtocol*) is:*

$$OT_1 = \frac{\{P\mapsto \texttt{p-send(m)}\}, [\texttt{m=m'}], (\texttt{s\_msg} \leftarrow \texttt{m})}{\triangleleft s_0, m_0, r_0 \triangleright \xrightarrow{\texttt{in(m')}} \triangleleft\, s_1, m_0, r_0 \triangleright}$$

*The global states here are triples, build as the product of states of the 3 pLTSs (remember the holes have no state). The assignment Post uses the variable* `m` *from the action of hole* `P` *to set the value of the sender variable named* `s_msg`*.*

We build the semantics of open pNets as an open automaton over the states given by Definition 8. The open transitions first project the global state into states of the leaves, then apply pLTS transitions on these states, and compose them with the sort of the holes. The semantics instantiates fresh variables using the predicate fresh$(x)$, additionally, for an action $\alpha$, fresh$(\alpha)$ means all variables in $\alpha$ are fresh.

**Definition 9** Semantics of open pNets**.** *The semantics of a pNet $P$ is an open automaton $A = \langle\!\langle Holes(P), States(P), InitState(P), vars(P), \mathcal{T}\rangle\!\rangle$ where $\mathcal{T}$ is the smallest set of open transitions such that $\mathcal{T} = \{OT \mid P \models OT\}$ and $P \models OT$ is defined by the following rules:*

- *The rule for a pLTS checks that the guard is verified and transforms assignments into post-conditions:*

$$\frac{s \xrightarrow{\langle \alpha,\ e_b,\ (x_j{:=}e_j)^{j\in J}\rangle} s' \in \rightarrow}{\langle\!\langle S, s_0, \rightarrow\rangle\!\rangle \models \dfrac{\emptyset, e_b, \{x_j \leftarrow e_j\}^{j\in J}}{\triangleleft s \triangleright \xrightarrow{\alpha} \triangleleft\, s' \triangleright}} \quad \textbf{\textit{Tr1}}$$

- *The second rule deals with pNet nodes: for each possible synchronisation vector (of index $k$) applicable to the rule subject, the premisses include one* open transition *for each sub-pNet involved, one possible* action *for each hole involved, and the predicate relating these with the resulting action of the vector. The sub-pNets involved are split between two sets, $I_2$ for sub-pNets that are pLTSs, and $I_1$ for the others, $J$ is the set of holes involved in the transition*[1]*.*

$$\frac{\begin{array}{c}
\text{Leaves}(\langle\!\langle P_m^{m\in I}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle) = pLTS_l^{l\in L} \quad k\in K \quad SV_k = (\alpha'_m)^{m\in I_1\uplus I_2\uplus J} \rightarrow \alpha'[e_b] \\
\forall m{\in}I_1.P_m \models \dfrac{\beta_j^{j\in J_m}, Pred_m, Post_m}{\triangleleft s_i^{i\in L_m} \triangleright \xrightarrow{\alpha_m} \triangleleft\, (s'_i)^{i\in L_m} \triangleright} \qquad \forall m{\in}I_2.P_m \models \dfrac{\emptyset, Pred_m, Post_m}{\triangleleft s_m \triangleright \xrightarrow{\alpha_m} \triangleleft\, s'_m \triangleright} \\
J' = \biguplus_{m\in I_1} J_m \uplus J \qquad Pred = \bigwedge_{m\in I_1\uplus I_2} Pred_m \wedge Pred_{sv}(SV_k, \alpha_m^{m\in I_1\uplus I_2}, \beta_j^{j\in J}, \alpha) \\
\forall i \in L\backslash \left(\biguplus_{m\in I_1} L_m \uplus I_2\right). s'_i = s_i \qquad \text{fresh}(\alpha'_m, \alpha', \beta_j^{j\in J}, \alpha)
\end{array}}{\langle\!\langle P_m^{m\in I}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle \models \dfrac{\beta_j^{j\in J'}, Pred, \biguplus_{m\in I_1\uplus I_2} Post_m}{\triangleleft s_i^{i\in L} \triangleright \xrightarrow{\alpha} \triangleleft\, (s'_i)^{i\in L} \triangleright}} \quad \textbf{\textit{Tr2}}$$

A key to understand this rule is that the open transitions are expressed in terms of the leaves and holes of the whole pNet structure, i.e. a flatten view of the pNet. For example, $L$

[1]Formally, if $SV_k = (\alpha')_m^{m\in M} \rightarrow \alpha'[e_b]$ is a synchronisation vector of P then $J = M \cap \text{Holes}(P)$, $I_2 = M \cap \text{Leaves}(P)$, $I_1 = M \setminus J \setminus I_2$

is the index set of the Leaves, $L_m$ the index set of the leaves of one sub-pNet indexed $m$, so all $L_m$ are disjoint subsets of $L$. Thus the states in the open transitions, at each level, are tuples including states of all the leaves of the pNet, not only those involved in the chosen synchronisation vector.

Note that the construction is symbolic, and each open-transition deduced expresses a whole family of behaviours, for any possible value of the variables.

In [10], we have shown a detailed example of the construction of a complex open transition, building a deduction tree using rules **Tr1** and **Tr2**.

We have shown in [10] that an open pNet with finite synchronisation sets, finitely many leaves and holes, and each pLTS at leaves having a finite number of states and (symbolic) transitions, has a finite automaton. The algorithm for building such an automaton can be found in [12].

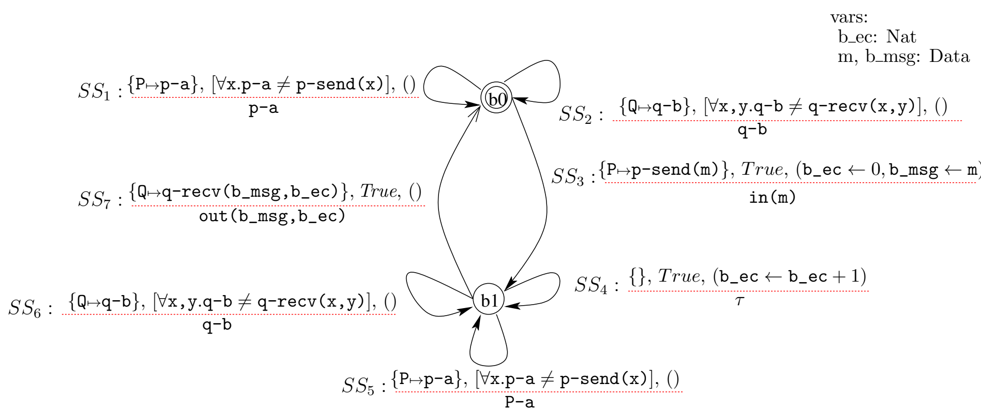



Figure 3: Specification Open Automaton

**Example.** Figure 3 shows the open automaton computed from the Specification pNet given in Figure 1. For later references, we give to the transitions of this (strong) Specification automaton as $SS_i$ while transitions of the Implementation automaton will be labelled $SI_i$. In the figure we have annotated each global state with the set of corresponding pLTS variables.

Figure 4 shows the open automaton of the implementation *SimpleSystem* (Figure 2). In this drawing, we have short labels for states, representing $\triangleleft s_0, m_0, r_0 \triangleright$ by 000. Note that open transitions are denoted $\mathtt{SI}_i$ and tau open transition by $\mathtt{SI}_\tau$. The resulting behaviour is quite simple: we have a main loop including receiving a message from $P$ and transmitting the same message to $Q$, with some intermediate $\tau$ actions from the internal communications between the protocol processes. In most of the transitions, you can observe that data is propagated between the successive plts variables (holding the message, and the error counter value). On the right of the figure, you have a loop of $\tau$ actions ($\mathtt{SI}_4$, $\mathtt{SI}_5$ and $\mathtt{SI}_6$) showing the handling of errors and corresponding increments of the error counter.

4.2. **pNet Composition Properties: composition of open transitions.** The semantics of open pNets allows us to prove two crucial properties relating pNet composition with pNet semantics: open transition of a composed pNet can be decomposed into open transitions

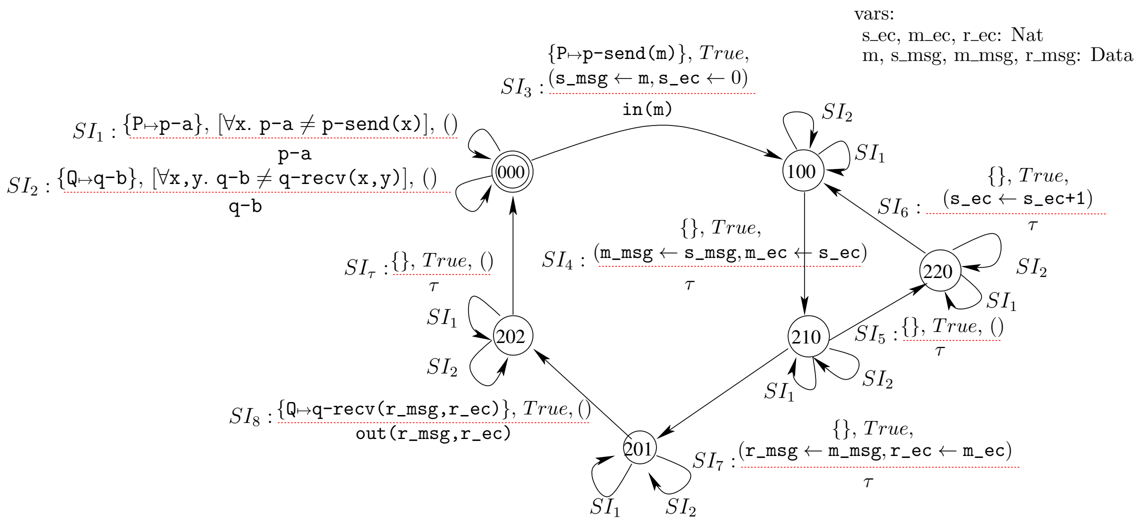



Figure 4: Open Automaton of the Simple Protocol Implementation

of its composing sub-pNets, and conversely, from the open transitions of sub-pNets, an open transition of the composed pNet can be built.

We start with decomposition: from one open transition of $P[Q]_{j_0}$, we exhibit corresponding behaviours of $P$ and $Q$, and determine the relation between their predicates:

**Lemma 1** Open transition decomposition. *Consider two pNets $P$ and $Q$ that are not pLTSs*[2]*. Let* $\text{Leaves}(Q) = p_l^{l \in L_Q}$*; suppose:*

$$P[Q]_{j_0} \models \frac{\beta_j^{j\in J}, Pred, Post}{\triangleleft s_i^{i\in L} \triangleright \xrightarrow{\alpha} \triangleleft\, s_i'^{\,i\in L} \triangleright}$$

*with* $J \cap \text{Holes}(Q) \neq \emptyset$ *or* $\exists i \in L_Q.\, s_i \neq s_i'$*, i.e. $Q$ takes part in the reduction. Then there exist $\alpha_Q$, $Pred'$, $Pred''$, $Post'$, $Post''$ s.t.:*

$$P \models \frac{\beta_j^{j\in (J\setminus \text{Holes}(Q))\cup\{j_0\}}, Pred', Post'}{\triangleleft s_i^{i\in L\setminus L_Q} \triangleright \xrightarrow{\alpha} \triangleleft\, s_i'^{\,i\in L\setminus L_Q} \triangleright}$$

$$\text{and } Q \models \frac{\beta_j^{j\in J\cap \text{Holes}(Q)}, Pred'', Post''}{\triangleleft s_i^{i\in L_Q} \triangleright \xrightarrow{\alpha_Q} \triangleleft\, s_i'^{\,i\in L_Q} \triangleright}$$

*and* $Pred \iff Pred' \wedge Pred'' \wedge \alpha_Q = \beta_{j_0}$*,* $Post = Post' \uplus Post''$ *where $Post''$ is the restriction of $Post$ over variables of $Q$.*

Lemma 2 is combining an open transition of $P$ with an open transition of $Q$, and building a corresponding transition of $P[Q]_{j_0}$, assembling their predicates.

**Lemma 2** Open transition composition. *Suppose $j_0 \in J$ and:*

$$P \models \frac{\beta_j^{j\in J}, Pred, Post}{\triangleleft s_i^{i\in L} \triangleright \xrightarrow{\alpha} \triangleleft\, s_i'^{\,i\in L} \triangleright} \quad \text{and } Q \models \frac{\beta_j^{j\in J_Q}, Pred', Post'}{\triangleleft s_i^{i\in L_Q} \triangleright \xrightarrow{\alpha_Q} \triangleleft\, s_i'^{\,i\in L_Q} \triangleright}$$

*Then, we have*

[2]A similar lemma can be proven for a pLTS $Q$

$$P[Q]_{j_0} \models \frac{\beta_j^{(j\in J\setminus\{j_0\})\uplus J_Q}, Pred \wedge Pred' \wedge \alpha_Q = \beta_{j_0}, Post \uplus Post'}{\triangleleft s_i^{i\in L\uplus L_Q} \triangleright \xrightarrow{\alpha} \triangleleft {s'}_i^{i\in L\uplus L_Q} \triangleright}$$

Note that this does not mean that any two pNets can be composed and produce an open transition. Indeed, the predicate $Pred \wedge Pred' \wedge \alpha_Q = \beta_{j_0}$ is often not satisfiable, in particular if the action $\alpha_Q$ cannot be matched with $\beta_{j_0}$. Note also that $\beta_{j_0}$ is only used as an intermediate term inside formulas in the composed open transition: it does not appear neither as global action nor as an action of a hole.

4.3. **Bisimulation for open pNets – a composable bisimulation theory.** As our symbolic operational semantics provides an open automaton, we can apply the notion of strong (symbolic) bisimulation on automata to open pNets:

**Definition 10** FH-bisimulation for open pNets**.** *Two pNets are FH-bisimilar if there exists a relation between their associated automata that is an FH-bisimulation and their initial states are in the relation (i.e. the predicate associated with the initial states is verifiable).*

We can now prove that pNet composition preserves FH-bisimulation. More precisely, one can define two preservation properties, namely 1) when one hole of a pNet is filled by two bisimilar other (open) pNets; and 2) when the same hole in two bisimilar pNets are filled by the same pNet, in other words, composing a pNet with two bisimilar contexts. The general case will be obtained by transitivity of the bisimulation relation (Theorem 1).

**Theorem 3** Congruence**.** *Consider an open pNet: $P = \langle\langle P_i^{i\in I}, Sort_j^{j\in J}, \overline{SV}\rangle\rangle$. Let $j_0 \in J$ be a hole. Let $Q$ and $Q'$ be two FH-bisimilar pNets such that*[3] $\mathrm{Sort}(Q) = \mathrm{Sort}(Q') = Sort_{j_0}$. *Then $P[Q]_{j_0}$ and $P[Q']_{j_0}$ are FH-bisimilar.*

**Theorem 4** Context equivalence**.** *Consider two open pNets $P = \langle\langle P_i^{i\in I}, Sort_j^{j\in J}, \overline{SV}\rangle\rangle$ and $P' = \langle\langle {P'}_i^{i\in I}, Sort_j^{j\in J}, \overline{SV}'\rangle\rangle$ that are FH-bisimilar (recall they must have the same holes to be bisimilar). Let $j_0 \in J$ be a hole, and $Q$ be a pNet such that* $\mathrm{Sort}(Q) = Sort_{j_0}$. *Then $P[Q]_{j_0}$ and $P'[Q]_{j_0}$ are FH-bisimilar.*

Finally, the previous theorems can be composed to state a general theorem about composability and FH-bisimilarity.

**Theorem 5** Composability**.** *Consider two FH-bisimilar pNets with an arbitrary number of holes, when replacing, inside those two original pNets, a subset of the holes by FH-bisimilar pNets, we obtain two FH-bisimilar pNets.*

This theorem is quite powerful. It somehow implies that the theory of open pNets is convenient to study properties of process composition. Open pNets can indeed be used to study process operators and process algebras, as shown in [10], or to study interaction protocols [13].

[3]Note that $\mathrm{Sort}(Q) = \mathrm{Sort}(Q')$ is ensured by strong bisimilarity.

## 5. Weak bisimulation

Weak symbolic bisimulation was introduced to relate transition systems that have indistinguishable behaviour, with respect to some definition of *internal actions* that are considered local to some subsystem, and consequently cannot be observed, nor used for synchronisation with their context. The notion of non-observable actions varies in different contexts, e.g. *tau* in CCS, and $i$ in Lotos, we could define classically a set of *internal/non-observable actions* depending on a specific action algebra. In this paper, to simplify the notations, we will simply use $\tau$ as the single non-observable action; the generalisation of our results to a set of non-observable actions is trivial. Naturally, a non-observable action cannot be synchronised with actions of other systems in its environment. We show here that under such assumption of non-observability of $\tau$ actions, see Definition 11, we can define a weak bisimulation relation that is compositional, in the sense of open pNet composition. In this section we will first define a notion of weak open transition similar to open transition. In fact a weak open transition is made of several open transitions labelled as non-observable transitions, plus potentially one observable open transition. This allows us to define weak open automata, and a weak bisimulation relation based on these weak open automata. Finally, we apply this weak bisimulation to open pNets, obtain a weak bisimilarity relationship for open pNets, and prove that this relation has compositional properties.

5.1. **Preliminary definitions and notations.** We first specify in terms of open transition, what it means for an action to be non-observable. Namely, we constraint ourselves to system where the emission of a $\tau$ action by a sub-pNet cannot be observed by the surrounding pNets. In other words, a pNet cannot change its state, or emit a specific observable action when one of its holes emits a $\tau$ action.

More precisely, we state that $\tau$ is not observable if the automaton always allows any $\tau$ transition from holes, and additionally the global transition resulting from a $\tau$ action of a hole is a $\tau$ transition not changing the pNet's state. We define $\mathrm{Id}(V)$ as the identity function on the set of variables $V$.

**Definition 11** Non-observability of $\tau$ actions for open automata**.** *An open automaton* $A = \langle\langle J, \mathcal{S}, s_0, V, \mathcal{T}\rangle\rangle$ cannot observe $\tau$ actions *if and only if for all $j$ in $J$ and $s$ in $\mathcal{S}$ we have:*

(1)
$$\frac{(j \mapsto \tau),\ \mathit{True}, \mathrm{Id}(V)}{s \xrightarrow{\tau} s} \in \mathcal{T}$$

*and*

(2) *for all $\beta_j$, $J$, $\alpha$, $s$, $s'$, Pred, Post such that*
$$\frac{\beta_j^{j\in J}, \mathit{Pred}, \mathit{Post}}{s \xrightarrow{\alpha} s'} \in \mathcal{T}$$

*If there exists $j$ such that $\beta_j = \tau$ then we have:*
$$\alpha = \tau \wedge s = s' \wedge \mathit{Pred} = \mathit{True} \wedge \mathit{Post} = \mathrm{Id}(V) \wedge J = \{j\}$$

The first statement of the definition states that the open automaton must allow a hole to do a silent action at any time, and must not observe it, i.e. cannot change its internal state because a hole did a $\tau$ transition. The second statement ensures that there cannot be in the open automaton other transitions that would be able to observe a $\tau$ action from a hole. The condition $J = \{j\}$ is a bit restrictive, it could safely be replaced by $\forall j \in J.\, \beta_j = \tau$, allowing the other holes to perform $\tau$ transitions too (because these $\tau$ actions cannot be observed).

By definition, one weak open transition contains several open transitions, where each open transition can require an observable action from a given hole, the same hole might have to emit several observable actions for a single weak open transition to occur. Consequently, for a weak open transition to trigger, a sequence of actions from a given hole may be required.

Thus, we let $\gamma$ range over sequences of action terms and use $\uplus$ as the concatenation operator that appends sequences of action terms: given two sequences of action terms $\gamma \uplus \gamma'$ concatenates the two sequences. The operation is lifted to indexed sets of sequences: at each index $i$, $\overline{\gamma_1} \uplus \overline{\gamma_2}$ concatenates the sequences of actions at index $i$ of $\overline{\gamma_1}$ and the one at index $i$ of $\overline{\gamma_2}$[4]. $[a]$ denotes a sequence with a single element.

As required actions are now sequences of observable actions, we need an operator to build them from set of actions that occur in open transitions, i.e. an operator that takes a set of actions performed by one hole and produces a sequence of observable actions.

Thus we define $(\overline{\beta})^{\nabla}$ as the mapping $\overline{\beta}$ with only observable actions of the holes in $I$, but where each element is either empty or a list of length 1:

$$(\beta_i^{i \in I})^{\nabla} = [\beta_i]^{i \in I'} \text{ where } I' = \{i | i \in I \wedge \beta_i \neq \tau\}$$

As an example the $(\overline{\beta})^{\nabla}$ built from the transition $OT_1$ in Example 1, page 12 is $\mathtt{P} \mapsto [\mathtt{p\text{-}send(m)}]$. Remark that in our simple example no $\tau$ transition involves any visible action from a hole, so we have no $\beta$ sequences of length longer than 1 in the weak automaton.

5.2. **Weak open transition definition.** Because of the non-observability property (Definition 11), it is possible to add any number of $\tau$ transitions of the holes before or after any open transition freely. This property justifies the fact that we can abstract away $\tau$ transitions from holes in the definition of a weak open transition.

**Definition 12** Weak open transition (WOT)**.** *A weak open transition over a set $J$ of holes with sorts $Sort_j^{j \in J}$ and a set of states $\mathcal{S}$ is a structure of the form:*

$$\frac{\gamma_j^{j \in J'}, Pred, Post}{s \stackrel{\alpha}{\Rightarrow} s'}$$

*Where $J' \subseteq J$, $s, s' \in \mathcal{S}$ and $\gamma_j$ is a list of transitions of the hole $j$, with each element of the list in $Sort_j$. $\alpha$ is an action label denoting the resulting action of this open transition. Pred and Post are defined similarly to Definition 5. We use $\mathcal{WT}$ to range over sets of weak open transitions.*

*A weak open automaton $\langle\langle J, \mathcal{S}, s_0, V, \mathcal{WT} \rangle\rangle$ is similar to an open automaton except that $\mathcal{WT}$ is a set of weak open transitions over $J$ and $\mathcal{S}$.*

A weak open transition labelled $\alpha$ can be seen as a sequence of open transitions that are all labelled $\tau$ except one that is labelled $\alpha$; however conditions on predicates, effects, and states must be verified for this sequence to be fired.

[4] One of the two sequences is empty when $i \notin \mathrm{dom}(\overline{\gamma_1})$ or $i \notin \mathrm{dom}(\overline{\gamma_2})$ .

We are now able to build a weak open automaton from an open automaton. This is done in a way that resembles the process of $\tau$ saturation: we add $\tau$ open transitions before or after another (observable or not) open transition.

**Definition 13** Building a weak open automaton**.** *Let $A = \langle\!\langle J, \mathcal{S}, s_0, V, \mathcal{T}\rangle\!\rangle$ be an open automaton. The weak open automaton* derived *from $A$ is an open automaton $\langle\!\langle J, \mathcal{S}, s_0, V, \mathcal{WT}\rangle\!\rangle$ where $\mathcal{WT}$ is derived from $\mathcal{T}$ by saturation, applying the following rules:*

$$\frac{\emptyset, \mathit{True}, \mathrm{Id}(V)}{s \overset{\tau}{\Rightarrow} s} \in \mathcal{WT} \qquad \textbf{\textit{WT1}}$$

*and*

$$\frac{\dfrac{\overline{\beta}, \mathit{Pred}, \mathit{Post}}{s \xrightarrow{\alpha} s'} \in \mathcal{T}}{\dfrac{(\overline{\beta})^{\nabla}, \mathit{Pred}, \mathit{Post}}{s \overset{\alpha}{\Rightarrow} s'} \in \mathcal{WT}} \qquad \textbf{\textit{WT2}}$$

*and*

$$\frac{\begin{array}{c}\dfrac{\overline{\gamma_1}, \mathit{Pred}_1, \mathit{Post}_1}{s \overset{\tau}{\Rightarrow} s_1} \in \mathcal{WT} \qquad \dfrac{\overline{\gamma_2}, \mathit{Pred}_2, \mathit{Post}_2}{s_1 \overset{\alpha}{\Rightarrow} s_2} \in \mathcal{WT} \qquad \dfrac{\overline{\gamma_3}, \mathit{Pred}_3, \mathit{Post}_3}{s_2 \overset{\tau}{\Rightarrow} s'} \in \mathcal{WT} \\ \mathit{Pred} = \mathit{Pred}_1 \wedge \mathit{Pred}_2\{\!\{\mathit{Post}_1\}\!\} \wedge \mathit{Pred}_3\{\!\{\mathit{Post}_2 \otimes \mathit{Post}_1\}\!\} \\ \overline{\gamma} = \overline{\gamma_1} \uplus \overline{\gamma_2}\{\!\{\mathit{Post}_1\}\!\} \uplus \overline{\gamma_3}\{\!\{\mathit{Post}_2 \otimes \mathit{Post}_1\}\!\} \qquad \alpha' = \alpha\{\!\{\mathit{Post}_1\}\!\}\end{array}}{\dfrac{\overline{\gamma}, \mathit{Pred}, \mathit{Post}_3 \otimes \mathit{Post}_2 \otimes \mathit{Post}_1}{s \overset{\alpha'}{\Rightarrow} s'} \in \mathcal{WT}} \ \textbf{\textit{WT3}}$$

Rule **WT1** states that it is always possible to do a non-observable transition, where the state is unchanged and the holes perform no action. Rule **WT2** states that each open transition can be considered as a weak open transition. The last rule is the most interesting: Rule **WT3** allows any number of $\tau$ transitions before or after any weak open transition. This rules carefully composes predicates, effects, and actions of the holes, indeed in the rule, predicate $\mathit{Pred}_2$ manipulates variables of $s_1$ that result from the first weak open transition. Their values thus depend on the initial state but also on the effect (as a substitution $\mathit{Post}_1$) of the first weak open transition. In the same manner, $\mathit{Pred}_3$ must be applied the joint substitution $\mathit{Post}_2 \otimes \mathit{Post}_1$. Similarly, effects on variables must be applied to obtain the global effect of the composed weak open transition, it must also be applied to observable actions of the holes, and to the global action of the weak open transition.

**Example 2** A weak open-transition**.** *Figure 5 shows the construction of one of the weak transitions of the Specification OA. On the top we show the subset of the original open automaton (from Figure 3) considered here, and at the bottom the generated weak transition. For readability, we abbreviate the weak open transitions encoded by $\dfrac{\{\}, \mathit{True}, ()}{s \overset{\tau}{\Rightarrow} s'}$ as $W_\tau$. The weak open transition shown here is the transition delivering the result of the algorithm to hole $Q$ by applying rules: **WT1**, **WT2**, and **WT3**. First rule **WT1** adds a $WT_\tau$ loop on each state. Rule **WT2** transforms each 3 OTs into WOTs. Then consider application of Rule **WT3** on a sequence 3 WOTs. $\dfrac{\{\}, \mathit{True}, (\mathtt{b\_ec} \leftarrow \mathtt{b\_ec} + 1)}{\mathtt{b1} \overset{\tau}{\Rightarrow} \mathtt{b1}}$; $\dfrac{\{\}, \mathit{True}, (\mathtt{b\_ec} \leftarrow \mathtt{b\_ec} + 1)}{\mathtt{b1} \overset{\tau}{\Rightarrow} \mathtt{b1}}$;*

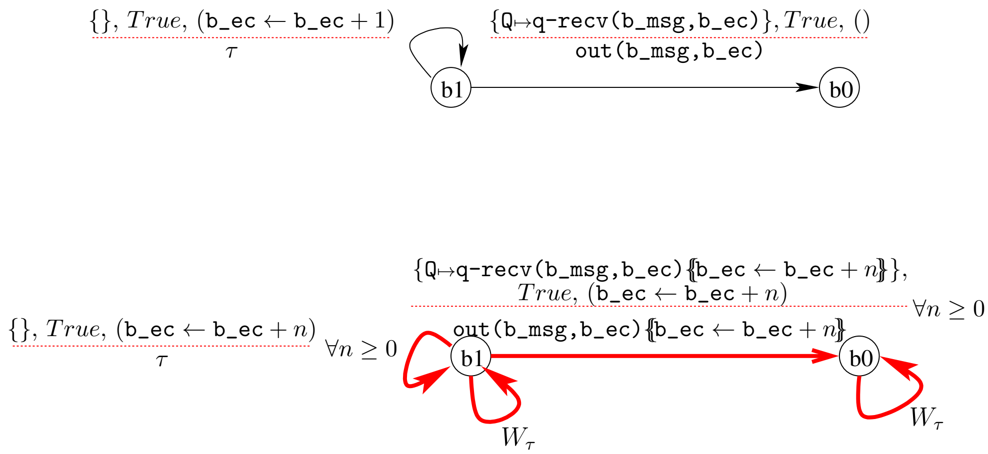



Figure 5: Construction of an example of weak open transition

$\dfrac{\{\}, True, ()}{\texttt{b1} \overset{\tau}{\Rightarrow} \texttt{b1}}$. *Then Rule **WT3** produces* $\dfrac{\{\}, True, (\texttt{b\_ec} \leftarrow \texttt{b\_ec} + 2)}{\texttt{b1} \overset{\tau}{\Rightarrow} \texttt{b1}}$. *We can iterate this construction an arbitrary number of times, getting for any natural number n a weak open transition:* $\dfrac{\emptyset, True, (\texttt{ec} \leftarrow \texttt{ec} + n)}{\texttt{b1} \overset{\tau}{\Rightarrow} \texttt{b1}}$ $\forall n \geq 0$. *Finally, applying again **WT3**, and using the central open transition having* **`out(b_msg,b_ec)`** *as* $\alpha$, *we get the resulting weak open transition between b1 and b0 (as shown in Figure 5). Applying the substitutions finally yields the weak transitions family* $WS_7$ *in Figure 6.*

**Example 3** Weak open automata**.** *Figures 6 and 7 respectively show the weak automata of the simple protocol specification and implementation. We encode weak open transitions by* $WS$ *on the specification model and by* $WI$ *on the implementation model.*

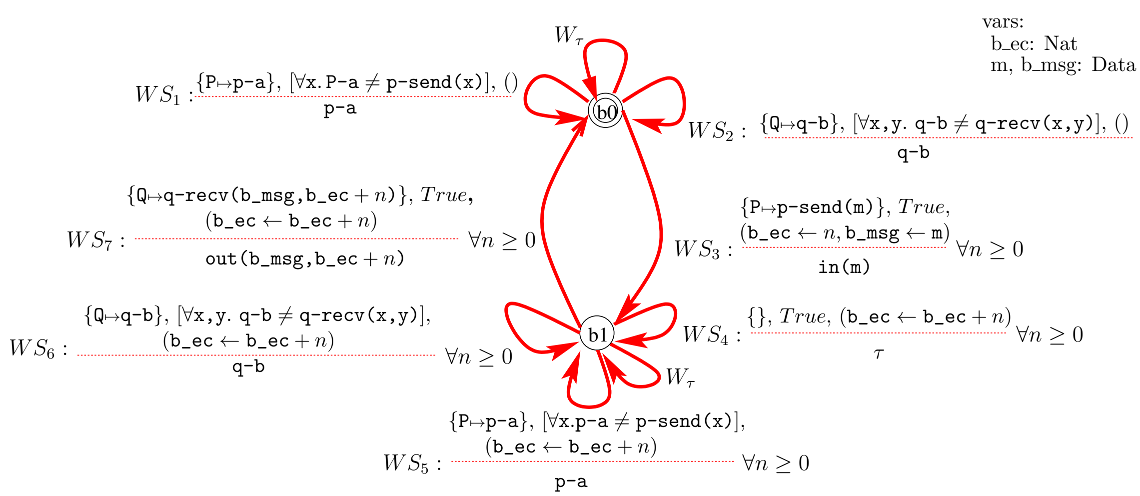



*Figure 6: Weak Open Automaton of the Specification*

*For readability, we do not include the full details of the Implementation weak open transitions in Figure 7, but we show some examples below and the full list is included in*

*Appendix C. First, let us point out that the weak OT loops ($WI_1$,$WI_2$ and $W_\tau$) on state 000 are also present in all other states, we did not repeat them. Then many WOTs are similar, and numbered accordingly as 3, 3a, 3b, 3c and 8, 8a, 8b, 8c respectively: they only differ by their respective source or target states; the "variant" WOTs appear in blue in Figure 7.*

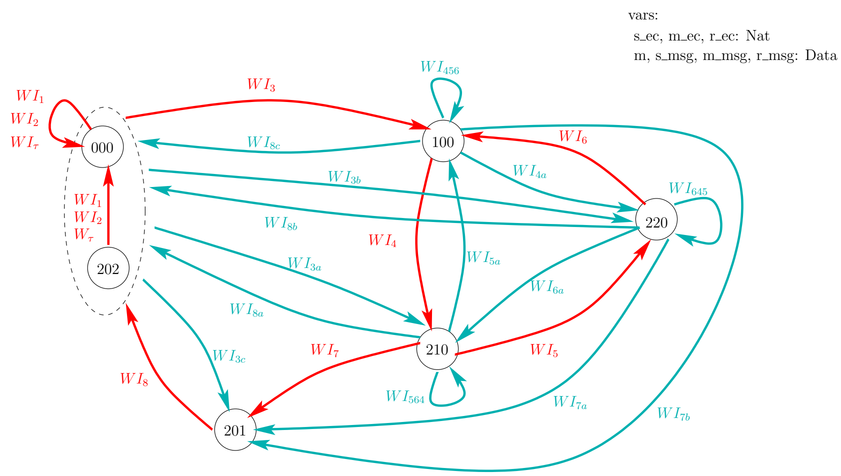



Figure 7: Weak Open Automaton of the implementation

Now let us give some details about the construction of the weak automaton of the implementation pNet, obtained by application of the weak rules as explained above. We concentrate on weak open transitions $WI_3$ and $WI_4$. Let us denote as $post_n$ the effect (substitution) of the strong open transitions $SI_n$ from Figure 4:

$post_3 = (\texttt{s\_msg} \leftarrow \texttt{m}, \texttt{s\_ec} \leftarrow 0)$
$post_4 = (\texttt{m\_msg} \leftarrow \texttt{s\_msg}, \texttt{m\_ec} \leftarrow \texttt{s\_ec})$
$post_5 = ()$
$post_6 = (\texttt{s\_ec} \leftarrow \texttt{s\_ec+1})$

Then the effect of one single $100 \xrightarrow{OT_4} 210 \xrightarrow{OT_5} 220 \xrightarrow{OT_6} 100$ loop is[5]:

$$post_{456} = post_6 \otimes\ post_5 \otimes\ post_4 = (\texttt{s\_ec} \leftarrow \texttt{s\_ec} + 1)$$

So if we denote $post_{456*}$ any iteration of this loop, we get $post_{456*} = (\texttt{s\_ec} \leftarrow \texttt{s\_ec} + n)$ for any $n \geq 0$, and the *Post* of the weak OT $WI_3$ is:
$Post_3 = post_{456*} \otimes\ post_3 = (\texttt{s\_msg} \leftarrow m, \texttt{s\_ec} \leftarrow n), \forall n \geq 0$ and *Post* of $WI_{3a}$ is:
$post_4 \otimes post_{456*} \otimes\ post_3 = (\texttt{m\_msg} \leftarrow m, \texttt{m\_ec} \leftarrow n), \forall n \geq 0.$

We can now show some of the weak OTs of Figure 7 (the full table is included in Appendix C). As we have seen above, the effect of rule $WT_3$ when a silent action have an effect on the variable *ec* will generate an infinite family of WOTs, depending on the number of iterations through the loops. We denote these families using a "meta-variable" $n$, ranging over `Nat`.

$$WI_1 = \frac{\{\texttt{P}\mapsto\texttt{p-a}\}, [\forall \texttt{x}.\texttt{p-a} \neq \texttt{p-send(x)}], ()}{s \overset{\texttt{p-a}}{\Longrightarrow} s} \text{ (for any } s \in S)$$

[5]when showing the result of *Posts* composition, we will omit the identity substitutions introduced by the $\otimes$ definition in page 4

$$\forall n \geq 0.\, WI_3(n) = \frac{\{\mathtt{P}{\mapsto}\mathtt{p\text{-}send(m)}\}, True, (\mathtt{s\_msg} \leftarrow \mathtt{m}, \mathtt{s\_ec} \leftarrow n)}{000 \xRightarrow{\mathtt{in(m)}} 100}$$

$$\forall n \geq 0.\, WI_4(n) = \frac{\{\}, True, (\mathtt{m\_msg} \leftarrow \mathtt{s\_msg}, \mathtt{m\_ec} \leftarrow \mathtt{s\_ec} + n, \mathtt{s\_ec} \leftarrow \mathtt{s\_ec} + n)}{100 \xRightarrow{\tau} 210}$$

$$\forall n \geq 0.\, WI_{456}(n) = \frac{\{\}, True, (\mathtt{s\_ec} \leftarrow \mathtt{s\_ec} + n)}{100 \xRightarrow{\tau} 100}$$

The *Post* of the weak OT $WI_{6a}$ is:

$$\begin{aligned} Post_{6a} &= post_4 \otimes post_{456*} \otimes post_6 \\ &= (\mathtt{m\_msg} \leftarrow \mathtt{s\_msg}, \mathtt{m\_ec} \leftarrow \mathtt{s\_ec}) \otimes (\mathtt{s\_ec} \leftarrow \mathtt{s\_ec} + n) \otimes (\mathtt{s\_ec} \leftarrow \mathtt{s\_ec} + 1) \\ &= (\mathtt{m\_msg} \leftarrow \mathtt{s\_msg}, \mathtt{m\_ec} \leftarrow \mathtt{s\_ec} + 1 + n, \mathtt{s\_ec} \leftarrow \mathtt{s\_ec} + 1 + n) \end{aligned}$$

So we get:

$$\forall n \geq 0.\, WI_{6a}(n) = \frac{\{\}, True, (\mathtt{m\_ec} \leftarrow \mathtt{s\_ec} + 1 + n, \mathtt{s\_ec} \leftarrow \mathtt{s\_ec} + 1 + n)}{220 \xRightarrow{\tau} 210}$$

5.3. **Composition properties: composition of weak open transitions.** We now have two different semantics for open pNets: a strong semantics, defined as an open automaton, and as a weak semantics, defined as a weak open automaton. Like the open automaton, the weak open automaton features valuable composition properties. We can exhibit a composition property and a decomposition property that relate open pNet composition with their semantics, defined as weak open automata. These are however technically more complex than the ones for open automata because each hole performs a set of actions, and thus a composed transition is the composition of one transition of the top-level pNet and a sequence of transitions of the sub-pNet that fills its hole. They can be found as Lemmas 6, Lemma 7, and Lemma 8 in Appendix B.2.

5.4. **Weak FH-bisimulation.** For defining a bisimulation relation between weak open automata, two options are possible. Either we define a simulation similar to the strong simulation but based on open automata, this would look like the FH-simulation but would need to be adapted to weak open transitions. Or we define directly and classically a weak FH-simulation as a relation between two open automata, relating the open transition of the first one with the transition of the weak open automaton derived from the second one.

The definition below specifies how a set of weak open transitions can simulate an open transition, and under which condition; this is used to relate, by weak FH-bisimulation, two open automata by reasoning on the weak open automata that can be derived from the strong ones. This is defined formally as follows.

**Definition 14** Weak FH-bisimulation**.**
*Let $A_1 = \langle\!\langle J, \mathcal{S}_1, s_0, V_1, \mathcal{T}_1 \rangle\!\rangle$ and $A_2 = \langle\!\langle J, \mathcal{S}_2, t_0, V_2, \mathcal{T}_2 \rangle\!\rangle$ be open automata with disjoint sets of variables. Let $\langle\!\langle J, \mathcal{S}_1, s_0, V_1, \mathcal{WT}_1 \rangle\!\rangle$ and $\langle\!\langle J, \mathcal{S}_2, t_0, V_2, \mathcal{WT}_2 \rangle\!\rangle$ be the weak open automata derived from $A_1$ and $A_2$ respectively. Let $\mathcal{R}$ a relation over $\mathcal{S}_1$ and $\mathcal{S}_2$, as in Definition 7.*

*Then $\mathcal{R}$ is a weak FH-bisimulation iff for any states $s \in \mathcal{S}_1$ and $t \in \mathcal{S}_2$ such that $(s, t|Pred) \in \mathcal{R}$, we have the following:*

- *For any open transition $OT$ in $\mathcal{T}_1$:*

$$\frac{\beta_j^{j\in J'}, Pred_{OT}, Post_{OT}}{s \xrightarrow{\alpha} s'}$$

*there exist weak open transitions $WOT_x^{x\in X} \subseteq \mathcal{WT}_2$:*

$$\frac{\gamma_{jx}^{j\in J_x}, Pred_{OT_x}, Post_{OT_x}}{t \xRightarrow{\alpha_x} t_x}$$

*such that* $\forall x. \{j \in J' | \beta_j \neq \tau\} = J_x, (s', t_x | Pred_{s',t_x}) \in \mathcal{R}$*; and*
$Pred \wedge Pred_{OT}$
$\Longrightarrow \bigvee_{x\in X} \Big(\forall j \in J_x.(\beta_j)^{\nabla} = \gamma_{jx} \wedge Pred_{OT_x} \wedge \alpha = \alpha_x \wedge Pred_{s',t_x}\{\!\{Post_{OT} \uplus Post_{OT_x}\}\!\}\Big)$

- *and symmetrically any open transition from $t$ in $\mathcal{T}_2$ can be covered by a set of weak transitions from $s$ in $\mathcal{WT}_1$.*

*Two pNets are weak FH-bisimilar if there exists a relation between their associated automata that is a Weak FH-bisimulation and their initial states are in the relation, i.e. the predicate associated to the relation between the initial states is True.*

Compared to strong bisimulation, except the obvious use of weak open transitions to simulate an open transition, the condition on predicate is slightly changed concerning actions of the holes. Indeed only the visible actions of the holes must be compared and they form a list of actions, but of length at most one.

Our first important result is that Weak FH-bisimilarity is an equivalence in the same way as strong FH-bisimilarity:

**Theorem 6** Weak FH-Bisimulation is an equivalence**.** *Suppose $\mathcal{R}$ is a weak FH-bisimulation. Then $\mathcal{R}$ is an equivalence, that is, $\mathcal{R}$ is reflexive, symmetric and transitive.*

The proof is detailed in Appendix B.1, it follows a similar pattern as the proof that strong FH-bisimulation is an equivalence, but technical details are different, and in practice we rely on a variant of the definition of weak FH-bisimilarity; this equivalent version simulates a *weak* open transition with a set of weak open transition. The careful use of the best definition of weak FH-bisimilarity makes the proof similar to the strong FH-bisimulation case.

*Proving bisimulation in practice.* In practice, we are dealing with finite representations of the (infinite) open automata. In [11], we defined a slightly modified definition of the "coverage" proof obligation, in the case of strong FH-Bisimulation. This modification is required to manage in a finite way all possible instantiations of an OT. In the case of weak FH-Bisimulation, the proof obligation from Definition 14 becomes:

$$\forall fv_{OT}. \Big\{ Pred \wedge Pred_{OT} \implies \bigvee_{x\in X} \Big[\exists fv_{OT_x}. \Big(\forall j \in J_x.(\beta_j)^{\nabla} = \gamma_{jx} \wedge Pred_{OT_x} \wedge \alpha = \alpha_x \wedge Pred_{s',t_x}\{\!\{Post_{OT} \uplus Post_{OT_x}\}\!\}\Big)\Big] \Big\}$$

In which $fv_{OT}$ denotes the set of free variables of all expressions in $OT$.

5.5. **Weak bisimulation for open pNets.** Before defining a weak open automaton for the semantics of open pNets, it is necessary to state under which condition a pNet is unable to observe silent actions of its holes. In the setting of pNets this can simply be expressed as a condition on the synchronisation vectors. Precisely, the set of synchronisation vectors must contain vectors that let silent actions go through the pNet, i.e. synchronisation vectors where one hole does a $\tau$ transition, and the global visible action is a $\tau$. Additionally, no other synchronisation vector must be able to react on a silent action from a hole, i.e. if a synchronisation vector observes a $\tau$ from a hole it cannot synchronise it with another action nor emit an action that is not $\tau$. This is formalised as follows:

**Definition 15** Non-observability of silent actions for pNets**.**
*A pNet* $\langle\langle P_i^{i\in I}, Sort_j^{j\in J}, \overline{SV}\rangle\rangle$ cannot observe silent actions *if it verifies:*
$\forall i \in I \uplus J.\, (i{\mapsto}\tau) \to \tau[True] \in \overline{SV}$ *and*

$$\forall \left((\alpha_i)^{i\in I'} \to \alpha'[e_b] \in \overline{SV}\right), \forall i \in I' \cap J.\, \alpha_i = \tau \implies \alpha' = \tau \ \wedge\ I' = \{i\}$$

With this definition, it is easy to check that the open automaton that gives the semantics of such an open pNet cannot observe silent actions in the sense of Definition 11:

**Property 1** Non-observability of silent actions**.** *The semantics of a pNet, as provided in Definition 9, that cannot observe silent actions is an open automaton that cannot observe silent actions.*

Under this condition, it is safe to define the weak open automaton that provides a weak semantics to a given pNet. This is simply obtained by applying Definition 13 to generate a weak open automaton from the open automaton that is the strong semantics of the open pNet, as provided by Definition 9.

**Definition 16** Semantics of pNets as a weak open automaton**.** *Let $A$ be the open automaton expressing the semantics of an open pNet $P$; let $\langle\langle J, \mathcal{S}, s_0, V, \mathcal{WT}\rangle\rangle$ be the weak open automaton derived from A; this weak open automaton defines the weak semantics of the pNet $P$. Then, we denote $P \models WOT$ whenever $WOT \in \mathcal{WT}$.*

From the definition of the weak open automata of pNets, we can now study the properties of weak bisimulation concerning open pNets.

5.6. **Properties of weak bisimulation for open pNets.** When silent actions cannot be observed, weak bisimulation is a congruence for open pNets: if $P$ and $Q$ are weakly bisimilar to $P'$ and $Q'$ then the composition of $P$ and $Q$ is weakly bisimilar to the composition of $P'$ and $Q'$, where composition is the hole replacement operator: $P[Q]_j$ and $P'[Q']_j$ are weak FH-bisimilar. This can be shown by proving the two following theorems. The detailed proof of these theorem can be found in Appendix B.2. The proof strongly relies on the fact that weak FH-bisimulation is an equivalence, but also on the composition properties for open automata.

**Theorem 7** Congruence for weak bisimulation**.** *Consider an open pNet $P$ that cannot observe silent actions, of the form $P = \langle\langle P_i^{i\in I}, Sort_j^{j\in J}, \overline{SV}\rangle\rangle$. Let $j_0 \in J$ be a hole. Let $Q$ and $Q'$ be two weak FH-bisimilar pNets such that*[6] $\mathrm{Sort}(Q) = \mathrm{Sort}(Q') = Sort_{j_0}$. *Then $P[Q]_{j_0}$ and $P[Q']_{j_0}$ are weak FH-bisimilar.*

[6]Note that $\mathrm{Sort}(Q) = \mathrm{Sort}(Q')$ is ensured by weak bisimilarity.

**Theorem 8** Context equivalence for weak bisimulation**.** *Consider two open pNets* $P = \langle\!\langle P_i^{i\in I}, Sort_j^{j\in J}, \overline{SV}\rangle\!\rangle$ *and* $P' = \langle\!\langle P'^{i\in I}_i, Sort_j^{j\in J}, \overline{SV}'\rangle\!\rangle$ *that are weak FH-bisimilar (recall they must have the same holes to be bisimilar) and that cannot observe silent actions. Let* $j_0 \in J$ *be a hole, and* $Q$ *be a pNet such that* $\mathrm{Sort}(Q) = Sort_{j_0}$*. Then* $P[Q]_{j_0}$ *and* $P'[Q]_{j_0}$ *are weak FH-bisimilar.*

Finally, the previous theorems can be composed to state a general theorem about composability and weak FH-bisimilarity.

**Theorem 9** Composability of weak bisimulation**.** *Consider two weak FH-bisimilar pNets with an arbitrary number of holes, such that the two pNets cannot observe silent actions. When replacing, inside those two original pNets, a subset of the holes by weak FH-bisimilar pNets, we obtain two weak FH-bisimilar pNets.*

*Running example.* In Section 5 we have shown the full satured weak automaton for both the specification and the implementation of the simple protocol. We will show here how we can check if some given relation between these two automata is a weak FH-Bisimulation.

Preliminary remarks:

- Both pNets trivialy verify the "non-observability" condition: the only vectors having $\tau$ as an action of a sub-net are of the form "$< -, \tau, - > \to \tau$".
- We must take care of variable name conflicts: in our example, the variables of the 2 systems already have different names, but the action parameters occurring in the transitions (`m`, `msg`, `ec`) are the same, that is not correct. In the tools, this is managed by the static semantic layer; in the following example, we have renamed all conflicting variables with 1 for the Spec, and 2 for the Impl.

Now consider the relation $\mathcal{R}$ defined by the following triples:

| Spec state | Impl state | Predicate |
|---|---|---|
| b0 | 000 | True |
| b0 | 202 | True |
| b1 | 100 | `b_msg = s_msg` $\wedge$ `b_ec = s_ec` |
| b1 | 210 | `b_msg = m_msg` $\wedge$ `b_ec = m_ec` |
| b1 | 220 | `b_msg = s_msg` $\wedge$ `b_ec = s_ec` |
| b1 | 201 | `b_msg = r_msg` $\wedge$ `b_ec = r_ec` |

Checking that $\mathcal{R}$ is a weak FH-Bisimulation means checking, for each of these triples, that each (strong) OT of one the states corresponds to a set of WOTs of the other, using the conditions from Definition 14. We give here one example: consider the second triple from the table, and transition $SS_3$ from state `b0`. Its easy to guess that it will correspond to $WI_3(0)$ of state 202.

$$SS_3 = \frac{\{\mathtt{P}\mapsto\mathtt{p\text{-}send(m1)}\}, True, (\mathtt{b\_msg} \leftarrow \mathtt{m1}, \mathtt{b\_ec} \leftarrow 0)}{\mathrm{b0} \xrightarrow{\mathtt{in(m1)}} \mathrm{b1}}$$

$$WI_3(0) = \frac{\{\mathtt{P}\mapsto\mathtt{p\text{-}send(m2)}\}, True, (\mathtt{s\_msg} \leftarrow \mathtt{m2}, \mathtt{s\_ec} \leftarrow 0)}{000 \xRightarrow{\mathtt{in(m2)}} 100}$$

Let us check formally the conditions:

- Their sets of active (non-silent) holes is the same: $J' = J_x = \{\mathtt{P}\}$.
- Triple $(\mathrm{b1}, 100, \mathtt{b\_msg\ =\ s\_msg} \wedge \mathtt{b\_ec\ =\ s\_ec})$ is in $\mathcal{R}$.

- The verification condition
  $\forall fv_{OT}.\,\{Pred \wedge Pred_{OT}$
  $\implies \bigvee_{x\in X}\Big[\exists fv_{OT_x}.\,\Big(\forall j \in J_x.(\beta_j)^{\nabla} = \gamma_{jx} \wedge Pred_{OT_x} \wedge \alpha = \alpha_x \wedge Pred_{s',t_x}\{\!\{Post_{OT} \uplus Post_{OT_x}\}\!\}\Big)\Big]\}$

  Gives us:
  $\forall \mathtt{m1}.\,\{True \wedge True \quad \implies \exists \mathtt{m2}.$
  $(\mathtt{[p\text{-}send(m1)]} = \mathtt{[p\text{-}send(m2)]} \wedge True \wedge \mathtt{in(m1)} = \mathtt{in(m2)} \wedge$
  $(\mathtt{b\_msg} = \mathtt{s\_msg} \wedge \mathtt{b\_ec} = \mathtt{s\_ec})\{\!\{(\mathtt{b\_msg} \leftarrow \mathtt{m1}, \mathtt{b\_ec} \leftarrow 0) \uplus (\mathtt{s\_msg} \leftarrow \mathtt{m2}, \mathtt{s\_ec} \leftarrow 0)\}\!\})\}$

  That is reduced to:
  $\forall \mathtt{m1}.\exists \mathtt{m2}.\,(\mathtt{p\text{-}send(m1)} = \mathtt{p\text{-}send(m2)} \wedge \mathtt{in(m1)} = \mathtt{in(m2)} \wedge \mathtt{m1} = \mathtt{m2} \wedge 0 = 0)$

  That is a tautology.

## 6. Related Works

To the best of our knowledge, there are not many research works on Weak Bisimulation Equivalences between such complicate system models (open, symbolic, data-aware, with loops and assignments). We give a brief overview of other related publications, focussing first on Open and Compositional approaches, then on Symbolic Bisimulation for data-sensitive systems.

*Open and Compositional systems.* In [14, 15], the authors investigate several methodologies for the compositional verification of software systems, with the aim to verify reconfigurable component systems. To improve scaling and compositionality, the authors decouple the verification problem that is to be resolved by a SMT (satisfiability modulo theory) solver into independent sub-problems on independent sets of variables. These works clearly highlight the interest of incremental and compositional verification in a very general setting. In our own work on open pNets, adding more structure to the composition model, we show how to enforce a compositional proof system that is more powerful than independent sets of variables. Our theory has also been encoded into an SMT solver and it would be interesting to investigate how the examples of evolving systems studied by the authors could be encoded into pNet and verified by our framework. However, the models of Johnson et al. are quite different from ours, in particular they are much less structured, and translating them is clearly outside the scope of this article. In previous work [16], we also have shown how (closed) pNet models could be used to encode and verify finite instances of reconfigurable component systems.

Methodologies for reasoning about abstract semantics of open systems can be found in [17, 18, 19], authors introduce behavioural equivalences for open systems from various symbolic approaches. Working in the setting of process calculi, some close relations exist with the work of the authors of [17, 18], where both approaches are based on some kinds of labelled transition systems. The distinguishing feature of their approach is the transitions systems are labelled with logical formulae that provides an abstract characterisation of the structure that a hole must possess and of the actions it can perform in order to allow a transition to fire. Logical formulae are suitable formats that capture the general class of components that can act as the placeholders of the system during its evolution. In our approach we purposely leave the algebra of action terms undefined but the only operation we allow on action of holes is the comparison with other actions. Defining properly the interaction between a logical formulae in the action and the logics of the pNet composition seems very difficult.

*Symbolic and data-sensitive systems.* As mentioned in the *Introduction*, the work that brought us a lot of inspirations are those of Lin et al. [3, 4, 5]. They developed the theory of symbolic transition graphs (STG), and the associated symbolic (early and late, strong and weak) bisimulations, they also study STGs with assignments as a model for message-passing processes. Our work extends these in several ways: first our models are compositional, and our bismulations come with effective conditions for being preserved by pNet composition (i.e. congruent), even for the weak version. This result is more general than the bisimulation congruences for value-passing CCS in [3]. Then our settings for management of data types are much less restrictive, thanks to our use of satisfiability engines, while Lin's algorithms were limited to data-independent systems.

In a similar way, [20] presents a notion of "data-aware" bisimulations on data graphs, in which computation of such bisimulations is studied based on XPath logical language extended with tests for data equality.

Research related to the keyword "Symbolic Bisimulation" refer to two very different domains, namely BDD-like techniques for modelling and computing finite-state bisimulations, that are not related to our topic; and symbolic semantics for data-dependant or high-order systems, that are very close in spirit to our approach. In this last area, we can mention Calder's work [21], that defines a symbolic semantic for full LOTOS, with a symbolic bisimulation over it; Borgstrom et al., Liu et al, Delaune et al. and Buscemi et al. providing symbolic semantic and equivalence for different variants of pi calculus respectively [22, 23, 24, 25]; and more recently Feng et al. provide a symbolic bisimulation for quantum processes [26]. All the above works, didn't give a complete approach for verification, and the models on which these works based are definitely different from ours.

## 7. Conclusion and Discussion

pNets (Parameterised Networks of Automata) is a formalism adapted to the representation of the behaviour of a parallel or distributed systems. One strength of pNets is their parameterised nature, making them adapted to the representation of systems of arbitrary size, and making the modelling of parameterised system possible. Parameters are also crucial to reason on interaction protocols that can address one entity inside an indexed set of processes. pNets have been successfully used to represent behavioural specification of parallel and distributed components and verify their correctness [8, 9]. VCE is the specification and verification platform that uses pNets as an intermediate representation.

Open pNets are pNets with holes; they are adapted to represent processes parameterised by the behaviour of other processes, like composition operators or interaction protocols that synchronize the actions of processes that can be plugged afterwards. Open pNets are hierarchical composition of automata with holes and parameters. We defined here a semantics for open pNets and a complete bisimulation theory for them. The semantics of open pNets relies on the definition of open automata that are automata with holes and parameters, but no hierarchy. Open automata are somehow labelled transition systems with parameters and holes, a notion that is useful to define semantics, but makes less sense when modelling a system, compared to pNets. To be precise, it is on open automata that we define our bisimulation relations.

This article defines a strong and a weak bisimulation principles that are adapted to parameterised systems and hierarchical composition. Our bisimulation principle handles pNet parameters in the sense that two states might be or not in relation depending on the value of parameters. Our strong bisimulation is compositional by nature in the sense that bisimulation is maintained when composing processes. We also identified a simple and realistic condition on the semantics of non-observable actions that allows weak bisimulation to be also compositional. Overall we believe that this article paved the way for a solid theoretical foundation for compositional verification of parallel and distributed systems.

We are currently extending this work, looking at further properties of FH-bisimulation, but also the relations with existing equivalences on both closed and open systems. In particular, our model being significantly different from those considered in [3], it would be interesting to compare our "FH" family of bisimulations with the hierarchy of symbolic bisimulations from these authors. We also plan to apply open pNets to the study of complex composition operators in a symbolic way, for example in the area of parallel skeletons, or distributed algorithms. We have developed tool support for computing the symbolic semantics in term of open automata [12], and have developed algorithms to check strong FH-bisimulation [11]. Naturally we are now working toward an implementation of weak FH-Bisimulation. The challenges here, in the context of our symbolic systems, is not so much algorithmic complexity, as was the case with classical weak bisimulation on finite models, but decidability and termination. Two main directions should be explored, either with an explicit construction of the weak transition, but this step in itself may introduce non-termination, or a direct implementation of the weak bisimulation definition, without constructing the weak automaton. We can expect that optimisations here in the symbolic setting would be quite different from the classical case.

## References


[1] De Simone, R.: Higher-level synchronising devices in MEIJE-SCCS. Theoretical Computer Science **37** (1985) 245–267

[2] Larsen, K.G.: A context dependent equivalence between processes. Theoretical Computer Science **49** (1987) 184–215

[3] Ingólfsdóttir, A., Lin, H.: A symbolic approach to value-passing processes. In Bergstra, J.A., Ponse, A., Smolka, S.A., eds.: Handbook of Process Algebra. North-Holland/Elsevier (2001) 427–478

[4] Hennessy, M., Lin, H.: Symbolic bisimulations. Theoretical Computer Science **138**(2) (1995) 353–389

[5] Lin, H.: Symbolic transition graph with assignment. In Montanari, U., Sassone, V., eds.: Concur'96. Volume 1119 of LNCS., Springer, Heidelberg (1996) 50–65

[6] Hennessy, M., Rathke, J.: Bisimulations for a calculus of broadcasting systems. Theoretical Computer Science **200**(1-2) (1998) 225–260

[7] Barros, T., Ameur-Boulifa, R., Cansado, A., Henrio, L., Madelaine, E.: Behavioural models for distributed fractal components. Annales des Télécommunications **64**(1-2) (2009) 25–43

[8] Ameur-Boulifa, R., Henrio, L., Kulankhina, O., Madelaine, E., Savu, A.: Behavioural semantics for asynchronous components. Journal of Logical and Algebraic Methods in Programming **89** (2017) 1 – 40

[9] Henrio, L., Kulankhina, O., Madelaine, E.: Integrated environment for verifying and running distributed components. In: in proc. of the 19th Int. Conf. on Fundamental Approaches to Software Engineering (FASE'16), Springer (2016)

[10] Henrio, L., Madelaine, E., Zhang, M.: A Theory for the Composition of Concurrent Processes. In Albert, E., Lanese, I., eds.: 36th International Conference on Formal Techniques for Distributed Objects, Components, and Systems (FORTE). Volume LNCS-9688 of Formal Techniques for Distributed Objects, Components, and Systems., Heraklion, Greece (June 2016) 175–194

[11] Hou, Z., Madelaine, E.: Symbolic Bisimulation for Open and Parameterized Systems. In: PEPM 2020 - ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation, New-Orleans, United States (January 2020)

[12] Qin, X., Bliudze, S., Madelaine, E., Zhang, M.: Using SMT engine to generate symbolic automata. In: 18th International Workshop on Automated Verification of Critical Systems (AVOCS 2018), Electronic Communications of the EASST (2018)

[13] Boulifa, R.A., Halalai, R., Henrio, L., Madelaine, E.: Verifying safety of fault-tolerant distributed components. In: International Symposium on Formal Aspects of Component Software (FACS 2011). Lecture Notes in Computer Science, Oslo, Springer (September 2011)

[14] Johnson, K., Calinescu, R., Kikuchi, S.: An incremental verification framework for component-based software systems. In: Proceedings of the 16th International ACM Sigsoft Symposium on Component-based Software Engineering. CBSE '13, New York, NY, USA, ACM (2013) 33–42

[15] Johnson, K., Calinescu, R.: Efficient re-resolution of smt specifications for evolving software architectures. In: Proceedings of the 10th International ACM Sigsoft Conference on Quality of Software Architectures. QoSA '14, New York, NY, USA, ACM (2014) 93–102

[16] Gaspar, N., Henrio, L., Madelaine, E.: Formally reasoning on a reconfigurable component-based system — a case study for the industrial world. In: The 10th International Symposium on Formal Aspects of Component Software, Nanchang, China (October 2013)

[17] Baldan, P., Bracciali, A., Bruni, R.: Bisimulation by unification. In Kirchner, H., Ringeissen, C., eds.: Algebraic Methodology and Software Technology, 9th International Conference, AMAST 2002, France. Volume 2422 of Lecture Notes in Computer Science., Springer (2002) 254–270

[18] Baldan, P., Bracciali, A., Bruni, R.: A semantic framework for open processes. Theor. Comput. Sci. **389**(3) (2007) 446–483

[19] Dubut, J.: Bisimilarity of diagrams. In Fahrenberg, U., Jipsen, P., Winter, M., eds.: Relational and Algebraic Methods in Computer Science - 18th International Conference, RAMiCS 2020, Palaiseau, France, April 8-11, 2020, Proceedings [postponed]. Volume 12062 of Lecture Notes in Computer Science., Springer (2020) 65–81

[20] Abriola, S., Barceló, P., Figueira, D., Figueira, S.: Bisimulations on data graphs. J. Artif. Intell. Res. **61** (2018) 171–213

[21] Calder, M., Shankland, C.: A symbolic semantics and bisimulation for full lotos. In: International Conference on Formal Techniques for Networked and Distributed Systems, Springer (2001) 185–200

[22] Borgström, J., Briais, S., Nestmann, U.: Symbolic bisimulation in the spi calculus. In: International Conference on Concurrency Theory, Springer (2004) 161–176

[23] Delaune, S., Kremer, S., Ryan, M.: Symbolic bisimulation for the applied pi calculus. In: International Conference on Foundations of Software Technology and Theoretical Computer Science, Springer (2007) 133–145

[24] Liu, J., Lin, H.: A complete symbolic bisimulation for full applied pi calculus. In: International Conference on Current Trends in Theory and Practice of Computer Science, Springer (2010) 552–563

[25] Buscemi, M.G., Montanari, U.: Open bisimulation for the concurrent constraint pi-calculus. In: European Symposium on Programming, Springer (2008) 254–268

[26] Feng, Y., Deng, Y., Ying, M.: Symbolic bisimulation for quantum processes. ACM Transactions on Computational Logic (TOCL) **15**(2) (2014) 14

## Appendix A. Proof on FH-bisimulation

### A.1. **Bisimulation is an equivalence: Proof of Theorem 1.**

*Suppose $\mathcal{R}$ is an FH-bisimulation. Then $\mathcal{R}$ is an equivalence, that is, $\mathcal{R}$ is reflexive, symmetric and transitive.*

*Proof.* It is trivial to check reflexivity and symmetry. Here we focus on the transitivity. To prove transitivity of strong FH-bisimulation on pNets it is sufficient to prove transitivity of the strong FH-bisimulation on states. Consider 3 open automata $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$ and states $s$, $t$, $u$ in those automata[7]. Suppose we have $\mathcal{R}$ an FH-bisimulation relation between states of $\mathcal{T}_1$ and of $\mathcal{T}_2$; members of $\mathcal{R}$ are of the form $(s, t|Pred_{s,t})$. Suppose we also have $\mathcal{R}'$ an FH-bisimulation relation between states of $\mathcal{T}_2$ and of $\mathcal{T}_3$; members of $\mathcal{R}'$ are of the form $(t, u|Pred_{t,u})$.

Let $\mathcal{R}''$ be the relation:

$$\mathcal{R}'' = \{(s, u|Pred_{s,u}) \;\Big|\; Pred_{s,u} = \bigvee_{\substack{(s,t|Pred_{s,t}) \in \mathcal{R} \\ (t,u|Pred_{t,u}) \in \mathcal{R}'}} Pred_{s,t} \wedge Pred_{t,u}\}$$

This relation is the adaptation of the transitivity to the conditional relationship that defines a bisimulation. Indeed the global disjunction together with the conjunction of predicates plays exactly the role of the intermediate element in a transitivity rule: "there exists an intermediate state" corresponds to the global disjunction, and the conjunction of states expresses the intermediate predicate is used to ensure satisfiability of the predicate relating the first state to the last one.

The relation is built as follows: for each pair of states $s$, $u$, for each state $t$ such that $\mathcal{R}$ relates $s$ and $t$, and $\mathcal{R}'$ relates $t$ and $u$, we take the conjunction of the two predicates. The predicates for different values of $t$ are collected by a disjunction.

We will show that $\mathcal{R}''$ is an FH-bisimulation. Consider $(s, u|Pred_{s,u}) \in \mathcal{R}''$. Then there is a set of states of $\mathcal{T}_2$ relating $s$ and $u$, let $(t_p)^{p\in P}$ be this family. We have $Pred_{s,u} = \bigvee_{p\in P} Pred_{s,p} \wedge Pred_{p,u}$.

For any $p \in P$ by definition of $\mathcal{R}''$, $(s, t_p|Pred_{s,p}) \in \mathcal{R}$, and $(t_p, u|Pred'_{p,u}) \in \mathcal{R}'$. We have the following by definition of bisimulation: For any open transition $OT$ in $\mathcal{T}_1$ originating from $s$.

$$\frac{\beta_j^{j\in J_1}, Pred_{OT}, Post_{OT}}{s \xrightarrow{\alpha} s'}$$

There exist open transitions $OT_{px}^{x\in X} \subseteq \mathcal{T}_2$:

$$\frac{\beta_{jpx}^{j\in J_{px}}, Pred_{OT_{px}}, Post_{OT_{px}}}{t_p \xrightarrow{\alpha_{px}} t_{px}} \qquad (*)$$

such that $\forall x, J_1 = J_{px}, (s', t_{px}|Pred_{px}) \in \mathcal{R}$; and

[7] We omit the constraints stating that each $s_x$, $t_x$, $u_x$ is in the states of $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$ for the sake of readability

$$\begin{aligned}&Pred_{s,p} \wedge Pred_{OT}\\ \Longrightarrow & \bigvee_{x\in X} \left(\forall j. \beta_j = \beta_{jpx} \wedge Pred_{OT_{px}} \wedge \alpha = \alpha_{px} \wedge Pred_{px}\{\!\{Post_{OT} \uplus Post_{OT_{px}}\}\!\}\right)\end{aligned}$$

For any open transition $OT_{px}$, since $(t_p, u|Pred_{p,u}) \in \mathcal{R}'$ there exist open transitions $OT_{pxy}^{y\in Y} \subseteq \mathcal{T}_3$:

$$\frac{\beta_{jpxy}^{j\in J_{pxy}}, Pred_{OT_{pxy}}, Post_{OT_{pxy}}}{u \xrightarrow{\alpha_{pxy}} u_{pxy}} \qquad (**)$$

such that $\forall y, J_{px} = J_{pxy}, (t_{px}, u_{pxy}|Pred_{pxy}) \in \mathcal{R}'$; and

$$\begin{aligned}&Pred'_{p,u} \wedge Pred_{OT_{px}}\\ \Longrightarrow & \bigvee_{y\in Y} \left(\forall j. \beta_{jpx} = \beta_{jpxy} \wedge Pred_{OT_{pxy}} \wedge \alpha_{px} = \alpha_{pxy} \wedge Pred_{pxy}\{\!\{Post_{OT_{px}} \uplus Post_{OT_{pxy}}\}\!\}\right)\end{aligned}$$

This is verified for each $p \in P$.

Overall, we have a family of open transitions $OT_{pxy}^{p\in P, x\in X, y\in Y} \subseteq \mathcal{T}_3$ that should simulate $OT$.

First, we have $\forall y, \forall x, \forall p, J_1 = J_{px} = J_{pxy}, (s', u_{pxy}|Pred'_{pxy}) \in \mathcal{R}''$ for some $Pred'_{pxy}$. Indeed for any $p$, $x$, and $y$, $t_{px}$ relates $s'$ and $u_{pxy}$, we have $(s', t_{px}|Pred_{px}) \in \mathcal{R}$ and $(t_{px}, u_{pxy}|Pred_{pxy}) \in \mathcal{R}'$. More precisely, $t_{px} \in (t'_p)^{p\in P'}$ where $(t'_p)^{p\in P'}$ and $P' \subseteq P$ is the set of states relating $s'$ and $u_{pxy}$ (the states used in the open transition must belong to the set of states ensuring the transitive relation). Additionally, for all $p$, $x$, $y$, $Pred_{px} \wedge Pred_{pxy} \implies Pred'_{pxy}$ (this is one element of the disjunction defining the predicate $Pred'_{pxy}$ relating $s'$ and $u_{pxy}$ in the definition of $\mathcal{R}''$).

One can notice that, as bisimulation predicates are used to relate states that belong to two different open automata, the free variables of these predicates that do not belong to the two related automata can safely be renamed to avoid any name clash. In practice, we can suppose that $Pred'_{pxy}$ does not contain the variables of $\mathcal{T}_2$ because it is used to relate states of $\mathcal{T}_1$ and $\mathcal{T}_3$ . Indeed if $Pred'_{pxy}$ uses variables of $\mathcal{T}_2$, we can consider instead another predicate that is equivalent to $Pred'_{pxy}$ and does not contain the variables of $\mathcal{T}_2$ (this is safe according to the semantic interpretation of open automata and relations). Similarly, we can suppose that $Pred_{px}$ contains no variable in $\mathcal{T}_3$, and $Pred_{pxy}$ contains no variable in $\mathcal{T}_1$.

Second, by definition of bisimulation we need (recall that $Pred_{s,u}$ is the original predicate relating $s$ and $u$ by definition of the transitive closure):

$$\begin{aligned}&Pred_{s,u} \wedge Pred_{OT}\\ \Longrightarrow & \bigvee_{x\in X}\bigvee_{y\in Y}\bigvee_{p\in P} \left(\forall j. \beta_j = \beta_{jpxy} \wedge Pred_{OT_{pxy}} \wedge \alpha = \alpha_{pxy} \wedge Pred'_{pxy}\{\!\{Post_{OT} \uplus Post_{OT_{pxy}}\}\!\}\right).\end{aligned}$$

From (*) and (**) we have, for all $p$ $Pred_{s,p} \wedge Pred_{OT} \wedge Pred_{p,u}$

$$\begin{aligned}\Longrightarrow & \bigvee_{x\in X} \left(\forall j. \beta_j = \beta_{jpx} \wedge Pred_{OT_{px}} \wedge \alpha = \alpha_{px} \wedge Pred_{px}\{\!\{Post_{OT} \uplus Post_{OT_{px}}\}\!\}\right) \wedge Pred_{p,u}\\
\Longrightarrow & \bigvee_{x\in X} \left(\forall j. \beta_j = \beta_{jpx} \wedge (Pred_{OT_{px}} \wedge Pred_{p,u}) \wedge \alpha = \alpha_{px} \wedge Pred_{px}\{\!\{Post_{OT} \uplus Post_{OT_{px}}\}\!\}\right)\\
\Longrightarrow & \bigvee_{x\in X} \left(\forall j. \beta_j = \beta_{jpx} \wedge ( \bigvee_{y\in Y} \left(\forall j'. \beta_{j'px} = \beta_{j'pxy} \wedge Pred_{OT_{pxy}} \wedge \alpha_{px} = \alpha_{pxy}\right.\right.\\
&\quad \left.\left. \wedge Pred_{pxy}\{\!\{Post_{OT_{px}} \uplus Post_{OT_{pxy}}\}\!\}\right)) \wedge \alpha = \alpha_{px} \wedge Pred_{px}\{\!\{Post_{OT} \uplus Post_{OT_{px}}\}\!\}\right)\\
\Longrightarrow & \bigvee_{x\in X}\bigvee_{y\in Y} \left(\forall j, j'. \beta_j = \beta_{jpx} \wedge \beta_{j'px} = \beta_{j'pxy} \wedge \left(Pred_{OT_{pxy}} \wedge \alpha = \alpha_{px} = \alpha_{pxy}\right.\right.\\
&\quad \left.\left. \wedge Pred_{pxy}\{\!\{Post_{OT_{px}} \uplus Post_{OT_{pxy}}\}\!\} \wedge Pred_{px}\{\!\{Post_{OT} \uplus Post_{OT_{px}}\}\!\}\right)\right)\end{aligned}$$

By construction, four substitutions $\{\!\{\ \}\!\}$ only have an effect on the variables of the open automaton they belong to, they also produce terms containing only variables of the open automaton they belong to. Finally, because of the domain of the substitutions of the predicates, we have:

$Pred_{px}\{\!\{Post_{OT} \uplus Post_{OT_{px}}\}\!\} \wedge Pred_{pxy}\{\!\{Post_{OT_{px}} \uplus Post_{OT_{pxy}}\}\!\} \Leftrightarrow$
$Pred_{px}\{\!\{Post_{OT} \uplus Post_{OT_{px}} \uplus Post_{OT_{pxy}}\}\!\} \wedge Pred_{pxy}\{\!\{Post_{OT} \uplus Post_{OT_{px}} \uplus Post_{OT_{pxy}}\}\!\}$
$\implies Pred'_{pxy}\{\!\{Post_{OT} \uplus Post_{OT_{px}} \uplus Post_{OT_{pxy}}\}\!\} \Leftrightarrow$
$Pred'_{pxy}\{\!\{Post_{OT} \uplus Post_{OT_{pxy}}\}\!\}$

This allows us to conclude, with $Pred_{s,u} = \bigvee_{p \in P} Pred_{s,p} \wedge Pred_{p,u}$:

$$
\begin{aligned}
& Pred_{s,u} \wedge Pred_{OT} \\
\implies & \bigvee_{p\in P} (Pred_{s,p} \wedge Pred_{p,u} \wedge Pred_{OT}) \\
\implies & \bigvee_{p\in P} \bigvee_{x\in X} \bigvee_{y\in Y} \big(\forall j, j'. \beta_j = \beta_{jpx} \wedge \beta_{j'px} = \beta_{j'pxy} \wedge Pred_{OT_{pxy}} \\
& \quad \wedge \alpha = \alpha_{px} = \alpha_{pxy} \wedge Pred'_{pxy}\{\!\{Post_{OT} \uplus Post_{OT_{pxy}}\}\!\}\big) \\
\implies & \bigvee_{x\in X} \bigvee_{y\in Y} \bigvee_{p\in P} \big(\forall j. \beta_j = \beta_{jpxy} \wedge Pred_{OT_{pxy}} \wedge \alpha = \alpha_{pxy} \wedge Pred'_{pxy}\{\!\{Post_{OT} \uplus Post_{OT_{pxy}}\}\!\}\big)
\end{aligned}
$$

Concerning the other direction of bisimulation, it is sufficient to notice that the role of $s$ and $u$ in the definition of $\mathcal{R}''$ is symmetrical, and thus the proof is similar.

$\square$

A.2. **Composition Lemmas.** The proofs of the composition theorems for FH-bisimulation rely on two main lemmas, dealing respectively with the decomposition of a composed behaviour between the context and the internal pNet, and with their recomposition.

**Lemma 1**: *Open transition decomposition.*
Consider two pNets $P$ and $Q$ that are not pLTSs[8]. Let Leaves$(Q) = p_l^{l\in L_Q}$; suppose:

$$P[Q]_{j_0} \models \frac{\beta_j^{j\in J}, Pred, Post}{\triangleleft s_i^{i\in L} \triangleright \xrightarrow{\alpha} \triangleleft\, s_i'^{\,i\in L} \triangleright}$$

with $J \cap \text{Holes}(Q) \neq \emptyset$ or $\exists i \in L_Q.\, s_i \neq s'_i$, i.e. $Q$ takes part in the reduction. Then, there exist $\alpha_Q$, $Pred'$, $Pred''$, $Post'$, $Post''$ s.t.:

$$P \models \frac{\beta_j^{j\in (J\setminus \text{Holes}(Q))\cup\{j_0\}}, Pred', Post'}{\triangleleft s_i^{i\in L\setminus L_Q} \triangleright \xrightarrow{\alpha} \triangleleft\, s_i'^{\,i\in L\setminus L_Q} \triangleright}$$

$$\text{and } Q \models \frac{\beta_j^{j\in J\cap \text{Holes}(Q)}, Pred'', Post''}{\triangleleft s_i^{i\in L_Q} \triangleright \xrightarrow{\alpha_Q} \triangleleft\, s_i'^{\,i\in L_Q} \triangleright}$$

and $Pred \iff Pred' \wedge Pred'' \wedge \alpha_Q = \beta_{j_0}$, $Post = Post' \uplus Post''$ where $Post''$ is the restriction of $Post$ over variables $vars(Q)$.

**Preliminary note:** The introduction of fresh variables introduce alpha-conversion at many points of the proof; we only give major arguments concerning alpha-conversion to make the proof readable; in general, fresh variables appear in each transition inside terms $\beta_j$, $v$, and *Pred*.

[8] A similar lemma can be proven for a pLTS $Q$

*Proof.* Consider rule **Tr2** in Definition 9, applied to the pNet $P[Q]_{j_0}$.

$$\dfrac{\begin{array}{c}\text{Leaves}(\langle\!\langle P_m^{m\in I}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle) = pLTS_l^{l\in L} \qquad k\in K \qquad SV_k = (\alpha'_m)^{m\in I_1\uplus I_2\uplus J} \to \alpha'[e_b] \\ \forall m\in I_1. P_m \models \dfrac{\beta_j^{j\in J_m}, Pred_m, Post_m}{\triangleleft s_i^{i\in L_m} \triangleright \xrightarrow{\alpha_m} \triangleleft (s'_i)^{i\in L_m}\triangleright} \qquad \forall m\in I_2. P_m \models \dfrac{\emptyset, Pred_m, Post_m}{\triangleleft s_m \triangleright \xrightarrow{\alpha_m} \triangleleft s'_m\triangleright} \\ J' = \biguplus_{m\in I_1} J_m \uplus J \qquad Pred = \bigwedge_{m\in I_1\uplus I_2} Pred_m \wedge Pred_{sv}(SV_k, \alpha_m^{m\in I_1\uplus I_2}, \beta_j^{j\in J}, \alpha) \\ \forall i\in L\backslash\left(\biguplus_{m\in I_1} L_m \uplus I_2\right). s'_i = s_i \qquad \text{fresh}(\alpha'_m, \alpha', \beta_j, \alpha)\end{array}}{\langle\!\langle P_m^{m\in I}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle \models \dfrac{\beta_j^{j\in J'}, Pred, \biguplus_{m\in I_1\uplus I_2} Post_m}{\triangleleft s_i^{i\in L} \triangleright \xrightarrow{\alpha} \triangleleft (s'_i)^{i\in L}\triangleright}} \ \textbf{Tr2}$$

We know each premise is *True* for $P[Q]_{j_0}$. $j_0 \in I_1$ because $Q$ is not a pLTS. We try to prove the equivalent premise for $P$.

First, $K$ and the synchronisation vector $SV_k$ are unchanged (however $j_0$ passes from the set of sub-pNets to the set of holes). We have $\text{Leaves}(P[Q]_{j_0}) = \text{Leaves}(P) \uplus \text{Leaves}(Q)$.

Now focus on the OTs of the sub-pNets. For each $m \in I_1 \uplus I_2$ we have one of the two following OT:
either $m$ in $I_1$

$$P_m \models \dfrac{\beta_j^{j\in J_m}, Pred_m, Post_m}{\triangleleft s_i^{i\in L_m} \triangleright \xrightarrow{\alpha_m} \triangleleft (s'_i)^{i\in L_m}\triangleright}$$

or, $m$ in $I_2$

$$P_m \models \dfrac{\emptyset, Pred_m, Post_m}{\triangleleft s_m \triangleright \xrightarrow{\alpha_m} \triangleleft s'_m\triangleright}$$

Only elements of $(I_1 \uplus I_2)\backslash\{j_0\}$ are useful to assert the premise for reduction of $P$; the last one ensures the open transition for the pNet $Q$ (note that $Q$ is at place $j_0$, and by definition of the open transition for $P[Q]_{j_0}$, $L_{j_0} = L_Q$, and $J_{j_0} = J \cap \text{Holes}(Q)$):

$$Q \models \dfrac{\beta_j^{j\in J\cap\text{Holes}(Q)}, Pred_{j_0}, Post''}{\triangleleft s_i^{i\in L_Q} \triangleright \xrightarrow{\alpha_{j_0}} \triangleleft (s'_i)^{i\in L_Q}\triangleright}$$

This already ensures the second part of the conclusion of the lemma, i.e. the OT for $Q$ if we choose $\alpha_Q = \alpha_{j_0}$ and $Pred'' = Pred_{j_0}$. Considering the OT of $P$ we have another $J'$ that is $J'_p = J' \setminus \text{Holes}(Q) \uplus \{j_0\}$; we denote $I'_1 = I_1 \setminus \{j_0\}$ the predicate is $Pred' = \bigwedge_{m\in I'_1\uplus I_2} Pred_m \wedge Pred_{sv}(SV_k, \alpha_m^{m\in I'_1\uplus I_2}, \beta_j^{j\in J\cup\{j_0\}}, \alpha)$
where

$$Pred_{sv}(SV_k, \alpha_m^{m\in I'_1\uplus I_2}, \beta_j^{j\in J\cup\{j_0\}}, \alpha) \Leftrightarrow$$
$$\forall i \in I'_1 \uplus I_2.\, \alpha_i = \alpha'_i \wedge \forall j \in J\cup\{j_0\}.\, \beta_j = \alpha'_j \wedge \alpha = \alpha' \wedge e_b$$

Modulo renaming of fresh variables, this is identical to the predicate that occurs in the source open transition except $\alpha_{j_0} = \alpha'_{j_0}$ has been replaced by $\beta_{j_0} = \alpha'_{j_0}$. As $\alpha_{j_0} = \alpha_Q$ and $\beta_{j_0}$ is free, we have $\beta_{j_0} = \alpha'_{j_0} \wedge \beta_{j_0} = \alpha_Q \iff \alpha_{j_0} = \alpha'_{j_0}$. Thus,

$Pred \iff (Pred' \wedge Pred'') \wedge \alpha_Q = \beta_{j_0}$. Finally, Post into conditions of the context $P$ and the pNet $Q$ (they are builts similarly as they only deal with leaves): $Post = Post' \uplus Post''$. This concludes the proof as we checked all the premises of the open transition for both $P$ and $Q$. We obtain the following reduction by the rule **Tr2**:

$$\dfrac{\begin{array}{c}\text{Leaves}(\langle\!\langle P_m^{m\in I\setminus\{j_0\}}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle) = pLTS_l^{l\in L} \qquad k\in K \qquad SV_k = (\alpha'_m)^{m\in I_1\uplus I_2\uplus J} \to \alpha'[e_b] \\ \forall m\in I_1\setminus\{j_0\}. P_m \models \dfrac{\beta_j^{j\in J_m}, Pred_m, Post_m}{\triangleleft s_i^{i\in L_m}\triangleright \xrightarrow{\alpha_m} \triangleleft (s'_i)^{i\in L_m}\triangleright} \qquad \forall m\in I_2. P_m \models \dfrac{\emptyset, Pred_m, Post_m}{\triangleleft s_m\triangleright \xrightarrow{\alpha_m} \triangleleft s'_m\triangleright} \\ J' = \biguplus_{m\in I_1\setminus\{j_0\}} J_m \uplus J \qquad Pred' = \bigwedge_{m\in I_1\uplus I_2\setminus\{j_0\}} Pred_m \wedge Pred_{sv}(SV_k, \alpha_m^{m\in I_1\uplus I_2\setminus\{j_0\}}, \beta_j^{j\in J\cup\{j_0\}}, \alpha) \\ \forall i\in L\setminus\left(\biguplus_{m\in I_1\setminus\{j_0\}} L_m \uplus I_2\right). s'_i = s_i \qquad \text{fresh}(\alpha'_m, \alpha', \beta_j, \alpha)\end{array}}{\langle\!\langle P_m^{m\in I\setminus\{j_0\}}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle \models \dfrac{\beta_j^{j\in J\setminus \mathrm{Holes}(Q)\uplus\{j_0\}}, Pred', \biguplus_{m\in I_1\setminus\{j_0\}\uplus I_2} Post_m}{\triangleleft s_i^{i\in L\setminus L_Q}\triangleright \xrightarrow{\alpha} \triangleleft (s'_i)^{i\in L\setminus L_Q}\triangleright}}$$

□

In general, the actions that can be emitted by $Q$ is a subset of the possible actions of the holes, and the predicate involving $v_Q$ and the synchronisation vector is more restrictive than the one involving only the variable $\beta_{j_0}$. This has no impact on the previous proof and this restriction results from the composition of predicates.

**Lemma 2: Open transition composition.**
Consider two pNets $P$ and $Q$ where $P$ is not a pLTS. Suppose $j_0 \in J$ and:

$$P \models \dfrac{\beta_j^{j\in J}, Pred, Post}{\triangleleft s_i^{i\in L}\triangleright \xrightarrow{\alpha} \triangleleft s_i'^{i\in L}\triangleright} \qquad \text{and} \qquad Q \models \dfrac{\beta_j^{j\in J_Q}, Pred', Post'}{\triangleleft s_i^{i\in L_Q}\triangleright \xrightarrow{\alpha_Q} \triangleleft s_i'^{i\in L_Q}\triangleright}$$

Then, we have:

$$P[Q]_{j_0} \models \dfrac{\beta_j^{(j\in J\setminus\{j_0\})\uplus J_Q}, Pred \wedge Pred' \wedge \alpha_Q = \beta_{j_0}, Post \uplus Post'}{\triangleleft s_i^{i\in L\uplus L_Q}\triangleright \xrightarrow{\alpha} \triangleleft s_i'^{i\in L\uplus L_Q}\triangleright}$$

Note that this does not mean that any two pNets can be composed and produce an open transition. Indeed, the predicate $Pred \wedge Pred' \wedge \alpha_Q = \beta_{j_0}$ will not be satisfiable if the action of $\alpha_Q$ cannot be matched with $\beta_{j_0}$. Note also that $\beta_{j_0}$ is now only used as an intermediate term inside formulas: it does not appear neither as global action nor as an action of a hole.

*Proof.* Let $P = \langle\!\langle P_m^{m\in I}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle$. Consider first the open transition derived from $P$. Consider each premise of the open transition (constructed by **Tr2** in Definition 9).

$$\dfrac{\begin{array}{c}\text{Leaves}(\langle\!\langle P_m^{m\in I}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle) = pLTS_l^{l\in L} \qquad k\in K \qquad SV_k = (\alpha'_m)^{m\in I_1\uplus I_2\uplus J} \to \alpha'[e_b] \\ \forall m\in I_1.P_m \models \dfrac{\beta_j^{j\in J_m}, Pred_m, Post_m}{\triangleleft s_i^{i\in L_m} \triangleright \xrightarrow{\alpha_m} \triangleleft\, (s'_i)^{i\in L_m} \triangleright} \qquad \forall m\in I_2.P_m \models \dfrac{\emptyset, Pred_m, Post_m}{\triangleleft s_m \triangleright \xrightarrow{\alpha_m} \triangleleft\, s'_m \triangleright} \\ J' = \biguplus_{m\in I_1} J_m \uplus J \qquad Pred = \bigwedge_{m\in I_1\uplus I_2} Pred_m \wedge Pred_{SV}(SV_k, \alpha_m^{m\in I_1\uplus I_2}, \beta_j^{j\in J}, \alpha) \\ \forall i\in L\backslash \left(\biguplus_{m\in I_1} L_m \uplus I_2\right).\, s'_i = s_i \qquad \text{fresh}(\alpha'_m, \alpha', \beta_j, \alpha)\end{array}}{\langle\!\langle P_m^{m\in I}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle \models \dfrac{\beta_j^{\,j\in J'}, Pred, \biguplus_{m\in I_1\uplus I_2} Post_m}{\triangleleft s_i^{i\in L} \triangleright \xrightarrow{\alpha} \triangleleft\, (s'_i)^{i\in L} \triangleright}}\ \textbf{Tr2}$$

We know each premise is *True* for $P$ and try to prove the equivalent premise for $P[Q]_{j_0}$ (using the open transition of $Q$). $P[Q]_{j_0}$ exhibits a similar **Tr2** rule where $K$ and the synchronisation vector are unchanged ($j_0$ is now in the set of sub-pNets); $SV_k = (\alpha'_j)^{j\in I\uplus\{j_0\}\uplus J} \to \alpha'[e_b]$. Leaves$(P[Q]_{j_0})$ = Leaves$(P) \uplus$ Leaves$(Q)$. $I$ and $J$ are the set of leaves and holes of $P$, $I_1 \uplus I_2$ and $J'$ are the sets of moving leaves and holes in the reduction of $P$. All sub-pNets of must be reduced, we need:

$$\forall m\in I_1 \uplus \{j_0\}.P_m \models \dfrac{\beta_j^{j\in J_m}, Pred_m, Post_m}{\triangleleft s_i^{i\in L_m} \triangleright \xrightarrow{\alpha_m} \triangleleft\, (s'_i)^{i\in L_m} \triangleright}$$

$$\forall m\in I_2.P_m \models \dfrac{\emptyset, Pred_m, Post_m}{\triangleleft s_m \triangleright \xrightarrow{\alpha_m} \triangleleft\, s'_m \triangleright}$$

the sub-pNet at position $j_0$ is the one filled by $Q$ (we define $P_{j_0} = Q$ and similarly $J_m = J_Q$, $Pred_m = Pred'$, $Post_m = Post'$,... are the elements of the OT of Q) which offers an open transition by hypothesis, the other open transitions are immediate consequence of the open transition that can be performed by $P$ (premises of **Tr2**). The set of moving leaves is the union of the moving leaves in the open transition for $P$ and the ones for $Q$; similarly the moving holes are the union of the moving holes, minus $j_0$: $J'_{PQ} = J \setminus \{j_0\} \uplus J_Q$. The predicate for the open transition is:
$Pred'' = \bigwedge_{m\in I_1\uplus I_2} Pred_m \wedge Pred' \wedge Pred(SV_k, v_i^{i\in I_1\uplus I_2} \uplus (j_0{\mapsto}v_Q), \beta_j^{j\in J}, \alpha)$.
By definition we have:
$Pred(SV_k, \alpha_i^{i\in I} \uplus (j_0{\mapsto}\alpha_Q), \beta_j^{j\in J}, v) \Leftrightarrow \forall i \in I.\, \alpha_i = \alpha'_i \wedge \forall j \in J.\, \beta_j = \alpha'_j \wedge \alpha = \alpha' \wedge \alpha_Q = \alpha_{j_0} \wedge e_b$, this is equivalent to $\forall i \in I.\, \alpha_i = \alpha'_i \wedge \forall j \in J.\, \beta_j = \alpha'_j \wedge \alpha = \alpha' \wedge \alpha_Q = \beta_{j_0} \wedge \beta_{j_0} = \alpha_{j_0} \wedge e_b$ and by definition of *Pred* (as obtained by **Tr2**), $Pred'' \iff Pred \wedge Pred' \wedge \alpha_Q = \beta_{j_0}$. The post-condition gathers the post-conditions related to all the leaves:
$\biguplus_{m\in I_1\cup\{j_0\}\uplus I_2} Post_m = Post \uplus Post'$.

Finally, the composed open transition can be built by **Tr2** as follows

$$\frac{\begin{array}{c}\text{Leaves}(\langle\!\langle P_m^{m\in I\cup\{j_0\}}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle) = pLTS_l^{l\in L} \qquad k\in K \qquad SV_k = (\alpha'_m)^{m\in I_1\uplus I_2\uplus J} \rightarrow \alpha'[e_b] \\ \forall m{\in}I_1\cup\{j_0\}.P_m \models \dfrac{\beta_j^{j\in J_m}, Pred_m, Post_m}{\triangleleft s_i^{i\in L_m} \triangleright \xrightarrow{\alpha_m} \triangleleft\, (s'_i)^{i\in L_m} \triangleright} \qquad \forall m{\in}I_2.P_m \models \dfrac{\emptyset, Pred_m, Post_m}{\triangleleft s_m \triangleright \xrightarrow{\alpha_m} \triangleleft\, s'_m \triangleright} \\ J'_{PQ} = J\setminus\{j_0\}\uplus J_Q \qquad Pred'' = \bigwedge_{m\in I_1\uplus I_2} Pred_m \wedge Pred' \wedge Pred(SV_k, v_i^{i\in I_1\uplus I_2} \uplus (j_0{\mapsto}v_Q), \beta_j^{j\in J}, \alpha) \\ \forall i \in L\setminus\left(\biguplus_{m\in I_1\cup\{j_0\}} L_m \uplus I_2\right). s'_i = s_i \qquad \text{fresh}(\alpha'_m, \alpha', \beta_j, \alpha)\end{array}}{\langle\!\langle P_m^{m\in I}, \overline{Sort}, SV_k^{k\in K}\rangle\!\rangle \models \dfrac{{\beta_j}^{j\in J'_{PQ}}, Pred'', Post \uplus Post'}{\triangleleft s_i^{i\in L\uplus L_Q} \triangleright \xrightarrow{\alpha} \triangleleft\, (s'_i)^{i\in L\uplus L_Q} \triangleright}}$$

This provides the desired conclusion. □

Note that we also have the following lemma (trivial):

**Lemma 3** Open transition composition – inactive**.**
*This lemma is the simple case where the pNet filling the hole is not involved in the transition. Suppose $j_0 \notin J$ and $L_Q = \text{Leaves}(Q)$:*

$$P \models \frac{\beta_j^{j\in J}, Pred, Post}{\triangleleft s_i^{i\in L} \triangleright \xrightarrow{\alpha} \triangleleft\, {s'}_i^{i\in L} \triangleright}$$

*Then, for any state $\triangleleft s_i^{i\in L_Q} \triangleright$ of $Q$, we have*

$$P[Q]_{j_0} \models \frac{\beta_j^{j\in J}, Pred, Post}{\triangleleft s_i^{i\in L} \uplus s_i^{i\in L_Q} \triangleright \xrightarrow{\alpha} \triangleleft\, {s'}_i^{i\in L} \uplus s_i^{i\in L_Q} \triangleright}$$

The proof is trivial.

### A.3. **Proof of Theorem 3.**

**Congruence**: *Consider an open pNet: $P = \langle\!\langle P_i^{i\in I}, Sort_j^{j\in J}, \overline{SV}\rangle\!\rangle$. Let $j_0 \in J$ be a hole. Let $Q$ and $Q'$ be two FH-bisimilar pNets such that* $\text{Sort}(Q) = \text{Sort}(Q') = Sort_{j_0}$[9]*. Then $P[Q]_{j_0}$ and $P[Q']_{j_0}$ are FH-bisimilar.*

*Proof.* The proof of Theorem 3 exhibits classically a bisimulation relation for a composed system. It considers then an open transition of $P[Q]_{j_0}$ that should be simulated. It then uses Lemma 1 to decompose the open transition of $P[Q]_{j_0}$ and obtain an open transition of $P$ and $Q$; the FH-bisimulation property can be applied to $Q$ to obtain an equivalent family of open transitions of $Q'$; this family is then recomposed by Lemma 2 to build a set of open transitions of $P[Q']_{j_0}$ that will simulate the original one.

Let $\text{Leaves}(Q) = p_l^{l\in L}$, $\text{Leaves}(Q') = {p'}_l^{l\in L'}$, $\text{Leaves}(P) = p_l^{l\in L_P}$. Consider $Q$ FH-bisimilar to $Q'$. It means that there is a relation $\mathcal{R}$ that is an FH-bisimulation between the open automata of the two pNets. We will consider the relation $\mathcal{R}' = \{(s, t|Pred_{s,t})|s = s' \uplus s'' \wedge t = t' \uplus s'' \wedge s'' \in \mathcal{S}_P \wedge (s', t'|Pred_{s,t}) \in \mathcal{R}\}$ where $\mathcal{S}_P$ is the set of states of the open automaton of $P$. We will prove that $\mathcal{R}'$ is an open FH-bisimulation. Consider a pair of

[9] Note that $\text{Sort}(Q) = \text{Sort}(Q')$ is ensured by strong bisimilarity.

FH-bisimilar states: $(\triangleleft s_i^{i\in L_P\uplus L}\triangleright, \triangleleft t_i^{i\in L'} \uplus s_i^{i\in L_P}\triangleright | Pred_{s,t}) \in \mathcal{R}'$. Consider an open transition $OT$ of $P[Q]_{j_0}$.

$$P[Q]_{j_0} \models \frac{\beta_j^{j\in J}, Pred_{OT}, Post_{OT}}{\triangleleft s_i^{i\in L_P\uplus L}\triangleright \xrightarrow{\alpha} \triangleleft s'^{\,i\in L_P\uplus L}_i\triangleright}$$

Let $J' = J \setminus \mathrm{Holes}(Q) \cup \{j_0\}$. By Lemma 1 we have :

$$P \models \frac{\beta_j^{j\in J'}, Pred', Post'}{\triangleleft s_i^{i\in L_P}\triangleright \xrightarrow{\alpha} \triangleleft s'^{\,i\in L_P}_i\triangleright}$$

$$Q \models \frac{\beta_j^{j\in J\cap \mathrm{Holes}(Q)}, Pred'', Post''}{\triangleleft s_i^{i\in L}\triangleright \xrightarrow{\alpha_Q} \triangleleft s'^{\,i\in L}_i\triangleright}$$

and $Pred_{OT} \iff Pred' \wedge Pred'' \wedge \alpha_Q = \beta_{j_0}$, $Post_{OT} = Post' \uplus Post''$ ($Post''$ is the restriction of $Post$ over $vars(Q)$). As $Q$ is FH-bisimilar to $Q'$ and $(\triangleleft s_i^{i\in L}\triangleright, \triangleleft t_i^{i\in L'}\triangleright | Pred_{s,t}) \in \mathcal{R}$ there is a family $OT'_x$ of open transitions of the automaton of $Q'$ such that

$$Q' \models \frac{\beta_{jx}^{j\in J\cap \mathrm{Holes}(Q)}, Pred_{OT_x}, Post_{OT_x}}{\triangleleft t_i^{i\in L'}\triangleright \xrightarrow{\alpha_x} \triangleleft t_{ix}^{i\in L'}\triangleright}$$

and $\forall x, (\triangleleft s_i^{i\in L}\triangleright, \triangleleft t_{ix}^{i\in L'}\triangleright | Pred_{sx}) \in \mathcal{R}$; and
$Pred_{s,t} \wedge Pred'' \implies$
$\bigvee_{x\in X} (\forall j \in J\cap \mathrm{Holes}(Q). \beta_j = \beta_{jx} \wedge Pred_{OT_x} \wedge \alpha_Q = \alpha_x \wedge Pred_{s,x}\{\{Post'' \uplus Post_{OT_x}\}\})$

We can now apply Lemma 2 on each of the $OT'_x$ together with the transition of $P$ and obtain a new family $OT_x$ of open transitions (where for $i \in L_P$, $t_i = s_i$ and $t_{ix} = s'_i$, and for $j \in Holes(P)$, $\beta_{jx} = \beta_j$):

$$P[Q']_{j_0} \models \frac{\beta_{jx}^{j\in J}, Pred' \wedge Pred_{OT_x} \wedge \alpha_x = \beta_{j_0x}, Post' \uplus Post_{OT_x}}{\triangleleft t_i^{i\in L'\uplus L_Q}\triangleright \xrightarrow{\alpha_x} \triangleleft t_{ix}^{i\in L'\uplus L_Q}\triangleright}$$

Observe that we used the fact that $J = (J \setminus \mathrm{Holes}(Q) \cup \{j_0\}) \setminus \{j_0\} \cup (J \cap \mathrm{Holes}(Q))$. Now we have to verify the conditions for the FH-bisimulation between $OT$ and $OT_x$. $\forall x, (\triangleleft s'^{\,i\in L_P\uplus L}_i\triangleright, \triangleleft t_{ix}^{i\in L_P\uplus L'}\triangleright | Pred_{s,x}) \in \mathcal{R}'$ (by definition of $\mathcal{R}'$) and in three steps we get:
$Pred_{s,t} \wedge Pred_{OT} \implies Pred_{s,t} \wedge Pred' \wedge Pred'' \wedge \alpha_Q = \beta_{j_0}$
$\implies \bigvee_{x\in X} (\forall j \in J\cap \mathrm{Holes}(Q). \beta_j = \beta_{jx} \wedge Pred_{OT_x} \wedge \alpha_Q = \alpha_x \wedge Pred_{s,x}\{\{Post'' \uplus Post_{OT_x}\}\}) \wedge$
$Pred' \wedge \alpha_Q = \beta_{j_0}$
$\implies \bigvee_{x\in X} (\forall j \in J\cap \mathrm{Holes}(Q). \beta_j = \beta_{jx} \wedge Pred' \wedge Pred_{OT_x} \wedge \alpha_Q = \alpha_x \wedge \alpha_Q = \beta_{j_0x} \wedge$
$Pred_{s,x}\{\{Post'' \uplus Post_{OT_x}\}\})$

Note that, $\beta_{j_0}$ can be transformed into $\beta_{j_0x}$ because of the implication hypothesis. The obtained formula reaches the goal except for two points:

- We need $\forall j \in J$ instead of $\forall j \in J \cap \mathrm{Holes}(Q)$ but adding prerequisite on more variables does not change the validity of the formula (those variables are not used).
- Concerning the last term, we need $Pred_{sx}\{\{Post_{OT} \uplus (Post' \uplus Post_{OT_x})\}\}$, i.e. $Pred_{s,x}\{\{(Post' \uplus Post'') \uplus (Post' \uplus Post_{OT_x})\}\}$. We can conclude by observing that $Pred_{s,x}$ does not use any variable of $P$ and thus the substitution $\{\{Post'\}\}$ has no effect on it.

Finally:

$Pred_{s,t} \wedge Pred_{OT} \implies$

$\bigvee_{x\in X} \left(\forall j \in J.\beta_j = \beta_{jx} \wedge \left(Pred' \wedge Pred_{OT_x} \wedge \alpha_Q = \beta_{j_0 x}\right) \wedge \alpha_Q = \alpha_x \wedge Pred_{s,x}\{\!\{Post'' \uplus Post_{OT_x}\}\!\}\right)$

This proves the condition of the FH-simulation, the other direction is similar. $\square$

A.4. **Proof of Theorem 4: Context equivalence.** *Consider two FH-bisimilar open pNets: $P = \langle\!\langle P_i^{i\in I}, Sort_j^{j\in J}, \overline{SV}\rangle\!\rangle$ and $P' = \langle\!\langle P'^{i\in I}_i, Sort_j^{j\in J}, \overline{SV'}\rangle\!\rangle$ (recall they must have the same holes to be bisimilar). Let $j_0 \in J$ be a hole, and $Q$ be a pNet such that* $\mathrm{Sort}(Q) = Sort_{j_0}$. *Then $P[Q]_{j_0}$ and $P'[Q]_{j_0}$ are FH-bisimilar.*

*Proof.* The proof of Theorem 4 exhibits a bisimulation relation for a composed system. It then uses Lemma 1 to decompose the open transition of $P[Q]_{j_0}$ and obtain an open transition of $P$ on which the FH-bisimulation property can be applied to obtain an equivalent family of open transitions of $P'$; this family is then recomposed by Lemma 2 to build a set of open transitions of $P'[Q]_{j_0}$ that will simulate the original one.

Let $\mathrm{Leaves}(Q) = p_l^{l\in L_Q}$, $\mathrm{Leaves}(P) = p_l^{l\in L}$, $\mathrm{Leaves}(P') = p'^{l\in L'}_l$. Consider $P$ FH-bisimilar to $P'$. It means that there is a relation $\mathcal{R}$ that is an FH-bisimulation between the open automata of the two pNets. We will consider the relation $\mathcal{R}' = \{(s,t|Pred_{s,t})|s = s' \uplus s'' \wedge t = t' \uplus s'' \wedge s \in \mathcal{S}_Q \wedge (s',t'|Pred_{s,t}) \in \mathcal{R}\}$ where $\mathcal{S}_Q$ is the set of states of the open automaton of $Q$. We will prove that $\mathcal{R}'$ is an open FH-bisimulation. Consider a pair of FH-bisimilar states: $(\triangleleft s_{1i}^{i\in L\uplus L_Q}\triangleright, \triangleleft s_{2i}^{i\in L'} \uplus s_{1i}^{i\in L_Q}\triangleright|Pred) \in \mathcal{R}'$. Consider an open transition $OT$ of $P[Q]_{j_0}$.

$$P[Q]_{j_0} \models \frac{\beta_j^{j\in J}, Pred_{OT}, Post_{OT}}{\triangleleft s_i^{i\in L\uplus L_Q}\triangleright \xrightarrow{\alpha} \triangleleft s'^{\,i\in L\uplus L_Q}_i\triangleright}$$

Let $J' = J \setminus \mathrm{Holes}(Q) \cup \{j_0\}$. By Lemma 1 we have :

$$P \models \frac{\beta_j^{j\in J'}, Pred', Post'}{\triangleleft s_1^{i\in L}\triangleright \xrightarrow{\alpha} \triangleleft s'^{\,i\in L}_i\triangleright}$$

$$Q \models \frac{\beta_j^{j\in J\cap \mathrm{Holes}(Q)}, Pred'', Post''}{\triangleleft s_i^{i\in L_Q}\triangleright \xrightarrow{\alpha_Q} \triangleleft s'^{\,i\in L_Q}_i\triangleright}$$

and $Pred_{OT} \iff Pred' \wedge Pred'' \wedge \alpha_Q = \beta_{j_0}$, $Post_{OT} = Post' \uplus Post''$ ($Post''$ is the restriction of $Post$ over $vars(Q)$). As $P$ is FH-bisimilar to $P'$ and $(\triangleleft s_i^{i\in L}\triangleright, \triangleleft t_i^{i\in L'}\triangleright|Pred_{s,t}) \in \mathcal{R}$ there is a family $OT'_x$ of open transitions of the automaton of $P'$ such that

$$P' \models \frac{\beta_{jx}^{j\in J'}, Pred_{OT_x}, Post_{OT_x}}{\triangleleft t_i^{i\in L'}\triangleright \xrightarrow{\alpha_x} \triangleleft t_{ix}^{i\in L'}\triangleright}$$

and $\forall x, (\triangleleft s_i^{i\in L}\triangleright, \triangleleft t_{ix}^{i\in L'}\triangleright|Pred_{sx}) \in \mathcal{R}$; and

$Pred_{s,t} \wedge Pred' \implies \bigvee_{x\in X} \left(\forall j \in J'.\beta_j = \beta_{jx} \wedge Pred_{OT_x} \wedge \alpha = \alpha_x \wedge Pred_{s,x}\{\!\{Post' \uplus Post_{OT_x}\}\!\}\right)$

We can now apply Lemma 2 on each of the $OT'_x$ together with the transition of $Q$ and obtain a new family $OT_x$ of open transitions (where for $i \in L_Q$, $t_i = s_i$ and $t_{ix} = s'_i$, and for $j \in Holes(Q)$, $b_{jx} = b_j$):

$$P'[Q]_{j_0} \models \dfrac{\beta_{jx}^{j\in J}, Pred_{OT_x} \wedge Pred'' \wedge \alpha_Q = \beta_{j_0 x}, Post_{OT_x} \uplus Post''}{\triangleleft t_i^{i\in L'\uplus L_Q} \triangleright \xrightarrow{\alpha_x} \triangleleft\, t_{ix}^{i\in L'\uplus L_Q} \triangleright}$$

Observe that $J = (J \setminus \mathrm{Holes}(Q) \cup \{j_0\}) \setminus \{j_0\} \cup (J \cap \mathrm{Holes}(Q))$. Now we have to verify the conditions for the FH-bisimulation between $OT$ and $OT_x$. $\forall x, (\triangleleft {s'}_i^{i\in L\uplus L_Q} \triangleright, \triangleleft t_{ix}^{i\in L'\uplus L_Q} \triangleright | Pred_{s,x}) \in \mathcal{R}'$ (by definition of $\mathcal{R}'$) and in four steps we get:

$Pred_{s,t} \wedge Pred_{OT} \implies Pred_{s,t} \wedge Pred' \wedge Pred'' \wedge \alpha_Q = \beta_{j_0}$

$\Longrightarrow \bigvee_{x\in X} (\forall j \in J'. \beta_j = \beta_{jx} \wedge Pred_{OT_x} \wedge \alpha_Q = \beta_{j_0} \wedge \alpha = \alpha_x \wedge Pred_{s,x}\{\!\{Post' \uplus Post_{OT_x}\}\!\}) \wedge Pred''$

$\Longrightarrow \bigvee_{x\in X} (\forall j \in J'. \beta_j = \beta_{jx} \wedge (Pred_{OT_x} \wedge Pred'' \wedge \alpha_Q = \beta_{j_0 x}) \wedge \alpha = \alpha_x \wedge Pred_{s,x}\{\!\{Post' \uplus Post_{OT_x}\}\!\})$

The obtained formula reaches the goal except for two points:

- We need $\forall j \in J$ instead of $\forall j \in J'$ with $J' = J \setminus \mathrm{Holes}(Q) \cup \{j_0\}$ but the formula under the quantifier does not depend on $b_{j_0}$ now (thanks to the substitution). Concerning $\mathrm{Holes}(Q)$, adding prerequisite on more variables does not change the validity of the formula (those variables are not used).
- We need $Pred_{s,x}\{\!\{Post_{OT} \uplus (Post_{OT_x} \uplus Post'')\}\!\}$, i.e., $Pred_{s,x}\{\!\{(Post' \uplus Post'') \uplus (Post_{OT_x} \uplus Post'')\}\!\}$. We can conclude by observing that $Pred_{s,x}$ does not use any variable of $Q$ and thus the substitution involving $Post''$ has no effect.

This proves the condition of the FH-simulation, the other direction is similar. $\square$

## Appendix B. Weak FH-bisimulation lemmas and proofs

We define a quantified composition operator for effects, i.e. *Post* elements of the open transitions. We use $\bigotimes_{i=n}^{0} Post_i$ to denote $Post_n \otimes .. \otimes Post_0$. By convention $\bigotimes_{i=-1}^{0} Post_i$ is the identity.

### B.1. Weak bisimulation is an equivalence.

In this section, we first define two alternative definitions, one for weak open transition, one for for weak bisimulation. We use these two alternative definitions to show that weak bisimulation is an equivalence, we will also re-use these alternative definitions in the proofs of the theorems in next sections.

**Lemma 4** Alternative definition of weak open transitions. *Let $A = \langle\!\langle J, \mathcal{S}, s_0, V_1, \mathcal{T} \rangle\!\rangle$ be an open automaton and $\langle\!\langle J, \mathcal{S}, s_0, V_2, \mathcal{WT} \rangle\!\rangle$ be the weak open automaton derived from $A$. The two following statements are equivalent*

(1) *Either $\alpha = \tau \wedge \overline{\gamma} = \emptyset \wedge Pred = True \wedge Post = \mathrm{Id}(s) \wedge s = s'$; or there exist $\overline{\beta_{1i}}$, $\overline{\beta_{2i}}$, and $\overline{\beta_{3i}}$, $Pred_{1i}$, $Pred_{3i}$, $Post_{1i}$, and $Pred_2$, $Post_2$, $n \geq -1$, $m \geq -1$ s.t.*[10]*:*

[10] $n = -1$ (resp. $m = -1$) corresponds to the case where there is no $\tau$ transition before (resp. after) the transition $\alpha$.

$$\forall i \in [0..n]. \frac{\overline{\beta_{1i}}, Pred_{1i}, Post_{1i}}{s_{1i} \xrightarrow{\tau} s_{1(i+1)}} \in \mathcal{T} \quad \wedge \quad \frac{\overline{\beta_2}, Pred_2, Post_2}{s_2 \xrightarrow{\alpha} s'_2} \in \mathcal{T} \quad \wedge$$

$$\forall i \in [0..m]. \frac{\overline{\beta_{3i}}, Pred_{3i}, Post_{3i}}{s_{3i} \xrightarrow{\tau} s_{3(i+1)}} \in \mathcal{T}$$

(2) *there exist $\overline{\gamma}$, Pred, Post s.t.*

$$\frac{\overline{\gamma}, Pred, Post}{s \xRightarrow{\alpha'} s'} \in \mathcal{WT}$$

*where*

$$\alpha' = \alpha\{\{\bigotimes_{j=n}^{0} Post_{1j}\}\}$$

$$s = s_{10} \wedge s_{1(n+1)} = s_2 \wedge s'_2 = s_{30} \wedge s_{3(m+1)} = s'$$

$$\overline{\gamma} = \biguplus_{i=0}^{n} (\overline{\beta_{1i}}\{\{\bigotimes_{j=i-1}^{0} Post_{1j}\}\})^{\nabla} \uplus (\overline{\beta_2}\{\{\bigotimes_{j=n}^{0} Post_{1j}\}\})^{\nabla} \uplus$$
$$\biguplus_{i=0}^{m} (\overline{\beta_{3i}}\{\{\bigotimes_{j=i-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n}^{0} Post_{1j}\}\})^{\nabla}$$

$$Pred = \bigwedge_{i=0}^{n} Pred_{1i}\{\{\bigotimes_{j=i-1}^{0} Post_{1j}\}\} \wedge Pred_2\{\{\bigotimes_{j=n}^{0} Post_{1j}\}\} \wedge$$
$$\Big(\bigwedge_{i=0}^{m} Pred_{3i}\{\{\bigotimes_{j=i-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n}^{0} Post_{1j}\}\}\Big)$$

$$Post = \bigotimes_{j=m}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n}^{0} Post_{1j}$$

*Proof.* ($\Rightarrow$) We present an induction on $n$ and $m$, focusing on the incrementation on $n$: we prove that the property is valid for $m = -1$, $n = -1$ apply a first induction proof for going from $n$ to $n+1$, a similar induction can be applied to go from $m$ to $m+1$ (omitted).

- The base case there is one transition, so $n = -1$ and $m = -1$, we have:

  $$\frac{\overline{\beta}, Pred, Post}{s \xrightarrow{\alpha} s'} \in \mathcal{T}$$

  by rule **WT2** we can directly conclude the implication:

  $$\frac{\overline{\beta}, Pred, Post}{s \xrightarrow{\alpha} s'} \in \mathcal{T} \Rightarrow \frac{(\overline{\beta})^{\nabla}, Pred, Post}{s \xRightarrow{\alpha} s'} \in \mathcal{WT}$$

- For the inductive step, first we have by induction hypothesis that the formula holds for some lengths $m$ and $n$. Induction step is to infer that formula holds for transitions of length $n+1$. We consider the case $n' = n+1$. We want to prove (1) $\Rightarrow$ (2) in the lemma, and in (1) we focus on the case where there is a set of open transitions (this is the case:

$s \neq s'$). In other words, we consider the sequence of $(n+m+4)$ open transitions:

$$\Big(\forall i \in [0..n+1].\frac{\overline{\beta_{1i}}, Pred_{1i}, Post_{1i}}{s_{1i} \xrightarrow{\tau} s_{1(i+1)}} \in \mathcal{T} \quad \wedge \quad \frac{\overline{\beta_{2}}, Pred_{2}, Post_{2}}{s_{2} \xrightarrow{\alpha} s'_{2}} \in \mathcal{T} \quad \wedge$$

$$\forall i \in [0..m].\frac{\overline{\beta_{3i}}, Pred_{3i}, Post_{3i}}{s_{3i} \xrightarrow{\tau} s_{3(i+1)}} \in \mathcal{T}\Big)$$

By recurrence hypothesis we suppose that (1) $\Rightarrow$ (2) holds for $n$ and $m$ (compared to the line above, we remove the first $\tau$ transition). We have:

$$\Big(\forall i \in [1..n+1].\frac{\overline{\beta_{1i}}, Pred_{1i}, Post_{1i}}{s_{1i} \xrightarrow{\tau} s_{1(i+1)}} \in \mathcal{T} \wedge \frac{\overline{\beta_{2}}, Pred_{2}, Post_{2}}{s_{2} \xrightarrow{\alpha} s'_{2}} \in \mathcal{T} \wedge$$

$$\forall i \in [0..m].\frac{\overline{\beta_{3i}}, Pred_{3i}, Post_{3i}}{s_{3i} \xrightarrow{\tau} s_{3(i+1)}} \in \mathcal{T}\Big) \Rightarrow \frac{\overline{\gamma}, Pred, Post}{s'' \xRightarrow{\alpha'} s'} \in \mathcal{WT} \text{ where}$$

$$s'' = s_{10} \wedge s_{1(n+2)} = s_2 \wedge s'_2 = s_{30} \wedge s_{3(m+1)} = s'$$

$$\alpha' = \alpha\{\!\{ \bigotimes_{j=n+1}^{1} Post_{1j} \}\!\}$$

$$\overline{\gamma} = \biguplus_{i=1}^{n+1} (\overline{\beta_{1i}}\{\!\{ \bigotimes_{j=i-1}^{1} Post_{1j} \}\!\})^{\nabla} \uplus (\overline{\beta_{2}}\{\!\{ \bigotimes_{j=n+1}^{1} Post_{1j} \}\!\})^{\nabla} \uplus$$

$$\biguplus_{i=0}^{m} (\overline{\beta_{3i}}\{\!\{ \bigotimes_{j=i-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n}^{0} Post_{1j} \}\!\})^{\nabla}$$

$$Pred = \bigwedge_{i=1}^{n+1} Pred_{1i}\{\!\{ \bigotimes_{j=i-1}^{1} Post_{1j} \}\!\} \wedge Pred_2\{\!\{ \bigotimes_{j=n+1}^{1} Post_{1j} \}\!\} \wedge$$

$$\Big( \bigwedge_{i=0}^{m} Pred_{3i}\{\!\{ \bigotimes_{j=i-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n+1}^{1} Post_{1j} \}\!\}\Big)$$

$$Post = \bigotimes_{j=m}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n+1}^{1} Post_{1j}$$

We need to prove that by adding the following open transition the implication remains true:

$$\frac{\overline{\beta_{10}}, Pred_{10}, Post_{10}}{s_{10} \xrightarrow{\tau} s_{11}} \in \mathcal{T}$$

First by using rule **WT2** we have:

$$\frac{\overline{\beta_{10}}, Pred_{10}, Post_{10}}{s_{10} \xrightarrow{\tau} s_{11}} \in \mathcal{T} \Rightarrow \frac{(\overline{\beta_{10}})^{\nabla}, Pred_{10}, Post_{10}}{s_{10} \xRightarrow{\tau} s_{11}} \in \mathcal{WT}$$

On the other hand, by rule **WT1** we have the following weak open transition:

$$\frac{\emptyset, True, \mathrm{Id}(s')}{s' \xRightarrow{\tau} s'} \in \mathcal{WT}$$

Finally by applying rule **WT3** on the above weak open transitions:

$$\dfrac{\dfrac{(\overline{\beta_{10}})^{\nabla}, Pred_{10}, Post_{10}}{s_{10} \overset{\tau}{\Rightarrow} s_{11}} \in \mathcal{WT} \qquad \dfrac{\overline{\gamma}, Pred, Post}{s'' \overset{\alpha'}{\Rightarrow} s'} \in \mathcal{WT} \qquad \dfrac{\emptyset, True, \mathrm{Id}(s'')}{s' \overset{\tau}{\Rightarrow} s'} \in \mathcal{WT} \\ \overline{\gamma''} = (\overline{\beta_{10}})^{\nabla} \uplus \overline{\gamma}\{\{Post_{10}\}\} \qquad Pred'' = Pred_{10} \wedge Pred\{\{Post_{10}\}\} \\ Post'' = \mathrm{Id}(s'') \otimes Post \otimes Post_{10} \qquad \alpha'' = \alpha'\{\{Post_{10}\}\}}{\dfrac{\overline{\gamma''}, Pred'', Post''}{s_{10} \overset{\alpha''}{\Longrightarrow} s'} \in \mathcal{WT}} \textbf{WT3}$$

where we obtain the conclusion of the lemma, as required with the following assertions (derived from previous assertions):

$$s_{10} = s_{10} \wedge s_{1(n+2)} = s_2 \wedge s_2' = s_{30} \wedge s_{3(m+1)} = s'$$

$$\alpha'' = \alpha\{\{\bigotimes_{j=n+1}^{1} Post_{1j}\}\}\{\{Post_{10}\}\} = \alpha\{\{\bigotimes_{j=n+1}^{0} Post_{1j}\}\}$$

$$\overline{\gamma''} = \biguplus_{i=0}^{n+1} (\overline{\beta_{1i}}\{\{\bigotimes_{j=i-1}^{1} Post_{1j}\}\})^{\nabla} \uplus (\overline{\beta_2}\{\{\bigotimes_{j=n+1}^{1} Post_{1j}\}\})^{\nabla} \uplus$$
$$\biguplus_{i=0}^{m} (\overline{\beta_{3i}}\{\{\bigotimes_{j=i-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n}^{0} Post_{1j}\}\})^{\nabla}$$

$$Pred = \bigwedge_{i=0}^{n+1} Pred_{1i}\{\{\bigotimes_{j=i-1}^{0} Post_{1j}\}\} \wedge Pred_2\{\{\bigotimes_{j=n+1}^{0} Post_{1j}\}\} \wedge$$
$$\Big(\bigwedge_{i=0}^{m} Pred_{3i}\{\{\bigotimes_{j=i-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n+1}^{0} Post_{1j}\}\}\Big)$$

$$Post = \bigotimes_{j=m}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n+1}^{0} Post_{1j}$$

The right part of the disjunction, i.e.

$$\Big(\alpha = \tau \wedge \overline{\gamma} = \emptyset \wedge Pred = True \wedge Post = \mathrm{Id}(s) \wedge s = s'\Big)$$

is handled trivially by rule **WT1**.

($\Leftarrow$) We proceed by structural induction on the rules building the weak transition (as described in the original definition). The recurrence hypothesis being that the the original definition implies the characterisation (1),with the conditions stated at the bottom of the theorem. We consider the different rules:

- Case rule **WT1**. We have:

$$\dfrac{\emptyset, True, \mathrm{Id}(s)}{s \overset{\tau}{\Rightarrow} s} \in \mathcal{WT}$$

We can directly conclude by the right part of the disjunction the following:

$$\frac{\emptyset,\, \mathit{True}, \mathrm{Id}(s)}{s \stackrel{\tau}{\Rightarrow} s} \in \mathcal{WT} \Rightarrow \Big(\alpha = \tau \wedge \overline{\gamma} = \emptyset \wedge \mathit{Pred} = \mathit{True} \wedge \mathit{Post} = \mathrm{Id}(s) \wedge s = s'\Big)$$

- Case rule **WT2**. We have:

$$\frac{\overline{\gamma}, \mathit{Pred}, \mathit{Post}}{s \stackrel{\alpha}{\Rightarrow} s'} \in \mathcal{WT} \Rightarrow \frac{\overline{\beta}, \mathit{Pred}, \mathit{Post}}{s \stackrel{\alpha}{\rightarrow} s'} \in \mathcal{T}$$

where $\overline{\gamma} = (\overline{\beta})^{\nabla}$.

These two cases above prove the implication with $n = -1$ and $m = -1$.

- Case rule **WT3**. We have:

$$\frac{\dfrac{\overline{\gamma_1}, \mathit{Pred}_1, \mathit{Post}_1}{s \stackrel{\tau}{\Rightarrow} s'} \in \mathcal{WT} \qquad \dfrac{\overline{\gamma_2}, \mathit{Pred}_2, \mathit{Post}_2}{s' \stackrel{\alpha}{\Rightarrow} s''} \in \mathcal{WT} \qquad \dfrac{\overline{\gamma_3}, \mathit{Pred}_3, \mathit{Post}_3}{s'' \stackrel{\tau}{\Rightarrow} s'''} \in \mathcal{WT} \qquad \mathit{Pred} = \mathit{Pred}_1 \wedge \mathit{Pred}_2\{\!\{\mathit{Post}_1\}\!\} \wedge \mathit{Pred}_3\{\!\{\mathit{Post}_2 \otimes \mathit{Post}_1\}\!\} \qquad \overline{\gamma} = \overline{\gamma_1} \mathbin{\dot\cup} \overline{\gamma_2}\{\!\{\mathit{Post}_1\}\!\} \mathbin{\dot\cup} \overline{\gamma_3}\{\!\{\mathit{Post}_2 \otimes \mathit{Post}_1\}\!\} \qquad \alpha' = \alpha\{\!\{\mathit{Post}_1\}\!\}}{\dfrac{\overline{\gamma}, \mathit{Pred}, \mathit{Post}_3 \otimes \mathit{Post}_2 \otimes \mathit{Post}_1}{s \stackrel{\alpha'}{\Rightarrow} s'''} \in \mathcal{WT}}$$

(1) By induction hypothesis this means each tau weak open transition can be written as a series of $n_1$ tau open transitions such $n_1 = n + m + 3$, hence by simplification we have (strictly speaking, by induction we might also have the case $\alpha = \tau \wedge \overline{\gamma} = \emptyset \wedge \ldots$ but in this case, rule **WT1** allows us to obtain a similar reduction with $n_1 = 1$):

$$\frac{\overline{\gamma_1}, \mathit{Pred}_1, \mathit{Post}_1}{s \stackrel{\tau}{\Rightarrow} s'} \in \mathcal{WT} \Rightarrow \forall i \in [0..n_1]. \frac{\overline{\beta_{1i}}, \mathit{Pred}_{1i}, \mathit{Post}_{1i}}{s_{1i} \stackrel{\tau}{\rightarrow} s_{1(i+1)}} \in \mathcal{T}$$

where

$$s = s_{10} \wedge s_{1(n_1+1)} = s', \quad \overline{\gamma_1} = \dot{\bigcup_{i=0}^{n_1}} (\overline{\beta_{1i}}\{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}_{1j} \}\!\})^{\nabla}$$

$$\mathit{Pred}_1 = \bigwedge_{i=0}^{n_1} (\mathit{Pred}_{1i}\{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}_{ij} \}\!\}), \quad \mathit{Post}_1 = \bigotimes_{i=n_1}^{0} \{\!\{\mathit{Post}_{1i}\}\!\}$$

(2) Similarly, a series of $m_1$ open transitions such that $m_1 = n + m + 3$ can be simplified as follows:

$$\frac{\overline{\gamma_3}, \mathit{Pred}_3, \mathit{Post}_3}{s'' \stackrel{\tau}{\Rightarrow} s'''} \in \mathcal{WT} \Rightarrow \forall i \in [0..m_1]. \frac{\overline{\beta_{3i}}, \mathit{Pred}_{3i}, \mathit{Post}_{3i}}{s_{3i} \stackrel{\tau}{\rightarrow} s_{3(i+1)}} \in \mathcal{T}$$

where

$$s'' = s_{30} \quad \wedge \quad s_{3(m_1+1)} = s''' \quad \wedge \quad \overline{\gamma_3} = \dot{\bigcup_{i=0}^{m_1}} (\overline{\beta_{3i}}\{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}_{3j} \}\!\})^{\nabla} \quad \wedge$$

$$\mathit{Pred}_3 = \bigwedge_{i=0}^{m_1} (\mathit{Pred}_{3i}\{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}_{3j} \}\!\}) \quad \wedge \quad \mathit{Post}_3 = \bigotimes_{i=m_1}^{0} \{\!\{\mathit{Post}_{3i}\}\!\}$$

(3) Concerning the middle reduction, by induction hypothesis there exists a set of open transitions in $\mathcal{T}$ such that:

$$\frac{\overline{\gamma_2}, \mathit{Pred}_2, \mathit{Post}_2}{s' \overset{\alpha'}{\Rightarrow} s''} \in \mathcal{WT} \Rightarrow \Big(\forall i \in [0..n_2]. \frac{\overline{\beta_{2i}}, \mathit{Pred}_{2i}, \mathit{Post}_{2i}}{s_{2i} \xrightarrow{\tau} s_{2(i+1)}} \in \mathcal{T} \land$$

$$\frac{\overline{\beta'}, \mathit{Pred}', \mathit{Post}'}{s_2 \xrightarrow{\alpha''} s'_2} \in \mathcal{T} \land \forall i \in [0..m_2]. \frac{\overline{\beta'_{2i}}, \mathit{Pred}'_{2i}, \mathit{Post}'_{2i}}{s'_{2i} \xrightarrow{\tau} s'_{2(i+1)}} \in \mathcal{T}\Big)$$

where

$$s' = s_{20} \land s_{2(n_2+1)} = s_2 \land s'_2 = s'_{20} \land s'_{2(m_2+1)} = s''$$

$$\alpha'' = \alpha' \{\!\{ \bigotimes_{j=n_2}^{0} \mathit{Post}_{2j} \}\!\}$$

$$\overline{\gamma_2} = \biguplus_{i=0}^{n_2} (\overline{\beta_{2i}} \{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}_{2j} \}\!\})^{\nabla} \uplus (\overline{\beta'} \{\!\{ \bigotimes_{j=n_2}^{0} \mathit{Post}_{2j} \}\!\})^{\nabla} \uplus$$

$$\biguplus_{i=0}^{m_2} (\overline{\beta'_{2i}} \{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}'_{2j} \otimes \mathit{Post}' \otimes \bigotimes_{j=n_2}^{0} \mathit{Post}_{2j} \}\!\})^{\nabla}$$

$$\mathit{Pred}_2 = \bigwedge_{i=0}^{n_2} \mathit{Pred}_{2i} \{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}_{2j} \}\!\} \land \mathit{Pred}' \{\!\{ \bigotimes_{j=n_2}^{0} \mathit{Post}_{2j} \}\!\} \land$$

$$\Big( \bigwedge_{i=0}^{m_2} \mathit{Pred}'_{2i} \{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}'_{2j} \otimes \mathit{Post}' \otimes \bigotimes_{j=n_2}^{0} \mathit{Post}_{2j} \}\!\} \Big)$$

$$\mathit{Post}_2 = \bigotimes_{j=m_2}^{0} \mathit{Post}'_{2j} \otimes \mathit{Post}' \otimes \bigotimes_{j=n_2}^{0} \mathit{Post}_{2j}$$

Therefore, we can deduce that we have:

$$\frac{\overline{\gamma}, \mathit{Pred}, \mathit{Post}}{s \overset{\alpha}{\Rightarrow} s'} \in \mathcal{WT} \Rightarrow \Big(\forall i \in [0..(n_1+n_2)]. \frac{\overline{\beta_{4i}}, \mathit{Pred}_{4i}, \mathit{Post}_{4i}}{s_{4i} \xrightarrow{\tau} s_{4(i+1)}} \in \mathcal{T} \land$$

$$\frac{\overline{\beta'}, \mathit{Pred}', \mathit{Post}'}{s_2 \xrightarrow{\alpha''} s'_2} \in \mathcal{T} \land \forall i \in [0..(m_1+m_2)]. \frac{\overline{\beta_{5i}}, \mathit{Pred}_{5i}, \mathit{Post}_{5i}}{s_{5i} \xrightarrow{\tau} s_{5(i+1)}} \in \mathcal{T}\Big)$$

such that

$$s_{4i} = \begin{cases} s_{1i} & \text{if } i < n_1 \\ s_{2i-n_1} & \text{if } i \geq n_1 \end{cases} \qquad s_{5i} = \begin{cases} s_{3i} & \text{if } i < m_1 \\ s'_{2i-m_1} & \text{if } i \geq m_1 \end{cases}$$

and similarly for $\mathit{Pred}_{4i}$, $\mathit{Pred}_{5i}$, $\mathit{Post}_{4i}$, and $\mathit{Post}_{5i}$.

Also, we have the following assertions:

$$s = s_{40} \;\wedge\; s_{4(n_1+n_2+1)} = s_2 \;\wedge\; s'_2 = s_{50} \;\wedge\; s_{5(m_1+m_2+1)} = s'$$

$$\alpha'' = \alpha\{\!\{ \bigotimes_{j=n_1+n_2}^{0} Post_{4j} \}\!\}$$

$$\overline{\gamma} = \biguplus_{i=0}^{n_1+n_2} (\overline{\beta_{4i}}\{\!\{ \bigotimes_{j=i-1}^{0} Post_{4j} \}\!\})^{\nabla} \uplus (\overline{\beta'}\{\!\{ \bigotimes_{j=n_1+n_2}^{0} Post_{4j} \}\!\})^{\nabla} \uplus$$
$$\biguplus_{i=0}^{m_1+m_2} (\overline{\beta_{5i}}\{\!\{ \bigotimes_{j=i-1}^{0} Post'_{5i} \otimes Post' \otimes \bigotimes_{j=n_1+n_2}^{0} Post_{4i} \}\!\})^{\nabla}$$

$$Pred = \bigwedge_{i=0}^{n_1+n_2} Pred_{4i}\{\!\{ \bigotimes_{j=i-1}^{0} Post_{4j} \}\!\} \wedge Pred'\{\!\{ \bigotimes_{j=n_1+n_2}^{0} Post_{4j} \}\!\} \wedge$$
$$\bigwedge_{i=0}^{m_1+m_2} Pred_{5i}\{\!\{ \bigotimes_{j=i-1}^{0} Post_{5j} \otimes Post' \otimes \bigotimes_{j=n_1+n_2}^{0} Post_{4j} \}\!\}$$

$$Post = \bigotimes_{j=m_1+m_2}^{0} Post_{5j} \otimes Post' \otimes \bigotimes_{j=n_1+n_2}^{0} Post_{4j}$$

This concludes the inductive step, showing that the decomposition expressed by the $\Leftarrow$ direction of the lemma is always possible with the right side conditions. $\square$

**Lemma 5** Alternative definition of weak bisimulation**.** *The definition of weak bisimulation given in Definition 14 is equivalent to the following one:*

*Let* $A_1 = \langle\!\langle J, \mathcal{S}_1, s_0, V_1, \mathcal{T}_1 \rangle\!\rangle$ *and* $A_2 = \langle\!\langle J, \mathcal{S}_2, t_0, V_2, \mathcal{T}_2 \rangle\!\rangle$ *be open automata;* $\langle\!\langle J, \mathcal{S}_1, s_0, V_1, \mathcal{WT}_1 \rangle\!\rangle$ *and* $\langle\!\langle J, \mathcal{S}_2, t_0, V_2, \mathcal{WT}_2 \rangle\!\rangle$ *be the weak open automaton derived from* $A_1$ *and* $A_2$ *respectively. For any states* $s \in \mathcal{S}_1$ *and* $t \in \mathcal{S}_2$ *such that* $(s, t | Pred_{s,t}) \in \mathcal{R}$*, we have:*

- *For any open transition WOT in* $\mathcal{WT}_1$*:*

$$\frac{\gamma_j^{j\in J'}, Pred_{OT}, Post_{OT}}{s \overset{\alpha}{\Rightarrow} s'}$$

*there exist weak open transitions* $WOT_x^{x\in X} \subseteq \mathcal{WT}_2$*:*

$$\frac{\gamma_{jx}^{j\in J_x}, Pred_{OT_x}, Post_{OT_x}}{t \overset{\alpha_x}{\Longrightarrow} t_x}$$

*such that* $\forall x, J' = J_x, (s', t_x | Pred_{s',x}) \in \mathcal{R}$*; and*

$$Pred_{s,t} \wedge Pred_{OT} \Longrightarrow \bigvee_{x\in X} (\forall j \in J_x. \gamma_j = \gamma_{jx} \wedge Pred_{OT_x} \wedge \alpha = \alpha_x \wedge Pred_{s',x}\{\!\{Post_{OT} \uplus Post_{OT_x}\}\!\})$$

- *and symmetrically any open transition from WOT in* $\mathcal{WT}_2$ *can be covered by a set of weak transitions from s in* $\mathcal{WT}_1$*.*

*Proof.* Note that Definition 14 is a particular case of the definition above, thus we only need to prove one direction of the equivalence between the two definitions, namely:
($\Rightarrow$) We prove that Definition 14 implies the definition above. In other words, suppose that

$Pred_{s,t} \in \mathcal{R}$ and suppose that the following statement holds:

$$\dfrac{\gamma_j^{j\in J'}, Pred_{OT}, Post_{OT}}{s \overset{\alpha}{\Rightarrow} s'} \in \mathcal{WT}_1$$

Moreover, by using Lemma 4 we know that:

$$\dfrac{\gamma_j^{j\in J'}, Pred_{OT}, Post_{OT}}{s \overset{\alpha}{\Rightarrow} s'} \in \mathcal{WT}_1 \Rightarrow \Big(\forall i \in [0..n].\dfrac{\beta_{1ij}^{j\in J'_1}, Pred_{1i}, Post_{1i}}{s_{1i} \xrightarrow{\tau} s_{1(i+1)}} \in \mathcal{T}_1 \wedge$$

$$\dfrac{\beta_{2j}^{j\in J'_2}, Pred_2, Post_2}{s_{20} \xrightarrow{\alpha'} s_{21}} \in \mathcal{T}_1 \wedge \forall i \in [0..m].\dfrac{\beta_{3ij}^{j\in J'_3}, Pred_{3i}, Post_{3i}}{s_{3i} \xrightarrow{\tau} s_{3(i+1)}} \in \mathcal{T}_1\Big)$$

where

$$s = s_{10} \wedge s_{1(n+1)} = s_{20} \wedge s_{21} = s_{30} \wedge s_{3(m+1)} = s'$$

$$\alpha = \alpha'\{\!\{\bigotimes_{j=n}^{0} Post_{1j}\}\!\}$$

$$\gamma_j^{j\in J'} = \biguplus_{i=0}^{n}(\overline{\beta_{1i}}\{\!\{\bigotimes_{j=i-1}^{0} Post_{1j}\}\!\})^{\nabla} \uplus (\overline{\beta_2}\{\!\{\bigotimes_{j=n}^{0} Post_{1j}\}\!\})^{\nabla} \uplus$$
$$\biguplus_{i=0}^{m}(\overline{\beta_{3i}}\{\!\{\bigotimes_{j=i-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n}^{0} Post_{1j}\}\!\})^{\nabla}$$

$$Pred_{OT} = \bigwedge_{i=0}^{n} Pred_{1i}\{\!\{\bigotimes_{j=i-1}^{0} Post_{1j}\}\!\} \wedge Pred_2\{\!\{\bigotimes_{j=n}^{0} Post_{1j}\}\!\} \wedge$$
$$\Big(\bigwedge_{i=0}^{m} Pred_{3i}\{\!\{\bigotimes_{j=i-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n}^{0} Post_{1j}\}\!\}\Big)$$

$$Post_{OT} = \bigotimes_{j=m}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=n}^{0} Post_{1j}$$

For the sake of simplicity, we prove the rule in the restricted case where $n$ and $m$ are equal to 0, hence a single tau open transition will be considered on each side of the potentially visible one. The proof may be easily generalized to the multiple tau open transitions by using the same reasoning and **WT3** rule. Consider each open transition separately:

(1) For the first open transition in $\mathcal{T}_1$:

$$\dfrac{\beta_{1j}^{j\in J'_1}, Pred_1, Post_1}{s_{10} \xrightarrow{\tau} s_{11}}$$

by hypothesis we have $(s, t|Pred_{s,t}) \in \mathcal{R}$ and $s = s_{10}$. Thus, by Definition 14 we can deduce there exist weak open transitions $WOT_a^{a\in A} \subseteq \mathcal{WT}_2$:

$$\dfrac{\gamma_{ja}^{j\in J_a}, Pred_{OT_a}, Post_{OT_a}}{t \overset{\alpha_{1a}}{\Longrightarrow} u_a}$$

such that $\forall a, J_a = \{j \in J_1' | \beta_{1j} \neq \tau\}, (s_{11}, u_a | Pred_{s_{11},a}) \in \mathcal{R}$ and
$Pred_{s,t} \wedge Pred_1 \implies$
$\bigvee_{a \in A} \Big(\forall j \in J_a.(\beta_{1j})^{\nabla} = \gamma_{ja} \wedge Pred_{OT_a} \wedge \alpha_{1a} = \tau \wedge Pred_{s_{11},a}\{\!\{Post_1 \uplus Post_{OT_a}\}\!\}\Big)$

Note that, because $\mathbb{E} \cap \mathbb{A} = \emptyset$ (actions and expressions are disjoint) and $\alpha_{1a} = \tau$ we have directly ($\alpha_{1a}$ cannot be a variable, and cannot contain expressions/variables because $\tau$ has no parameter):

$$\frac{\gamma_{ja}^{j \in J_a}, Pred_{OT_a}, Post_{OT_a}}{t \overset{\tau}{\Rightarrow} u_a}$$

(2) Concerning the middle open transition in $\mathcal{T}_1$:

$$\frac{\beta_{2j}^{j \in J_2'}, Pred_2, Post_2}{s_{20} \xrightarrow{\alpha'} s_{21}}$$

we have $(s_{11}, u_a | Pred_{s_{11},a}) \in \mathcal{R}$ and $s_{11} = s_{20}$. Again by Definition 14 we can deduce there exist weak open transitions $WOT_b^{b \in B} \subseteq \mathcal{WT}_2$:

$$\frac{\gamma_{jb}^{j \in J_b}, Pred_{OT_b}, Post_{OT_b}}{u_a \overset{\alpha_{2b}}{\Longrightarrow} v_b}$$

such that $\forall b, J_b = \{j \in J_2' | \beta_{2j} \neq \tau\}, (s_{21}, v_b | Pred_{s_{21},b}) \in \mathcal{R}$;
$Pred_{s_{11},a} \wedge Pred_2 \implies$
$\bigvee_{b \in B} \Big(\forall j \in J_b.(\beta_{2j})^{\nabla} = \gamma_{jb} \wedge Pred_{OT_b} \wedge \alpha' = \alpha_{2b} \wedge Pred_{s_{21},b}\{\!\{Post_2 \uplus Post_{OT_b}\}\!\}\Big)$

(3) Similarly to the case 1, we consider the third open transition in $\mathcal{T}_1$:

$$\frac{\beta_{3j}^{j \in J_3'}, Pred_3, Post_3}{s_{30} \xrightarrow{\tau} s_{31}} \in \mathcal{T}$$

From previous case, we have $(s_{21}, v_b | Pred_{s_{21},b}) \in \mathcal{R}$, and we have $s_{21} = s_{30}$. Then, by Definition 14 there exist weak open transition $WOT_a^{c \in C} \subseteq \mathcal{WT}_2$:

$$\frac{\gamma_{jc}^{j \in J_c}, Pred_{OT_c}, Post_{OT_c}}{v_b \overset{\tau}{\Rightarrow} w_c}$$

such that $\forall c, J_c = \{j \in J_3' | \beta_{3j} \neq \tau\}, (s_{31}, w_c | Pred_{s_{31},c}) \in \mathcal{R}$ and
$Pred_{s_{21},b} \wedge Pred_3 \implies \bigvee_{c \in C} \Big(\forall j \in J_c.(\beta_{3j})^{\nabla} = \gamma_{jc} \wedge Pred_{OT_c} \wedge Pred_{s_{31},c}\{\!\{Post_3 \uplus Post_{OT_c}\}\!\}\Big)$

Based on cases described above by applying **WT3** rule on the resulting *WOT*s we have:

$$\dfrac{\dfrac{\gamma_{ja}^{j\in J_a}, Pred_{OT_x}, Post_{OT_a}}{t \overset{\tau}{\Rightarrow} u_a} \qquad \dfrac{\gamma_{jb}^{j\in J_b}, Pred_{OT_y}, Post_{OT_b}}{u_a \overset{\alpha_2}{\Longrightarrow} v_b} \qquad \dfrac{\gamma_{jc}^{j\in J_c}, Pred_{OT_c}, Post_{OT_c}}{v_b \overset{\tau}{\Rightarrow} w_c} \quad \begin{array}{c} \overline{\gamma'} = \gamma_{ja}^{j\in J_a} \uplus \gamma_{jb}^{j\in J_b}\{\!\{Post_{OT_a}\}\!\} \uplus \gamma_{jc}^{j\in J_c}\{\!\{Post_{OT_b} \otimes Post_{OT_a}\}\!\} \\ Pred = Pred_{OT_a} \wedge Pred_{OT_b}\{\!\{Post_{OT_a}\}\!\} \wedge Pred_{OT_c}\{\!\{Post_{OT_b} \otimes Post_{OT_a}\}\!\} \\ Post = Post_{OT_c} \otimes Post_{OT_b} \otimes Post_{OT_a} \quad \alpha'' = \alpha_2\{\!\{Post_{OT_a}\}\!\} \end{array}}{\dfrac{\overline{\gamma}, Pred, Post}{t \overset{\alpha''}{\Longrightarrow} w_c}}$$

It remains to be proven that the following statement holds:

$Pred_{s,t} \wedge Pred \implies \bigvee_{x\in X} (\forall j \in J.\gamma'_j = \gamma_j \wedge Pred \wedge \alpha = \alpha'' \wedge Pred_{s',x}\{\!\{Post_{OT} \uplus Post\}\!\})$

We have:

$Pred_{OT} = Pred_1 \wedge Pred_2\{\!\{Post_1\}\!\} \wedge Pred_3\{\!\{Post_2 \otimes Post_1\}\!\}$

$Post_{OT} = Post_3 \otimes Post_2 \otimes Post_1$

Moreover, we have the following statement:

$Pred_{s,t} \wedge Pred_1 \implies \bigvee_{a\in A} \Big(\forall j \in J_a.(\beta_{1j})^{\nabla} = \gamma_{ja} \wedge Pred_{OT_a} \wedge Pred_{s_{11},a}\{\!\{Post_1 \uplus Post_{OT_a}\}\!\}\Big)$

With the conjunction of the predicate $Pred_2\{\!\{Post_1\}\!\}$ on both sides of the implication, we get:

$$(Pred_{s,t} \wedge Pred_1) \wedge Pred_2\{\!\{Post_1\}\!\} \implies$$
$$\bigvee_{a\in A}(\forall j \in J_a.(\beta_{1j})^{\nabla} = \gamma_{ja} \wedge Pred_{OT_a} \wedge Pred_{s_{11},a}\{\!\{Post_1 \uplus Post_{OT_a}\}\!\}) \wedge Pred_2\{\!\{Post_1 \uplus Post_{OT_a}\}\!\}$$

Note that on the right side of the implication we added the substitution of $Post_{OT_x}$ without affecting the validity of the statement, because the domain of the substitution $Post_{OT_x}$ is disjoint from the others. Hence a little rewriting gives:

$$(Pred_{s,t} \wedge Pred_1) \wedge Pred_2\{\!\{Post_1\}\!\} \implies$$
$$\bigvee_{a\in A}\Big(\forall j \in J_a.(\beta_{1j})^{\nabla} = \gamma_{ja} \wedge Pred_{OT_a} \wedge (Pred_{s_{11},a} \wedge Pred_2)\{\!\{Post_1 \uplus Post_{OT_a}\}\!\}\Big)$$

By replacing the inner predicate $(Pred_{s_{11},u_a} \wedge Pred_2)$ by the conclusion of the statement given in case 2, the formula becomes:

$$(Pred_{s,t} \wedge Pred_1) \wedge Pred_2\{\!\{Post_1\}\!\} \implies$$
$$\bigvee_{a\in A}(\forall j \in J_a.(\beta_{1j})^{\nabla} = \gamma_{ja} \wedge Pred_{OT_a} \wedge (\bigvee_{b\in B} (\forall j \in J_b.(\beta_{2j})^{\nabla} = \gamma_{jb} \wedge Pred_{OT_b} \wedge \alpha' = \alpha_{2b} \wedge$$
$$Pred_{s_{21},b}\{\!\{Post_2 \uplus Post_{OT_b}\}\!\})\{\!\{Post_1 \uplus Post_{OT_a}\}\!\})$$

This can be rewritten into:

$$
\begin{aligned}
&(Pred_{s,t} \wedge Pred_1) \wedge Pred_2\{\!\{Post_1\}\!\} \implies \\
&\bigvee_{a\in A}\bigvee_{b\in B}(\forall j \in J_a.(\beta_{1j})^{\nabla}=\gamma_{ja} \wedge \forall j \in J_b.(\beta_{2j})^{\nabla}=\gamma_{jb}\{\!\{Post_1 \uplus Post_{OT_a}\}\!\} \wedge Pred_{OT_a} \wedge \\
&\qquad Pred_{OT_b}\{\!\{Post_1 \uplus Post_{OT_a}\}\!\} \wedge (\alpha'=\alpha_{2b})\{\!\{Post_1 \uplus Post_{OT_a}\}\!\} \wedge \\
&\qquad\qquad Pred_{s_{21},b}\{\!\{Post_2 \otimes Post_1 \uplus Post_{OT_b} \otimes Post_{OT_a}\}\!\})
\end{aligned}
$$

Since $Post_1$ does not act on $\gamma_{jb}$, nor on $Pred_{OT_b}$ and $\alpha_2$. As well $Post_{OT_a}$ does not act on $\alpha'$, nor on $\beta_{2j}$ the formula can be simplified as follows:

$$
\begin{aligned}
&(Pred_{s,t} \wedge Pred_1) \wedge Pred_2\{\!\{Post_1\}\!\} \implies \\
&\bigvee_{a\in A}\bigvee_{b\in B}(\forall j \in J_a.(\beta_{1j})^{\nabla}=\gamma_{ja} \wedge \forall j \in J_b.(\beta_{2j})^{\nabla}\{\!\{Post_1\}\!\} = \gamma_{jb}\{\!\{Post_{OT_a}\}\!\} \wedge Pred_{OT_a} \wedge \\
&\qquad Pred_{OT_b}\{\!\{Post_{OT_a}\}\!\} \wedge \alpha'\{\!\{Post_1\}\!\} = \alpha_{2b}\{\!\{Post_{OT_a}\}\!\} \wedge \\
&\qquad\qquad Pred_{s_{21},b}\{\!\{Post_2 \otimes Post_1 \uplus Post_{OT_b} \otimes Post_{OT_a}\}\!\})
\end{aligned}
$$

Finally, the conjunction with the term $Pred_3\{\!\{Post_2 \otimes Post_1\}\!\}$ of the both sides of the implication and rewriting, we get:

$$
\begin{aligned}
&(Pred_{s,t} \wedge Pred_1) \wedge Pred_2\{\!\{Post_1\}\!\} \wedge Pred_3\{\!\{Post_2 \otimes Post_1\}\!\} \implies \\
&\bigvee_{a\in A}\bigvee_{b\in B}(\forall j \in J_a.(\beta^1_j)^{\nabla}=\gamma_{ja} \wedge \forall j \in J_b.(\beta_{2j})^{\nabla}\{\!\{Post_1\}\!\}=\gamma_{jb}\{\!\{Post_{OT_a}\}\!\} \wedge Pred_{OT_a} \wedge \\
&\qquad Pred_{OT_b}\{\!\{Post_{OT_a}\}\!\} \wedge \alpha'\{\!\{Post_1\}\!\} = \alpha_{2b}\{\!\{Post_{OT_a}\}\!\} \wedge (Pred_{s_{21},v_b} \wedge Pred_3) \\
&\qquad\qquad \{\!\{Post_2 \otimes Post_1 \uplus Post_{OT_b} \otimes Post_{OT_a}\}\!\})
\end{aligned}
$$

Again note that because of the domain of the substitution is independent from some predicates and expressions, we removed $Post_1$ and we added the term $Post_{OT_b} \otimes Post_{OT_a}$ in the substitution of the right side of the implication.
Finally, by replacing the predicate $(Pred_{s_{21},b} \wedge Pred_3)$ by the conclusion of the implication given in case 3, we get:

$$
\begin{aligned}
&Pred_{s,t} \wedge \underbrace{Pred_1 \wedge Pred_2\{\!\{Post_1\}\!\} \wedge Pred_3\{\!\{Post_2 \otimes Post_1\}\!\}}_{Pred_{OT}} \implies \\
&\bigvee_{a\in A}\bigvee_{b\in B}\bigvee_{c\in C}(\forall j \in J_a.(\beta_{1j})^{\nabla}=\gamma_{ja} \wedge \forall j \in J_b.(\beta_{2j}\{\!\{Post_{OT_a}\}\!\})^{\nabla}=\gamma_{jb}\{\!\{Post_{OT_a}\}\!\} \wedge \\
&\qquad \forall j \in J_c.(\beta_{3j}\{\!\{Post_2 \otimes Post_1\}\!\})^{\nabla}=\gamma_{jc}\{\!\{Post_{OT_b} \otimes Post_{OT_a}\}\!\} \wedge \\
&\underbrace{Pred_{OT_a} \wedge Pred_{OT_b}\{\!\{Post_{OT_a}\}\!\} \wedge Pred_{OT_c}\{\!\{Post_{OT_b} \otimes Post_{OT_a}\}\!\}}_{Pred} \wedge \underbrace{\alpha'\{\!\{Post_1\}\!\}}_{\alpha} = \underbrace{\alpha_{2b}\{\!\{Post_{OT_a}\}\!\}}_{\alpha''} \\
&\qquad \wedge Pred_{s_{31},c}\{\!\{\underbrace{Post_3 \otimes Post_2 \otimes Post_1}_{Post_{OT}} \uplus \underbrace{Post_{OT_c} \otimes Post_{OT_b} \otimes Post_{OT_a}}_{Post}\}\!\})
\end{aligned}
$$

The three for all statements (on $J_a$, $J_b$ and $J_c$) can be concatenated using $\uplus$, the list union lifted to indexed sets (if $\gamma = \gamma'$ and $\gamma'' = \gamma'''$ then $\gamma\uplus\gamma'' = \gamma'\uplus\gamma'''$).

$$\forall j \in J_a \uplus J_b \uplus J_c.(\beta_{1j})^\nabla \uplus (\beta_{2j}\{\{Post_{OT_a}\}\})^\nabla \uplus (\beta_{3j}\{\{Post_2 \otimes Post_1\}\})^\nabla = \gamma_{ja} \uplus \gamma_{jb}\{\{Post_{OT_a}\}\} \uplus \gamma_{jc}\{\{Post_{OT_b} \otimes Post_{OT_a}\}\}$$

We have $s_{31} = s'$, so can rewrite the formula:
$Pred_{s,t} \wedge Pred_{OT} \implies \bigvee_{a\in A} \bigvee_{b\in B} \bigvee_{c\in C} (\forall j \in J.\gamma'_j = \gamma_j \wedge Pred \wedge \alpha = \alpha'' \wedge Pred_{s',c}\{\{Post_{OT} \uplus Post\}\}$

All the combinations of elements in $A$, $B$, and $C$ provides a set $X$ of weak open transitions (each combination of one transition in $A$, one in $B$, and one in $C$ provides one weak open transition in the set $X$, i.e. each $x \in X$ corresponds to a triple $(a, b, c) \in A \times B \times C$); this defines a set of weak open transitions indexed over $X$; each such open transition leads to a $w_c$ that we call $t_x$. This re-indexing allows us to conclude:
$Pred_{s,t} \wedge Pred_{OT} \implies \bigvee_{x\in X} (\forall j \in J.\gamma'_j = \gamma_j \wedge Pred \wedge \alpha = \alpha'' \wedge Pred_{s',x}\{\{Post_{OT} \uplus Post\}\}$

□

**Theorem 6**. *Weak FH-Bisimulation is an equivalence.* Suppose $\mathcal{R}$ is a weak FH-bisimulation. Then $\mathcal{R}$ is an equivalence, that is, $\mathcal{R}$ is reflexive, symmetric and transitive.

With the above lemma, we can use the same technique as for Theorem 1 to prove that a weak FH-bisimulation is an equivalence. Indeed, we essentially use the same proof-scheme the main difference concerns $\beta$ and $\gamma$. Indeed, while the schema of the proof of transitivity was not directly applicable on the definition of weak bisimulation, Lemma 5 provides a characterisation of weak bisimulation similar to the definition of strong bisimulation, and thus the same proof scheme is directly applicable.

B.2. **Composition properties.** This section gives decomposition/composition lemmas and their proofs, these are the equivalent of the composition lemmas for open transitions, but applied to weak open automata.

**Lemma 6** Weak open transition decomposition**.** *Let* Leaves$(Q) = pLTS_l^{l\in L_Q}$*; suppose*[11]*:*

$$P[Q]_{j_0} \models \frac{\gamma_j^{j\in J}, Pred, Post}{\triangleleft s_i^{i\in L} \triangleright \overset{\alpha}{\Rightarrow} \triangleleft s_i'^{i\in L} \triangleright}$$

*with* $J \cap \text{Holes}(Q) \neq \emptyset$ *or* $\exists i \in L_Q.\, s_i \neq s'_i$*, i.e. $Q$ takes part in the reduction. Then there exist $n$, $Pred'$, $Post'$, and for all $p \in [0..n]$ there exist $\beta_p$, $\alpha_p$, $Pred_p$, $Post_p$ and a family $\gamma_{pj}^{j\in J_p}$ and for all $p \in [0..n+1]$ $s_{pi}^{i\in L_Q}$. s.t.:*

[11]Note that the hypotheses of the lemma imply that $Q$ is not a pLTS but a similar lemma can be proven for a pLTS $Q$

$$P \models \frac{\gamma_j^{j\in(J_p\setminus\mathrm{Holes}(Q))\cup\{j_0\}}, \mathit{Pred}', \mathit{Post}'}{\lhd s_i^{i\in L\setminus L_Q} \rhd \overset{\alpha}{\Rightarrow} \lhd s_i'^{\,i\in L\setminus L_Q} \rhd} \qquad \mathit{and} \qquad \gamma_{j_0} = [\beta_0..\beta_n]$$

$$\mathit{and\ for\ all}\ p \in [0..n]\ Q \models \frac{\gamma_{pj}^{j\in J_p}, \mathit{Pred}_p, \mathit{Post}_p}{\lhd s_{pi}^{i\in L_Q} \rhd \overset{\alpha_p}{\Longrightarrow} \lhd s_{(p+1)i}^{i\in L_Q} \rhd}$$

$$\mathit{such\ that} \qquad \bigcup_{p=0}^{n} J_p = J \cap \mathrm{Holes}(Q),\ \gamma_j^{j\in J\cap\mathrm{Holes}(Q)} = \bigcup_{p=0}^{n}{}^{\!\cdot}\, (\gamma_{pj}^{j\in J_p})\{\!\{ \bigotimes_{i=p-1}^{0} \mathit{Post}_i \}\!\},$$

$$\mathit{Pred} \iff \mathit{Pred}' \wedge \bigwedge_{p=0}^{n} (\alpha_p\{\!\{ \bigotimes_{i=p-1}^{0} \mathit{Post}_i \}\!\} = \beta_p \wedge \mathit{Pred}_p\{\!\{ \bigotimes_{i=p-1}^{0} \mathit{Post}_i \}\!\}),$$

$$\mathit{Post} = \mathit{Post}' \uplus \bigotimes_{p=n}^{0} \mathit{Post}_p,\ \mathit{and} \quad \forall i \in L_Q.\, s_{(n+1)i} = s_i' \wedge s_{0i} = s_i$$

*where for any p, $\mathit{Post}_p$ only acts upon variables vars(Q).*

*Proof.* Suppose that we have:

$$P[Q]_{j_0} \models \frac{\gamma_j^{j\in J}, \mathit{Pred}, \mathit{Post}}{\lhd s_l^{l\in L} \rhd \overset{\alpha}{\Rightarrow} \lhd s_l'^{\,l\in L} \rhd}$$

By Lemma 4 this implies the following:

$$\forall p\in [0..m_1]\ P[Q]_{j_0} \models \frac{\overline{\beta_{1p}}, \mathit{Pred}_{1p}, \mathit{Post}_{1p}}{\lhd s_{pl}^{l\in L} \rhd \xrightarrow{\tau} \lhd s_{(p+1)l}^{l\in L} \rhd}, \qquad P[Q]_{j_0} \models \frac{\overline{\beta_2}, \mathit{Pred}_2, \mathit{Post}_2}{\lhd t_l^{l\in L} \rhd \xrightarrow{\alpha'} \lhd t_l'^{\,l\in L} \rhd}$$

and $\forall p \in [0..m_2]\ P[Q]_{j_0} \models \frac{\overline{\beta_{3p}}, \mathit{Pred}_{3p}, \mathit{Post}_{3p}}{\lhd u_{pl}^{l\in L} \rhd \xrightarrow{\tau} \lhd u_{(p+1)l}^{l\in L} \rhd}$

where

$$\forall l \in L.\, s_l = s_{0l} \wedge s_{(m_1+1)l} = t_l \wedge t_l' = u_{0l} \wedge u_{(m_2+1)l} = s_l'$$

$$\alpha = \alpha'\{\!\{ \bigotimes_{j=m_1}^{0} \mathit{Post}_{1j} \}\!\}$$

$$\gamma_j^{j\in J} = \bigcup_{i=0}^{m_1}{}^{\!\cdot}\, (\overline{\beta_{1i}}\{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}_{1j} \}\!\})^{\nabla} \cup\!\!\cdot\ (\overline{\beta_2}\{\!\{ \bigotimes_{j=m_1}^{0} \mathit{Post}_{1j} \}\!\})^{\nabla} \cup\!\!\cdot \bigcup_{i=0}^{m_2}{}^{\!\cdot}\, (\overline{\beta_{3i}}\{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}_{3j} \otimes \mathit{Post}_2 \otimes \bigotimes_{j=m_1}^{0} \mathit{Post}_{1j} \}\!\})^{\nabla}$$

$$\mathit{Pred} = \bigwedge_{i=0}^{m_1} \mathit{Pred}_{1i}\{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}_{1j} \}\!\} \wedge \mathit{Pred}_2\{\!\{ \bigotimes_{j=m_1}^{0} \mathit{Post}_{1j} \}\!\} \wedge \bigwedge_{i=0}^{m_2} \mathit{Pred}_{3i}\{\!\{ \bigotimes_{j=i-1}^{0} \mathit{Post}_{3j} \otimes \mathit{Post}_2 \otimes \bigotimes_{j=m_1}^{0} \mathit{Post}_{1j} \}\!\}$$

$$\mathit{Post} = \bigotimes_{j=m_2}^{0} \mathit{Post}_{3j} \otimes \mathit{Post}_2 \otimes \bigotimes_{j=m_1}^{0} \mathit{Post}_{1j}$$

We can apply Lemma 1 on each $OT$:

(1) For each open transition $OT_p$ in the form $(\overline{\beta_{1p}} = \beta_{1pj}^{j\in J_{1p}})$:

$$P[Q]_{j_0} \models \frac{\beta_{1pj}^{j\in J_{1p}}, Pred_{1p}, Post_{1p}}{\triangleleft s_{pl}^{l\in L} \triangleright \xrightarrow{\tau} \triangleleft\, s_{(p+1)l}^{l\in L} \triangleright}$$

If $Q$ moves then we obtain by Lemma 1:
$P \models \dfrac{(\beta_{1pj})^{j\in(J_{1p}\setminus \mathrm{Holes}(Q))\cup\{j_0\}}, Pred'_{1p}, Post'_{1p}}{\triangleleft s_{pl}^{l\in L\setminus L_Q} \triangleright \xrightarrow{\tau} \triangleleft\, (s_{(p+1)l})^{l\in L\setminus L_Q} \triangleright}$ and $Q \models \dfrac{(\beta_{1pj})^{j\in J_{1p}\cap \mathrm{Holes}(Q)}, Pred''_{1p}, Post''_{1p}}{\triangleleft s_{pl}^{l\in L_Q} \triangleright \xrightarrow{\alpha_{1p}} \triangleleft\, s_{(p+1)l}^{l\in L_Q} \triangleright}$
such that
$Pred_{1p} \iff Pred'_{1p} \wedge Pred''_{1p} \wedge \alpha_{1p} = \beta_{1pj_0}$, $Post_{1p} = Post'_{1p} \uplus Post''_{1p}$ where $Post''_{1p}$ is the restriction of $Post_{1p}$ over $vars(Q)$.
Else $Q$ does not move and we have:
$P \models \dfrac{(\beta_{1pj})^{j\in(J_{1p}\setminus \mathrm{Holes}(Q))}, Pred'_{1p}, Post'_{1p}}{\triangleleft (s_{pl})^{l\in L\setminus L_Q} \triangleright \xrightarrow{\tau} \triangleleft\, (s_{(p+1)l})^{l\in L\setminus L_Q} \triangleright}$ and $\triangleleft (s_p)_l^{l\in L_Q} \triangleright = \triangleleft (s_{(p+1)l})^{l\in L_Q} \triangleright$

(2) Similarly, we have similar open transitions on states $u_{pl}$ (for the final $\tau$ transitions).

(3) Finally, for the open transition in the form $(\overline{\beta_2} = \beta_{2j}^{j\in J_2})$:

$$P[Q]_{j_0} \models \frac{\beta_{2j}^{j\in J_2}, Pred_2, Post_2}{\triangleleft t_l^{l\in L_Q} \triangleright \xrightarrow{\alpha'} \triangleleft\, (t'_l)^{l\in L_Q} \triangleright}$$

If $Q$ moves then we obtain by Lemma 1:
$P \models \dfrac{(\beta_{2j})^{j\in(J_2\setminus \mathrm{Holes}(Q))\cup\{j_0\}}, Pred'_2, Post'_2}{\triangleleft t_l^{l\in L\setminus L_Q} \triangleright \xrightarrow{\alpha'} \triangleleft\, t_l'^{\,l\in L\setminus L_Q} \triangleright}$ and $Q \models \dfrac{(\beta_{2j})^{j\in J_2\cap \mathrm{Holes}(Q)}, Pred''_2, Post''_2}{\triangleleft t_l^{l\in L_Q} \triangleright \xrightarrow{\alpha_{20}} \triangleleft\, t_l'^{\,l\in L_Q} \triangleright}$
such that $Pred_2 \iff Pred'_2 \wedge Pred''_2 \wedge \alpha_2 = \beta_{2j_0}$, $Post_2 = Post'_2 \uplus Post''_2$ where $Post''_2$ is the restriction of $Post_2$ over variables $vars(Q)$.
Else $Q$ does not move and we have
$P \models \dfrac{(\beta_{2j})^{j\in(J_2\setminus \mathrm{Holes}(Q))\cup\{j_0\}}, Pred'_2, Post'_2}{\triangleleft t_l^{l\in L\setminus L_Q} \triangleright \xrightarrow{\alpha'} \triangleleft\, (t_l)'^{\,l\in L\setminus L_Q} \triangleright}$ and $\triangleleft\, t_l^{l\in L_Q} \triangleright = \triangleleft (t'_l)^{l\in L_Q} \triangleright$

By using Lemma 4, and denoting $J = \bigcup_{i=0}^{m_1} J_{1i} \cup J_2 \cup \bigcup_{i=0}^{m_2} J_{3i}$, we can conclude from cases (1), (2) and (3) that we have:

$$P \models \frac{(\gamma'_j)^{j\in(J\setminus \mathrm{Holes}(Q))\cup\{j_0\}}, Pred', Post'}{\triangleleft s_l^{l\in L\setminus L_Q} \triangleright \xRightarrow{\alpha''} \triangleleft\, s_l'^{\,l\in L\setminus L_Q} \triangleright}$$

where $\alpha'' = \alpha' \{\!\{ \bigotimes_{j=m_1}^{0} Post'_{1j} \}\!\}$

On the other hand, we have: $\alpha = \alpha' \{\!\{ \bigotimes_{j=m_1}^{0} Post'_{1j} \}\!\} \{\!\{ \bigotimes_{j=m_1}^{0} Post''_{1j} \}\!\} = \alpha'' \{\!\{ \bigotimes_{j=m_1}^{0} Post''_{1j} \}\!\}$.

As $\{\!\{ \bigotimes_{j=m_1}^{0} Post''_{1j} \}\!\}$ has no effect on variables of $P$ and thus on variables of $\alpha''$, so we have $\alpha = \alpha''$.

$$\forall l \in L.\, s_l = s_{0l} \wedge s_{(m_1+1)l} = t_l \wedge t'_l = u_{0l} \wedge u_{(m_2+1)l} = s'_l$$

$$(\gamma'_j)^{j\in(J\setminus\text{Holes}(Q))\cup\{j_0\}} = \dot{\bigcup_{i=0}^{m_1}}(\overline{\beta_{1i}}\{\!\{\bigotimes_{j=i-1}^{0} Post'_{1j}\}\!\})^{\nabla} \dot\cup (\overline{\beta_2}\{\!\{\bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\})^{\nabla} \dot\cup$$
$$\dot{\bigcup_{i=0}^{m_2}}(\overline{\beta_{3i}}\{\!\{\bigotimes_{j=i-1}^{0} Post'_{3j} \otimes Post'_2 \otimes \bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\})^{\nabla}$$

$$Pred' = \bigwedge_{i=0}^{m_1} Pred'_{1i}\{\!\{\bigotimes_{j=i-1}^{0} Post'_{1j}\}\!\} \wedge Pred'_2\{\!\{\bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\} \wedge$$
$$\bigwedge_{i=0}^{m_2} Pred'_{3i}\{\!\{\bigotimes_{j=i-1}^{0} Post'_{3j} \otimes Post'_2 \otimes \bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\}$$

$$Post' = \bigotimes_{j=m_2}^{0} Post'_{3j} \otimes Post'_2 \otimes \bigotimes_{j=m_1}^{0} Post'_{1j}$$

Note that for all $j \in J \setminus \text{Holes}(Q)$, $\gamma'_j = \gamma_j$ because for all $l$ $Post'_{1l}$ coincide with $Post_{1l}$ on the variables of $\beta_{1ij}$, and similarly for $Post'_2$ and $Post'_{3l}$.

We introduce the following predicate (we will need it for reasoning on the global predicate and will reason on it along the proof):

$$Pred_\beta = \bigwedge_{p=0}^{m_1} (\beta_{1pj_0} = \alpha_{1p})\{\!\{\bigotimes_{j=p-1}^{0} Post_{1j}\}\!\} \wedge (\beta_{2j_0} = \alpha_{20})\{\!\{\bigotimes_{j=m_1}^{0} Post_{1j}\}\!\} \wedge$$
$$\bigwedge_{p=0}^{m_2} (\beta_{3pj_0} = \alpha_{3p})\{\!\{\bigotimes_{j=p-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=m_1}^{0} Post_{1j}\}\!\}$$

Concerning $Q$, we reduce the sequence of OTs to a path for which it moves in all steps. In other words, if $Q$ does not move at step $q$, then we have $\triangleleft s_{ql}^{l\in L_Q}\triangleright = \triangleleft s_{(q+1)l}^{l\in L_Q}\triangleright$ , then we skip the state $\triangleleft s_{(q+1)l}^{l\in L_Q}\triangleright$, i.e. we rename all the following states $\triangleleft s_{pl}^{l\in L_Q}\triangleright$ where $p \geq q+1$ into $\triangleleft s_{(p-1)l}^{l\in L_Q}\triangleright$. Note that self-loops where $Q$ does an action but stays at the same state are not removed. We proceed in the same way for states named $u$. To simplify the proof, we suppose that in case 3, $Q$ moves, else transition 3 of $Q$ should be skipped and the last $s_{pl}$ are equal to the first $u_{0l}$. So we have:

$\forall p \in [0..n_1]\ Q \models \dfrac{(\beta'_{1pj})^{j\in J\cap\text{Holes}(Q)}, Pred''_{1p}, Post''_{1p}}{\triangleleft s_{pl}^{l\in L_Q}\triangleright \xrightarrow{\alpha_{1p}} \triangleleft s_{(p+1)l}^{l\in L_Q}\triangleright}$, $Q \models \dfrac{(\beta'_{2j})^{j\in J\cap\text{Holes}(Q)}, Pred''_2, Post''_2}{\triangleleft t_l^{l\in L_Q}\triangleright \xrightarrow{\alpha_{20}} \triangleleft t'^{\,l\in L_Q}_l\triangleright}$ and

$\forall p \in [0..n_2]\ Q \models \dfrac{(\beta'_{3pj})^{j\in J\cap\text{Holes}(Q)}, Pred''_{3p}, Post''_{3p}}{\triangleleft u_{pl}^{l\in L_Q}\triangleright \xrightarrow{\alpha_{3p}} \triangleleft u_{(p+1)l}^{l\in L_Q}\triangleright}$

such that $n_1 \leq m_1$ and $n_2 \leq m_2$.

By renaming all state names ($s$, $u$ and $t$) with the same state name $v$. We have:

$$\forall p \in [0..(n_1+n_2+2)]\ Q \models \frac{\beta_{pj}^{j\in J\cap\text{Holes}(Q)}, Pred''_p, Post''_p}{\triangleleft v_{pl}^{l\in L_Q}\triangleright \xrightarrow{\alpha'_p} \triangleleft v_{(p+1)l}^{l\in L_Q}\triangleright}$$

In this equation, and using case 1 above for all $k \in [0 \dots n_1]$ there is a $p \in [0..m_1]$ such that $\alpha_{1p} = \alpha'_k$ (following the re-indexing done in the removal of steps where $Q$ does not move), we know that $Pred_{1p}$ contains the predicate $(\alpha_{1p} = \beta_{1pj_0})$. Because $\beta_{1pj_0}$ only contains variables

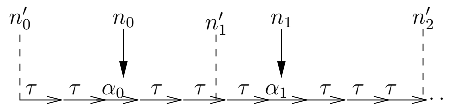



Figure 8: Composition of the subsequences

of $P$ and $\alpha'_k$ only variables of $Q$, we have:

$$(\alpha_{1p} = \beta_{1pj_0})\{\!\{\bigotimes_{j=p-1}^{0} Post_{1j}\}\!\} \iff \alpha_{1p}\{\!\{\bigotimes_{j=p-1}^{0} Post''_{1j}\}\!\} = \beta_{1pj_0}\{\!\{\bigotimes_{j=p-1}^{0} Post'_{1j}\}\!\}$$
$$\iff \alpha'_k\{\!\{\bigotimes_{j=k-1}^{0} Post''_{j}\}\!\} = \beta_{1pj_0}\{\!\{\bigotimes_{j=p-1}^{0} Post'_{1j}\}\!\}$$

We can obtain similar equations for $\alpha'_{n_1+1}$ related with $\beta_{2j_0}$ and the $\alpha'_k$ for $k \geq n_1+2$ related with $\beta_{3pj_0}$ for some $p$. Note that the substitutions are however more complex in the other cases. Overall we obtain (we skip here the details about the three cases 1, 2, and 3 above that all fall into the same equation because of the re-indexing we perform):

$$Pred_\beta \iff \gamma_{j_0} = ([\alpha'_p\{\!\{\bigotimes_{j=p-1}^{0} Post''_j\}\!\} | p \in [0..n_1+n_2+2]])^{\nabla} \tag{B.1}$$

Let us consider the sequence of $(n_1 + n_2 + 3)$ actions $\alpha'_p$ some of them may be non-observable (they are $\tau$ transitions). By considering the sequence of $\tau$ and non-$\tau$ actions we split the sequence of actions into $n+1$ sub-sequences, such that each subsequence is a sequence of actions containing only one observable action that will be named $\alpha_p$, and possibly many non-observable ($\tau$) ones.

We can decompose each of the $n+1$ sub-sequences in the following way (see Figure 8). For $k \in [0..n]$ the position of the $k^{\text{th}}$ visible action is $n_k$. For $l \in [1..n]$, $n'_l$ is any index between $n_{l-1}$ and $n_l$, additionally $n'_0 = 0$ and $n'_{n+1} = n_1 + n_2 + 3$. We obtain $n+1$ sub-sequences made of the following OTs, for all $k \in [0..n]$ :

$\forall p \in [n'_k..(n_k - 1)]\ Q \models \dfrac{\beta_{pj}^{j\in J\cap \text{Holes}(Q)}, Pred''_p, Post''_p}{\triangleleft v_{pl}^{l\in L_Q} \triangleright \xrightarrow{\tau} \triangleleft\, v_{(p+1)l}^{l\in L_Q} \triangleright}$, $\quad Q \models \dfrac{\beta_{n_kj}^{j\in J\cap \text{Holes}(Q)}, Pred''_{n_k}, Post''_{n_k}}{\triangleleft v_{n_kl}^{l\in L_Q} \triangleright \xrightarrow{\alpha'_{n_k}} \triangleleft\, v_{(n_k+1)l}^{l\in L_Q} \triangleright}$ and

$\forall p \in [(n_k+1)..(n'_{k+1}-1)]\ Q \models \dfrac{\beta_{pj}^{j\in J\cap \text{Holes}(Q)}, Pred''_p, Post''_p}{\triangleleft v_{pl}^{l\in L_Q} \triangleright \xrightarrow{\tau} \triangleleft\, v_{(p+1)l}^{l\in L_Q} \triangleright}$

Thereafter, by Lemma 4 we can deduce the following weak open transition:

$$Q \models \frac{(\gamma_{kj})^{j\in J\cap \text{Holes}(Q)}, Pred_k, Post_k}{\triangleleft v_{kl}^{l\in L_Q} \triangleright \xRightarrow{\alpha_k} \triangleleft\, (v'_{kl})^{l\in L_Q} \triangleright}$$

with:

$$\forall l \in L_Q . v_{kl} = v_{(n'_k)l} \wedge v'_{kl} = v_{(n'_k)l}$$

$$\alpha_k = \alpha'_{n_k} \{\!\{ \bigotimes_{j=n_k-1}^{n'_k} Post''_j \}\!\}$$

$$\gamma_{kj}^{j\in J\cap \mathrm{Holes}(Q)} = \dot{\bigcup}_{i=n'_k}^{(n_k-1)} (\beta_{ij}^{j\in J\cap \mathrm{Holes}(Q)} \{\!\{ \bigotimes_{l=i-1}^{n'_k} Post''_l \}\!\})^{\nabla} \dot{\cup} (\beta_{n_k j}^{j\in J\cap \mathrm{Holes}(Q)} \{\!\{ \bigotimes_{l=n_k-1}^{n'_k} Post''_l \}\!\})^{\nabla} \dot{\cup}$$
$$\dot{\bigcup}_{i=n_k+1}^{n'_{k+1}-1} (\beta_{ij}^{j\in J\cap \mathrm{Holes}(Q)} \{\!\{ \bigotimes_{l=i-1}^{n_k+1} Post''_l \otimes Post''_{n_k} \otimes \bigotimes_{l=n_k-1}^{n'_k} Post''_l \}\!\})^{\nabla}$$

$$Pred_k = \bigwedge_{i=n'_k}^{n_k-1} Pred''_i \{\!\{ \bigotimes_{j=i-1}^{n'_k} Post''_j \}\!\} \wedge Pred''_{n_k} \{\!\{ \bigotimes_{j=n_k-1}^{n'_k} Post''_j \}\!\} \wedge$$
$$\bigwedge_{i=n_k+1}^{n'_{k+1}-1} Pred''_i \{\!\{ \bigotimes_{j=i-1}^{n'_k} Post''_j \otimes Post''_{n_k} \otimes \bigotimes_{j=n_k-1}^{n'_k} Post''_j \}\!\}$$

$$Post_k = \bigotimes_{j=n'_{k+1}-1}^{n'_k} Post''_j$$

Note that for all $k \in [0..n-1]$, $v'_{kl} = v_{(k+1)l}$, $v_{0l} = s_{0l} = s_l$, and $v'_{nl} = v_{(n_1+n_2+3)l} = u_{(n_2+1)l} = s'_l$.

By definition of $Post_k$, we have $\bigotimes_{j=n'_k-1}^{0} Post''_j = \bigotimes_{j=k-1}^{0} Post_j$. Consequently, we have:

$$\alpha'_{n_k} \{\!\{ \bigotimes_{j=n_k-1}^{0} Post''_j \}\!\} = \alpha'_{n_k} \{\!\{ \bigotimes_{j=n_k-1}^{n'_k} Post''_j \otimes \bigotimes_{j=n'_k-1}^{0} Post''_j \}\!\} = \alpha_k \{\!\{ \bigotimes_{j=n'_k-1}^{0} Post''_j \}\!\} = \alpha_k \{\!\{ \bigotimes_{j=k-1}^{0} Post_j \}\!\}$$

From equation B.1, we obtain the following equation (we recall that the actions $\alpha_k$ are the actions $\alpha'_p$ that are observable):

$$\begin{aligned} Pred_\beta \quad &\Longleftrightarrow \quad \gamma_{j_0} = \dot{\bigcup}_{p=0}^{p=n_1+n_2+2} (\alpha'_p \{\!\{ \bigotimes_{j=p-1}^{0} Post_j \}\!\})^{\nabla} \\ &\Longleftrightarrow \quad \gamma_{j_0} = [\alpha_p \{\!\{ \bigotimes_{j=p-1}^{0} Post_j \}\!\} | p \in [0..n]] \end{aligned}$$

We need now to show that the set of WOT obtained above verifies the conditions of the lemma, i.e. it is a set of WOT of the form:

$$Q \models \frac{\gamma_{pj}^{j\in J_p}, Pred_p, Post_p}{\lhd s_{pi}^{i\in L_Q} \rhd \overset{\alpha_p}{\Longrightarrow} \lhd s_{(p+1)i}^{i\in L_Q} \rhd}$$

with

$$\bigcup_{p=0}^{n} J_p = J \cap \mathrm{Holes}(Q) \qquad \text{trivial}$$

$$\gamma_j^{j\in J\cap \mathrm{Holes}(Q)} = \dot{\bigcup_{p=0}^{n}}(\gamma_{pj}^{j\in J_p})\{\!\{\bigotimes_{i=p-1}^{0} Post_i)\}\!\}$$

Indeed we have:

$$\begin{aligned}\gamma_j^{j\in J} = &\dot{\bigcup_{i=0}^{m_1}}(\overline{\beta_{1i}}\{\!\{\bigotimes_{j=i-1}^{0} Post_{1j}\}\!\})^{\nabla}\dot{\cup}(\overline{\beta_2}\{\!\{\bigotimes_{j=m_1}^{0} Post_{1j}\}\!\})^{\nabla}\dot{\cup}\\ &\dot{\bigcup_{i=0}^{m_2}}(\overline{\beta_{3i}}\{\!\{\bigotimes_{j=i-1}^{0} Post_{3j}\otimes Post_2\otimes \bigotimes_{j=m_1}^{0} Post_{1j}\}\!\})^{\nabla}\end{aligned}$$

And thus, because $\beta_{pj}^{j\in J\cap \mathrm{Holes}(Q)}$ are equal to the concatenation of $(\beta'_{1pj})^{j\in J\cap \mathrm{Holes}(Q)}$, $(\beta'_{2j})^{j\in J\cap \mathrm{Holes}(Q)}$, and $(\beta'_{3pj})^{j\in J\cap \mathrm{Holes}(Q)}$ (re-indexed because we skipped some transitions), and additionally $(\beta'_{1pj})^{j\in J\cap \mathrm{Holes}(Q)}$, $(\beta'_{2j})^{j\in J\cap \mathrm{Holes}(Q)}$, and $(\beta'_{3pj})^{j\in J\cap \mathrm{Holes}(Q)}$ are identical to the hole labels $\beta_{1kj}^{j\in J\cap \mathrm{Holes}(Q)}$, $\beta_{2j}^{j\in J\cap \mathrm{Holes}(Q)}$, and $\beta_{3kj}^{j\in J\cap \mathrm{Holes}(Q)}$ (re-indexed) when $Q$ moves[12]. We can assert a similar equality on post-conditions, i.e. between $Post''_p$ and $Post''_{1k}$, $Post''_2$, $Post''_{3k}$ where $Post''_{1p}$ is the restriction of $Post_{1p}$ over $vars(Q)$ (see initial decomposition, case 1, 2, and 3 above). Overall, we have $\forall i \in L_Q.\, s_{(n+1)i} = s'_i \wedge s_{0i} = s_i$ (see above):

$$\begin{aligned}\gamma_j^{j\in J\cap \mathrm{Holes}(Q)} &= \dot{\bigcup_{i=0}^{m_1}}(\overline{(\beta'_{1i})}\{\!\{\bigotimes_{j=i-1}^{0} Post''_{1j}\}\!\})^{\nabla}\dot{\cup}(\overline{\beta'_2}\{\!\{\bigotimes_{j=m_1}^{0} Post''_{1j}\}\!\})^{\nabla}\dot{\cup}\dot{\bigcup_{i=0}^{m_2}}(\overline{\beta'_{3i}}\{\!\{\bigotimes_{j=i-1}^{0} Post''_{3j}\otimes Post''_2\otimes\bigotimes_{j=m_1}^{0} Post''_{1j}\}\!\})^{\nabla}\\ &= \dot{\bigcup_{k=0}^{n}}\left(\dot{\bigcup_{i=n'_k}^{n'_{k+1}-1}}(\beta_{ij}^{j\in J\cap \mathrm{Holes}(Q)}\{\!\{\bigotimes_{j=i-1}^{n'_k} Post''_j \bigotimes_{j=n'_k-1}^{0} Post''_j\}\!\})^{\nabla}\right)\\ &= \dot{\bigcup_{k=0}^{n}}\left(\gamma_{kj}^{j\in J\cap \mathrm{Holes}(Q)}\{\!\{\bigotimes_{j=k-1}^{0} Post_k\}\!\}\right)\end{aligned}$$

Next, we have

$$Pred = Pred' \wedge \bigwedge_{p=0}^{n}\Big((\alpha_p\{\!\{\bigotimes_{i=p-1}^{0} Post_i\}\!\}) = \beta_p \wedge (Pred_p\{\!\{\bigotimes_{i=p-1}^{0} Post_i\}\!\})\Big)$$

Indeed,

$$\begin{aligned}Pred \iff &\bigwedge_{i=0}^{m_1} Pred_{1i}\{\!\{\bigotimes_{j=i-1}^{0} Post_{1j}\}\!\} \wedge Pred_2\{\!\{\bigotimes_{j=m_1}^{0} Post_{1j}\}\!\} \wedge \bigwedge_{i=0}^{m_2} Pred_{3i}\{\!\{\bigotimes_{j=i-1}^{0} Post_{3j}\otimes Post_2\otimes\bigotimes_{j=m_1}^{0} Post_{1j}\}\!\}\\ \iff &\bigwedge_{i=0}^{m_1}\left(Pred'_{1i}\wedge Pred''_{1i}\wedge \alpha_{1i} = \beta_{1ij_0}\right)\{\!\{\bigotimes_{j=i-1}^{0} Post_{1j}\}\!\}\\ &\wedge\left(Pred'_2\wedge Pred''_2\wedge\alpha_{20} = \beta_{2j_0}\right)\{\!\{\bigotimes_{j=m_1}^{0} Post_{1j}\}\!\}\\ &\wedge\bigwedge_{i=0}^{m_2}\left(Pred'_{3i}\wedge Pred''_{3i}\wedge\alpha_{3i} = \beta_{3ij_0}\right)\{\!\{\bigotimes_{j=i-1}^{0} Post_{3j}\otimes Post_2\otimes\bigotimes_{j=m_1}^{0} Post_{1j}\}\!\}\end{aligned}$$

[12] more precisely, when $Q$ moves either $\beta_{1kj}^{j\in J\cap \mathrm{Holes}(Q)}$ is not empty and thus $(\beta'_{1pj})^{j\in J\cap \mathrm{Holes}(Q)} = \beta_{1kj}^{j\in J\cap \mathrm{Holes}(Q)}$, or both are empty if the holes of $Q$ perform no action

$$\iff \bigwedge_{i=0}^{m_1}\left((Pred'_{1i}\{\!\{\bigotimes_{j=i-1}^{0} Post'_{1j}\}\!\}) \wedge (Pred''_{1i}\{\!\{\bigotimes_{j=i-1}^{0} Post''_{1j}\}\!\}) \wedge (\alpha_{1i} = \beta_{1ij_0})\{\!\{\bigotimes_{j=i-1}^{0} Post_{1j}\}\!\}\right)$$

$$\wedge \left(Pred'_2\{\!\{\bigotimes_{j=m_1}^{0} Post_{1j}\}\!\} \wedge Pred''_2\{\!\{\bigotimes_{j=m_1}^{0} Post_{1j}\}\!\} \wedge (\alpha_{20} = \beta_{2j_0})\{\!\{\bigotimes_{j=m_1}^{0} Post_{1j}\}\!\}\right)$$

$$\wedge \bigwedge_{i=0}^{m_2}\left(Pred'_{3i}\{\!\{\bigotimes_{j=i-1}^{0} Post_{3j}\otimes Post_2 \otimes \bigotimes_{j=m_1}^{0} Post_{1j}\}\!\} \wedge Pred''_{3i}\{\!\{\bigotimes_{j=i-1}^{0} Post_{3j}\otimes Post_2 \otimes \bigotimes_{j=m_1}^{0} Post_{1j}\}\!\}\right.$$

$$\left.\wedge (\alpha_{3i} = \beta_{3ij_0})\{\!\{\bigotimes_{j=i-1}^{0} Post_{3j}\otimes Post_2 \otimes \bigotimes_{j=m_1}^{0} Post_{1j}\}\!\}\right)$$

$$\iff Pred' \wedge \bigwedge_{k=0}^{n} Pred_k\{\!\{\bigotimes_{j=k-1}^{0} Post_j\}\!\} \wedge Pred_\beta$$

$$\iff Pred' \wedge \bigwedge_{k=0}^{n} Pred_k\{\!\{\bigotimes_{j=k-1}^{0} Post_j\}\!\} \wedge (\gamma_{j_0} = [\alpha_i\{\!\{\bigotimes_{j=i-1}^{0} Post_j\}\!\} | i \in [0..n]])$$

which is exactly what is needed with $\gamma_{j_0} = [\beta_0..\beta_n]$.

Finally we have $Post = Post' \uplus \bigotimes_{p=n}^{0} Post_p$ because

$$Post = \bigotimes_{j=m_2}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=m_1}^{0} Post_{1j}$$

$$= \bigotimes_{j=m_2}^{0} Post'_{3j} \otimes Post'_2 \otimes \bigotimes_{j=m_1}^{0} Post'_{1j} \uplus \bigotimes_{j=m_2}^{0} Post''_{3j} \otimes Post''_2 \otimes \bigotimes_{j=m_1}^{0} Post''_{1j}$$

$$= Post' \uplus \bigotimes_{j=n'_{n+1}-1}^{0} Post''_j$$

Which concludes because we have we have $\bigotimes_{j=n'_k-1}^{0} Post''_j = \bigotimes_{j=k-1}^{0} Post_j$. $\square$

**Lemma 7** Weak open transition composition**.** *Suppose that we have a weak open automaton such that the WOTs cannot observe silent actions (see Definition 11). Suppose $j_0 \in J$ and:*

$$P \models \frac{\beta_j^{j\in J}, Pred, Post}{\triangleleft s_i^{i\in L} \triangleright \xrightarrow{\alpha} \triangleleft (s'_i)^{i\in L} \triangleright} \quad \textit{and} \quad Q \models \frac{\overline{\gamma}, Pred_Q, Post_Q}{\triangleleft s_i^{i\in L_Q} \triangleright \xRightarrow{\alpha_Q} \triangleleft (s'_i)^{i\in L_Q} \triangleright}$$

*Let* $\quad Pred' = Pred \wedge (\beta_{j_0} = \alpha_Q \wedge Pred_Q)$ *and* $\quad Post' = Post \uplus Post_Q$
*Then, we have*

$$P[Q]_{j_0} \models \frac{\overline{\gamma} \uplus (\beta_j^{j\in J\setminus\{j_0\}})^{\nabla}, Pred', Post'}{\triangleleft s_i^{i\in L\uplus L_Q} \triangleright \xRightarrow{\alpha} \triangleleft (s'_i)^{i\in L\uplus L_Q} \triangleright}$$

*Proof.* By Lemma 4 we can decompose the WOT of $Q$ into a series of $k+1$ and $k'+1$ tau open transitions and an $\alpha'_Q$ open transition (observable or not depending on $\alpha_Q$):

$$\forall h \in [0..k].Q \models \frac{\overline{\beta_{1h}}, Pred_{1h}, Post_{1h}}{\triangleleft(s_{1h}) \triangleright \xrightarrow{\tau} \triangleleft (s_{1(h+1)}) \triangleright}, \quad Q \models \frac{\overline{\beta_2}, Pred_2, Post_2}{\triangleleft s_{20} \triangleright \xrightarrow{\alpha'_Q} \triangleleft s_{21} \triangleright},$$

$$\text{and} \quad \forall h \in [0..k'].Q \models \frac{\overline{\beta_{3h}}, Pred_{3h}, Post_{3h}}{\triangleleft(s_{3h}) \triangleright \xrightarrow{\tau} \triangleleft (s_{3(h+1)}) \triangleright}$$

such that

$$s_i^{i\in L_Q} = s_{10} \wedge s_{1(k+1)i} = s_{20} \wedge s_{21} = s_{30} \wedge s_{3(k'+1)i} = s'^{i\in L_Q}_i$$

$$\alpha_Q = \alpha'_Q \{\!\{ \bigotimes_{j=k}^{0} Post_{1j} \}\!\}$$

$$\overline{\gamma} = \biguplus_{h=0}^{k} (\overline{\beta_{1h}} \{\!\{ \bigotimes_{j=h-1}^{0} Post_{1j} \}\!\})^{\nabla} \uplus (\overline{\beta_2} \{\!\{ \bigotimes_{j=k}^{0} Post_{1j} \}\!\})^{\nabla} \uplus \biguplus_{h=0}^{k'} (\overline{\beta_{3h}} \{\!\{ \bigotimes_{j=h-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{j=k}^{0} Post_{1j} \}\!\})^{\nabla}$$

$$Pred_Q = \bigwedge_{h=0}^{k} Pred_{1h} \{\!\{ \bigotimes_{j=h-1}^{0} Post_{1j} \}\!\} \wedge Pred_2 \{\!\{ \bigotimes_{h=k}^{0} Post_{1h} \}\!\} \wedge \bigwedge_{h=0}^{k'} Pred_{3h} \{\!\{ \bigotimes_{j=h-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{h=k}^{0} Post_{1h} \}\!\}$$

$$Post_Q = \bigotimes_{h=k'}^{0} Post_{3h} \otimes Post_2 \otimes \bigotimes_{j=k}^{0} Post_{1j}$$

(1) For the first $k$ open tau transitions, by Definition 11 $P$ can necessarily make a tau open transition if the hole indexed $j_0$ makes a tau action. So by Lemma 2 we obtain $k$ open transitions in the form:

$$P[Q]_{j_0} \models \frac{\overline{\beta_{1h}}, Pred_{1h}, Post_{1h}}{\triangleleft s_{1h} \uplus s_i^{i\in L} \triangleright \xrightarrow{\tau} \triangleleft s_{1(h+1)} \uplus s_i^{i\in L} \triangleright}$$

(2) For the possibly observable open transition. By Lemma 2 with the lemma hypotheses we obtain:

$$P[Q]_{j_0} \models \frac{\beta_j^{(j\in J\setminus\{j_0\})} \uplus \overline{\beta_2}, Pred \wedge Pred_2 \wedge \alpha_Q = \beta_{j_0}, Post \uplus Post_2}{\triangleleft s_i^{i\in L} \uplus s_{20} \triangleright \xrightarrow{\alpha} \triangleleft s'^{i\in L}_i \uplus s_{21} \triangleright}$$

(3) We proceed in the same way as the first item for $k'$ last weak open transitions, and we obtain $k'$ open tau transitions.

Using Lemma 4, from cases (1), (2) and (3) we get:

$$P[Q]_{j_0} \models \frac{\overline{\gamma_c}, Pred_c, Post_c}{\triangleleft s_{10} \uplus s_i^{i\in L} \triangleright \xRightarrow{\alpha'} \triangleleft s'^{i\in L}_i \uplus s_{3(k'+1)} \triangleright}$$

where $\alpha' = \alpha\{\!\{\bigotimes_{j=k}^{0} Post_{1j}\}\!\}$ and $\alpha = \alpha'$ because $Post_{1j}$ acts on variables of $Q$ and $\alpha$ contains only variables of $P$.

$$\begin{aligned}
\overline{\gamma_c} =& \biguplus_{h=0}^{k} (\overline{\beta_{1h}}\{\!\{\bigotimes_{i=h-1}^{0} Post_{1i}\}\!\})^{\nabla} \uplus ((\beta_j{}^{(j\in J\setminus\{j_0\})} \uplus \overline{\beta_2})\{\!\{\bigotimes_{i=k}^{0} Post_{1i}\}\!\})^{\nabla} \uplus \\
& \biguplus_{h=0}^{k'} (\overline{\beta_{3h}}\{\!\{\bigotimes_{i=h-1}^{0} Post_{3i} \otimes Post_2 \otimes \bigotimes_{i=k}^{0} Post_{1i}\}\!\})^{\nabla} \\
=& \overline{\gamma} \uplus (\beta_j^{\in J\setminus\{j_0\}})^{\nabla} \qquad \text{because } Post_{1j} \text{ does not act on variables of } \beta_j. \\
Pred_c =& \bigwedge_{h=0}^{k} Pred_{1h}\{\!\{\bigotimes_{i=h-1}^{0} Post_{1i}\}\!\} \wedge \left(Pred \wedge Pred_2 \wedge \alpha'_Q = \beta_{j_0}\right)\{\!\{\bigotimes_{i=k}^{0} Post_{1i}\}\!\} \wedge \\
& \bigwedge_{h=0}^{k'} Pred_{3h}\{\!\{\bigotimes_{i=h-1}^{0} Post_{3i} \otimes (Post \uplus Post_2) \otimes \bigotimes_{i=k}^{0} Post_{1i}\}\!\} \\
Post_c =& (\bigotimes_{i=k'}^{0} Post_{3i}) \otimes (Post \uplus Post_2) \otimes \bigotimes_{i=k}^{0} Post_{1i}
\end{aligned}$$

Note that we have $s_i^{i\in L_Q} = s_{10} \wedge s_{1(k+1)i} = s_{20} \wedge s_{21} = s_{30} \wedge s_{3(k'+1)i} = s'^{i\in L_Q}_i$.

Note also that $Post$ only acts on variables of $P$ while $Post_{1i}$ only acts on variables of $Q$. We conclude on predicate and posts as follows[13]:

$$\begin{aligned}
Pred_c &= Pred_Q \wedge Pred\{\!\{\bigotimes_{i=k}^{0} Post_{1i}\}\!\} \wedge (\alpha'_Q = \beta_{j_0})\{\!\{\bigotimes_{i=k}^{0} Post_{1i}\}\!\} \\
&= Pred_Q \wedge Pred \wedge \alpha_Q = \beta_{j_0} \\
Post_c &= Post \uplus Post_Q
\end{aligned}$$

□

**Lemma 8** Weak open transition composition**.** *Suppose that we have a weak open automaton such that the WOTs cannot observe silent actions (see Definition 11). Suppose* $j_0 \in J$ *and* $\gamma_{j_0} = [\beta_0..\beta_n]$ *and additionally:*

$$P \models \frac{\gamma_j^{j\in J}, Pred, Post}{\triangleleft s_i^{i\in L} \triangleright \overset{\alpha}{\Longrightarrow} \triangleleft\, s'^{i\in L}_i \triangleright} \qquad \textit{and for all } p \in [0..n] \qquad Q \models \frac{\gamma_{pj}^{j\in J_p}, Pred_p, Post_p}{\triangleleft s_{pi}^{i\in L_Q} \triangleright \overset{\alpha_p}{\Longrightarrow} \triangleleft\, s_{(p+1)i}^{i\in L_Q} \triangleright}$$

*Let*

$$J_Q = \bigcup_{p=0}^{n} J_p \qquad \forall i \in L_Q.\, s_i = s_{0i} \;\wedge\; s'_i = s_{(n+1)i} \qquad \forall j \in J_p,\, \gamma_j = \biguplus_{p=0}^{n} \gamma_{pj}\{\!\{\bigotimes_{k=p}^{0} Post_k\}\!\}$$

$$Pred' = Pred \wedge \bigwedge_{p=0}^{n} (\alpha_p = \beta_p \wedge Pred_p)\{\!\{\bigotimes_{i=p-1}^{0} Post_i\}\!\} \qquad Post' = Post \uplus \bigotimes_{p=n}^{0} Post_p$$

[13] $Post_{1i}$ only has an effect on variables of $Q$ and thus does not modify $Pred$ or $\beta_{j_0}$

*Then, we have*

$$P[Q]_{j_0} \models \frac{\gamma_j^{j\in(J\setminus\{j_0\})\uplus J_Q}, Pred', Post'}{\triangleleft s_i^{i\in L\uplus L_Q} \triangleright \overset{\alpha}{\Rightarrow} \triangleleft\, {s'_i}^{i\in L\uplus L_Q} \triangleright}$$

*Proof.* Suppose we have:

$$P \models \frac{\gamma_j^{j\in J}, Pred, Post}{\triangleleft s_i^{i\in L} \triangleright \overset{\alpha}{\Rightarrow} \triangleleft\, {s'_i}^{i\in L} \triangleright}$$

By Lemma 4 this implies the following:

$\forall p\in[0\,..\,m_1].P \models \dfrac{\beta_{1pj}^{j\in J_{1p}}, Pred_{1p}, Post_{1p}}{\triangleleft(s_{1pi})^{i\in L} \triangleright \xrightarrow{\tau} \triangleleft\,(s_{1(p+1)i})^{i\in L} \triangleright}, \quad P \models \dfrac{\beta_{2j}^{j\in J_2}, Pred_2, Post_2}{\triangleleft(s_{20i})^{i\in L} \triangleright \xrightarrow{\alpha'} \triangleleft\,(s_{21i})^{i\in L} \triangleright}$

and $\forall p \in [0\,..\,m_2].P \models \dfrac{\beta_{3pj}^{j\in J_{3p}}, Pred_{3p}, Post_{3p}}{\triangleleft(s_{3pi})^{i\in L} \triangleright \xrightarrow{\tau} \triangleleft\,(s_{3(p+1)i})^{i\in L} \triangleright}$

where:

$\forall i \in L.\ s_i = s_{10i} \wedge s_{1(m_1+1)i} = s_{20i} \wedge s_{21i} = s_{30i} \wedge s_{3(m_2+1)i} = s'_i$

$$\alpha = \alpha'\{\{\bigotimes_{j=m_1}^{0} Post_{1j}\}\}$$

$$\gamma_j^{j\in J} = \bigcup_{i=0}^{m_1}\!\!\cdot\,(\beta_{1ij}^{j\in J_{1p}}\{\{\bigotimes_{k=i-1}^{0} Post_{1k}\}\})^{\nabla} \uplus (\beta_{2j}^{j\in J_2}\{\{\bigotimes_{k=m_1}^{0} Post_{1k}\}\})^{\nabla} \uplus$$
$$\bigcup_{i=0}^{m_2}\!\!\cdot\,(\beta_{3ij}^{j\in J_{3p}}\{\{\bigotimes_{k=i-1}^{0} Post_{3k} \otimes Post_2 \otimes \bigotimes_{k=m_1}^{0} Post_{1k}\}\})^{\nabla}$$

$$Pred = \bigwedge_{p=0}^{m_1}\big(Pred_{1p}\{\{\bigotimes_{j=p-1}^{0} Post_{1j}\}\}\big) \wedge Pred_2\{\{\bigotimes_{p=m_1}^{0} Post_{1p}\}\} \wedge \bigwedge_{p=0}^{m_2} Pred_{3p}\{\{\bigotimes_{j=p-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{p=m_1}^{0} Post_{1p}\}\}$$

$$Post = \bigotimes_{p=m_2}^{0} Post_{3p} \otimes Post_2 \otimes \bigotimes_{j=m_1}^{0} Post_{1j}$$

Note that, for $l \in \{1,3\}$ if $\beta_{lpj_0} = \tau$, then, because of Definition 11, $P$ necessarily makes a $\tau$ open transition and remains in the same state, e.g. $s_{1pi} = s_{1(p+1)i}$. Thus without loss of generality, we can bypass such an open transition and obtain another decomposition of the WOT without the open transition that requires $\beta_{lpj_0} = \tau$. We can thus suppose that for all $p$ and $l$ we have $\beta_{lpj_0} \neq \tau$ or $j_0 \notin J_{1p}$. To avoid a special case, we suppose that the hole $j_0$ moves during the OT $\alpha'$, i.e. $\beta_{2j_0} = \beta_m$ for some $m$. Additionally, $\beta_m \neq \tau$, else we would have $\alpha = \alpha' = \tau$ and the $\alpha'$ OT could be also removed from the reduction, leading to a particular and simpler case.

We introduce $n_i^{i\in[0..m-1]}$, and $(n'_i)^{i\in[m+1..n]}$ the indices of the steps in which the hole $j_0$ moves in the 3 sets of OTs above ($\beta_m$ is the action that matches the hole $j_0$ in the OT $\alpha'$), in other words, we have for all $i$, $\beta_{1n_ij_0}$ a visible action, as additionally:

$\gamma_{j_0} = [\beta_0..\beta_n]$

$$= \bigcup_{\substack{i=0\\ j_0\in J_{1i}}}^{m_1} (\beta_{1ij_0}\{\!\{\bigotimes_{k=i-1}^{0} Post_{1k}\}\!\})^{\nabla} \cup (\beta_{2j_0}\{\!\{\bigotimes_{k=m_1}^{0} Post_{1k}\}\!\})^{\nabla} \cup \bigcup_{\substack{i=0\\ j_0\in J_{3i}}}^{m_2} (\beta_{3ij_0}\{\!\{\bigotimes_{k=i-1}^{0} Post_{3k} \otimes Post_2 \otimes \bigotimes_{k=m_1}^{0} Post_{1k}\}\!\})^{\nabla}$$

We have, by definition of $n_i$ and $n'_i$:

$\forall i \in [0\,..\,m-1], \beta_{1n_ij_0}\{\!\{ \bigotimes_{k=n_i-1}^{0} Post_{1k}\}\!\} = \beta_i, \qquad \beta_{2j_0}\{\!\{ \bigotimes_{k=m_1}^{0} Post_{1k}\}\!\} = \beta_m$, and

$\forall i \in [m+1\,..\,n], \beta_{3n'_ij_0}\{\!\{ \bigotimes_{k=n'_i-1}^{0} Post_{3k} \otimes Post_2 \otimes \bigotimes_{k=m_1}^{0} Post_{1k}\}\!\} = \beta_i$

Now, we compose OTs for each of the case above (depending on the OT of $P$):

(1) For the first $\tau$ OTs, i.e. $p \in [0\,..\,m_1]$. We have:
Either there is $i$ such that $p = n_i$, and thus $\beta_i$ and $\beta_{1pj_0}$ are defined. In this case by Lemma 7, we have:

$$P[Q]_{j_0} \models \frac{\overline{\gamma'_{1p}}, Pred'_{1p}, Post'_{1p}}{\triangleleft s_{1pj}^{j\in L} \uplus s_{ij}^{j\in L_Q} \triangleright \xRightarrow{\tau} \triangleleft\, s_{1(p+1)j}^{j\in L} \uplus s_{(i+1)j}^{j\in L_Q} \triangleright}$$

with

$$\overline{\gamma'_{1p}} = \gamma_{ij}^{j\in J_i} \uplus (\beta_{1pj}^{j\in J_{1p}\setminus\{j_0\}})^{\nabla} \qquad Pred'_{1p} = Pred_{1p} \wedge (\beta_{1pj_0} = \alpha_i \wedge Pred_i)$$

$$Post'_{1p} = Post_{1p} \uplus Post_i$$

Or $j_0 \notin \mathrm{dom}(\beta_{1p})$ and $Q$ does not move in the composed reduction. In this case there is no $i$ such that $p = n_i$, but there is $i$ such that $p \in ]n_i..n_{i+1}[$, and

$$P[Q]_{j_0} \models \frac{\overline{\beta_{1p}}, Pred_{1p}, Post_{1p}}{\triangleleft s_{1pj}^{j\in L} \uplus s_{ij}^{j\in L_Q} \triangleright \xrightarrow{\tau} \triangleleft\, s_{1(p+1)j}^{j\in L} \uplus s_{ij}^{j\in L_Q} \triangleright}$$

and thus we also have a weak OT by Definition 13 (rule (**WT2**)):

$$P[Q]_{j_0} \models \frac{\overline{\gamma'_{1p}}, Pred'_{1p}, Post'_{1p}}{\triangleleft s_{1pj}^{j\in L} \uplus s_{ij}^{j\in L_Q} \triangleright \xRightarrow{\tau} \triangleleft\, s_{1(p+1)j}^{j\in L} \uplus s_{ij}^{j\in L_Q} \triangleright}$$

with $\overline{\gamma'_{1p}} = \overline{(\beta_{1p})^{\nabla}}$, $Pred'_{1p} = Pred_{1p}$, $Post'_{1p} = Post_{1p}$

(2) Similarly, for the middle OT with label $\alpha$:

$$P[Q]_{j_0} \models \frac{\overline{\gamma'_2}, Pred'_2, Post'_2}{\triangleleft (s_{20j})^{j\in L} \uplus (s_{mj})^{j\in L_Q} \triangleright \xRightarrow{\alpha'} \triangleleft\, (s_{21j})^{j\in L} \uplus (s_{(m+1)j})^{j\in L_Q} \triangleright}$$

with

$$\overline{\gamma'_2} = \gamma_{mj}^{j\in J_m} \uplus (\beta_{2j}^{j\in J_2\setminus\{j_0\}})^{\nabla} \qquad Pred'_2 = Pred_2 \wedge (\beta_{2j_0} = \alpha_m \wedge Pred_m)$$

$$Post'_2 = Post_2 \uplus Post_m$$

(3) For the last $\tau$ OTs, i.e. $p \in [0\,..\,m_2]$. We have similarly to the first case:
Either there is $i$ such that $p = n'_i$, and thus $\beta_i$ and $\beta_{1pj_0}$ are defined. In this case by Lemma 7, we have:

$$P[Q]_{j_0} \models \frac{\overline{\gamma'_{3p}}, Pred'_{3p}, Post'_{3p}}{\triangleleft s_{3pj}^{j\in L} \uplus s_{ij}^{j\in L_Q} \triangleright \xRightarrow{\tau} \triangleleft\, s_{3(p+1)j}^{j\in L} \uplus s_{(i+1)j}^{j\in L_Q} \triangleright}$$

with

$$\overline{\gamma'_{3p}} = \gamma_{ij}^{j\in J_i} \uplus (\beta_{3pj}^{j\in J_{3p}\setminus\{j_0\}})^{\nabla} \qquad Pred'_{3p} = Pred_{3p} \wedge (\beta_{3pj_0} = \alpha_i \wedge Pred_i)$$

$$Post'_{3p} = Post_{3p} \uplus Post_i$$

Or $j_0 \notin \mathrm{dom}(\beta_{3p})$ and $Q$ does not move in the composed reduction. In this case there is no $i$ such that $p = n'_i$, but there is $i$ such that $p \in ]n'_i..n'_{i+1}[$, and

$$P[Q]_{j_0} \models \frac{\overline{\beta_{3p}}, Pred_{3p}, Post_{3p}}{\triangleleft s_{3pj}^{j\in L} \uplus s_{ij}^{j\in L_Q} \triangleright \xrightarrow{\tau} \triangleleft\, s_{3(p+1)j}^{j\in L} \uplus s_{ij}^{j\in L_Q} \triangleright}$$

and thus we also have a weak OT by definition 13 (rule **WT2**):

$$P[Q]_{j_0} \models \frac{\overline{\gamma'_{3p}}, Pred'_{3p}, Post'_{3p}}{\triangleleft s_{3pj}^{j\in L} \uplus s_{ij}^{j\in L_Q} \triangleright \xRightarrow{\tau} \triangleleft\, s_{3(p+1)j}^{j\in L} \uplus s_{ij}^{j\in L_Q} \triangleright}$$

with $\overline{\gamma'_{3p}} = \overline{\beta_{3p}}, Pred'_{3p} = Pred_{3p}, Post'_{3p} = Post_{3p}$

By definition of weak open transition (Definition 13, rule **WT3**), we obtain:

$$P[Q]_{j_0} \models \frac{\overline{\gamma'}, Pred'', Post''}{\triangleleft s_{10j}^{j\in L} \uplus s_{0j}^{j\in L_Q} \triangleright \xRightarrow{\alpha''} \triangleleft\, s_{3(m_2+1)j}^{j\in L} \uplus s_{(n+1)j}^{j\in L_Q} \triangleright}$$

Where

$$\begin{aligned}
\alpha'' =& \alpha'\{\!\{\bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\}\\
\overline{\gamma'} =& \dot{\bigcup_{i=0}^{m_1}} \overline{\gamma'_{1i}}\{\!\{\bigotimes_{k=i-1}^{0} Post'_{1k}\}\!\} \cup \overline{\gamma'_2}\{\!\{\bigotimes_{k=m_1}^{0} Post'_{1k}\}\!\} \cup \dot{\bigcup_{i=0}^{m_2}} \overline{\gamma'_{3i}}\{\!\{\bigotimes_{k=i-1}^{0} Post'_{3k} \otimes Post'_2 \otimes \bigotimes_{k=n}^{0} Post'_{1k}\}\!\}\\
Pred'' =& \bigwedge_{i=0}^{m_1} Pred'_{1i}\{\!\{\bigotimes_{j=i-1}^{0} Post'_{1j}\}\!\} \wedge Pred'_2\{\!\{\bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\} \wedge\\
&\bigwedge_{i=0}^{m_2} Pred'_{3i}\{\!\{\bigotimes_{j=i-1}^{0} Post'_{3j} \otimes Post'_2 \otimes \bigotimes_{j=m_2}^{0} Post'_{1j}\}\!\}\\
Post'' =& \bigotimes_{j=m_2}^{0} Post'_{3j} \otimes Post'_2 \otimes \bigotimes_{j=m_1}^{0} Post'_{1j}
\end{aligned}$$

However it must be noticed that in steps 1 and 3, we have two kinds of WOTs with different signatures (depending on whether $Q$ moves or not). It is still possible to glue them together in a global rule with two more terms for *Pred* and *Post* terms. This global merge is possible because the post-conditions of $P$ only act on variables of $P$ and those of $Q$ on variables of $Q$ (for example $Post_i$ has no effect on $Pred_{1p}$ and thus does not need to be taken into account when dealing with WOTs where $Q$ does not move).

We now compare each element of the obtained WOT with the conclusion of the lemma:

$$\begin{aligned}
\alpha'' &= \alpha'\{\!\{\bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\} \\
&= \alpha'\{\!\{\bigotimes_{j=m_1}^{0} Post_{1j}\}\!\} && \alpha' \text{ only contains variables of } P \text{ untouched by } Post_i \\
&= \alpha
\end{aligned}$$

For $\overline{\gamma'}$ we distinguish elements in the holes of $P$ and of $Q$.

First suppose $j \in J \setminus \{j_0\}$ we have $\gamma'_j = \gamma_j$ because $Post'_{ij}$ has no effect on variables of $P$ and on $\beta_{1pj}$, consequently

$$\gamma'_j = \dot{\bigcup_{i=0}^{m_1}}(\beta_{1ij}\{\!\{\bigotimes_{k=i-1}^{0} Post_{1k}\}\!\})^{\nabla} \dot{\cup} (\beta_{2j}\{\!\{\bigotimes_{k=m_1}^{0} Post_{1k}\}\!\})^{\nabla} \dot{\cup} \dot{\bigcup_{i=0}^{m_2}}(\beta_{3ij}\{\!\{\bigotimes_{k=i-1}^{0} Post_{3k} \otimes Post_2 \otimes \bigotimes_{k=m_1}^{0} Post_{1k}\}\!\})^{\nabla}$$

Second, when $j \in J_t$ for some $t$, $\gamma'_j$ is the concatenation of elements of $\gamma'_{1ij}$, $\gamma'_{2j}$, $\gamma'_{3ij}$ that are not empty. By construction the concatenation of these elements is $\gamma_{tj}$, for $t \in [0..n]$. $Post_{ik}$ has no effect on $\gamma_{tj}$ but $Post_k$ has. We obtain:

$$\begin{aligned}
\gamma'_j &= \dot{\bigcup_{i=0}^{m_1}} \gamma'_{1ij}\{\!\{\bigotimes_{k=i-1}^{0} Post'_{1k}\}\!\} \dot{\cup} \gamma'_{2j}\{\!\{\bigotimes_{k=m_1}^{0} Post'_{1k}\}\!\} \dot{\cup} \dot{\bigcup_{i=0}^{m_2}} \gamma'_{3ij}\{\!\{\bigotimes_{k=i-1}^{0} Post'_{3k} \otimes Post'_2 \otimes \bigotimes_{k=n}^{0} Post'_{1k}\}\!\} \\
&= \dot{\bigcup_{t=0}^{n}} \gamma_{tj}\{\!\{\bigotimes_{k=t-1}^{0} Post_k\}\!\}
\end{aligned}$$

Concerning predicates, we also separate predicates on $P$ from predicates on $Q$, and from the equality on the action filling the hole:

$$\begin{aligned}
Pred'' = &\Big(\bigwedge_{i=0}^{m_1} Pred'_{1i}\{\!\{\bigotimes_{j=i-1}^{0} Post'_{1j}\}\!\} \wedge Pred'_2\{\!\{\bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\} \wedge \bigwedge_{i=0}^{m_3} Pred'_{3i}\{\!\{\bigotimes_{j=i-1}^{0} Post'_{3j} \otimes Post'_2 \otimes \bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\}\Big) \\
= &\Big(\bigwedge_{p=0}^{m_1} Pred_{1p}\{\!\{\bigotimes_{j=p-1}^{0} Post_{1j}\}\!\} \wedge Pred_2\{\!\{\bigotimes_{p=m_1}^{0} Post_{1p}\}\!\} \wedge \bigwedge_{p=0}^{m_2} Pred_{3p}\{\!\{\bigotimes_{j=p-1}^{0} Post_{3j} \otimes Post_2 \otimes \bigotimes_{p=m_1}^{0} Post_{1p}\}\!\}\Big) \\
&\wedge \bigwedge_{t=0}^{n} Pred_t \bigotimes_{i=t-1}^{0} Post_i \wedge \Big(\bigwedge_{i=0}^{m-1} (\beta_{1n_ij_0} = \alpha_i)\{\!\{\bigotimes_{j=n_i-1}^{0} Post'_{1j}\}\!\} \wedge (\beta_{2j_0} = \alpha_m)\{\!\{\bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\} \wedge \\
&\bigwedge_{i=m+1}^{n} (\beta_{3n'_ij_0} = \alpha_i)\{\!\{\bigotimes_{j=n'_i-1}^{0} Post'_{3j} \otimes Post'_2 \otimes \bigotimes_{j=m_1}^{0} Post'_{1j}\}\!\}\Big)
\end{aligned}$$

$$
\begin{aligned}
&= \Big(\bigwedge_{p=0}^{m_1} \mathit{Pred}_{1p}\{\!\{\bigotimes_{j=p-1}^{0} \mathit{Post}_{1j}\}\!\}\Big) \wedge \mathit{Pred}_2\{\!\{\bigotimes_{p=m_1}^{0} \mathit{Post}_{1p}\}\!\} \wedge \bigwedge_{p=0}^{m_2} \mathit{Pred}_{3p}\{\!\{\bigotimes_{j=p-1}^{0} \mathit{Post}_{3j} \otimes \mathit{Post}_2 \otimes \bigotimes_{p=m_1}^{0} \mathit{Post}_{1p}\}\!\}\Big) \\
&\quad \wedge \bigwedge_{t=0}^{n} \mathit{Pred}_t \bigotimes_{i=t-1}^{0} \mathit{Post}_i \wedge \Big(\bigwedge_{i=0}^{m-1} (\beta_i = \alpha_i)\{\!\{\bigotimes_{j=i-1}^{0} \mathit{Post}_j\}\!\} \wedge (\beta_m = \alpha_m)\{\!\{\bigotimes_{j=m}^{0} \mathit{Post}_j\}\!\} \wedge \\
&\quad \bigwedge_{i=m+1}^{n} (\beta_i = \alpha_i)\{\!\{\bigotimes_{j=i-1}^{m} \mathit{Post}_j \otimes \mathit{Post}_m \otimes \bigotimes_{j=m-1}^{0} \mathit{Post}_j\}\!\}\Big) \\
&= \mathit{Pred}
\end{aligned}
$$

Finally, concerning post-conditions:

$$
\begin{aligned}
\mathit{Post}'' &= \bigotimes_{j=m_2}^{0} \mathit{Post}'_{3j} \otimes \mathit{Post}'_2 \otimes \bigotimes_{j=m_1}^{0} \mathit{Post}'_{1j} \\
&= \left(\bigotimes_{j=m_2}^{0} \mathit{Post}'_{3j} \otimes \mathit{Post}'_2 \otimes \bigotimes_{j=m_1}^{0} \mathit{Post}'_{1j}\right) \uplus \bigotimes_{j=n}^{0} \mathit{Post}_j \\
&= \mathit{Post} \uplus \bigotimes_{j=n}^{0} \mathit{Post}_j
\end{aligned}
$$

This allows us to conclude concerning the lemma. □

**Theorem 7.** *Congruence.* Consider an open pNet: $P = \langle\!\langle P_i^{i\in I}, \mathit{Sort}_j^{j\in J}, \overline{SV}\rangle\!\rangle$. Let $j_0 \in J$ be a hole. Let $Q$ and $Q'$ be two weak FH-bisimilar pNets such that $\mathrm{Sort}(Q) = \mathrm{Sort}(Q') = \mathit{Sort}_{j_0}$. Then $P[Q]_{j_0}$ and $P[Q']_{j_0}$ are weak FH-bisimilar.

*Proof.* Consider $Q$ weak FH-bisimilar to $Q'$. It means that there exist a FH-bisimulation $\mathcal{R}_{Q,Q'}$ relating the two pNets $Q$ and $Q'$. We define a relation $\mathcal{R}$ relating states of $P[Q]_{j_0}$ with states of $P[Q']_{j_0}$:

$$
\mathcal{R} = \{(\triangleleft S_P \uplus S_Q \triangleright, \triangleleft S_P \uplus S_{Q'} \triangleright, \mathit{Pred}_{Q,Q'}) |\, (S_Q, S_{Q'}, \mathit{Pred}_{Q,Q'}) \in \mathcal{R}_{Q,Q'}\}
$$

To prove weak FH-bisimulation of $P[Q]_{j_0}$ and $P[Q']_{j_0}$, we consider an open transition $OT$ of $P[Q]_{j_0}$, and an equivalent state of $P[Q']_{j_0}$, and we try to find a family of WOT of $P[Q']_{j_0}$ that simulates $OT$. Consider an OT of $P[Q]_{j_0}$ it is of the form (notations introduced to prepare the decomposition):

$$
P[Q]_{j_0} \models \frac{\beta_j^{j\in(J_P \uplus J_Q)}, \mathit{Pred}_P \wedge \mathit{Pred}_Q, \mathit{Post}_P \uplus \mathit{Post}_Q}{\triangleleft S_P \uplus S_Q \triangleright \xrightarrow{\alpha} \triangleleft S'_P \uplus S'_Q \triangleright}
$$

By the decomposition lemma for OTs (Lemma 1), we obtain the 2 following OTs (equality side-conditions have been inlined for clarity):

$$
P \models \frac{\beta_j^{j\in J_P} \uplus (j_0 \mapsto \alpha_Q), \mathit{Pred}_P, \mathit{Post}_Q}{\triangleleft S_P \triangleright \xrightarrow{\alpha} \triangleleft S'_P \triangleright} \quad \text{and} \quad Q \models \frac{\beta_j^{j\in J_Q}, \mathit{Pred}_Q, \mathit{Post}_Q}{\triangleleft S_Q \triangleright \xrightarrow{\alpha_Q} \triangleleft S'_Q \triangleright}
$$

By definition of $\mathcal{R}$ we have $(S_Q, S_{Q'} | \mathit{Pred}_{Q,Q'}) \in \mathcal{R}_{Q,Q'}$. And thus, by definition of weak FH-bisimulation, there exist a family of weak open transitions $WOT_x$:

$$
\frac{\gamma_{jx}^{j\in J_Q}, \mathit{Pred}_{Q'x}, \mathit{Post}_{Q'x}}{\triangleleft S_{Q'} \triangleright \stackrel{\alpha_x}{\Longrightarrow} \triangleleft S'_{Q'x} \triangleright}
$$

where

$$\forall x.\,(S'_Q, S'_{Q'x}|Pred_{Q,Q'x}) \in \mathcal{R}_{Q,Q'}$$

and

$$Pred_{Q,Q'} \wedge Pred_Q \implies \bigvee_{x\in X} (\forall j \in J_Q.\,(\beta_j)^\nabla = \gamma_{jx}) \Rightarrow (Pred_{Q_x} \wedge \alpha_Q = \alpha_x \wedge Pred_{Q,Q'x}\{\!\{Post_{Q'x} \uplus Post_Q\}\!\})$$

Composing the OT of $P$ with the WOTs of $Q'$ by Lemma 7 we obtain:

$$P[Q']_{j_0} \models \frac{(\beta_j^{j\in J_P})^\nabla \uplus \gamma_{jx}^{j\in J_Q}, Pred_P \wedge Pred_{Q'x}, Post_P \uplus Post_{Q'x}}{\triangleleft S_P \uplus S_{Q'} \triangleright \overset{\alpha}{\Rightarrow} \triangleleft\, S'_P \uplus S'_{Q'x} \triangleright}$$

with $\bigvee_{x\in X} (\forall j \in J_Q.\,(\beta_j)^\nabla = \gamma_{jx} \implies \alpha_Q = \alpha_x)$ that ensures that the open transitions can be recomposed when the OT fires.

Side conditions necessary to prove weak-FH bisimulations are:

$$\forall x.\,(S'_P \uplus S'_Q, S'_P \uplus S'_{Q'x}|Pred_{Q,Q'x}) \in \mathcal{R}$$

which is true, and

$$Pred_{Q,Q'} \wedge Pred_P \wedge Pred_Q \implies \Big(\bigvee_{x\in X} (\forall j \in J_Q.\,(\beta_j)^\nabla = \gamma_{jx} \wedge \forall j \in J_P.\,(\beta_j)^\nabla = (\beta_j)^\nabla)) \Rightarrow (Pred_P \wedge Pred_{Q'x} \wedge \alpha = \alpha \wedge Pred_{Q,Q'x}\{\!\{Post_P \uplus Post_{Q'x} \uplus Post_Q\}\!\})\Big)$$

We conclude by observing that $Post_P$ has no effect on variables of $Q$ and $Q'$, and thus on $Pred_{Q,Q'x}$ □

**Theorem 8**. *Context equivalence.* Consider two FH-bisimilar open pNets: $P = \langle\!\langle P_i^{i\in I}, Sort_j^{j\in J}, \overline{SV}\rangle\!\rangle$ and $P' = \langle\!\langle P'^{i\in I}_i, Sort_j^{j\in J}, \overline{SV}'\rangle\!\rangle$ (recall they must have the same holes to be bisimilar). Let $j_0 \in J$ be a hole, and $Q$ be a pNet such that $\mathrm{Sort}(Q) = Sort_{j_0}$. Then $P[Q]_{j_0}$ and $P'[Q]_{j_0}$ are FH-bisimilar.

*Proof.* Consider $P$ weak FH-bisimilar to $P'$. There exist a FH-bisimulation $\mathcal{R}_{P,P'}$ relating $P$ and $P'$. We define a relation $\mathcal{R}$ relating states of $P[Q]_{j_0}$ with states of $P'[Q]_{j_0}$:

$$\mathcal{R} = \{(\triangleleft S_P \uplus S_Q \triangleright, \triangleleft S_{P'} \uplus S_Q \triangleright, Pred_{P,P'})|\,(S_P, S_{P'}, Pred_{P,P'}) \in \mathcal{R}_{P,P'}\}$$

To prove weak FH-bisimulation of $P[Q]_{j_0}$ and $P'[Q]_{j_0}$, we consider an open transition $OT$ of $P[Q]_{j_0}$, and an equivalent state of $P'[Q]_{j_0}$, and we try to find a family of WOT of $P'[Q]_{j_0}$ that simulates $OT$. Consider an OT of $P[Q]_{j_0}$ it is of the form (notations introduced to prepare the decomposition):

$$P[Q]_{j_0} \models \frac{\beta_j^{j\in(J_P \uplus J_Q)}, Pred_P \wedge Pred_Q \wedge Pred, Post_P \uplus Post_Q}{\triangleleft S_P \uplus S_Q \triangleright \xrightarrow{\alpha} \triangleleft\, S'_P \uplus S'_Q \triangleright}$$

By the decomposition lemma for OTs (Lemma 1), we obtain the 2 following OTs (equality side-conditions have been inlined for clarity):

$$P \models \frac{\beta_j^{j\in J_P} \uplus (j_0 \mapsto \alpha_Q), Pred_P, Post_Q}{\triangleleft S_P \triangleright \xrightarrow{\alpha} \triangleleft S'_P \triangleright} \quad \text{and} \quad Q \models \frac{\beta_j^{j\in J_Q}, Pred_Q, Post_Q}{\triangleleft S_Q \triangleright \xrightarrow{\alpha_Q} \triangleleft S'_Q \triangleright}$$

With $Pred \iff \alpha_Q = \beta_{j_0}$

By definition of $\mathcal{R}$ we have $(S_P, S_{P'}, Pred_{P,P'}) \in \mathcal{R}_{P,P'}$. And thus, by definition of weak FH-bisimulation, there exist a family of weak open transitions $WOT_x$:

$$\frac{\gamma_{jx}^{j\in J_P \uplus \{j_0\}}, Pred_{P'x}, Post_{P'x}}{\triangleleft S_{P'} \triangleright \xRightarrow{\alpha_x} \triangleleft S'_{P'x} \triangleright}$$

where

$$\forall x. \, (S'_P, S'_{P'x}, Pred_{P,P'x}) \in \mathcal{R}_{P,P'}$$

and

$$Pred_{P,P'} \wedge Pred_P \implies \Big( \bigvee_{x\in X} (\forall j \in J_P. \, (\beta_j)^{\nabla} = \gamma_{jx} \wedge (\alpha_Q)^{\nabla} = \gamma_{j_0}) \Rightarrow (Pred_{P'_x} \wedge$$
$$\alpha = \alpha_x \wedge Pred_{P,P'x}\{\!\{Post_{P'x} \uplus Post_P\}\!\}) \Big)$$

We here need a special case of Lemma 8 where the inner pNet $Q$ does a simple OT. This is just a particular case of the theorem but where notations get simplified because the inner pnet does a single transition. This way we can compose the WOTs of $P'$ with the OT of $Q$ and obtain, with $\gamma_{j_0} = [\beta]$:

$$P'[Q]_{j_0} \models \frac{(\beta_j^{j\in J_Q})^{\nabla} \uplus \gamma_{jx}^{j\in J_P}, Pred_{P'x} \wedge Pred_Q \wedge \alpha_Q = \beta, Post_{P'x} \uplus Post_Q}{\triangleleft S_{P'} \uplus S_Q \triangleright \xRightarrow{\alpha_x} \triangleleft S'_{P'x} \uplus S'_Q \triangleright}$$

Side conditions necessary to prove weak-FH bisimulations are:

$$\forall x. \, (S'_P \uplus S'_Q, S'_{P'x} \uplus S'_Q, Pred_{P,P'x}) \in \mathcal{R}$$

which is true, and

$$Pred_{P,P'} \wedge Pred_P \wedge Pred_Q Pred \implies$$
$$\Big( \bigvee_{x\in X} (\forall j \in J_P. \, (\beta_j)^{\nabla} = \gamma_{jx} \wedge \forall j \in J_Q. \, (\beta_j)^{\nabla} = (\beta_j)^{\nabla}) \Rightarrow$$
$$(Pred_{P'x} \wedge Pred_Q \wedge \alpha_Q = \beta \wedge \alpha_x = \alpha \wedge Pred_{P,P'x}\{\!\{Post_{P'x} \uplus Post_P \uplus Post_Q\}\!\}) \Big)$$

We conclude by observing that $Post_Q$ has no effect on variables of $P$ and $P'$, and thus on $Pred_{P,P'x}$ and $Pred$ leading to the conclusion about $\alpha_Q = \beta$. $\square$

## Appendix C. Full details of the Simple Protocol Example

The first piece of code is the textual definition of the SimpleSystemSpec pNet (named SimpleProtSpec2 in the code here), that was drawn in Figure 2, page 8. This code should be intuitive enough to read, with the following language conventions, that brings some user-friendly features, mapped by the editor into pure pNet constructs.

- Constants of any type (including Action) must be declared as "const". They are used either as functions with argument, as typically `in(msg)`, or constants without argument, typically as "tau()".
- Variables can be declared as global variables of a pLTS (e.g. `m_msg` in `PerfectBuffer`), or a pNet Node in the case of synchronisation vector variables (e.g. `p_a`), or as input variables in a pLTS, as `?msg` in `PerfectBuffer`.
- The variables in the guards of synchronisation vectors (e.g. in SV1) do not need to be explicitly quantified: by convention, all variables in a guard that do not appear inside the vector actions will be recognised as bound by a *forall* quantifier inside the guard.
- The tools will check that everything is correctly declared, that variables are used properly and do not conflict between different objects, that vectors have coherent length, etc.

```
SimpleProtSpec2:
import "Data_Alg.algp"
root SimpleProtSpec2
const in, out:Action
const p_send, q_recv: Action
const tau:Action

pLTS PerfectBuffer
initial b0
vars ?m:Data
vars b_msg:Data b_ec:Nat

state b0
transition in(m) -> b1 {b_msg:=m, b_ec:=0}

state b1
transition out(b_msg, b_ec) -> a0
transition synchro(tau())  -> b1 {b_ec:=b_ec+1}

pNet SimpleProtSpec2
holes P,Q
subnets P,PerfectBuffer,Q
vars p_a,q_b:Action m:Data ec:Nat

vector SV0 <p_send(m),in(m),_>->synchro(in(m))
vector SV1 <p_a,_,_>->p_a [p_a != p_send(x)]
vector SV2 <_,out(m,ec),q_recv(m,ec)>->synchro(out(m,ec))
vector SV3 <_,_,q_b>->q_a [q_b != q_recv(x,y)]
```

The corresponding generated Open Automaton was given in Figure 3, page 14.

Next is the code for the SimpleSystem implementation pNet (named SimpleProtImpl2):

```
SimpleProtImpl2:
  import "Data_Alg.algp"
  root SimpleProtImpl2
const in,out:Action
const tau,p_send,q_recv,m_recv,m_send,m_error: Action
const s_recv,s_send,s_ack,s_error,r_recv,r_ack,r_send: Action

pLTS Sender
  initial s0
  vars ?m:Data
  vars  s_msg:Data s_ec:Nat
state s0
  transition s_recv(m) -> s1 {s_msg:=m, s_ec:=0}
state s1
  transition s_send(s_msg, s_ec) -> s2
state s2
  transition s_ack() -> s0
  transition s_error() -> s1 {s_ec:=s_ec+1}

pLTS Medium
  initial m0
  vars ?m:Data ?ec:Nat
  vars m_msg:Data m_ec:Nat
state m0
  transition m_recv(m,ec) -> m1 {m_msg:=m, m_ec:=ec}
state m1
  transition m_send(m_msg, m_ec) -> m0
  transition synchro(tau()) -> m2
state m2
  transition m_error() -> m0

pLTS Receiver
  initial r0
  vars ?m:Data ?ec:Nat
  vars r_msg:Data r_ec:Nat
state r0
  transition r_recv(m,ec) -> r1 {r_msg:=msg, r_ec:=ec}
state r1
  transition r_send(r_msg, r_ec) -> r2
state r2
  transition r_ack() -> r0

pNet SimpleProtocol
  subnets Sender,Medium,Receiver
  vars m:Data c:Nat
vector SV0 <s_recv(m),_, _>->in(m)
vector SV1 <s_send(m,ec),m_recv(m,ec),_>->synchro(tau())
vector SV2 <_,m_send(m,ec),r_recv(m,ec)>->synchro(tau())
vector SV3 <s_ack(),_,r_ack()>->synchro(tau())
vector SV4 <s_error(),m_error(),_>->synchro(tau())
vector SV5 <_,_,r_send(m,ec)>->out(m,ec)

pNet SimpleProtImpl2
  holes P,Q
  subnets P,SimpleProtocol,Q
  vars p_a,q_a:Action  m:Data c:Nat
vector SV0 <p_send(m),in(m),_>->synchro(in(m))
vector SV1 <p_a,_,_>->p_a [p_a != p_send(x)]
vector SV2 <_,out(m,ec),q_recv(m,ec)>->synchro(out(m,ec))
vector SV3 <_,_,q_b>->q_b [q_b != q_recv(x,y)]
```

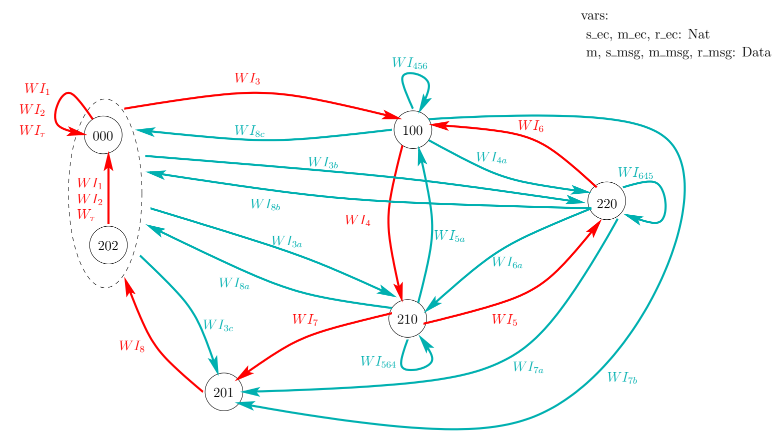



Figure 9: Weak Open Automaton of the implementation

In Figure 9 we recall the weak open automaton of our implementation pNet.

This drawing is based on the observation that states 202 and 000 are only linked by a "pure $\tau$" transition, and have exactly the same possible behaviours. In this configuration we can guarantee that they are Weak bisimilar, and we have merged their (incoming and outgoing) transitions in the Figure. We denote this equivalence class of states as $\{000, 202\}$.

Full details of the weak transitions is listed here:

In the first 3 weak transitions, $S$ denotes the set of all global states.

$$W_\tau = \frac{\{\}, True, ()}{S \overset{\tau}{\Rightarrow} S}$$

$$WI_1 = \frac{\{\mathtt{P}\mapsto\mathtt{p\text{-}a}\}, [\forall \mathtt{x}.\mathtt{p\text{-}a} \neq \mathtt{p\text{-}send(x)}], ()}{S \overset{\mathtt{p\text{-}a}}{\Longrightarrow} S}$$

$$WI_2 = \frac{\{\mathtt{Q}\mapsto\mathtt{q\text{-}b}\}, [\forall \mathtt{x,y}.\mathtt{q\text{-}b} \neq \mathtt{q\text{-}recv(x,y)}], ()}{S \overset{\mathtt{q\text{-}b}}{\Longrightarrow} S}$$

All the following transitions are parameterised by an integer $n \in \mathtt{Nat}$, meaning they stand for the corresponding (infinite) set of weak OTs. In some cases, this set is further restricted (see e.g. $WI_{7b}(n)$), in which cases we have added an explicit quantifier.

$$WI_3(n) = \frac{\{\mathtt{P}\mapsto\mathtt{p\text{-}send(m)}\}, True, (\mathtt{s\_msg} \leftarrow \mathtt{m}, \mathtt{s\_ec} \leftarrow n)}{\{000, 202\} \overset{\mathtt{in(m)}}{\Longrightarrow} 100}$$

$$WI_{3a}(n) = \frac{\{\mathtt{P}\mapsto\mathtt{p\text{-}send(m)}\}, True, (\mathtt{m\_msg} \leftarrow \mathtt{m}, \mathtt{m\_ec} \leftarrow n, \mathtt{s\_ec} \leftarrow n)}{\{000, 202\} \overset{\mathtt{in(m)}}{\Longrightarrow} 210}$$

$$WI_{3b}(n) = \frac{\{\mathtt{P}\mapsto\mathtt{p\text{-}send(m)}\}, True, (\mathtt{s\_ec} \leftarrow n)}{\{000, 202\} \overset{\mathtt{in(m)}}{\Longrightarrow} 220}$$

$$WI_{3c}(n) = \frac{\{\mathtt{P}\mapsto\mathtt{p\text{-}send(m)}\}, True, (\mathtt{r\_msg} \leftarrow \mathtt{m}, \mathtt{r\_ec} \leftarrow n)}{\{000, 202\} \overset{\mathtt{in(m)}}{\Longrightarrow} 201}$$

$$WI_4(n) = \frac{\{\}, True, (\mathtt{m\_msg} \leftarrow \mathtt{s\_msg}, \mathtt{m\_ec} \leftarrow \mathtt{s\_ec} + n, \mathtt{s\_ec} \leftarrow \mathtt{s\_ec} + n)}{100 \overset{\tau}{\Rightarrow} 210}$$

$$WI_{4a}(n) = \frac{\{\}, True, (\mathtt{s\_ec} \leftarrow \mathtt{s\_ec} + n)}{100 \overset{\tau}{\Rightarrow} 220}$$

$$WI_5(n) = \frac{\{\}, True, (\texttt{s\_ec} \leftarrow \texttt{s\_ec} + n)}{210 \overset{\tau}{\Rightarrow} 220}$$

$$WI_{5a}(n) = \frac{\{\}, True, (\texttt{s\_ec} \leftarrow \texttt{s\_ec} + 1 + n)}{210 \overset{\tau}{\Rightarrow} 100}$$

$$WI_6(n) = \frac{\{\}, True, (\texttt{s\_ec} \leftarrow \texttt{s\_ec} + 1 + n)}{220 \overset{\tau}{\Rightarrow} 100}$$

$$WI_{6a}(n) = \frac{\{\}, True, (\texttt{m\_msg} \leftarrow \texttt{s\_msg}, \texttt{m\_ec} \leftarrow \texttt{s\_ec} + 1 + n, \texttt{s\_ec} \leftarrow \texttt{s\_ec} + 1 + n)}{220 \overset{\tau}{\Rightarrow} 210}$$

Because

$$\begin{aligned}
Post_{6a} &= \ post_4 \otimes \ post^*_{456} \otimes \ post_6 \\
&= ((\texttt{m_msg} \leftarrow \texttt{s_msg}, \texttt{m_ec} \leftarrow \texttt{s_ec}) \otimes (\texttt{s_ec} \leftarrow \texttt{s_ec} + n)) \otimes (\texttt{s_ec} \leftarrow \texttt{s_ec} + 1) \\
&= (\texttt{m_msg} \leftarrow \texttt{s_msg}, \texttt{m_ec} \leftarrow (\texttt{s_ec} + 1) + n, \texttt{s_ec} \leftarrow (\texttt{s_ec} + 1) + n)
\end{aligned}$$

$$WI_{456*}(n) = \frac{\{\}, True, (\texttt{s\_ec} \leftarrow \texttt{s\_ec} + n)}{100 \overset{\tau}{\Rightarrow} 100}$$

$$WI_{564*}(n) = \frac{\{\}, True, (\texttt{m\_msg} \leftarrow \texttt{s\_msg}, \texttt{s\_ec} \leftarrow \texttt{s\_ec} + 1 + n, \texttt{m\_ec} \leftarrow \texttt{s\_ec} + 1 + n)}{210 \overset{\tau}{\Rightarrow} 210}$$

$$WI_{645*}(n) = \frac{\{\}, True, (\texttt{s\_ec} \leftarrow \texttt{s\_ec} + 1 + n)}{220 \overset{\tau}{\Rightarrow} 220}$$

$$WI_7(n) = \frac{\{\}, True, (\texttt{r\_msg} \leftarrow \texttt{s\_msg}, \texttt{r\_ec} \leftarrow \texttt{s\_ec} + n)}{210 \overset{\tau}{\Rightarrow} 201}$$

$$WI_{7a}(n) = \frac{\{\}, True, (\texttt{r\_msg} \leftarrow \texttt{s\_msg}, \texttt{r\_ec} \leftarrow \texttt{m\_ec} + n)}{220 \overset{\tau}{\Rightarrow} 201}$$

$$\forall n \geq 1. WI_{7b}(n) = \frac{\{\}, True, (\texttt{r\_msg} \leftarrow \texttt{m\_msg}, \texttt{r\_ec} \leftarrow \texttt{s\_ec} + n)}{100 \overset{\tau}{\Rightarrow} 201}$$

$$WI_8 = \frac{\{\texttt{Q}{\mapsto}\texttt{q-recv(r1-msg,r1-ec)}\}, True, ()}{201 \xRightarrow{\texttt{out(r1-msg,r1-ec)}} \{202, 000\}}$$

$$\forall n \geq 1. WI_{8a}(n) = \frac{\{\texttt{Q}{\mapsto}\texttt{q-recv(m\_msg,m\_ec} + n)\}, True, ()}{210 \xRightarrow{\texttt{out(m\_msg,m\_ec}+n)} \{202, 000\}}$$

$$\forall n \geq 1. WI_{8b}(n) = \frac{\{\texttt{Q}{\mapsto}\texttt{q-recv(??\_msg,s\_ec} + n)\}, True, ()}{220 \xRightarrow{\texttt{out(??\_msg,m\_ec}+n)} \{202, 000\}}$$

$$\forall n \geq 1. WI_{8c}(n) = \frac{\{\texttt{Q}{\mapsto}\texttt{q-recv(s\_msg,s\_ec} + n)\}, True, ()}{100 \xRightarrow{\texttt{out(s\_msg,s\_ec}+n)} \{202, 000\}}$$

Then for all $\tau$ transitions above we have a similar WOT that include a non-$\tau$ move from an external action of P or Q, like for example:

$$WI_4P(n) = \frac{\{\texttt{P}{\mapsto}\texttt{p-a}\}, [\forall \texttt{x.p-a} \neq \texttt{p-send(x)}], (\texttt{m\_msg} \leftarrow \texttt{s\_msg}, \texttt{m\_ec} \leftarrow \texttt{s\_ec} + n, \texttt{s\_ec} \leftarrow \texttt{s\_ec} + n)}{100 \xRightarrow{\texttt{p-a}} 210}$$

and $WI_4Q(n) = \dfrac{\{\texttt{Q}\mapsto\texttt{q-b}\}, [\forall \texttt{x,y}.\texttt{q-b} \neq \texttt{q-recv(x,y)}], (\texttt{m\_msg} \leftarrow \texttt{s\_msg}, \texttt{m\_ec} \leftarrow \texttt{s\_ec} + n, \texttt{s\_ec} \leftarrow \texttt{s\_ec} + n)}{100 \xRightarrow{\texttt{q-b}} 210}$

but also e.g.:

$WI_{456*}P(n) = \dfrac{\{\texttt{P}\mapsto\texttt{p-a}\}, [\forall \texttt{x}.\texttt{p-a} \neq \texttt{p-send(x)}], (\texttt{s\_msg} \leftarrow \texttt{s\_msg}, \texttt{s\_ec} \leftarrow \texttt{s\_ec} + n)}{100 \xRightarrow{\texttt{p-a}} 100}$

The following table give a summary of WOTs, when sharing their names as much as possible.

| WOT name | Pairs of source states and target states | # WOTs |
|---|---|---|
| $WI_1$ $WI_2$ $WI_\tau$ | $\{(s,s) \vert s \in \texttt{States of WOA}\} \cup \{(202,000)\}$ | 21 |
| $WI_3(n)$ | {(202,100),(000,100)} | 2 |
| $WI_{3a}(n)$ | {(202,210),(000,210)} | 2 |
| $WI_{3b}(n)$ | {(202,220),(000,220)} | 2 |
| $WI_{3c}(n)$ | {(202,201),(000,201)} | 2 |
| $WI_4(n)$ $WI_4P(n)$ $WI_4Q(n)$ | {(100,210)} | 3 |
| $WI_{4a}(n)$ $WI_4aP(n)$ $WI_4aQ(n)$ | {(100,220)} | 3 |
| $WI_{456*}(n)$ $WI_{456*}P(n)$ $WI_{456*}Q(n)$ | {(100,100)} | 3 |
| $WI_5(n)$ $WI_5P(n)$ $WI_5Q(n)$ | {(210,220)} | 3 |
| $WI_{5a}(n)$ $WI_{5a}P(n)$ $WI_{5a}Q(n)$ | {(210,100)} | 3 |
| $WI_{564*}(n)$ $WI_{564*}P(n)$ $WI_{564*}Q(n)$ | {(210,210)} | 3 |
| $WI_6(n)$ $WI_6P(n)$ $WI_6Q(n)$ | {(220,100)} | 3 |
| $WI_{6a}(n)$ $WI_{6a}P(n)$ $WI_{6a}Q(n)$ | {(220,210)} | 3 |
| $WI_{645*}(n)$ $WI_{645*}P(n)$ $WI_{645*}Q(n)$ | {(220,220)} | 3 |
| $WI_7(n)$ $WI_7P(n)$ $WI_7Q(n)$ | {(210,201)} | 3 |
| $WI_{7a}(n)$ $WI_{7a}P(n)$ $WI_{7a}Q(n)$ | {(220,201)} | 3 |
| $WI_{7b}(n)$ $WI_{7b}P(n)P$ $WI_{7b}Q(n)$ | {(100,201)} | 3 |
| $WI_8(n)$ | {(201,202),(201,000)} | 2 |
| $WI_{8a}(n)$ | {(210,202),(210,000)} | 2 |
| $WI_{8b}(n)$ | {(220,202),(220,000)} | 2 |
| $WI_{8c}(n)$ | {(100,202),(100,000)} | 2 |

That makes a total of 73 WOTs in the SimpleImpl WOA.

C.1. **Details of the Bisimulation Checking.** We recall here the relation $\mathcal{R}$ that is the candidate for our weak bisimulation relation:

| Spec state | Impl state | Predicate |
|---|---|---|
| b0 | 000 | True |
| b0 | 202 | True |
| b1 | 100 | `b_msg = s_msg` $\wedge$ `b_ec = s_ec` |
| b1 | 210 | `b_msg = m_msg` $\wedge$ `b_ec = m_ec` |
| b1 | 220 | `b_msg = s_msg` $\wedge$ `b_ec = s_ec` |
| b1 | 201 | `b_msg = r_msg` $\wedge$ `b_ec = r_ec` |

Consider the first triple <b0, 000, True>, we have to prove the following 6 properties, in which $OT << WOT$ means that the (strong) open transition $OT$ is covered, in the sense of definition 14, by the weak transition $WOT$ (it could be a set, but this will not be used here):

$$\begin{array}{ll} SS_1 << WI_1 & SI_1 << WS_1 \\ SS_2 << WI_2 & SI_2 << WS_2 \\ SS_3 << WI_3 & SI_3 << WS_3 \end{array}$$

Note: if we were using the alternative Weak bisim from Appendix B.1, Lemma 4, that is checking strong bisimulation between the corresponding weak automaton, we would have a more transitions coverage to examine, as we have 4 weak transitions for $b0$ in the specification weak automaton, and 7 WOTs (including 4 parameterised WOTs) from 000 in the implementation automaton.

Preliminary remarks:

- Both pNets trivialy verify the "non-observability" condition: the only vectors having $\tau$ as an action of a sub-net are of the form "$< -, \tau, - > - > \tau$".
- We must take care of variable name conflicts: in our example, the variables of the 2 systems already have different names, but the action parameters occurring in the transitions (m, msg, ec) are the same, that is not correct. In the tools, this is managed by the static semantic layer; in the following example, we have renamed all conflicting variables using suffix 1 for the Specification, and 2 for the Implementation.

In our running example in page 25, we have shown the proof for one of the transitions of $(\mathrm{b0}, 202, True)$, namely that $SS_3$ is covered by $WI_3(0)$. We give here another example with $SS_1 << WI_1$, from the first triple $(\mathrm{b0}, 000, True)$. It includes less trivial predicates in the OTs:

$$SS_1 = \frac{\{\mathtt{P}\mapsto\mathtt{p\text{-}a1}\}, [\forall \mathtt{m1}.\mathtt{p\text{-}a1} \neq \mathtt{p\text{-}send(m1)}], ()}{\mathrm{b0} \xrightarrow{\mathtt{p\text{-}a1}} \mathrm{b0}}$$

$$WI_1 = \frac{\{\mathtt{P}\mapsto\mathtt{p\text{-}a2}\}, [\forall \mathtt{m2}.\mathtt{p\text{-}a2} \neq \mathtt{p\text{-}send(m2)}], ()}{000 \xRightarrow{\mathtt{p\text{-}a2}} 000}$$

Let us check formally the conditions:

- Their sets of active (non-silent) holes is the same: $J' = J_x = \{\mathtt{P}\}$.
- Triple $(\mathrm{b0}, 000, True)$ is in $\mathcal{R}$.
- The verification condition
$$\forall fv_{OT}.\{Pred \wedge Pred_{OT}$$
$$\Longrightarrow \bigvee_{x\in X} \Big[\exists fv_{OT_x}.\,\Big(\forall j \in J_x.(\beta_j)^{\nabla} = \gamma_{jx} \wedge Pred_{OT_x} \wedge \alpha = \alpha_x \wedge Pred_{s',t_x}\{\!\{Post_{OT} \uplus Post_{OT_x}\}\!\}\Big)\Big]\}$$

Gives us:

$$\forall \mathtt{p\text{-}a1}.\{True \wedge \forall \mathtt{m1}.\mathtt{p\text{-}a1} \neq \mathtt{p\text{-}send(m1)}$$
$$\Longrightarrow \exists \mathtt{p\text{-}a2}.(\mathtt{p\text{-}a1} = \mathtt{p\text{-}a2} \wedge \forall \mathtt{m2}.\mathtt{p\text{-}a2} \neq \mathtt{p\text{-}send(m2)} \wedge \mathtt{p\text{-}a1} = \mathtt{p\text{-}a2} \wedge True\}$$

That is trivially true, choosing `p-a2=p-a1` for each given `p-a1`.

All others pairs from this set are just as easily proven true.



This work is licensed under the Creative Commons Attribution License. To view a copy of this license, visit https://creativecommons.org/licenses/by/4.0/ or send a letter to Creative Commons, 171 Second St, Suite 300, San Francisco, CA 94105, USA, or Eisenacher Strasse 2, 10777 Berlin, Germany