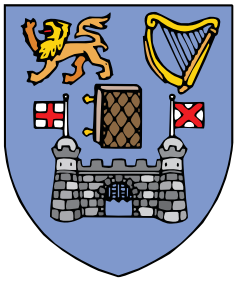


TRINITY COLLEGE DUBLIN
COLÁISTE NA TRÍONÓIDE, BAILE ÁTHA CLIATH

A simple probabilistic broadcast language

Andrea Cerone and Matthew Hennessy

Computer Science Department Technical Report **TCD-CS-2012-02**
Foundations and Methods Research Group **February 3, 2012**

A SIMPLE PROBABILISTIC BROADCAST LANGUAGE

ANDREA CERONE AND MATTHEW HENNESSY

e-mail address: ceronea@cs.tcd.ie, matthew.hennessy@cs.tcd.ie

Abstract. We propose a process calculus to model high level wireless systems, where the topology of a network is described by a graph. The calculus enjoys features which are proper of wireless networks, namely broadcast communication and probabilistic behaviour.

We first focus on the problem of composing wireless networks, then we present a compositional theory based on a generalisation of the well known may-testing preorder. Also, we define an extensional semantics for our calculus, and the respective simulation preorder induced. We prove that our simulation preorder is sound with respect to the may testing preorder, thus providing a proof method for establishing whether two networks cannot be distinguished by any test.
However, we also provide a counterexample showing that completeness of the simulation preorder, with respect to the may testing one, does not hold. We conclude the paper with an application of our theory to routing protocols.

Contents

The financial support of SFI is gratefully acknowledged.

1. Introduction

Wireless networks have widespread worldwide in the last decades; nowadays they are used in many areas, from domestic use to mobile phone networks, to the more new sensor networks. Further, in the last few years there has been a growing interest in the development of formal methods for the analysis of wireless systems [7, 8, 10].

While different process calculi have been developed to describe wireless systems, each of them uses local broadcast as the only way of communication; roughly speaking, locations (or nodes) are introduced to model the topology of a network. Processes are assigned to locations, and rules are defined to establish which locations are affected when one message is sent from another one. The way such rules are defined is different according to the nature of the calculus. Different approaches have been developed, for example by using *metric spaces* [8], using allocation environments [10] or semantic tags for locations [7].

In this report we propose a different process calculus for modeling wireless systems; in our framework, the topology of a network is described by an undirected graph. Intuitively, vertices of the graph represent nodes, while edges establish the capability of their endpoints to communicate each other. A probabilistic process calculus is then introduced to assign code to locations. The need for probabilistic behaviour arise for there has been a growing interest in the research of probabilistic protocols, especially in the field of wireless networks; see for example [17, 1, 13].

We remark that the mapping that associates processes to locations is partial, meaning that some location could have no code assigned. Nodes with no running code (or external nodes) will play an important role to develop a compositional theory of wireless systems; intuitively, in our framework networks can be composed each other by merging the graphs that describe their topology, while the code that is assigned to nodes of the merged graph is preserved by that of the original networks. As we will point out throughout the paper, the composition of two networks will not always be defined, as conflict situations can arise (for example one in which both networks have code running at the same location).

In our compositional framework, external nodes can be seen as nodes where code can be placed to test the behaviour of a network. This approach leads to the definition of a probabilistic generalisation of the *Hennessy-De Nicola*'s may testing preorder, whose theory has been developed in [2] for a probabilistic version of *CSP*. Here the authors show that, in the probabilistic setting, the may-testing preorder coincides with the simulation preorder.

In the same style, we develop a compositional theory for the may-testing preorder for probabilistic networks. However, the presence of local broadcast in our calculus creates some difficulties when providing a proof methodology for the may-testing preorder by using simulation preorder.

The main problem arise, roughly speaking, because the broadcast of a message to a set of nodes can be simulated by a multicast of more copies of the same message, which will be detected by exactly the same set of nodes. To solve this problem, much of the theory has to be redefined; specifically, in our framework we will use a non-standard concept of weak actions to define the simulation preorder. This preorder is shown to be sound with respect to the may-testing. However it fails to be complete, rather surprisingly, in view of the completeness result in [2].

The rest of the paper is organised as follows: in Section 2 we will give a brief introduction to the mathematical tools needed for the development and the analysis of systems which present both probabilistic and non-deterministic behaviour. In Section 3 we present our calculus for wireless systems, and we provide some simple examples of networks which it can model.
In Section 4 we develop the testing preorder for our calculus. This is done by addressing several topics; first, we discuss the concept of network composition in Section 4.1. The compositional

theory developed is then used in Section 4.2 to define our testing framework, while the behavioural preorder it induces is defined in Section 4.3.
In Section 5 we present another behavioural preorder for comparing networks, based on simulations over actions which can be detected by external nodes. To this end, we first define an extensional semantics for wireless networks in Section 5.1, then we introduce the formal definition of simulation preorder at the end of the same Section.

Finally, in section 5.2 we prove that the simulation preorder is included in the may testing one. However, this inclusion turns out to be strict; in Section 5.4 we provide an example of two networks which can be related via the may testing preorder, but for which no simulation relation can be exhibited.

Despite our proof methodology fails to be complete, it is powerful enough to analyse real world situations; we provide an application of our results by performing an analysis of routing models in Section 6.
Finally we review some related work and discuss future directions of research in Section 7.

2. Background

In this Section we will summarise the mathematical concepts, taken from [2], that will be needed throughout the paper. First we will introduce some basic concepts from probability theory; then we will show how these can be used to model concurrent systems which exhibit both probabilistic and non-deterministic behaviour.

Let S be a set; a function $\Delta : S \to [0,1]$ is called a (probability) sub-distribution over S if $\sum_{s\in S} \Delta(s) \leq 1$. This quantity, $\sum_{s\in S} \Delta(s)$, is called the mass of the sub-distribution, denoted as $|\Delta|$. If $|\Delta| = 1$, then we say that Δ is a (full) distribution. The support of a distribution Δ, denoted $\lceil\Delta\rceil$, is the subset of S consisting of all those elements which contribute to its mass, namely $\lceil\Delta\rceil = \{s \in S \mid \Delta(s) > 0\}$.

For each $s \in S$, the point distribution $\overline{s}$ is defined to be the distribution which takes value 1 at s, and 0 elsewhere. The set of sub-distributions and distributions over a set S are denoted by $\mathcal{D}_{sub}(S)$ and $\mathcal{D}(S)$, respectively.

Given a family of sub-distributions $\{\Delta_k \mid k \in K\}$, $\sum_{k\in K} \Delta_k$ is the partial real-valued function in $S \to \mathbb{R}$ defined by $(\sum_{k\in K} \Delta_k)(s) := \sum_{k\in K} \Delta_k(s)$. This is a partial operation because for a given $s \in S$ this sum might not exist; it is also a partial operation on sub-distributions because even if the sum does exist it may be greater than 1.

Similarly, if $p \leq 1$ and Δ is a sub-distribution , then $p \cdot \Delta$ is the sub-distribution over S such that

$$(p \cdot \Delta)(s) = p \cdot \Delta(s).$$

It is not difficult to show that if $\{p_k\}_{k\in K}$ is a sequence of positive real numbers such that $\sum_{k\in K} p_k \leq 1$, and $\{\Delta_k\}_{k\in K}$ is a family of sub-distributions over a set S, then $\sum_{i=1}^{n} p_i \cdot \Delta_i$ always defines a sub-distribution over S.

Finally, if $f : X \to Y$ and Δ is a sub-distribution over X then we use $f(\Delta)$ to be the sub-distribution over Y defined by:

$$f(\Delta)(y) = \sum_{x\in X} \{\, \Delta(x) \mid f(x) = y \,\}. \tag{2.1}$$

This definition can be generalised to two arguments functions; if $f : X_1 \times X_2 \to Y$ is a function, and Δ, Θ are two sub-distributions respectively over X_1 and X_2, then $f(\Delta, \Theta)$ denotes the sub-distribution

over Y defined as

$$f(\Delta, \Theta)(y) = \sum_{x_1 \in X_1, x_2 \in X_2} \{\, \Delta(x_1) \cdot \Theta(x_2) \mid f(x_1, x_2) = y \,\}. \tag{2.2}$$

Now we turn our attention to probabilistic concurrent systems. The formal model we use to represent them is a generalisation to a probabilistic setting of Labelled Transition Systems (LTSs) [9].

Definition 2.1. A *probabilistic labelled transition system* (pLTS) is a 4-tuple $\langle S, \mathsf{Act}_\tau, \rightarrow, \omega \rangle$, where

(i) S is a set of states,
(ii) Act_τ is a set of transition labels with a distinguished label τ,
(iii) the relation $\rightarrow$ is a subset of $S \times \mathsf{Act}_\tau \times \mathcal{D}(S)$,
(iv) $\omega : S \mapsto \{ \text{ true }, \text{ false } \}$ is a (success) predicate over the states S.

As usual, we will write $s \xrightarrow{\mu} \Delta$ in lieu of $(s, \alpha, \Delta) \in \longrightarrow$. □

Before discussing pLTSs, some definitions first: a pLTS whose state space is finite is said to be finite state; further, we say that a pLTS $\langle S, \mathsf{Act}_\tau, \rightarrow, \omega \rangle$ is finite branching if, for every $s \in S$, the set $\{\, \Delta \mid s \xrightarrow{\mu} \Delta \text{ for some } \mu \in \mathsf{Act}_\tau \,\}$ is finite. Finally, a finitary pLTS is one which is both finite state and finite branching.

We have included in the definition of a pLTS a success predicate ω over states, which will be used when testing processes. Apart from this, the only difference between LTSs and pLTSs is given by the definition of the transition relation; in the latter this is defined to be a relation (parametric in some action μ) between states and distribution of states, thus capturing the concept of probabilistic behaviour.

However, this modification introduces some difficulties when sequences of transitions performed by a given pLTS have to be considered, as the domain and the image of the transition relation do not coincide. To avoid this problem, we will focus only on distributions of states by defining transitions between distributions of states. The following Definition serves to this purpose:

Definition 2.2 (Lifted Relations)**.** Let $\mathcal{R} \subseteq S \times \mathcal{D}_{sub}(S)$ be a relation from states to subdistributions. Then $\overline{R} \subseteq \mathcal{D}_{sub}(S) \times \mathcal{D}_{sub}(S)$ is the smallest relation which satisfies

- $s \mathcal{R} \Delta$ implies $\overline{s}\ \overline{\mathcal{R}}\ \Delta$
- If I is a finite index set and $\Delta_i\ \overline{\mathcal{R}}\ \Theta_i$ for each $i \in I$ then $(\sum_{i \in I} p_i \cdot \Delta_i)\ \mathcal{R}\ (\sum_{i \in I} p_i \cdot \Theta_i)$ whenever $\sum_{i \in I} p_i \leq 1$. □

Lifting of relations can also be defined for probability distributions, by simply requiring $\sum_{i \in I} p_i = 1$ in the last constraint of the definition above.

In a pLTS $\langle S, \mathsf{Act}_\tau, \rightarrow, \omega \rangle$, each transition relation $\xrightarrow{\mu} \subseteq S \times \mathcal{D}(S)$ can be lifted to $\overline{(\xrightarrow{\mu})} \subseteq \mathcal{D}(S) \times \mathcal{D}(S)$. With an abuse of notation, the latter will still be denoted as $\xrightarrow{\mu}$.

Lifted transition relations allow us to reason about the behaviour of pLTSs in terms of sequences of transitions; here we are mainly interested in the behaviour of a pLTS in the long run; that is, given a pLTS $\langle S, \mathsf{Act}_\tau, \rightarrow, \omega \rangle$ and a distribution $\Delta \subseteq \mathcal{D}(S)$, we are interested in distributions $\Theta \subseteq \mathcal{D}(S)$ which can be reached by Δ after an indefinite number of transitions.

For the moment we will focus only on internal actions of a pLTS, in which case the behaviour of a pLTS in the long run is captured by the concept of hyper-derivation:

Definition 2.3. [Hyper-derivations] In a pLTS a hyper-derivation consists of a collection of sub-distributions $\Delta, \Delta_k^{\rightarrow}, \Delta_k^{\times}$, for $k \geq 0$, with the following properties:

$$\begin{array}{lcccl} \Delta & = & \Delta_0^{\rightarrow} & + & \Delta_0^{\times} \\ \Delta_0^{\rightarrow} & \xrightarrow{\tau} & \Delta_1^{\rightarrow} & + & \Delta_1^{\times} \\ & \vdots & & & \\ \Delta_k^{\rightarrow} & \xrightarrow{\tau} & \Delta_{k+1}^{\rightarrow} & + & \Delta_{k+1}^{\times} \\ & \vdots & & & \end{array}$$

If $\omega(s) =$ false for each $s \in \lceil \Delta_k^{\rightarrow} \rceil$ and $k \geq 0$ we call $\Delta' = \sum_{k=0}^{\infty} \Delta_k^{\times}$ a *hyper-derivative* of Δ, and write $\Delta \Longrightarrow \Delta'$. □

Hyper-derivations can be viewed as the probabilistic counterpart of the weak $\stackrel{\tau}{\Longrightarrow}$ action in LTSs; see [2] for a detailed discussion. Intuitively speaking, they represent fragments of computations obtained by performing only internal actions. The last constraint in Definition 2.3 is needed since we introduced a success predicate in our model; we require that a computation cannot proceed in the case that a state s such that $\omega(s) =$ true has been reached; this is for we are only interested in detecting if such states can be reached in a computation. States in which the predicate $\omega(\cdot)$ is true are called ω-successful.

Further, we are mainly interested in maximal computations of distributions. That is, we require a computation to proceed as long as some internal activity can be performed. To this end, we say that $\Delta \Longrightarrow\!\!\!\!\gg \Delta'$ if

- $\Delta \Longrightarrow \Delta'$,
- for every $s \in \lceil \Delta_k^{\times} \rceil$, $s \xrightarrow{\tau}$ implies $\omega(s) =$ true. □

This is a mild generalisation of the notion of *extreme derivative* from [2]. Note that the last constraint models exactly the requirement of performing some internal activity whenever it is possible; In other words extreme derivatives correspond to a probabilistic version of maximal computations.

Theorem 2.4. In an arbitrary pLTS

(1) $\Longrightarrow$ is reflexive and transitive
(2) if $\Delta \Longrightarrow \Delta'$ and $\Delta' \Longrightarrow\!\!\!\!\gg \Delta''$, then $\Delta \Longrightarrow\!\!\!\!\gg \Delta''$; this is a direct consequence of the previous statement, and the definition of extreme derivatives
(3) suppose $\Delta = \sum_{i \in I} p_i \cdot \Delta_i$, where I is an index set and $\sum_{i \in I} p_i \leq 1$. If for any $i \in I, \Delta_i \Longrightarrow \Theta_i$ for some Θ_i, then $\Delta \Longrightarrow \Theta$, where $\Theta = \sum_{i \in I} p_i \cdot \Theta_i$.
(4) for all distributions Δ, there exists a sub-distribution Θ such that $\Delta \Longrightarrow\!\!\!\!\gg \Theta$

Proof. See [2] for detailed proofs. □

3. Networks and their computations

The calculus we present is designed to model broadcast systems, particularly wireless networks, at a high level. We do not deal with low level issues, such as collisions of broadcast messages or multiplexing mechanisms [19]; instead, we assume that network nodes use protocols at the *MAC level* [6] to achieve both perfect and dedicated communication between nodes.

Basically, the language will contain both primitives for sending and receiving messages and will enjoy the following features:

$M,\ N$	::=		**Systems**
		$n[\![s]\!]$	Nodes
		$M \mid N$	Composition
		$\mathbf{0}$	Identity
$p,\ q$	::=		(probabilistic) Processes
		s	
		$p \,{}_p\!\oplus\, q$	probabilistic choice
$s,\ t$	::=		States
		$c!\langle e\rangle.p$	broadcast
		$c?(x).p$	receive
		$\omega.\mathbf{0}$	test
		$s + t$	choice
		if b then s else t	branch
		$\tau.p$	preemption
		$A(\tilde{x})$	definitions
		$\mathbf{0}$	terminate

Figure 1: Syntax

(i) communication can be obtained through the use of different channels; although the physical medium for messages exchange in wireless networks is unique, it is reasonable to assume that network nodes use some multiple access technique, such as *TDMA* or *FDMA* [19], to setup and communicate through virtual channels,

(ii) communication is broadcast; whenever a node of a given network sends a message, it will be detected by all nodes in its range,

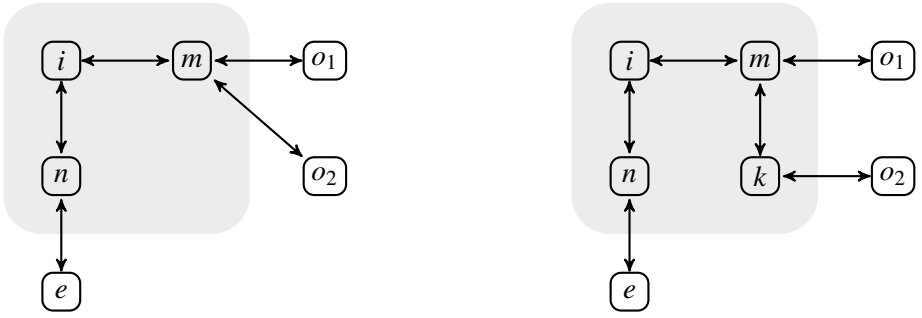
(iii) communication is perfect: whenever a node broadcasts a message and a neighbouring node (that is, a node in the sender's range) is waiting to receive a message on the same channel, then the message will be delivered to the receiver. This is not ensured if low level issues are considered, as problems such as message collisions [6] and nodes synchronisation [15] arise .

The language for system terms, ranged over by M, N is given in Figure 1. Basically a system consists of a collection of named nodes at each of which there is some running code. The syntax for this code a fairly straightforward instance of a standard process calculus, augmented by a probabilistic choice; code descriptions have the usual constructs for channel based comunication, with input $c?(x).p$ being the unique binder.

We only consider the sub-language of well-formed system terms in which all node names have at most one occurrence. We use sSys to range over all closed well-formed terms. A (well formed) system term can be viewed as a mapping that assigns to node names the code they are executing. A subterm $n[\![s]\!]$ appearing in a system term M represents node n running code s.

Additional information such as the connectivity between nodes of a network is needed to formalise communications between nodes, as well as the probabilistic behaviour of systems. Network connectivity is represented by a graph $\Gamma = \langle \Gamma_V, \Gamma_E \rangle$; here Γ_V is a finite set of nodes and $\Gamma_E \subseteq (\Gamma_V \times \Gamma_V)$ satisfying

(i) (symmetry) $(n, m) \in \Gamma_E$ implies $(m, n) \in \Gamma_E$,



$\mathcal{M} = \Gamma_M \rhd n[\![A_n]\!] \mid i[\![A_i]\!] \mid m[\![A_m]\!]$ $\quad$ $\mathcal{N} = \Gamma_N \rhd n[\![A_n]\!] \mid i[\![A_i]\!] \mid m[\![A_k]\!] \mid k[\![A_m]\!]$

Figure 2: Example networks

(ii) (irreflexive) $(n, m) \in \Gamma_E$ implies $n \neq m$.

Symmetry is needed to model the fact that all nodes have the same transmission range; that is, if a broadcast from node m can be detected by another node n, then the opposite will also hold. We use the more graphic notation $\Gamma \vdash m$ to mean $v \in \Gamma_V$ and $\Gamma \vdash n \leftrightarrow m$ for $(m, n) \in \Gamma_E$. Intuitively $\Gamma \vdash n \leftrightarrow m$ means that messages broadcast from node n can be received by node m, and vice-versa.

A *network* consists of a pair $(\Gamma \rhd M)$, representing the system M, from sSys, executing relative to the connectivity graph Γ. All nodes occurring in M, nodes(M), will appear in Γ and the effect of running the code at $n \in \text{nodes}(M)$ will depend on the connectivity of n in Γ. But in general there will be nodes in Γ which do not occur in M; let $\mathsf{Int}(\Gamma \rhd M) = \Gamma_V \setminus \text{nodes}(M)$; we call this set the *interface* of the network $\Gamma \rhd M$, and its elements are called *external nodes*. Intuitively these are nodes which may be used to compose the network $\Gamma \rhd M$ with other networks, or to place code for testing the behaviour of M.

However, there can be situations where a network $\Gamma \rhd M$ contains unnecessary information to model the behaviour of a network, or the network connectivity and the system term are related in a way which is inconsistent with our concept of wireless networks. Therefore, we place some requirements on the structure of a network.

Definition 3.1. The network $\Gamma \rhd M$ is well-formed if:

(i) $\text{nodes}(M) \subseteq \Gamma_V$
(ii) $M \in \mathsf{sSys}$
(iii) whenever $k \in \mathsf{Int}(\Gamma \rhd M)$, there exists some $m \in \text{nodes}(M)$ such that $\Gamma \vdash k \leftrightarrow m$
(iv) whenever $k_1, k_2 \in \mathsf{Int}(\Gamma \rhd M)$, $\Gamma \vdash k_1 \nleftrightarrow k_2$. □

Condition (iii) is a natural sanity requirement, as there is no point in having an unconnected node in the interface of a network. As we will see, in Example 5.8, requirement (iv) is needed to ensure the soundness of our proof technique. We use Nets to denote the set of well-formed networks, and in the sequel we will assume that a network is well formed, unless otherwise stated. We will sometimes use $\mathcal{M}$, $\mathcal{N}$ etc. to range over arbitrary (well-formed) networks, and apply operations such as nodes($\mathcal{M}$) in the obvious manner.

Example 3.2. Consider $\mathcal{M}$ described in Figure 2. There are six nodes, three occupied by code n, i and m, and three in the interface $\mathsf{Int}(\mathcal{M})$, e, o_1 and o_2. Here, and in future example, we differentiate between the interface and the occupied nodes using shading. Suppose the code at nodes are given

by

$$A_n \Leftarrow c?(x).d!\langle x\rangle.\mathbf{0} \qquad A_i \Leftarrow d?(x).d!\langle f(x)\rangle.\mathbf{0} \qquad A_m \Leftarrow d?(x).(d!\langle x\rangle.\mathbf{0} \;{}_{0.8}\oplus\; \mathbf{0})$$

Then $\mathcal{M}$ can receive input from node e at its interface along the channel c; this is passed on to the internal node i using channel d, where it is transformed in some way, described by the function f, and then forwarded to node m, where 80% of the time it is broadcast to the external nodes o_1 and o_2. The remainder of the time the message is lost.

The network $\mathcal{N}$ has the same interface as $\mathcal{M}$, but has an extra internal node k connected to o_2, and m is only connected to one interface node o_1 and the internal node k. The nodes i and n have the same code running as in $\mathcal{M}$, while nodes m and k will run the code

$$A_k \Leftarrow d?(x).(d!\langle x\rangle.\mathbf{0} \,{}_{0.9}\oplus\, \mathbf{0})$$

Intuitively, the behaviour of $\mathcal{N}$ is more complex than that of $\mathcal{M}$; indeed, there is the possibility for a computation of $\mathcal{N}$ to deliver a value only to one between the external nodes o_1 and o_2, while this is not possible in $\mathcal{N}$. However, 81% of the times this message will be delivered to both these nodes, and thus it is more reliable than $\mathcal{M}$. Suppose now that we change the code at the intermediate code m in $\mathcal{M}$,

$$\mathcal{M}_1 = \Gamma_M \rhd \ldots \mid m[\![B_m]\!] \qquad \text{where } B_m \Leftarrow d?(x).(\tau.(d!\langle x\rangle.\mathbf{0} \,{}_{0.5}\oplus\, \mathbf{0}) + \tau.d!\langle x\rangle.\mathbf{0})$$

In $\mathcal{M}_1$ the behaviour at the node m is non-deterministic; it may act like a perfect forwarder, or one which is only 50% reliable. Optimistically it could be more reliable than $\mathcal{M}$, or pessimistically it could be less reliable than the latter. Further, there is no possibility for the network $\mathcal{M}_1$ to forward the message to only one of the external nodes o_1, o_2, so that its behaviour is somewhat less complex than that of $\mathcal{N}$.

As a further variation let $\mathcal{M}_2$ be the result of replacing the code at m with

$$C_m \Leftarrow d?(x).D$$
$$D \Leftarrow \tau.(d!\langle x\rangle.\mathbf{0} \;{}_{0.5}\oplus\; \tau.D)$$

Here the behaviour is once more deterministic, with the probability that the message will be eventually transmitted successfully through node k approaching 1 in the limit. Thus, this network is as reliable as $\mathcal{M}_1$, when the latter is viewed optimistically. □

We now turn our attention on the operational semantics of networks. Following [3, 2], processes will be interpreted as probability distributions of states; such an interpretation is encoded by the function $[\![\cdot]\!]$ defined below:

$$\begin{aligned} [\![s]\!] &= \overline{s} \\ [\![p_1 \,{}_p\oplus\, p_2]\!] &= p \cdot [\![p_1]\!] + (1 - p \cdot [\![p_2]\!]). \end{aligned}$$

Judgements in the intensional semantics of networks take the form

$$\Gamma \rhd M \xrightarrow{\mu} \Delta$$

where Γ is a network connectivity, M is a (well formed) system from sSys, and Δ is a distribution over sSys; intuitively this means that relative to the connectivity Γ the system M can perform the action μ, and with probability $\Delta(N)$ be transformed into the system N, for every $N \in \lceil\Delta\rceil$. The action labels can take the form

(i) receive, $c.n?v$, where $n \in \Gamma_V$: meaning that the value v is detected on channel c by all nodes in nodes(M) which are reachable from n in Γ

$$\text{(B-BROAD)}\quad \frac{s \xrightarrow{c!v} p}{\Gamma \rhd n[\![s]\!] \xrightarrow{c.n!v} n[\![\Delta]\!]}\,[\![p]\!] = \Delta \qquad \text{(B-REC)}\quad \frac{s \xrightarrow{c?v} p}{\Gamma \rhd n[\![s]\!] \xrightarrow{c.m?v} n[\![\Delta]\!]}\,[\![p]\!] = \Delta, \Gamma \vdash n \leftrightarrow m$$

$$\text{(B-DEAF)}\quad \frac{s \overset{c?v}{\not\longrightarrow}}{\Gamma \rhd n[\![s]\!] \xrightarrow{c.m?v} \overline{n[\![s]\!]}}\,\Gamma \vdash m \leftrightarrow n \qquad \text{(B-DISC)}\quad \frac{}{\Gamma \rhd n[\![s]\!] \xrightarrow{c.m?v} \overline{n[\![s]\!]}}\,\Gamma \vdash n \nleftrightarrow m$$

$$\text{(B-}\mathbf{0}\text{)}\quad \frac{}{\mathbf{0} \xrightarrow{c.m?v} \overline{\mathbf{0}}}$$

$$\text{(B-}\tau\text{)}\quad \frac{s \xrightarrow{\tau} p}{\Gamma \rhd n[\![s]\!] \xrightarrow{n.\tau} n[\![\Delta]\!]}\,[\![p]\!] = \Delta \qquad \text{(B-}\tau\text{.PROP)}\quad \frac{\Gamma \rhd M \xrightarrow{n.\tau} \Delta}{\Gamma \rhd M \mid N \xrightarrow{n.\tau} \Delta \mid \overline{N}}$$

$$\text{(B-PROP)}\quad \frac{\Gamma \rhd M \xrightarrow{c.m?v} \Delta,\ \Gamma \rhd N \xrightarrow{c.m?v} \Theta}{\Gamma \rhd M \mid N \xrightarrow{c.m?v} \Delta \mid \Theta} \qquad \text{(B-SYNC)}\quad \frac{\Gamma \rhd M \xrightarrow{c.m!v} \Delta,\ \Gamma \rhd N \xrightarrow{c.m?v} \Theta}{\Gamma \rhd M \mid N \xrightarrow{c.m!v} \Delta \mid \Theta}$$

Figure 3: Intensional semantics of networks

(ii) broadcast, $c.n!v$: meaning the node n (occurring in nodes(M), and therefore in Γ) broadcasts the value v on channel c to all nodes directly connected to n in Γ

(iii) internal activity, $n.\tau$, meaning either some internal (housekeeping) or preempting activity performed by node n.

The rules for inferring judgements are given in Figure 3; they rely on the pre-semantics for the code, which is discussed presently. Thus Rule (B-BROAD) models the capability for a node to broadcast a value v through channel c, assuming the code running there is capable of broadcasting along c. Here the term $n[\![\Delta]\!]$ represents a distribution over sSys, obtained by a direct application of Equation (2.1) to the function $n[\![\cdot]\!]$ which maps states into system terms. The distribution Δ is in turn obtained from the residual of the state s after the broadcast action.

Example 3.3. Consider the simple network $\Gamma \rhd n[\![s]\!]$ where the code s has the form $c!\langle v\rangle.(s_1 \,{}_{\frac{1}{4}}\!\oplus s_2)$.

As we will see the pre-semantics of states determines that $s \xrightarrow{c!v} (s_1 \,{}_{\frac{1}{4}}\!\oplus s_2)$; also $[\![(s_1 \,{}_{\frac{1}{4}}\!\oplus s_2)]\!]$ is the distribution $\frac{1}{4} \cdot \overline{s_1} + \frac{3}{4} \cdot \overline{s_2}$. Thus according to the rule (B-BROAD) we have the judgement

$$\Gamma \rhd n[\![s]\!] \xrightarrow{c.n!v} \frac{1}{4} \cdot n[\![s_1]\!] + \frac{3}{4} \cdot n[\![s_2]\!]$$

□

Rules (B-REC), (B-DEAF) and (B-DISC) express how a node reacts when a message is broadcast; the first essentially models the capability of a node which is listening to a channel c, and which appears in the sender's range of transmission, to receive the message correctly. The other two rules

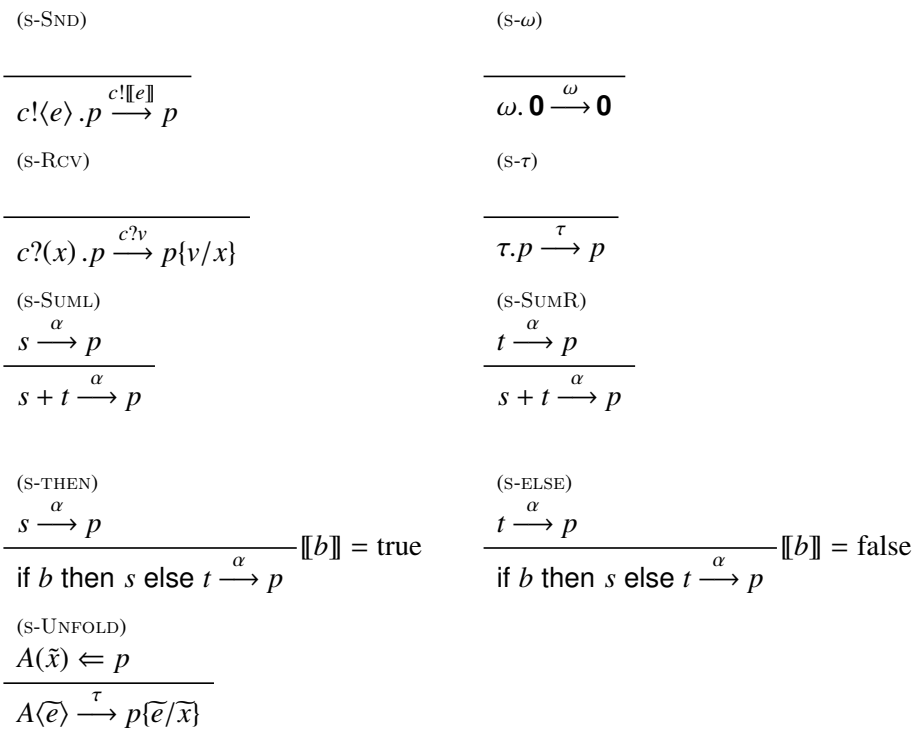


Figure 4: Pre-semantics of states

model situations in which a node is not listening to the channel used to broadcast a message, or it is not in the range of the sender; In both these cases this node cannot detect the transmission at all.

The rules (B-τ) and (B-τ.PROP) model internal activities performed by some node of a system term; the latter expresses the inability for a node which performs an internal activity to affect other nodes in a system term. Here again, $\Delta \mid \Theta$ is a distribution over sSys, this time obtained by instantiating Equation (2.2) to the function $(\cdot \mid \cdot) : \mathsf{sSys} \times \mathsf{sSys} \to \mathsf{sSys}$.

Finally, rules (B-SYNC) and (B-PROP) describe how communication between nodes of a network is handled; notice that if a system term M performs a broadcast action of the form $c.n!v$, while a second system term N receives such a value by performing a $c.n?v$ action, the action performed by the composed system $M \mid N$ will still be $c.n!v$. Informally speaking, this means that the output action performed by node m will be available to other nodes, thus implementing the behaviour of broadcast communication.

We should emphasise that we do not require a network to be well formed in order to infer judgements for it; in practice it is always the case that, when inferring a judgement of the form $\Gamma \rhd (M|N) \xrightarrow{\mu} \Delta$, the former being a well formed network, it will be required first to infer judgements for the networks $\Gamma \rhd M$ and $\Gamma \rhd N$, which in general are not well formed.

The pre-semantics for states takes the form

$$s \xrightarrow{\mu} p$$

where s is a closed state, that is containing no free occurrence of a variable, p is a process and μ can take one of the forms $c!v$, $c?v$ or τ. The deductive rules for inferring these judgements are

given in Figure 4 and should be self-explanatory. It assumes some mechanism for evaluating closed data-expressions e to values $\mathrm{val}(e)$.

State $c!\langle e\rangle.p$ performs the $c!v$ action, where $v = \mathrm{val}(e)$, before evolving in process p. Note that the latter is not necessarily a state, as it can contain the probabilistic choice operator.

The state $c?(x).p$ can receive an arbitrary value v, then evolving in the process obtained by substituting all the free occurrences of variable x in p with v.

A state of the form $\tau.p$ will perform some internal activity and then will proceed as p. Standard rules for nondeterministic choice, matching and guarded recursion are also given. Finally the language includes code of the form $\omega.\mathbf{0}$ where ω is a special action, used to indicate the success of tests.

We conclude this Section by performing some sanity checks for the operational semantics of networks:

(1) A distribution Δ over sSys is called *(node)-stable* if $\Delta(N) > 0$ and $\Delta(M) > 0$ implies $\mathrm{nodes}(N) = \mathrm{nodes}(M)$.

If $\Gamma \rhd M \xrightarrow{\mu} \Delta$ can be deduced from the rules in Figure 3 then Δ is a stable distribution. Intuitively this means that probabilistic choices are only ever made at the level of processes, not systems.

(2) $\Gamma \rhd M \xrightarrow{c.m?v} \Delta$ implies
   (a) $m \notin \mathrm{nodes}(M)$,
   (b) for every value w there exists some Δ' such that $\Gamma \rhd M \xrightarrow{c.m?w} \Delta'$,
   (c) if $\Gamma \nvdash m$, then Δ is $\overline{M}$.

(3) $\Gamma \rhd M \xrightarrow{\mu} \Delta$, with μ being equal either to $c.m!v$ or $m.\tau$, implies $m \in \mathrm{nodes}(M)$ and therefore $\Gamma \vdash m$.

4. Testing networks

Here we develop a preorder

$$(\Gamma_M \rhd M) \sqsubseteq_{behav} (\Gamma_N \rhd N)$$

Intuitively this means that the network $(\Gamma_M \rhd M)$ can be replaced by $(\Gamma_N \rhd N)$, as a part of a larger overall network, without any loss of behaviour. Note that we will be able to compare networks with different nodes and different connectivity graphs.

To formalise this concept we need to say how networks are composed to form larger networks, the topic of Section 4.1, and then say how behaviour is determined, explained in Section 4.2. The formal definition of the preorder is given in Section 4.3, together with examples.

4.1. Composing networks.

Here we discuss how networks are to be combined to form a composite network. The general form of the definition is as follows:

Definition 4.1. [Composing networks] Let iscons be a partial binary predicate on networks in Nets, intuitively saying that they can be safely combined to obtain a well-formed net. Then we define the associated partial composition relation by:

$$(\Gamma_M \rhd M) \;|||_{\mathrm{iscons}}\; (\Gamma_N \rhd N) = \begin{cases} (\Gamma_M \cup \Gamma_N) \rhd M \mid N, & \text{if } (\Gamma_M \rhd M)\ \mathsf{iscons}\ (\Gamma_N \rhd N) \\ \text{undefined}, & \text{otherwise} \end{cases}$$

Figure 5: A problem with composition

The connectivity graph $\Gamma_M \cup \Gamma_N$ is defined by letting $(\Gamma_M \cup \Gamma_N)_V = (\Gamma_M)_V \cup (\Gamma_N)_V$, $(\Gamma_M \cup \Gamma_N)_E = (\Gamma_M)_E \cup (\Gamma_N)_E$. □

So the form of the composite network is determined completely by that of the components, and the only requirement on the consistency predicate iscons is that whenever it is defined we get a well-formed network. This amounts to requiring that

$$(\Gamma_M \rhd M)\ \mathsf{iscons}\ (\Gamma_N \rhd N) \ \text{ implies } \ \mathrm{nodes}(M) \cap \mathrm{nodes}(N) = \emptyset \tag{4.1}$$

Example 4.2. Let m be the partial binary predicate defined by letting $(\Gamma_M \rhd M)\ \mathsf{m}\ (\Gamma_N \rhd N)$ whenever

- $\mathrm{nodes}(M) \cap \mathrm{nodes}(N) = \emptyset$
- $\Gamma_M \vdash m \leftrightarrow n$ if and only if $\Gamma_N \vdash m \leftrightarrow n$, for every $m \in \mathrm{nodes}(M)$ and $n \in \mathrm{nodes}(N)$

By definition this satisfies the requirement (4.1) above, and intuitively it only allows the composition whenever the two individual networks agree on the interconnections between internal and external nodes.

It is easy to check that the resulting operator $|||_{\mathrm{m}}$ is both associative and commutative. □

One use of composition operators is to enable compositional reasoning. For example the task of establishing

$$\mathcal{N}_1 \sqsubseteq_{behav} \mathcal{N}_2 \tag{4.2}$$

can be simplified if we can discover a common component, that is some $\mathcal{N}$ such that $\mathcal{N}_1 = \mathcal{M}_1 ||| \mathcal{N}$ and $\mathcal{N}_2 = \mathcal{M}_2 ||| \mathcal{N}$. Then (4.2) can be reduced to establishing

$$\mathcal{M}_1 \sqsubseteq_{behav} \mathcal{M}_2,$$

assuming that the behavioural preorder in question, $\sqsubseteq_{behav}$, is preserved by the composition operator $|||$.

However another use of a composition operator is in the definition of the behavioural preorder $\sqsubseteq_{behav}$ itself. Intuitively we can define

$$\mathcal{N}_1 \sqsubseteq_{behav} \mathcal{N}_2 \tag{4.3}$$

to be true if for every component $\mathcal{T}$ which can be composed with both $\mathcal{N}_1$ and $\mathcal{N}_1$ the external observable behaviour of the composite networks $\mathcal{N}_1 ||| \mathcal{T}$ and $\mathcal{N}_2 ||| \mathcal{T}$ are related in some appropriate way; we will shortly see precisely what this might mean.

Unfortunately with this use in mind the composition operator $|||_{\mathrm{m}}$ defined via Example 4.2 is not appropriate.

Example 4.3. Consider the networks in Figure 5, where the code running at node l in M is very different than that running at k in N; say $c!\langle 0\rangle$ and $c!\langle 1\rangle$ respectively. Then intuitively $\mathcal{M}$ and $\mathcal{N}$ should have different observable behaviour, observable by placing a test at the node o. However if the operator $|||_{\mathrm{m}}$ is used to combine the test with the network being observed they are indistinguishable.

This is because if there is a network $\mathcal{T}$ such that $\mathcal{M} \,|||_{\mathrm{m}}\, \mathcal{T}$ and $\mathcal{N} \,|||_{\mathrm{m}}\, \mathcal{T}$ are well-defined then o can not be in nodes($\mathcal{T}$). For if o were in nodes($\mathcal{T}$), then since $\Gamma_M \vdash l \leftrightarrow o$ the definition of the operator implies that $\Gamma_T \vdash l \leftrightarrow o$. This in turn implies that $\Gamma_N \vdash l \leftrightarrow o$, which is not true.

Now since no testing network which can be applied to both $\mathcal{M}$ and $\mathcal{N}$ can place any code at o, no difference can be discovered between them. □

We want to be able to compare networks with different connectivity graphs, and possibly different nodes, such as $\mathcal{M}$ and $\mathcal{N}$ in Figure 5. We also should not be able to change the connectivity of a network when we test it; we wish to implement *black-box* testing, where the nodes containing running code cannot be accessed directly.

Rather than presenting another arbitrary composition operator for composing a testing network with a network to be tested, let us place natural requirements on such operators. The first says that the composed network is completely determined by the components:

(I) Merge: the operator $|||$ should be determined by some partial predicate iscons using Definition 4.1.

Intuitively the interface of a network is how their external behaviour is to be observed. Since our aim is to enable compositional reasoning over networks, we would expect composition to preserve interfaces:

(II) Interface preservation: If $\mathsf{Int}(\mathcal{M}) = \mathsf{Int}(\mathcal{N})$ and $\mathcal{T}$ can be composed with both, that is both $\mathcal{M} \,|||\, \mathcal{T}$ and $\mathcal{N} \,|||\, \mathcal{T}$ are well-defined, then $\mathsf{Int}(\mathcal{M} \,|||\, \mathcal{T}) = \mathsf{Int}(\mathcal{N} \,|||\, \mathcal{T})$.

The final requirement captures the intuitive idea that reorganising the internal structure of a network should not affect the ability to perform a test; in fact the reorganisation is simply a renaming of nodes. Let σ be a permutation of node names. We use $(\Gamma \rhd M)\sigma$ to denote the result of applying σ to the node names in M and in the connectivity graph Γ.

(III) Renaming: Suppose $\mathcal{M} \,|||\, \mathcal{T}$ is defined. Then $\mathcal{M}\sigma \,|||\, \mathcal{T}$ is also defined, provided σ is a node permutation which satisfies

- $\sigma(o) = o$ for every $o \in \mathsf{Int}(\mathcal{M})$
- no $n \in \mathrm{nodes}(\mathcal{T})$ appears in the range of σ; that is $n \in \mathrm{nodes}(\mathcal{T})$ implies $\sigma(n) = n$.

Example 4.4. The operator $|||_{\mathrm{m}}$ does not satisfy **(III)**, as can be seen using the simple networks in Figure 6; $\mathcal{M} \,|||_{\mathrm{m}}\, \mathcal{T}$ is obviously well-defined. However, consider the renaming σ which swaps node names l to k, which is valid with respect to $\mathcal{T}$; the network $\mathcal{M}\sigma \,|||_{\mathrm{m}}\, \mathcal{T}$ is not defined, as $\Gamma_T \nvdash k \leftrightarrow o$.

A slight modification will demonstrate that interfaces are also not preserved by this operator. □

Proposition 4.5. Suppose $|||$ satisfies the conditions (I) - (III) above. Then $\mathrm{nodes}(\mathcal{M}) \cap \mathsf{Int}(\mathcal{N}) = \emptyset$ whenever $\mathcal{M} \,|||\, \mathcal{N}$ is defined.

Proof. By contradiction; let $\mathcal{M} = \Gamma_M \rhd M$ and $\mathcal{N} = \Gamma_N \rhd N$. Assume that m is a node included in $\mathrm{nodes}(M) \cap \mathsf{Int}(\Gamma_N \rhd N)$, and that $\mathcal{M} \,|||\, \mathcal{N}$ is defined. By condition (I), $\mathcal{M} \,|||\, \mathcal{N} = (\Gamma_1 \cup \Gamma_2) \rhd M|N$.

Since $m \in \mathsf{Int}(\mathcal{N})$, it has to be $\Gamma_2 \vdash m$.
Since $m \in \mathrm{nodes}(M)$, it holds $m \in \mathrm{nodes}(M|N)$. By definition of interface, we obtain $m \notin \mathsf{Int}(\mathcal{M} \,|||\, \mathcal{N})$.

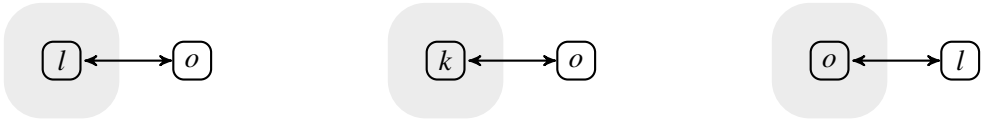


Figure 6: A problem with the composition operator

Now let l be a node which is not contained in Γ_M, nor in Γ_N. Consider the permutation σ which swaps nodes m and n; that is $\sigma(m) = l, \sigma(l) = m$ and $\sigma(k) = k$ for all $k \neq m, l$. Condition (i) of Definition 3.1 ensures that $l \notin \text{nodes}(\mathcal{M})$, so that $m \notin \text{nodes}(\mathcal{M}\sigma)$. Further, the permutation σ is consistent with condition (III), renaming, when applied to networks $\mathcal{M}$ and $\mathcal{N}$; therefore $\mathcal{M}\sigma \mid\mid\mid \mathcal{N}$ is defined. Also, we have $\mathsf{Int}(\mathcal{M}) = \mathsf{Int}(\mathcal{M}\sigma)$.

Far by now, we established the following:

(1) $m \notin \mathsf{Int}(\mathcal{M} \mid\mid\mid \mathcal{N})$,
(2) $m \notin \text{nodes}(\mathcal{M}\sigma)$,
(3) $m \notin \text{nodes}(\mathcal{N})$,
(4) $\Gamma_N \vdash m$,
(5) $\mathsf{Int}(\mathcal{M}) = \mathsf{Int}(\mathcal{M}\sigma)$.

By (2) and (3) above, it holds that $m \notin \text{nodes}(\mathcal{M}\sigma \mid\mid\mid \mathcal{N})$; By (4), we obtain that $(\Gamma_1)\sigma \cup \Gamma_2 \vdash n$. These two statements ensure that $m \in \mathsf{Int}(\mathcal{M}\sigma \mid\mid\mid \mathcal{N})$.
As a direct consequence of (5) and condition (II), type preservation, it also holds $m \in \mathsf{Int}(\mathcal{M} \mid\mid\mid \mathcal{N})$, but this contradicts (1).

□

Corollary 4.6. Let $\mid\mid\mid$ be any symmetric composition operator which satisfies the conditions (I) - (III). Suppose $\mathcal{M}_1 \mid\mid\mid \mathcal{M}_2$ is well-defined, and of the form $\Gamma \rhd M$. Then $\Gamma \vdash m_1 \leftrightarrow m_2$ whenever $m_i \in \text{nodes}(\mathcal{M}_i)$.

Proof. A simple consequence of the previous result. □

What this means is if we use such a symmetric operator when applying a test to a network, as in Definition 4.9 below, then the resulting testing preorder will be degenerate; it will not distinguish between any pair of nets. In some sense this result is unsurprising. For $\mathcal{T}$ to test $\mathcal{M}$ in $\mathcal{M} \mid\mid\mid \mathcal{T}$ it must have code running at the interface of $\mathcal{M}$. But, as we have seen, condition (III) more or less forbids $\mathcal{T}$ to have code running at the interface of $\mathcal{M}$.

We would like our composition operator to satisfy these three conditions, and since there are no further obvious requirements we choose the largest such operator.

Definition 4.7. Let $\between$ be the partial composition operator obtained by using in Definition 4.1 the predicate $(\Gamma_M \rhd M)$ iscons $(\Gamma_N \rhd N)$ whenever $\text{nodes}(M) \cap (\Gamma_N)_V = \emptyset$. □

It is straightforward to show that $\between$ satisfies the minimum requirement (4.1) above, and the conditions (I) - (III). Proposition 4.5 then gives that it is the largest such operator.
Informally speaking, the operator $\between$ can be viewed as an extension operator; basically, the network $\mathcal{M} \between \mathcal{N}$ is defined by integrating the information of network $\mathcal{M}$ with that of $\mathcal{N}$, provided that the latter does not interfere with the connectivities of $\mathcal{M}$.

Despite not being symmetric the operator $\between$ has interesting properties, the most important of which concerns decomposition. Let $\mathbf{G}$ be the collection of networks which contain exactly one occupied node, that is of the form $\Gamma \rhd n[\![s]\!]$. It turns out that, modulo a simple structural equivalence, all networks can be generated from $\mathbf{G}$ using the composition operator $\between$. This structural equivalence is generated by the commutativity and associativity of the parallel operator $|$ and the axiom $M \mid \mathbf{0} = M$.

Proposition 4.8.

(i) $\between$ is associative; that is, $\mathcal{M}_1 \between (\mathcal{M}_2 \between \mathcal{M}_3) \equiv (\mathcal{M}_1 \between \mathcal{M}_2) \between \mathcal{M}_3$, where $\equiv$ is Kleene equality.
(ii) Suppose $\mathcal{M}$ is a network such that $n \in \mathrm{nodes}(\mathcal{M})$. Then, modulo the structural equivalence, $\mathcal{M} = \mathcal{N} \between \mathcal{G}$ for some $\mathcal{G} \in \mathbf{G}$.

Proof. Part (i) follows by simple calculation. To prove part (ii) let the connectivity graph Γ_n be defined by

- $(\Gamma_n)_V = \{n\} \cup I$ where $I = \{\, k \in \mathsf{Int}(\mathcal{M}) \mid \mathcal{M} \vdash n \leftrightarrow k \,\}$.
- The only connections are between n and the nodes in I.

The network $(\Gamma_n \rhd n[\![p]\!])$ is obviously well-formed and in $\mathcal{G}$.

Even if n is the only node in the network we can assume, using the structural equivalence, that $\mathcal{M}$ has the form $\Gamma \rhd M \mid n[\![p]\!]$, and so the other component $\mathcal{N}$ will be of the form $(\Gamma_M \rhd M)$ where

- $(\Gamma_M)_V = \Gamma_V \setminus I$
- The connectivity is determined by $\Gamma_M \vdash k_1 \leftrightarrow k_2$ if and only if at least one k_i is in $\mathrm{nodes}(M)$ and $\Gamma \vdash k_1 \leftrightarrow k_2$.

Again it is easy to check that $(\Gamma_M \rhd M)$ is well-formed. Moreover $\mathrm{nodes}(M) \cap (\Gamma_n)_V = \emptyset$ and so $(\Gamma_M \rhd M) \between \Gamma \rhd M \mid n[\![p]\!]$ is defined, and it is equal to $\mathcal{M}$.

□

4.2. **Testing structures.** The introduction of the composition operator $\between$ allows to introduce a behavioural theory based on a probabilistic generalisation of the *Hennessy* and *De Nicola* testing preorders [12]. In order to develop such a framework, we will exploit the mathematical tools introduced in Section 2; our aim is to be able to relate networks with different connectivities.

In our framework, testing can be summarised as follows; a network is composed with another one, which takes the name of *testing network*. The composition of these two networks is then isolated from the external environment, in the sense that no external agent (in our case nodes in the interface of the composed network) can interfere with its behaviour; we will shortly present how such a task can be accomplished. The composition of the two networks, isolated from the external environment, takes the name of *experiment*.

Once these two operations (composition with a test and isolation from the external environment) have been performed, the behaviour of the resulting eperiment is analysed to check whether there exists a computation that yields to a state which is successful. This task can be accomplished by relying on testing structures, which will be presented shortly.

At an informal level, successful states in our languages coincide with those associated with networks where at least the code running in one node can perform the action ω. For network have probabilistic behaviour, each computation will be associated with the probability of reaching a successful state; thus, every experiment will be associated with a set of success probabilities, one for each of its computation.

Let us now look at how the procedure explained above can be formalised; the topic of composing networks has already been addressed in detail in Section 4.1, in which we defined the operator

$\nmid$ and proved basilar properties for it. To model experiments and their behaviour, we rely on the following mathematical structure.

Definition 4.9. A *Testing structure* (TS) is a triple $\langle S, \rightarrow, \omega \rangle$ where

(i) S is a set of states,
(ii) the relation $\rightarrow$ is a subset of $S \times \mathcal{D}(S)$,
(iii) ω is a success predicate over S, that is $\omega : S \rightarrow \{\mathsf{true}, \mathsf{false}\}$. □

Testing structures can be seen as (degenerate) pLTSs where the only possible action corresponds to the internal activity τ, and the transition $\xrightarrow{\tau}$ is defined to coincide with the reduction relation. Conversely every pLTS automatically determines a testing structure, by concentrating on the relation $\xrightarrow{\tau}$.

Our goal is to turn a network in a testing structure. This amounts to defining, for networks, a reduction relation and the success predicate ω. As we have mentioned in the beginning of this Section, when converting a network in a testing structure, we want to make it isolated from the external environment.

When considering simpler process languages, like *CCS* or *CSP* (and, more generally, their probabilistic counterparts), processes are converted into testing structures by isolating them from the external environment through the use of some restriction operator.

Networks, however, have a more complicated structure; here the external environment consists of their interface. The only possibility for the behaviour of a network to be corrupted by an external agent coincides with a node in its interface sending messages to it; this is the same as inferring that such a network can perform an action of the form $\xrightarrow{m.c?v}$. All other actions (outputs and internal actions) are allowed, as they do not originate from any external agent.

Thus, isolation can be achieved by defining the reduction relation, for the testing structure associated with networks, to coincide with the union of all internal and output actions.

Finally, the success predicate ω is defined to be true for all, and only all, those networks in which a node can perform the success action $\xrightarrow{\omega}$.

Example 4.10. The main example of a TS is given by

$$\langle \mathsf{Nets}, \longmapsto, \Omega \rangle$$

where

(i) $(\Gamma \rhd M) \longmapsto (\Gamma \rhd \Delta)$, with $\Gamma \rhd M$ being a state based network, whenever
  (a) $\Gamma \rhd M \xrightarrow{m.\tau} \Delta$ for some $m \in \mathrm{nodes}(M)$
  (b) or, $\Gamma \rhd M \xrightarrow{c.m!v} \Delta$ for some value v, node name m and channel c

(ii)

$$\omega(M) = \begin{cases} \mathsf{true}, & M = M' \mid n[\![s]\!],\ s \xrightarrow{\omega} \text{ for some n} \\ \mathsf{false}, & \text{otherwise} \end{cases}$$

If $\omega(M) = \mathsf{true}$ for some system term M, we say that a network $\Gamma \rhd M$, where Γ is an arbitrary conncectivity graph, is ω-successful. Note that when recording an ω-success we do not take into account the node involved. □

As TSs can be seen as pLTSs, we can use in an arbitrary TS the various constructions introduced in Section 2. Thus the reduction relation $\longmapsto$ can be lifted to $\mathcal{D}(\mathsf{sSys}) \times \mathcal{D}(\mathsf{sSys})$ and we can make use of the concepts of hyper-derivatives and extreme-derivatives, introduced in Section 2, to model fragments of executions and maximal executions of a testing structure, respectively. Below we

provide two simple examples that show how to reason about the behaviour of the testing structures presented in Example 4.10.

Example 4.11. Consider the testing structure associated with the network $\mathcal{N}$ in the center of Figure 6, where the code q is given by the definition $q \Leftarrow q_{0.5}\oplus\, c!\langle v\rangle\,.\,\mathbf{0}$. We can show that, in the long run, this network will broadcast message v to the external location o by exhibiting a hyper derivation for it which terminates in the pointed distribution $\overline{\Gamma_N \rhd k[\![c!\langle v\rangle\,.\,\mathbf{0}]\!]}$. If we let $\mathcal{N}_1$ denote the configuration $\Gamma_N \rhd \Gamma_N \rhd k[\![c!\langle v\rangle\,.\,\mathbf{0}]\!]$, we have the following hyper-derivation:

$$\begin{array}{lcll}
\overline{\mathcal{N}} & = & \frac{1}{2}\cdot\overline{\mathcal{N}} & + \ \frac{1}{2}\cdot\overline{\mathcal{N}_1} \\
\frac{1}{2}\cdot\overline{\mathcal{N}} & \longmapsto & \frac{1}{2^2}\cdot\overline{\mathcal{N}} & + \ \frac{1}{2^2}\cdot\overline{\mathcal{N}_1} \\
\vdots & & & \\
\frac{1}{2^n}\cdot\overline{\mathcal{N}} & \longmapsto & \frac{1}{2^{n+1}}\cdot\overline{\mathcal{N}} & + \ \frac{1}{2^{n+1}}\cdot\overline{\mathcal{N}_1} \\
\vdots & & & \\
 & & & \Delta' = \sum_{n=1}^{\infty}\ \frac{1}{2^n}\ \cdot\ \overline{\mathcal{N}_1}
\end{array}$$

Now it is straightforward to check that $\Delta' = \overline{\mathcal{N}_1}$ and therefore we have the hyper-derivation $\mathcal{N} \Longrightarrow \overline{\mathcal{N}_1}$. □

An arbitrary network $\mathcal{N}$ can be tested by another (testing) network $\mathcal{T}$ provided $\mathcal{N} \between \mathcal{T}$ is well-defined. Executions of the resulting testing structure will then be checked to establish whether the network $\mathcal{M}$ satisfies a property the test was designed for; in such a case, the testing component of an experiment will reach a ω-successful state.

Executions, or maximal computations, correspond to extreme derivatives of $\mathcal{N} \between \mathcal{T}$, as defined in Section 2. Since the framework is probabilistic, each execution (that is extreme derivative) will be associated with a probability value, representing the probability that it will lead to an ω-successful state. Since the framework is also nondeterministic the possible results of this test application is given by a non-empty set of probability values.

Definition 4.12. [Tabulating results] The *value* of a subdistribution in a TS is given by the function $\mathcal{V} : \mathcal{D}_{sub}(S) \to [0,1]$, defined by $\mathcal{V}(\Delta) = \sum\{\,\Delta(s) \mid \omega(s) = \mathsf{true}\,\}$. Then the set of possible results from a sub-distribution Δ is defined by $\mathcal{O}(\Delta) = \{\,\mathcal{V}(\Delta') \mid \Delta \Longrightarrow\!\!\!\!\gg \Delta'\,\}$. □

Example 4.13. Consider the testing network $\mathcal{T}$ given in Figure 6, where the code is determined by $t \Leftarrow c?(x)\,.\omega.\,\mathbf{0}$. It is easy to check that $\mathcal{N} \between \mathcal{T}$ is well-defined, and is equal $\Gamma \rhd k[\![q]\!] \mid o[\![t]\!]$. So consider the testing structure associated with it; recall that we have the definition $q \Leftarrow q_{0.5}\oplus\, c!\langle v\rangle\,.\,\mathbf{0}$. For convenience let $\mathcal{N}_1 = \Gamma_N \rhd k[\![c!\langle v\rangle\,.\,\mathbf{0}]\!]$ as in the previous example, $\mathcal{N}_2 = \Gamma_N \rhd k[\![\mathbf{0}]\!]$ and $\mathcal{T}_\omega = \Gamma_T \rhd o[\![\omega.\,\mathbf{0}]\!]$. Then we have the following hyper-derivation for $\mathcal{N} \between \mathcal{T}$:

$$\begin{array}{lcll}
\overline{\mathcal{N} \between \mathcal{T}} & \longmapsto & (\frac{1}{2}\cdot\overline{\mathcal{N} \between \mathcal{T}} + \frac{1}{2}\cdot\overline{\mathcal{N}_1 \between \mathcal{T}}) & + \ \varepsilon \\
\frac{1}{2}\cdot\overline{\mathcal{N} \between \mathcal{T}} + \frac{1}{2}\cdot\overline{\mathcal{N}_1 \between \mathcal{T}} & \longmapsto & (\frac{1}{2^2}\cdot\overline{\mathcal{N} \between \mathcal{T}} + \frac{1}{2^2}\cdot\overline{\mathcal{N}_1 \between \mathcal{T}}) & + \ \frac{1}{2}\cdot\overline{\mathcal{N}_2 \between \mathcal{T}_\omega} \\
\vdots & & \vdots & \vdots \\
\frac{1}{2^n}\cdot\overline{\mathcal{N} \between \mathcal{T}} + \frac{1}{2^n}\cdot\overline{\mathcal{N}_1 \between \mathcal{T}} & \longmapsto & (\frac{1}{2^{n+1}}\cdot\overline{\mathcal{N} \between \mathcal{T}} + \frac{1}{2^{n+1}}\cdot\overline{\mathcal{N}_1 \between \mathcal{T}}) & + \ \frac{1}{2^n}\cdot\overline{\mathcal{N}_2 \between \mathcal{T}_\omega} \\
\vdots & & \vdots & \vdots
\end{array}$$

were ε denotes the empty sub-distribution, that is the one with $\lceil\varepsilon\rceil = \emptyset$. We have therefore the hyper-derivative

$$\overline{\mathcal{N} \between \mathcal{T}} \Longrightarrow \varepsilon + \sum_{n=1}^{\infty}\frac{1}{2^n}\overline{\mathcal{N}_2 \between \mathcal{T}_\omega} = \overline{\mathcal{N}_2 \between \mathcal{T}_\omega}.$$

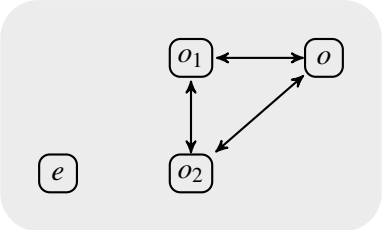


Figure 7: A test

Further, the above hyper-derivation satisfies the constraints required by $\Longrightarrow$, defined in Section 2, and therefore we have the extreme derivative $\overline{\mathcal{N} \between \mathcal{T}} \Longrightarrow \overline{\mathcal{N}_2 \between \mathcal{T}_\omega}$. Since $\mathcal{V}(\overline{\mathcal{N}_2 \between \mathcal{T}_\omega}) = 1$ we can therefore deduce that $1 \in \mathcal{O}(\mathcal{N} \between \mathcal{T})$. □

4.3. **The behavioural preorder.** We now combine the concepts of the previous two sections to obtain our behavioural preorder. We have seen how to associate a non-empty set of probabilities, tabulating the possible outcomes from applying the test $\mathcal{T}$ to the network $\mathcal{N}$. As explained in [3] there are two natural ways to compare such sets, optimistically or pessimistically.

Here, for the sake of simplicity, we consider only the opimistic version. In this case, two sets of outcomes O_1 and O_2 can be related by looking if every outcome o_1 which appears in O_1 is less than at least an outcome o_2 in the set O_2; in this case we write $O_1 \sqsubseteq_{\text{may}} O_2$. This notion coincides with the Hoare preorder defined for sets of values contained in the interval $[0, 1]$.

Definition 4.14. [Testing networks] For $\mathcal{M}_1, \mathcal{M}_2 \in \mathsf{Nets}$ we write $\mathcal{M}_1 \sqsubseteq_{\text{may}} \mathcal{M}_2$ if

(i) $\mathsf{Int}(\mathcal{M}_1) = \mathsf{Int}(\mathcal{M}_2)$,

(ii) for every (testing) network $\mathcal{T} \in \mathsf{Nets}$ such that both $\mathcal{M}_1 \between \mathcal{T}$ and $\mathcal{M}_2 \between \mathcal{T}$ are defined, $\mathcal{O}(\mathrm{cl}(\mathcal{M}_1 \between \mathcal{T})) \sqsubseteq_{\text{may}} \mathcal{O}(\mathrm{cl}(\mathcal{M}_2 \between \mathcal{T}))$.

We use $\mathcal{M}_1 =_{\text{may}} \mathcal{M}_2$ as an abbreviation for $\mathcal{M}_1 \sqsubseteq_{\text{may}} \mathcal{M}_2$ and $\mathcal{M}_2 \sqsubseteq_{\text{may}} \mathcal{M}_1$. □

The first requirement in Definition 4.14 has been introduced to ensure that, whenever $\mathcal{M}_1 \sqsubseteq_{\text{may}} \mathcal{M}_2$, if m is a node which can be used to test the behaviour of $\mathcal{M}_1$, then it also has to be available to test the behaviour of $\mathcal{M}_2$ as well. In other words, we have no interest in relating two networks $\mathcal{M}_1$ and $\mathcal{M}_2$ if there exists a node which can be used to observe only the behaviour of the former network.

Example 4.15. Consider the testing network

$$\mathcal{T} = \Gamma_T \rhd e[\![c!\langle 0\rangle]\!] \mid o_1[\![d?(x).c!\langle x\rangle]\!] \mid o_2[\![d?(x).c!\langle x\rangle]\!] \mid o[\![c?(x).c?(y).\omega]\!]$$

where the connectivity is described in Figure 7. This can be used to test the networks from Figure 2 in the testing structure of Example 4.10. Intuitively the test send the value 0 along the channel c at the node e, awaits for results along the channel d at the nodes o_1 and o_2. These results are processed at node o, where success might be announced.

The combined network $(\mathcal{M} \between \mathcal{T})$ is deterministic in this TS, although probabilistic, and so has only one extreme derivative; $\mathcal{O}(\mathcal{M} \between \mathcal{T}) = \{0.8\}$. A similar calculation shows that $\mathcal{O}(\mathcal{N} \between \mathcal{T}) = \{0.81\}$; it therefore follows that $\mathcal{N} \not\sqsubseteq_{\text{may}} \mathcal{M}$.

However $(\mathcal{M}_1 \between \mathcal{T})$ is both probabilistic and nondeterministic, and $\mathcal{O}(\mathcal{M}_1 \between \mathcal{T}) = \{\, p \mid 0.5 \leq p \leq 1 \,\}$. Moreover because of the optimistic manner in which results sets are compared we have $\mathcal{O}(\mathcal{M} \between \mathcal{T}) \sqsubseteq_{\text{may}} \mathcal{O}(\mathcal{M}_1 \between \mathcal{T})$.

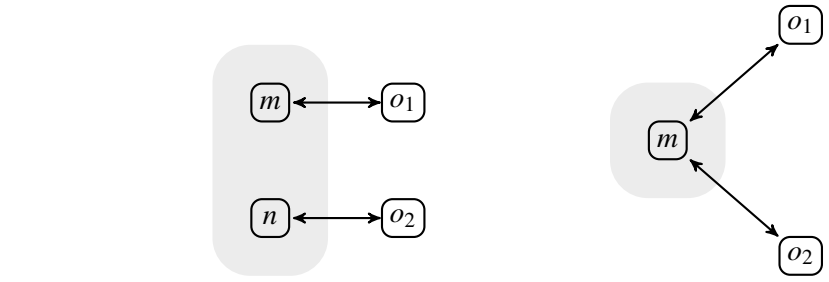


Figure 8: Broadcast vs Multicast

Finally the combined network $(\mathcal{M}_2 \between \mathcal{T})$ is also deterministic, although it has *limiting behaviour*; $O(\mathcal{M}_2 \between \mathcal{T}) = \{1\}$. Thus, in this case we have both $O(\mathcal{M} \between \mathcal{T}) \sqsubseteq_{\text{may}} O(\mathcal{M}_2 \between \mathcal{T})$ and $O(\mathcal{M}_1 \between \mathcal{T}) \sqsubseteq_{\text{may}} O(\mathcal{M}_2 \between \mathcal{T})$. □

Example 4.16. [Broadcast vs Multicast] Consider the networks $\mathcal{M}$ and $\mathcal{N}$ in Figure 8. Intuitively in $\mathcal{N}$ the value v is (simultaneously) broadcast to both nodes o_1 and o_2 while in $\mathcal{M}$ there is a multicast. More specifically o_1 receives v from mode m while in an independent broadcast o_2 receives it from n.

This difference in behaviour can be detected by testing network

$$\mathcal{T} \quad = \quad \Gamma_T \triangleright o_1[\![c?(x)\,.c!\langle 0\rangle \,.\, \mathbf{0}]\!] \mid o_2[\![c?(x)\,.c?(y)\,.\text{if } y = 0 \text{ then } \mathbf{0} \text{ else } \omega]\!]$$

assuming v is different than 0; here we assume Γ_T is the simple network which connects o_1 with o_2. Both $\mathcal{M} \between \mathcal{T}$ and $\mathcal{N} \between \mathcal{T}$ are well-formed and note that they are both non-probabilistic.

Because $\mathcal{N}$ simultaneously broadcasts to o_1 and o_2 the second value received by o_2 is always 0 and therefore the test never succeeds; $\mathcal{V}(\mathcal{N} \between \mathcal{T}) = \{0\}$. On the other-hand there is a possibility for the test succeeding when applied to $\mathcal{M}$, $1 \in \mathcal{V}(\mathcal{M} \between \mathcal{T})$. This is because in $\mathcal{M}$ node m might first transmit v to o_1 after which n transmits 0 to o_2; now node n might transmit the value v to o_2 and asuming it is different than 0 we reach a success state. It follows that $\mathcal{M} \not\sqsubseteq_{\text{may}} \mathcal{N}$.

One might also think it possible to use the difference between broadcast and multicast to design a test which $\mathcal{N}$ passes and $\mathcal{M}$ does not. For example $\mathcal{N}$ passes the test

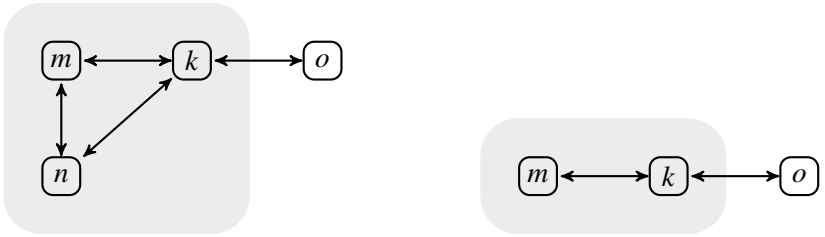
$$\mathcal{T}' \quad = \quad \Gamma_T \triangleright o_1[\![c?(x)\,.c!\langle 0\rangle \,.\, \mathbf{0}]\!] \mid o_2[\![c?(x)\,.c?(y)\,.\text{if } y = 0 \text{ then } \omega \text{ else } \mathbf{0}]\!]$$

because in $\mathcal{N} \between \mathcal{T}'$ the second value received by o_2 is always 0. However $\mathcal{M}$ also passes this test, since the simultaneous broadcast in $\mathcal{N}$ can be simulated by a multicast in $\mathcal{M}$, by node m first broadcasting to o_1 followed by n broadcasting to o_2. As this line of reasoning is independent from the test $\mathcal{T}'$, it also applies to all those networks that can be used to test the behaviour of $\mathcal{M}$ and $\mathcal{N}$; this leads to the intuition that $\mathcal{N} \sqsubseteq_{\text{may}} \mathcal{M}$, which will be proved formally later as a consequence of Example 5.5 and Theorem 5.6. □

One pleasing property of the behavioural preorder $\sqsubseteq_{\text{may}}$ is that it allows compositional reasoning over networks.

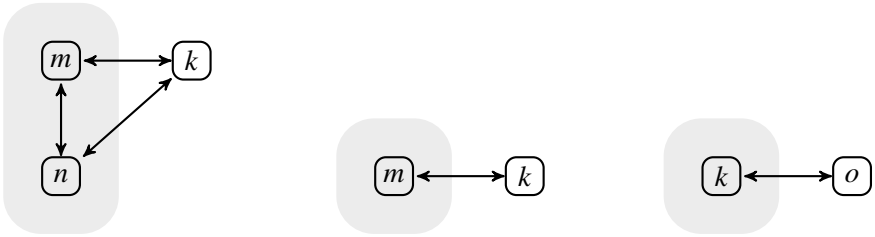
Proposition 4.17 (Compositionality)**.** *Let $\mathcal{M}_1, \mathcal{M}_2$ be two networks such that $\mathcal{M}_1 \sqsubseteq_{may} \mathcal{M}_2$, and let $\mathcal{N}$ be another network such that both $\mathcal{M}_1 \between \mathcal{N}$ and $\mathcal{M}_2 \between \mathcal{N}$ are defined. Then $\mathcal{M}_1 \between \mathcal{N} \sqsubseteq_{may} \mathcal{M}_2 \between \mathcal{N}$.*

Proof. A direct consequence of $\between$ being both associative and interface preserving. □



$\mathcal{M} = \Gamma_M \rhd m[\![A_m]\!] \mid n[\![A_n]\!] \mid k[\![A_k]\!]$ $\quad$ $\mathcal{N} = \Gamma_N \rhd m[\![B_m]\!] \mid k[\![A_k]\!]$

Figure 9: Two networks with a common sub-network



$\mathcal{M}_1 = \Gamma_1 \rhd m[\![A_m]\!] \mid n[\![A_n]\!]$ $\quad$ $\mathcal{N}_1 = \Gamma_2 \rhd m[\![B_m]\!]$ $\quad$ $\mathcal{K} = \Gamma_K \rhd k[\![B_k]\!]$

Figure 10: Decomposition of the networks $\mathcal{M}$ and $\mathcal{N}$

An application of this Compositionality result is given by the following Example:

Example 4.18. Consider the networks $\mathcal{M}$ and $\mathcal{N}$ in Figure 9, where the codes at the various nodes are given by

$$\begin{aligned} A_m &\Leftarrow c!\langle v\rangle\,.\,\mathbf{0} \\ A_n &\Leftarrow c?(x)\,.d!\langle w\rangle\,.\,\mathbf{0} \\ A_k &\Leftarrow c?(x)\,.d?(y)\,.e!\langle u\rangle\,.\,\mathbf{0} \\ B_n &\Leftarrow c!\langle v\rangle\,.d!\langle w\rangle\,.\,\mathbf{0} \end{aligned}$$

It is possible to write both of them respectively as $\mathcal{M}_1 \nparallel \mathcal{K}$ and $\mathcal{N}_1 \nparallel \mathcal{K}$, where the networks $\mathcal{M}_1, \mathcal{N}_1$ and $\mathcal{K}$ are depicted in Figure 10. In order to prove that $\mathcal{M} \sqsubseteq_{\text{may}} \mathcal{N}$, it is therefore sufficient to focus on their respective subnetworks $\mathcal{M}_1$ and $\mathcal{N}_1$, and prove $\mathcal{M}_1 \sqsubseteq_{\text{may}} \mathcal{N}_1$. The equivalence of the two originary networks will then follow from a direct application of Proposition 4.17. □

5. Simulations

The aim is to exhibit a sound proof method to check whether two networks can be related via the may-testing preorder. In [3] it was shown that the may-testing preorder over the process calculus pCSP can be characterised in terms of certain kinds of simulations over a probabilistic labelled transition system.

Here we consider the simulation preorder which is induced by a non-standard definition of weak extensional actions, which are defined in order to deal with the local broadcast features which

characterize our calculus. We show that our simulation preorder provides a sound proof methodology for checking whether two networks can be compared with respect to the $\sqsubseteq_{\text{may}}$ preorder; however our simulations fail to be complete.

In the first section we concentrate on the activities of a network which can be observed by nodes in their interface, thus defining both strong and weak extensional actions for a network. This leads to a definition of (weak probabilistic) simulation between networks. The next two sections are devoted to the proof of soundness. Then we explain, in Section 5.4, why the proof methodolgy is not complete.

5.1. **Extensional semantics.** Here the idea is to design a pLTS over the collection of networks Nets such that whenever two networks are simulation related as defined in [3, 18], then they will also be testing related, $\mathcal{M} \sqsubseteq_{\text{may}} \mathcal{N}$. The intensional semantics in Section 3 already provides a pLTS and it is instructive to see why this is not appropriate.

Consider $\mathcal{M}$ and $\mathcal{N}$ from Figure 6, and suppose further that the code p and q, running at l and k respectively, is identical, $c!\langle v\rangle.\mathbf{0}$. Then we would expect $\mathcal{M}$ and $\mathcal{N}$ to be behaviourally indistinguishable. However $\mathcal{M}$ will have an output action, labelled $c.l!v$, which is not possible for $\mathcal{N}$. So output actions cannot record their source node. What turns out to be important is the target nodes. For example if in $\mathcal{M}$ we added a new node m to the interface, with a connection to l then we would be able to distinguish $\mathcal{M}$ from $\mathcal{N}$; the required test would simply place some appropriate testing code at the new node m.

We now present an extensional semantics for networks; here the visible actions consist of activities which can be detected (hence tested) by placing code at the interface of a network. In this semantics we have internal, input and output actions.

Definition 5.1. [Extensional actions] The actions of the extensional semantics are defined as follows:

(1) **internal**, $(\Gamma \rhd M) \xrightarrow{\tau} (\Gamma \rhd \Delta)$; some internal activity reduces the system M, relative to the connectivity Γ, to some system N, where $N \in \lceil\Delta\rceil$. Here the internal activity of a network coincides either with some node performing a silent move $m.\tau$ or broadcasting a value which cannot be detected by any node in the interface of the network itself.

Formally, $(\Gamma \rhd M) \xrightarrow{\tau} (\Gamma \rhd \Delta)$ whenever

(a) $\Gamma \rhd M \xrightarrow{m.\tau} \Delta$

(b) or $\Gamma \rhd M \xrightarrow{c.n!v} \Delta$ for some value v, channel c and node name n satisfying $\Gamma \vdash n \leftrightarrow m$ implies $m \in \text{nodes}(M)$

Note that we are using the notation given in Section 3 for defining distributions. Here Δ is a distribution over sSys and so $(\Gamma \rhd \Delta)$ is a distribution over networks; however all networks in its support use the same network connectivity Γ.

(2) **input**, $(\Gamma \rhd M) \xrightarrow{c.n?v} (\Gamma \rhd \Delta)$; an observer placed at node n can send the value v along the channel c to the network $(\Gamma \rhd M)$. For the observer to be able to place the code at node n we must have $n \in \mathsf{Int}(\Gamma \rhd M)$.

Formally $(\Gamma \rhd M) \xrightarrow{c.n?v} (\Gamma \rhd \Delta)$ whenever

(a) $\Gamma \rhd M \xrightarrow{c.n?v} \Delta$

(b) $n \in \mathsf{Int}(\Gamma \rhd M)$

(3) **output**, $(\Gamma \rhd M) \xrightarrow{c!v \rhd \eta} (\Gamma \rhd \Delta)$, where η is a non-empty set of nodes; an observer placed at any node $n \in \eta$ can receive the value v along the channel c. For this to happen each node $n \in \eta$ must

be in $\mathsf{Int}(\Gamma \rhd M)$, and there must be some code running at some node in M which can broadcast along channel c to each such n.

Formally, $(\Gamma \rhd M) \xrightarrow{c!v \rhd \eta} (\Gamma \rhd \Delta)$ whenever

(i) $(\Gamma \rhd M) \xrightarrow{c.m!v} \Delta$ for some node m

(ii) $\eta = \{ n \in \mathsf{Int}(\Gamma \rhd M) \mid \Gamma \vdash m \leftrightarrow n \} \neq \emptyset$. □

These extensional actions endow the set of networks with the structure of a pLTS. Thus the terminology used for pLTSs is extended to networks, so that in the following we will use terms such as finitary networks or finite branching networks. Also, there is a close relationship between extensional τ-actions and the reduction relation of testing structures.

Remark 5.2 (Testing structures formally unnecessary)**.** We could have defined experiments in our testing framework by relying on the extensional semantics introduced above. For this to be possible, it is necessary to introduce an operator $\mathrm{cl}(\cdot) : \mathsf{Nets} \to \mathsf{Nets}$ that isolates networks from the external environment. This operator, when applied to a network $\mathcal{M}$, simply deletes the nodes in its interface, together with the associated connectivities. Given a network $\Gamma \rhd M$, the network $\mathrm{cl}(\Gamma \rhd M) = \Gamma_M \rhd M$ is defined as

$$(\Gamma_M)_V \quad = \quad \Gamma_V \setminus \mathsf{Int}(\Gamma \rhd M)$$
$$\Gamma_M \vdash m, \Gamma_M \vdash n, \Gamma \vdash m \leftrightarrow n \quad \text{implies} \quad \Gamma_M \vdash m \leftrightarrow n$$

It is straightforward to show that the operator $\mathrm{cl}(\cdot)$ preserves well formedness, and that the reduction relation of a TS associated with an arbitrary (well formed) network $\mathcal{M}$ coincides with the extensional action $\xrightarrow{\tau}$ in $\mathrm{cl}(\mathcal{M})$.

While defining experiments by using the above operator and the extensional semantics is relatively simpler than introducing testing structures, we preferred to avoid this approach, as it leads to the introduction of complications in the proof of Theorem 5.6, below. □

In the following we will need *weak* versions of extensional actions, which abstract from internal activity, provided by the relation $\xrightarrow{\tau}$. Internal activity can be modeled by the hyper-derivation relation $\Longrightarrow$, which is a probabilistic generalisation of the more standard weak internal relation $\xrightarrow{\tau}{}^*$.

Definition 5.3. [Weak extensional actions]

(1) Let $\mathcal{M} \overset{\tau}{\Longrightarrow} \Delta$ whenever we have the hyper-derivation $\mathcal{M} \Longrightarrow \Delta$

(2) $\mathcal{M} \overset{c.n?v}{\Longrightarrow} \Delta$ whenever $\mathcal{M} \Longrightarrow \xrightarrow{c.n?v} \Longrightarrow \Delta$

(3) Let $\mathcal{M} \overset{c!v \rhd \eta}{\Longrightarrow} \mathcal{N}$ be the least relation satisfying:

(a) $\mathcal{M} \Longrightarrow \xrightarrow{c!v \rhd \eta} \Longrightarrow \Delta$ implies $\mathcal{M} \overset{c!v \rhd \eta}{\Longrightarrow} \mathcal{N}$

(b) $\mathcal{M} \overset{c!v \rhd \eta_1}{\Longrightarrow} \Delta'$, $\Delta' \overset{c!v \rhd \eta_2}{\Longrightarrow} \Delta$, where $\eta_1 \cap \eta_2 = \emptyset$, implies $\mathcal{M} \overset{c!v \rhd (\eta_1 \cup \eta_2)}{\Longrightarrow} \Delta$ □

These weak actions endow the set of networks Nets with the structure of another pLTS, called the *extensional pLTS* and denoted by $\mathsf{pLTS}_{\mathsf{Nets}}$. We can therefore use such weak actions to define a simulation preorder between networks, as in [3].

Definition 5.4. [Simulation preorder] In $\mathsf{pLTS}_{\mathsf{Nets}}$ we let $\lhd_{sim}$ denote the largest relation in $\mathsf{Nets} \times \mathcal{D}(\mathsf{Nets})$ such that if $s \lhd_{sim} \Theta$ then:

- if $\omega(s) = \text{ true}$, then $\Theta \overset{\tau}{\Longrightarrow} \Theta'$ such that for every $t \in \lceil \Theta' \rceil$, $\omega(t) = \text{ true}$

Figure 11: Ensuring soundness

- otherwise, whenever $\overline{s} \stackrel{\mu}{\Longrightarrow} \Delta'$, for $\mu \in \mathsf{Act}_\tau$, then there is a $\Theta' \in \mathcal{D}(S)$ with $\Theta \stackrel{\mu}{\Longrightarrow} \Theta'$ and $\Delta' \; \overline{\lhd_{sim}} \; \Theta'$.

We often use $s_1 \lhd_{sim} s_2$ in place of $s_1 \lhd_{sim} \overline{s_2}$. □

This is a mild generalisation of the corresponding definition in [3] where we factor in the presence of the success predicate $\omega(\,)$.

Some explanation is necessary for the non-standard definition of output actions in Definition 5.3(3). Informally speaking, the definition of weak extensional output actions expresses the capability of simulating broadcast through multicast; that is, a single broadcast action detected by a set of nodes η can be matched by a sequence of broadcast actions (possibly interrupted by internal actions), detected respectively by $\eta_1, \cdots, \eta_i \subseteq \eta$, provided that the collection $\{\eta_1, \cdots, \eta_i\}$ is a partition of η. This constraint is needed to ensure that

(i) every node in η will detect the transmitted value and
(ii) no node in η will detect the value more than once.

Example 5.5. Consider the networks $\mathcal{M}$ and $\mathcal{N}$ in Figure 8, discussed already in Example 4.16. It is easy to show that both of them can perform the weak extensional action $\stackrel{c!v \rhd \{o_1,o_2\}}{\Longrightarrow}$. However, the inference of the action is different for the individual networks; while in network $\mathcal{N}$ it is implied by the execution of a single broadcast action, detected by both nodes o_1 and o_2 simultaneously, in $\mathcal{M}$ this is implied by a sequence of weak extensional actions $\mathcal{M} \stackrel{c!v \rhd \{o_1\}}{\Longrightarrow} \stackrel{c!v \rhd \{o_2\}}{\Longrightarrow}$.

It is therefore possible to exhibit a simulation between $\mathcal{N}$ and $\mathcal{M}$, thus showing that $\mathcal{N} \lhd_{sim} \mathcal{M}$; Theorem 5.6, coming up, will show that this will implies $\mathcal{N} \sqsubseteq_{\text{may}} \mathcal{M}$.

However, in Example 4.16 we have already seen that $\mathcal{M} \not\sqsubseteq_{\text{may}} \mathcal{N}$. And indeed $\mathcal{M} \not\lhd_{sim} \mathcal{N}$ because the weak action $\mathcal{M} \stackrel{c!v \rhd \{o_1\}}{\Longrightarrow}$ cannot be matched by $\mathcal{N}$. □

We can now state the main result of the paper:

Theorem 5.6. [Soundness] Suppose $\mathcal{N}_1$, $\mathcal{N}_2$ are finitary networks. Then $\mathcal{N}_1 \lhd_{sim} \mathcal{N}_2$ in $\mathsf{pLTS}_{\mathsf{Nets}}$ implies $\mathcal{N}_1 \sqsubseteq_{\text{may}} \mathcal{N}_2$.

Section 5.2 is devoted to the proof of this theorem. Also in Section 5.4 we explain why, rather surprisingly, the converse, completeness, does not hold.

We end this section with two examples which re-inforce the delicacy of the issues involved in achieving soundness.

Example 5.7. Soundness requires that the extensional output actions records the set of target nodes, rather than single nodes. Consider $\mathcal{M} = \Gamma \rhd k_1[\![\mathbf{0}]\!] \,|\, k_2[\![c!\langle 1\rangle]\!]$ and $\mathcal{N} = \Gamma \rhd k_1[\![c!\langle 1\rangle]\!] \,|\, k_2[\![\mathbf{0}]\!]$, where

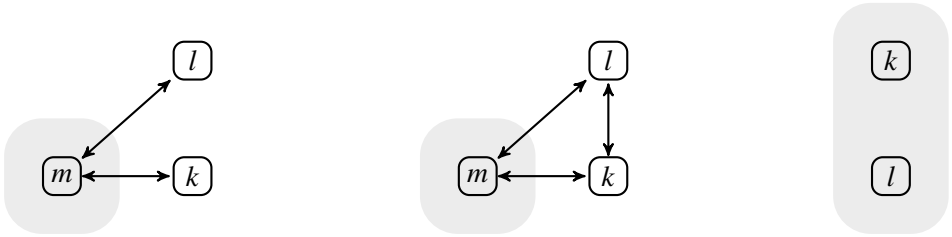


Figure 12: Ensuring soundness

the connectivity is given in Figure 11. $\mathcal{N} \not\sqsubseteq_{\text{may}} \mathcal{M}$ because of the test $\mathcal{T} = \Gamma_T \rhd T$, where T is the code $l_1[\![c?.c!]\!] | l_2[\![c?.c!]\!] | o[\![c?.c?.\omega]\!]$. Moreover $\mathcal{N} \not\lhd_{sim} \mathcal{M}$ because $\mathcal{N}$ can perform the output action labelled $c!\langle 1\rangle \rhd \{l_1, l_2\}$, which can not be matched by $\mathcal{M}$.

However suppose we were to restrict η in the definition of extensional output actions, part (3) of Definition 5.1, to be singleton sets of node names. Then in the resulting pLTS it is easy to check that $\mathcal{M}$ can simulate $\mathcal{N}$. In other words with this simplification the resulting simulations would not be sound; that is, Theorem 5.6 would no longer hold. □

Recall from Definition 3.1 we required that for $\Gamma \rhd M$ to be well-formed, it is necessary that $\Gamma \vdash k_1 \leftrightarrow k_2$ whenever k_1 and k_2 are in the interface of $\Gamma \rhd M$. The next example shows why this is also necessary to achieve soundness.

Example 5.8. Consider $\mathcal{N}_1$ and $\mathcal{N}_2$ in Figure 12, and note that the latter is not actually well-formed; it violates condition (3) in Definition 3.1. Nevertheless it is immediate that $\mathcal{N}_2 \lhd_{sim} \mathcal{N}_1$.

However $\mathcal{N}_2 \not\sqsubseteq_{\text{may}} \mathcal{N}_1$ because of the test $\mathcal{T} = \Gamma_T \rhd T$ where T is the code $l[\![c?(x).c!]\!] \mid k[\![c?(x).c?(x).c!(\omega)]\!]$. In the composite network $\mathcal{N}_2 \nparallel \mathcal{T}$ two inputs can be received at k, from m and l; but this is not possible if $\mathcal{N}_2$ is replaced by $\mathcal{N}_1$. □

5.2. **Soundness.** The proof of Soundness relies on an alternative characterisations of the simulation preorder in $\mathsf{pLTS}_{\mathsf{Nets}}$, which is essentially a simplification of what it means to be a simulation. With simulations in $\mathsf{pLTS}_{\mathsf{Nets}}$ weak actions are matched against weak actions; an alternative would be simply to require that strong actions are matched by weak actions.

Definition 5.9. [Simple simulations] In $\mathsf{pLTS}_{\mathsf{Nets}}$ we let $\lhd^s$ denote the largest relation in $\mathsf{Nets} \times \mathcal{D}(\mathsf{Nets})$ such that if $\mathcal{M} \lhd^s \Theta$ then:

- if $\omega(\mathcal{M}) = \mathbf{tt}$ for any $\omega \in \Omega$ then $\Theta \stackrel{\tau}{\Longrightarrow} \Theta'$ such that $\omega(\Theta') = \mathbf{tt}$
- otherwise,
  (i) whenever $\overline{\mathcal{M}} \stackrel{\mu}{\longrightarrow} \Delta'$ there is a $\Theta' \in \mathcal{D}(\mathsf{Nets})$ with $\Theta \stackrel{\mu}{\Longrightarrow} \Theta'$ and $\Delta' \; \overline{\lhd^s} \; \Theta'$.

Theorem 5.10. [Alternative characterisation] In $\mathsf{pLTS}_{\mathsf{Nets}}$, $\mathcal{M} \lhd_{sim} \Theta$ if and only if $\mathcal{M} \lhd^s \Theta$, provided that $\mathcal{M}$ is a finitary network, and Θ is a finitary distribution of networks (that is, every network in its support is finitary),

Proof. (Outline) Practically identical to the corresponding proof in [2]. The difficulty is to check that if $s \lhd_{sim} \Theta$ and $s \Longrightarrow\!\!> \Delta'$ then $\Theta \Longrightarrow\!\!> \Theta'$ such that $\Delta' \; \overline{\lhd^s} \; \Theta'$; see Theorem 7.20 of [2]. □

Theorem 5.11. Suppose $\mathsf{Int}(\mathcal{M}) = \mathsf{Int}(\mathcal{N})$ and both $\mathcal{M} \between (\Gamma \rhd n[\![p]\!])$ and $\mathcal{N} \between (\Gamma \rhd n[\![p]\!])$ are defined. Then $\mathcal{M} \lhd^s \mathcal{N}$ implies $\mathcal{M} \between (\Gamma \rhd n[\![p]\!]) \lhd^s \mathcal{N} \between (\Gamma \rhd n[\![p]\!])$.

The proof of this result is quite long and technical, and is therefore relegated to an independent subsection, Section 5.3 below.

Corollary 5.12. [Compositionality] Suppose $\mathsf{Int}(\mathcal{M}_1) = \mathsf{Int}(\mathcal{M}_2)$. Then $\mathcal{M}_1 \lhd^s \mathcal{M}_2$ implies $(\mathcal{M}_1 \between \mathcal{N}) \lhd_{sim} (\mathcal{M}_1 \between \mathcal{N})$ whenever both these networks are defined.

Proof. By induction on the number of nodes in $\mathcal{N}$, using the previous theorem, Theorem 5.10 and Proposition 4.8. □

We also need to relate the simulation preorder $\lhd_{sim}$ to the valuation of distributions, as given in Definition 4.12. First an auxiliary result.

Lemma 5.13. Let Δ, Θ be distributions in $\mathsf{pLTS}_{\mathsf{Nets}}$ such that $\Delta \; \overline{\lhd_{sim}} \; \Theta$; then $\Theta \Longrightarrow\!\!\!\!\gg \Theta'$ such that $\mathcal{V}(\Delta) \leq \mathcal{V}(\Theta')$.

Proof. There are two cases.

(i) First suppose Δ is a point distribution $\overline{s}$. If the predicate $\omega(s)$ is equal to false, $\mathcal{V}(\overline{s}) = 0$. In this case, we recall that Theorem 2.4 (4) ensures that there exists at least one extreme derivative Θ' of Θ, for which $0 \leq \mathcal{V}(\Theta')$ trivially holds.

Otherwise the predicate $\omega(s)$ is satisfied and $\mathcal{V}(\overline{s})$ has to be 1. Since $s \lhd_{sim} \Theta$ we know $\Theta \Longrightarrow \Theta'$ such that for all $s' \in \Theta', \omega(s') = \text{ true }$. This means that $\mathcal{V}(\Theta') = 1$; moreover, as every state in $\lceil \Theta' \rceil$ is a successful state, we also have that $\Theta \Longrightarrow\!\!\!\!\gg \Theta'$

(ii) Otherwise Θ can be written as $\sum_{s \in \lceil \Delta \rceil} \Delta(s) \cdot \Theta_s$ where $s \lhd_{sim} \Theta_s$ for each s in the support of Δ. By part (i) each $\Theta_s \Longrightarrow\!\!\!\!\gg \Theta'_s$ such that $\mathcal{V}(\overline{s}) \leq \mathcal{V}(\Theta'_s)$. As an extreme derivative is also a hyper-derivative, we can combine these to obtain a hyper derivation for Θ, using Theorem 2.4 (3). This leads to

$$\Theta = \sum_{s \in \lceil \Delta \rceil} \Delta(s) \cdot \Theta_s \Longrightarrow \sum_{s \in \lceil \Delta \rceil} \Theta'_s = \Theta'$$

As for every $s \in \lceil \Delta \rceil$, $t \in \lceil \Theta'_s \rceil$ we have that $t \xrightarrow{\tau}$ implies $\omega t = \text{ true }$, this condition is respected also by all states in $\lceil \Theta' \rceil$. Thus, the hyper derivation $\Theta \Longrightarrow \Theta'$ is also an extreme derivation. Finally, the quantity $\mathcal{V}(\Delta) = \sum\{\Delta(s) \mid \omega(s) = \text{ true }\}$ can be rewritten as $\sum_{s \in \lceil \Delta \rceil} \mathcal{V}(\overline{s})$, leading to

$$\mathcal{V}(\Delta) = \sum_{s \in \lceil \Delta \rceil} \mathcal{V}(\overline{s}) \leq \sum_{s \in \lceil \Delta \rceil} \mathcal{V}(\Theta'_s) = \mathcal{V}(\Theta') \,.$$

□

Theorem 5.14. In $\mathsf{pLTS}_{\mathsf{Nets}}$, $\Delta \; \overline{\lhd_{sim}} \; \Theta$ implies $O(\Delta) \sqsubseteq_{\text{may}} O(\Theta)$.

Proof. Suppose $\Delta \Longrightarrow\!\!\!\!\gg \Delta'$. We have to find a derivation $\Theta \Longrightarrow\!\!\!\!\gg \Theta'$ such that $\mathcal{V}(\Delta') \leq \mathcal{V}(\Theta')$. We can use the definition of $\lhd_{sim}$ to find a derivation $\Theta \Longrightarrow \Theta''$ such that $\Delta' \; \overline{\lhd_{sim}} \; \Theta''$. Applying the previous lemma we obtain $\Theta'' \Longrightarrow\!\!\!\!\gg \Theta'$ such that $\mathcal{V}(\Delta') \leq \mathcal{V}(\Theta')$. The result follows since Theorem 2.4 gives $\Theta \Longrightarrow\!\!\!\!\gg \Theta'$.

□

Proof of Theorem 5.6:

This is now a straightforward application of Compositionality and Theorem 5.14.

Let us assume that $\mathcal{N}_1 \lhd_{sim} \mathcal{N}_2$. To prove the conclusion, $\mathcal{N}_1 \sqsubseteq_{\text{may}} \mathcal{N}_2$, we must show that $\mathcal{O}(\mathcal{N}_1 \between \mathcal{T}) \sqsubseteq_{\text{may}} \mathcal{O}(\mathcal{N}_2 \between \mathcal{T})$ for an arbitrary testing network $\mathcal{T}$ such that both $\mathcal{N}_1 \between \mathcal{T}$ and $\mathcal{N}_2 \between \mathcal{T}$ are defined. For such a $\mathcal{T}$ Compositionality entails $(\mathcal{N}_1 \between \mathcal{T}) \lhd_{sim} (\mathcal{N}_2 \between \mathcal{T})$, and now we can apply Theorem 5.14. □

5.3. **Single node compositionality.** The aim of this section is to outline the proof of Theorem 5.11; it may be safely skipped by the reader uninterested in the detail. The standard approach to compositionality [16, 5] for a behavioural preorder involves proving decomposition and composition results for the actions on which the pre-order depends. As an example decomposition would involve showing that an action $P_1 \mid\mid\mid P_2 \xrightarrow{\mu} \Delta_1 \mid\mid\mid \Delta_2$ can be decomposed into two components

$$P_1 \xrightarrow{\mu_1} \Delta_1 \qquad\qquad P_2 \xrightarrow{\mu_1} \Delta_2$$

where μ_i are such that any other pairs of actions $Q_1 \xrightarrow{\mu_1} \Theta_1, \quad Q_2 \xrightarrow{\mu_1} \Theta_2$ can be recomposed into $Q_1 \mid\mid\mid Q_2 \xrightarrow{\mu} \Theta_1 \mid\mid\mid \Theta_2$.

Unfortunately such decomposition results do not hold in $\mathsf{pLTS}_{\mathsf{Nets}}$ for our operator $\between$.

Example 5.15. Let $\mathcal{M}$, $\mathcal{N}$ be defined by $(\Gamma_M \rhd m[\![c!\langle 0\rangle]\!])$, $(\Gamma_N \rhd n[\![c?.\omega]\!])$, where Γ_N is the trivial graph containing only one node, and Γ_M is determined by $\Gamma_M \vdash m \leftrightarrow n$.

Then in $\mathsf{pLTS}_{\mathsf{Nets}}$ $\mathcal{M} \between \mathcal{N} \xrightarrow{\tau} (\Gamma_M \rhd m[\![\mathbf{0}]\!]) \between (\Gamma_N \rhd n[\![\omega]\!])$. But this move can not be decomposed into individual actions in $\mathsf{pLTS}_{\mathsf{Nets}}$ from $\mathcal{M}$ and $\mathcal{N}$ respectively, as $\mathcal{N}$ cannot perform the transition $\mathcal{N} \xrightarrow{c?v} \overline{\Gamma_N \rhd n[\![\omega]\!]}$. □

Luckily, because we are only considering composition on the right hand side by single node networks, we can work with the symmetric operator $|||_{\text{m}}$ introduced in Example 4.2; as we will see this operator will support the decomposition and recomposition of actions in $\mathsf{pLTS}_{\mathsf{Nets}}$.

Proposition 5.16. Suppose $\mathcal{M} \between (\Gamma_n \rhd n[\![p]\!])$ is well-defined. Then there exists a Γ such that $(\Gamma \rhd n[\![p]\!])$ is well defined, and $\mathcal{M} \between (\Gamma_n \rhd n[\![p]\!])$ coincides with $\mathcal{M} \mid\mid\mid_{\text{m}} (\Gamma \rhd n[\![p]\!])$.

Proof. Γ can be constructed by adding to Γ_n all nodes in $\mathcal{M}$ which are connected to n; also, we require all those nodes to be connected to n in Γ. □

Proposition 5.17. [Strong decomposition in $\mathsf{pLTS}_{\mathsf{Nets}}$]

(1) If $(\Gamma_M \rhd M) \mid\mid\mid_{\text{m}} (\Gamma_n \rhd n[\![s]\!]) \xrightarrow{\tau} \Delta$
then
- either $(\Gamma_M \rhd M) \xrightarrow{\tau} (\Gamma_M \rhd \Delta_M)$ and $\Delta = (\Gamma_M \rhd \Delta_M) \mid\mid\mid_{\text{m}} \overline{\Gamma_n \rhd n[\![s]\!]}$
- or $(\Gamma_n \rhd n[\![s]\!]) \xrightarrow{\tau} (\Gamma_n \rhd n[\![\Delta_n]\!])$ and $\Delta = \overline{(\Gamma_M \rhd M)} \mid\mid\mid_{\text{m}} (\Gamma_n \rhd \Delta_n)$
- or $(\Gamma_M \rhd M) \xrightarrow{c!v\rhd\{n\}} (\Gamma_M \rhd \Delta_M)$, $(\Gamma_n \rhd n[\![s]\!]) \xrightarrow{c.m?v} (\Gamma_n \rhd n[\![\Delta_n]\!])$, with $\Gamma_n \vdash m \leftrightarrow n$ and $\Delta = (\Gamma_M \rhd \Delta_M) \mid\mid\mid_{\text{m}} \Gamma_n \rhd n[\![\Delta_n]\!]$,
- or $(\Gamma_M \rhd M) \xrightarrow{c.n?v} (\Gamma_M \rhd \Delta_M)$, $(\Gamma_n \rhd n[\![s]\!]) \xrightarrow{c!v\rhd\eta} (\Gamma_n \rhd n[\![\Delta_n]\!])$, with $\eta \subseteq \text{nodes}(M)$ and $\Delta = (\Gamma_M \rhd \Delta_M) \mid\mid\mid_{\text{m}} \Gamma_n \rhd n[\![\Delta_n]\!]$,

(2) If $(\Gamma_M \rhd M) \mid\mid\mid_{\text{m}} (\Gamma_n \rhd n[\![s]\!]) \xrightarrow{c.m?v} \Delta$ then
- either $(\Gamma_M \rhd M) \xrightarrow{c.m?v} (\Gamma_M \rhd \Delta_M)$, and $\Delta = (\Gamma_M \rhd \Delta_M) \mid\mid\mid_{\text{m}} \overline{\Gamma_n \rhd n[\![s]\!]}$,

- or $(\Gamma_n \rhd n[\![s]\!]) \xrightarrow{c.m?v} (\Gamma_n \rhd n[\![\Delta_n]\!])$ and $\Delta = \overline{\Gamma_M \rhd M} \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Delta_n]\!])$,
- or $(\Gamma_M \rhd M) \xrightarrow{c.m?v} (\Gamma_M \rhd \Delta_M)$, $(\Gamma_n \rhd n[\![s]\!]) \xrightarrow{c.m?v} (\Gamma_n \rhd n[\![\Delta_M]\!])$ and $\Delta = (\Gamma_M \rhd \Delta_M) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Delta_n]\!])$,

(3) If $(\Gamma_M \rhd M) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![s]\!]) \xrightarrow{c!v \rhd \eta} \Delta$ then

- either $\Delta = (\Gamma \rhd \Delta_M) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Delta_n]\!])$ where $(\Gamma_n \rhd n[\![s]\!]) \xrightarrow{c.m?v} (\Gamma_n \rhd n[\![\Delta_n]\!])$ for some $m \in \mathsf{Int}(\Gamma_n \rhd n[\![p]\!])$, and $(\Gamma_M \rhd M) \xrightarrow{c!v \rhd \eta \cup \{n\}} (\Gamma_M \rhd \Delta_M)$
- or $\Delta = (\Gamma_M \rhd \Delta_M) \;|||_{\mathrm{m}}\; \overline{(\Gamma_n \rhd n[\![s]\!])}$ where $(\Gamma_M \rhd M) \xrightarrow{c!v \rhd \eta} (\Gamma_M \rhd \Delta_M)$ and $n \notin \eta$.
- or $\Delta = (\Gamma_M \rhd \Delta_M) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Delta_n]\!])$, where $(\Gamma_M \rhd M) \xrightarrow{c.n?v} (\Gamma_M \rhd \Delta_M)$ and $(\Gamma_n \rhd n[\![s]\!]) \xrightarrow{c!v \rhd \eta'} (\Gamma \rhd n[\![\Delta_n]\!])$, and $\eta = \eta' \cap \mathrm{nodes}(M)$.

Proof. See the appendix. □

Next we consider how the weak actions performed by a node-stable distribution of the form $\Gamma_M \rhd \Delta \;|||_{\mathrm{m}}\; \Gamma_n \rhd n[\![\Theta]\!]$ can be inferred from those performed by $\Gamma_M \rhd \Delta$ and $\Gamma_n \rhd n[\![\Theta]\!]$ respectively.

Proposition 5.18. [Weak composition in $\mathsf{pLTS}_{\mathsf{Nets}}$] Suppose $(\Gamma_M \rhd \Delta) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Theta]\!])$ is well-defined.

(i) $(\Gamma_M \rhd \Delta) \stackrel{\tau}{\Longrightarrow} (\Gamma_M \rhd \Delta_M), \Gamma_n \rhd n[\![\Theta]\!] \stackrel{\tau}{\Longrightarrow} \Gamma_n \rhd n[\![\Theta_n]\!]$ implies $(\Gamma_M \rhd \Delta) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Theta]\!]) \stackrel{\tau}{\Longrightarrow} (\Gamma_M \rhd \Delta_M) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Theta_n]\!])$,

(ii) $(\Gamma_M \rhd \Delta) \stackrel{c!v \rhd \eta}{\Longrightarrow} \Gamma_M \rhd \Delta_M$, $n \notin \eta$ implies $(\Gamma_M \rhd \Delta) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Theta]\!]) \stackrel{c!v \rhd \eta}{\Longrightarrow} (\Gamma_M \rhd \Delta_M) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Theta]\!])$,

(iii) $(\Gamma_M \rhd \Delta) \stackrel{c!v \rhd \eta}{\Longrightarrow} (\Gamma_M \rhd \Delta_M)$, $n \in \eta$ and $(\Gamma_n \rhd n[\![\Theta]\!]) \stackrel{c.m?v}{\Longrightarrow} (\Gamma_n \rhd n[\![\Theta_n]\!])$ for some $m \in \mathsf{Int}(\Gamma_n \rhd n[\![s]\!])$ implies $(\Gamma_M \rhd \Delta) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Theta]\!]) \stackrel{c!v \rhd \eta \setminus \{n\}}{\Longrightarrow} (\Gamma_M \rhd \Delta_M) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Theta_n]\!])$,

(iv) $(\Gamma_M \rhd \Delta) \stackrel{c.n?v}{\Longrightarrow} \Gamma_M \rhd \Delta_M$ and $(\Gamma_n \rhd n[\![\Theta]\!]) \stackrel{c!v \rhd \eta}{\Longrightarrow}$ implies $(\Gamma_M \rhd \Delta) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Theta]\!]) \stackrel{c!v \rhd \eta'}{\Longrightarrow} (\Gamma_M \rhd \Delta_M) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Theta_n]\!])$, where $\eta' = \eta \setminus \mathrm{nodes}(\Delta)$.

(v) $(\Gamma_M \rhd \Delta) \stackrel{c.m?v}{\Longrightarrow} (\Gamma_M \rhd \Delta_M)$ and $(\Gamma_N \rhd n[\![s]\!]) \stackrel{c.m?v}{\Longrightarrow} (\Gamma_n \rhd p[\![\Delta_n]\!])$ implies $(\Gamma_M \rhd M) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![s]\!]) \stackrel{c.m?v}{\Longrightarrow} (\Gamma_M \rhd \Delta_M) \;|||_{\mathrm{m}}\; (\Gamma_n \rhd n[\![\Delta_n]\!])$.

Proof. See the appendix. □

Before proving Theorem 5.11 we need another result that allows us to relate the actions performed by two single node networks with different connectivities; this is because the fact that the operator $\between$ being asymmetric is reflected in the fact that the application of Proposition 5.16 to a network of the form $(\Gamma_M \rhd M) \between (\Gamma_n \rhd p[\![n]\!])$ leads to a change in the connectivity graph of the network appearing in the right hand side of the composition.

Formally, let $(\Gamma_M \rhd M)$, $(\Gamma_N \rhd N)$ and $(\Gamma_n \rhd n[\![s]\!])$ be three state based networks, and suppose both $(\Gamma_M \rhd M) \between (\Gamma_n \rhd n[\![s]\!])$ and $(\Gamma_N \rhd N) \between (\Gamma_n \rhd n[\![s]\!])$ are defined. Then

$$\begin{aligned}(\Gamma_M \rhd M) \between (\Gamma_n \rhd n[\![s]\!]) &= (\Gamma_M \rhd M) \;|||_{\mathrm{m}}\; (\Gamma_1 \rhd n[\![s]\!])\\ (\Gamma_N \rhd N) \between (\Gamma_n \rhd n[\![s]\!]) &= (\Gamma_N \rhd N) \;|||_{\mathrm{m}}\; (\Gamma_2 \rhd n[\![s]\!])\end{aligned}$$

Since the definitions of the connectivity graphs Γ_1 and Γ_2 depend on those of Γ_M and Γ_N, respectively, it is possible to obtain $\Gamma_1 \neq \Gamma_2$. Thus, when proving Theorem 5.11, we will need to deal with situations in which the two networks $\Gamma_M \rhd M$ and $\Gamma_N \rhd N$ are composed (via the $|||_{\mathrm{m}}$ operator) with networks having different connectivities. The following result, however, allows us to relate such networks:

Proposition 5.19 (Single node inputs)**.** *Let $n[\![s]\!] \in \mathsf{sSys}$, and let Γ_1, Γ_2 be two connectivity graphs such that both $\Gamma_1 \rhd n[\![s]\!]$ and $\Gamma_2 \rhd n[\![s]\!]$ are well formed. If $(\Gamma_1 \rhd n[\![s]\!]) \xrightarrow{c.m?v} (\Gamma_1 \rhd n[\![\Delta_n]\!])$ with $m \in \mathsf{Int}(\Gamma_1 \rhd n[\![s]\!])$ then $(\Gamma_2 \rhd n[\![s]\!]) \xrightarrow{c.l?v} \Gamma_2 \rhd n[\![\Delta_n]\!]$ for every $l \in \mathsf{Int}(\Gamma_2 \rhd n[\![s]\!])$.*

Proof. Straightforward from the definitions of both extensional and intensional input actions. □

Corollary 5.20 (Single node simulations)**.** *Let $\Gamma_1 \rhd n[\![s]\!]$ be a single node network with $\mathsf{Int}(\Gamma_1 \rhd n[\![s]\!]) \neq \emptyset$, and suppose $\Gamma_1 \rhd n[\![s]\!] \lhd_{sim} \Gamma_1 \rhd n[\![\Theta]\!]$; then, for any Γ_2 such that $\Gamma_2 \rhd n[\![s]\!]$ is well formed, $\Gamma_2 \rhd n[\![s]\!] \lhd_{sim} \Gamma_2 \rhd n[\![\Theta]\!]$.*

Proof. Follows directly from the definition of extensional and intensional actions and from Proposition 5.19. The constraint that $\mathsf{Int}(\Gamma_1 \rhd n[\![s]\!])$ be non-empty is needed when considering the case $(\Gamma_2 \rhd n[\![s]\!]) \xrightarrow{c.m?v} \Gamma_2 \rhd n[\![\Theta]\!]$. □

With an abuse of notation, we write $s \lhd_{sim} \Theta$ whenever $(\Gamma \rhd n[\![s]\!]) \lhd_{sim} (\Gamma \rhd \Theta[\![s]\!])$ for any Γ such that $\mathsf{Int}(\Gamma \rhd n[\![s]\!]) \neq \emptyset$.
We are now ready to prove the main result of this section.

Proof of Theorem 5.11: We actually prove a more general result. Recall that a distribution Δ over sSys is called node-stable if $\mathrm{nodes}(N) = \mathrm{nodes}(M)$ whenever $\Delta(N) > 0$ and $\Delta(M) > 0$. For such a distribution it makes sense to define $\mathsf{Int}(\Gamma \rhd \Delta)$ to be $\mathsf{Int}(\Gamma \rhd M)$ for any M such that $\Delta(M) > 0$. Now let

Let $\mathcal{R} \subseteq \mathsf{Nets} \times \mathcal{D}(\mathsf{Nets})$ be given by

$$(\Gamma_M \rhd M) \nparallel (\Gamma_n \rhd n[\![s]\!]) \;\; \mathcal{R} \;\; (\Gamma \rhd \Delta_1) \nparallel (\Gamma_n \rhd n[\![\Theta_1]\!])$$

whenever

(a) Θ_1, a distribution over sSys, is node stable
(b) $\mathsf{Int}(\Gamma_M \rhd M) = \mathsf{Int}(\Gamma \rhd \Delta_1)$
(c) $(\Gamma_M \rhd M) \lhd_{sim} (\Gamma \rhd \Delta_1)$
(d) $s \lhd_{sim} \Theta_1$; here Θ_1 is a distribution over states, from the syntax in Figure 1.

Theorem 5.11 will follow if we can show that $\mathcal{R}$ is a simple simulation, in the sense of Definition 5.9.

The proof proceeds by considering a strong extensional action

$$(\Gamma_M \rhd M) \nparallel (\Gamma_n \rhd n[\![s]\!]) \xrightarrow{\mu} \Delta \tag{5.1}$$

We must find a corresponding weak extensional action

$$(\Gamma \rhd \Delta_1) \nparallel (\Gamma_n \rhd n[\![\Theta_1]\!]) \stackrel{\mu}{\Longrightarrow} \Theta$$

such that $\Delta \; \overline{\mathcal{R}} \; \Theta$.

The first step is to employ Proposition 5.16 so as to write $(\Gamma_M \rhd M) \nparallel (\Gamma_n \rhd n[\![s]\!])$ in the form $(\Gamma_M \rhd M) \;|||_{\mathrm{m}}\; (\Gamma_1 \rhd n[\![s]\!])$ for some network connectivity Γ_n. After this translation has been carried out, we may apply Decomposition , Proposition 5.17, to the action (5.1) above. There are three cases, depending on μ. We only consider the case $\mu = c!v \rhd \eta$.

Suppose then $(\Gamma_M \rhd M) \;|||_{\mathrm{m}}\; (\Gamma_1 \rhd n[\![s]\!]) \xrightarrow{c!v \rhd \eta} (\Gamma_M \rhd \Delta_M) \;|||_{\mathrm{m}}\; (\Gamma_1 \rhd n[\![\Theta_n]\!])$. According to Proposition 5.17 we have three different sub-cases to consider; again, we will consider only the most interesting

one, namely

$$\begin{array}{rcl}(\Gamma_M \rhd M) & \stackrel{c!v\rhd\eta'}{\longrightarrow} & (\Gamma_M \rhd \Delta_M)\\ (\Gamma_1 \rhd n[\![s]\!]) & \stackrel{c.m?v}{\longrightarrow} & \Gamma_1 \rhd n[\![\Theta_n]\!]\\ \eta' = \eta \cup \{n\} & & m \in \mathsf{Int}(\Gamma_1 \rhd n[\![\Theta_n]\!])\end{array}$$

We have that $(\Gamma_M \rhd M) \lhd_{sim} (\Gamma \rhd \Delta_1)$ by hypothesis, so that $(\Gamma \rhd \Delta_1) \stackrel{c!v\rhd\eta'}{\Longrightarrow} (\Gamma \rhd \Delta_2)$ with $(\Gamma_M \rhd \Delta_M) \overline{\lhd_{sim}} (\Gamma \rhd \Delta_2)$. We can now rewrite $(\Gamma \rhd \Delta_1) \nparallel (\Gamma_n \rhd n[\![\Theta_1]\!])$ as $(\Gamma \rhd \Delta_1) \mid\mid\mid_{\mathrm{m}} (\Gamma_2 \rhd n[\![\Theta_1]\!])$. Notice also that, since $n \in \eta'$, there exists a node m in nodes(M) such that $\Gamma_M \vdash n \leftrightarrow m$. By the definition of $\mid\mid\mid_{\mathrm{m}}$, we obtain therefore that $m \in \mathsf{Int}(\Gamma_1 \rhd n[\![s]\!])$. Thus, we can apply both Proposition 5.19 and Corollary 5.20 to infer $(\Gamma_2 \rhd n[\![\Theta_1]\!]) \stackrel{c.l?v}{\Longrightarrow} (\Gamma_2 \rhd n[\![\Theta_2]\!])$ for some $l \in \mathsf{Int}(\Gamma_2 \rhd n[\![\Theta_n]\!])$, and $\Theta_n \overline{\lhd_{sim}} \Theta_2$.
Thus we have proved

$$\begin{array}{rcl}\Gamma \rhd \Delta_1 & \stackrel{c!v\rhd\eta'}{\Longrightarrow} & \Gamma \rhd \Delta_2,\\ \Gamma_2 \rhd n[\![\Theta_1]\!] & \stackrel{c.l?v}{\Longrightarrow} & \Gamma_2 \rhd n[\![\Theta_2]\!], l \in \mathsf{Int}(\Gamma_2 \rhd n[\![\Theta_2]\!]),\\ \Gamma_M \rhd \Delta_M & \overline{\lhd_{sim}} & \Gamma \rhd \Delta_2,\\ \Theta_n & \overline{\lhd_{sim}} & \Theta_2.\end{array}$$

The first two results can be used together with Decomposition to prove $\Gamma \rhd \Delta_1 \mid\mid\mid_{\mathrm{m}} \Gamma_2 \rhd n[\![\Theta_n]\!] \stackrel{c!v\rhd\eta}{\Longrightarrow} \Gamma \rhd \Delta_2 \mid\mid\mid_{\mathrm{m}} \Gamma_2 \rhd n[\![\Theta_2]\!]$, while the last two allow us to infer $\Gamma_M \rhd \Delta_M \mid\mid\mid_{\mathrm{m}} \Gamma_1 \rhd n[\![\Theta_n]\!] \overline{\mathcal{R}}$ $\Gamma_M \rhd \Delta_2 \mid\mid\mid_{\mathrm{m}} \Gamma_2 \rhd n[\![\Theta_2]\!]$. □

5.4. **Simulation preorder fails to be complete.** Although the simulation preorder $\lhd_{sim}$ provides a proof methodology for establishing that two networks are be related via the testing preorder $\sqsubseteq_{\mathrm{may}}$, it is not complete. That is, it is possible to find two networks $\mathcal{M}, \mathcal{N}$ such that $\mathcal{M} \sqsubseteq_{\mathrm{may}} \mathcal{N}$ holds, but $\mathcal{M}$ cannot be simulated by $\mathcal{N}$. This is quite surprising, as simulation preorder has been already proved to provide a characterisation of the may-testing preorder for more standard process calculi, such as pCSP [2].

However, in our setting a problem arises; the mathematical basis of simulation preorders rely on (full) probability distributions, which are a suitable tool in a framework where a weak action from a process term has to be matched with the same action performed by a distribution of processes.

This is not true in our calculus; we have already shown that, due to the presence of local broadcast communication, it is possible to match a weak broadcast action with a sequence of outputs whose sets of target nodes are pairwise disjoint. This behaviour has been formalised by giving a non-standard definition of weak extensional actions in Definition 5.3.

Such a definition captures the possibility of simulating a broadcast through a multicast only when the former action is performed with probability 1. However, when comparing distributions of networks we have to also match actions which are performed with probabilities less than 1, at least informally; here the simulation of broadcast using multicast runs into problems, as the following example shows.

Example 5.21. Consider the two network distributions $\Gamma_M \rhd \Delta$, $\Gamma_N \rhd \Theta$ depicted in Figure 13; let

$$\begin{array}{rcl}\Delta & = & 0.81 \cdot m[\![c!\langle v\rangle\,.\,\mathbf{0}]\!] + 0.19 \cdot m[\![\mathbf{0}]\!]\\ \Theta & = & 0.9 \cdot m[\![c!\langle v\rangle\,.\,\mathbf{0}]\!] \mid n[\![P]\!] + 0.1 \cdot m[\![\mathbf{0}]\!] \mid n[\![P]\!]\end{array}$$

$\Delta = \Gamma_M \rhd (0.81 \cdot m[\![c!\langle v\rangle \,.\, \mathbf{0}]\!] + 0.19 \cdot m[\![\mathbf{0}]\!])$

$\Theta = \Gamma_N \rhd (0.9 \cdot m[\![c!\langle v\rangle \,.\, \mathbf{0}]\!] \mid n[\![P]\!] + 0.1 \cdot m[\![\mathbf{0}]\!] \mid n[\![P]\!])$

Figure 13: Two testing related networks

where P is the process $c?(x)\,.(c!\langle x\rangle\,.\,\mathbf{0}_{0.9}\oplus\,\mathbf{0}) + c?(x)\,.P$. In $\Gamma_M \rhd \Delta$ a message is broadcast to nodes o_1, o_2 with probability 0.81, while in $\Gamma_N \rhd \Theta$ two different broadcasts happen in sequence (first to node o_1, then to o_2). Each of this broadcast happens with probability 0.9, so that the overall probability of message v to be detected by both nodes o_1, o_2 is again 0.81.

We first show that $\Gamma_M \rhd \Delta \sqsubseteq_{\text{may}} \Gamma_N \rhd \Theta$, then we prove that $\Gamma_M \rhd \Delta \not\simeq_{sim} \Gamma_N \rhd \Theta$.
For the first statement, we only supply informal details, as a complete proof would be rather long and technical. Consider a test distribution $\Gamma_T \rhd \Lambda$, such that both $\Gamma_M \rhd \Delta \between \Gamma_T \rhd \Lambda$ and $\Gamma_N \rhd \Theta \between \Gamma_T \rhd \Lambda$ are defined. Without loss of generality, suppose that both $o_1, o_2 \in \text{nodes}(\Gamma_T \rhd \Lambda)$, thus every $T \in \lceil \Lambda \rceil$ can be written in $o_1[\![t_1]\!] \mid o_2[\![t_2]\!] \mid T$. Also, we provide details only for the most interesting case, that is when the testing component reaches (with some probability p) an ω-successful configuration after network $\mathcal{M}$ broadcasts the message v. In this case, a computation fragment of $\Gamma_M \rhd \Delta \between \Gamma_T \rhd \Lambda$ can be summarised as follows:

(1) The testing component $\Gamma_T \rhd \Lambda$ performs some internal activity, thus leading to $\Gamma_T \rhd \Lambda \stackrel{\tau}{\Longrightarrow} \Gamma_T \rhd o_1[\![\Lambda_1]\!] \mid o_2[\![\Lambda_2]\!] \mid \Lambda_T$
(2) At this point, the distribution Δ will broadcast the message with probability 0.81, causing the testing component to evolve in $\Gamma_T \rhd o_1[\![\Lambda'_1]\!] \mid o_2[\![\Lambda'_2]\!] \mid \Lambda_T$[1]. The tested component, at this point, will be in a deadlocked configuration, that is it cannot perform any action.

Consider now the distribution $\Gamma_N \rhd \Theta \between \Gamma_T \rhd \Lambda$. For such a network, a matching computation will proceed as follows:

(1) The testing component $\Gamma_T \rhd \Lambda$ performs the same sequence of internal activities as before, thus it will end up in the distribution $\Gamma_T \rhd o_1[\![\Lambda_1]\!] \mid o_2[\![\Lambda_2]\!] \mid \Lambda_T$.
(2) At this point, message v will be broadcast by Θ to node o_1. This happens with probability 0.9, and it causes the testing network to evolve in $\Gamma_T \rhd o_1[\![\Lambda'_1]\!] \mid o_2[\![\Lambda_2]\!] \mid \Theta_T$. Here note that, since the broadcast can not be heard by node o_2, the probability distribution for such a node in the testing component has not been affected.
(3) Before allowing the testing component to perform any activity, we require the tested network to perform the second broadcast, which will be heard by node o_2; again, this will happen with probability 0.9 and it will not affect the probability distribution of processes running at node o_1. Thus, after the second message has been broadcast by the tested network, the testing component will have the form $\Gamma_T \rhd o_1[\![\Lambda'_1]\!] \mid o_2[\![\Lambda'_2]\!] \mid \Lambda_T$, which is exactly the same configuration obtained in the first experiment, after $\Gamma_M \rhd \Delta$ has broadcast the

[1]Note that only nodes o_1 and o_2 are affected by the broadcast performed by node m

message to both locations. Further, note that the overall probability of Θ delivering both messages is again 0.81, and that the tested network is now in a deadlocked configuration.

Thus we have shown that, whenever the broadcast of message v by $\Gamma_M \rhd \Delta$ affects the testing network $\Gamma_T \rhd \Lambda$ in some way, then $\Gamma_N \rhd \Theta$ is able to multicast the message to both o_1 and o_2, causing $\Gamma_T \rhd \Lambda$ to behave in the same way. Note also that in $\Gamma_N \rhd \Theta$ we introduced a non-deterministic choice in process P; this choice has been introduced because it is possible for node n to receive messages from the external node o_2. Since in $\Gamma_M \rhd \Delta$ messages received from external nodes do not affect the behaviour of the network, we require $\Gamma_N \rhd \Theta$ to have at least an extreme derivative in which this policy is respected. In fact, when $\Gamma_N \rhd \Theta$ receives a message from o_2, code P running at node n can decide to ignore the message by evolving to the process P itself. At this point, the reader should be convinced that $\Gamma_M \rhd \Delta \sqsubseteq_{\text{may}} \Gamma_N \rhd \Theta$.

Now we show that it is the case that $\Gamma_M \rhd \Delta$ cannot be simulated by $\Gamma_N \rhd \Theta$. The proof is obtained by contradiction. Suppose then that $\Gamma_M \rhd \Delta \ \overline{\lhd_{sim}}\ \Gamma_N \rhd \Theta$. Let M_1 be the system term $m[\![c!\langle v\rangle]\!]$. As $\Gamma_M \rhd M_1 \xrightarrow{c!v\rhd\{o_1,o_2\}}$, and $\Delta = 0.81 \cdot \overline{\Gamma_M \rhd M_1} + 0.19 \cdot \overline{\Gamma_M \rhd m[\![\mathbf{0}]\!]}$, we can rewrite Θ as

$$\Theta = 0.81 \cdot \Theta_1 + 0.19 \cdot \Theta_2$$

such that $\Theta_1 \overset{c!v\rhd\{o_1,o_2\}}{\Longrightarrow}$. Let now N_1 and N_2 be the state based terms $m[\![c!\langle v\rangle]\!] \,|\, n[\![P]\!]$, and $m[\![\mathbf{0}]\!] \,|\, n[\![P]\!]$, respectively. These terms have been defined so that $\lceil \Gamma_N \rhd \Theta \rceil = \{\Gamma_N \rhd N_1, \Gamma_N \rhd N_2\}$. Since $\Gamma_N \rhd N_2$ is a deadlocked network (hence it cannot perform any output action), the only network in the support of Θ_1 has to be $\Gamma_N \rhd N_1$, for a distribution can perform an action only if all the networks in its support can perform the same action.

It is easy to show that the only possible action for $\Gamma_N \rhd N_1$ is $\Gamma_N \rhd N_1 \overset{c!v\rhd\{o_1\}}{\Longrightarrow} \Gamma_N \rhd \Theta''$, where $\Theta'' = \overline{m[\![\mathbf{0}]\!]} \mid \Theta_N$ and $\Theta_N = 0.9 \cdot \overline{n[\![c!\langle v\rangle]\!]} + 0.1 \cdot \overline{n[\![\mathbf{0}]\!]}$. Since the latter is a deadlock state, we can conclude that the action $\Gamma_N \rhd \Theta'' \overset{c!v\rhd\{o_2\}}{\Longrightarrow}$ is not possible, so neither is $\Gamma_N \rhd N_1 \overset{c!v\rhd\{o_1,o_2\}}{\Longrightarrow}$. It follows that the broadcast action performed by $\Gamma_M \rhd M_1$ cannot be matched by $\Gamma_N \rhd N_1$, and hence $\Gamma_M \rhd \Delta \ \overline{\not\lhd_{sim}}\ \Gamma_N \rhd \Theta$. □

The example above has more serious consequences than just showing that simulation preorder is not complete with respect to the may testing preorder. In fact, distributions $\Gamma_M \rhd \Delta$, $\Gamma_N \rhd \Theta$ from Example 5.21 can be used to prove that the relation $\sqsubseteq_{\text{may}}$ does not enjoy the decomposition property of Definition 2.2, a property inherent to lifted relations. In other words, it is not possible to obtain $\sqsubseteq_{\text{may}}$ as the lifting of any relation between networks and distribution of networks.

Theorem 5.22. *There exists no relation $\mathcal{R} \subseteq \mathsf{Nets} \times \mathcal{D}(\mathsf{Nets})$ such that the testing preorder $\sqsubseteq_{may}$ coincides with $\overline{\mathcal{R}}$.*

Proof. The proof is carried out by contradiction. Suppose $\mathcal{R} \subseteq \mathsf{Nets} \times \mathcal{D}(\mathsf{Nets})$ is a relation which coincides with the $\sqsubseteq_{\text{may}}$ testing preorder, and consider the distributions $\Gamma_M \rhd \Delta, \Gamma_N \rhd \Theta$ from Example 5.21. We have already proved that $\Gamma_M \rhd \Delta \sqsubseteq_{\text{may}} \Gamma_N \rhd \Theta$ and so by the hypothesis we have $\Gamma_M \rhd \Delta \ \overline{\mathcal{R}}\ \Gamma_N \rhd \Theta$. Let $M_1 = m[\![c!\langle v\rangle.\,\mathbf{0}]\!]$, $M_2 = m[\![\mathbf{0}]\!]$; we can rewrite Δ as $0.81M_1 + 0.19 \cdot M_2$.

By the Definition of lifting 2.2, we can rewrite Θ as $0.81 \cdot \Theta_1 + 0.19 \cdot \Theta_2$, for some distributions Θ_1 and Θ_2 such that $\Gamma_M \rhd M_i \ \mathcal{R}\ \Gamma_N \rhd \Theta_i$, $i = 1, 2$. In particular this leads to $\overline{\Gamma_M \rhd M_1}\ \overline{\mathcal{R}}\ \Gamma_N \rhd \Theta_1$, or equivalently $\Gamma_M \rhd M_1 \sqsubseteq_{\text{may}} \Gamma_N \rhd \Theta_1$. The contradiction is provided by showing that this is not possible for any possible Θ_1.

Consider the test $\mathcal{T} = \Gamma_T \rhd T$, where Γ_T is the connectivity graph depicted in Figure 14 and $T = o_1[\![c?(x).d!\langle\cdot\rangle]\!] \,|\, o_2[\![c?(x).d?(x).\omega]\!]$. It is easy to show that $1 \in O(\overline{\Gamma_M \rhd M_1} \not\parallel \overline{\mathcal{T}})$; as $\Gamma_M \rhd M_1 \sqsubseteq_{\text{may}} \Gamma_N \rhd \Theta_1$, we also have $1 \in O(\Gamma_N \rhd \Theta_1 \not\parallel \overline{\mathcal{T}})$. For this to happen, each network in $\lceil \Gamma_N \rhd \Theta_1 \rceil$ has to be

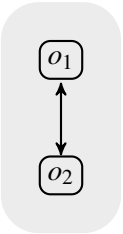


Figure 14: A connectivity graph for testing distributions $\Gamma_M \rhd \Delta$ and $\Gamma_N \rhd \Theta$

able to broadcast a value to node o_1. Since $\Theta = 0.81 \cdot \Theta_1 + 0.19 \cdot \Theta_2$, we also have that $\lceil \Theta_1 \rceil \subseteq \lceil \Theta \rceil$. As the only network in $\lceil \Theta \rceil$ which is able to broadcast a value to node o_1 is $N = m[\![c!v.\mathbf{0}]\!] \mid n[\![P]\!]$, we have $\Theta_1 = \overline{N}$.

Now note that the only transition from $\Gamma_N \rhd N \between \mathcal{T}$ is given by $\Gamma_N \rhd N \between \mathcal{T} \xrightarrow{\tau} \Gamma_N \rhd \Theta'_1 \between \mathcal{T}'$, where $\Theta'_1 = 0.9 \cdot m[\![\mathbf{0}]\!] \mid n[\![c!\langle v\rangle]\!] + 0.1 \cdot m[\![\mathbf{0}]\!] \mid n[\![\mathbf{0}]\!]$ and $\mathcal{T}' = \Gamma_T \rhd o_1[\![d!\langle\cdot\rangle]\!] \mid o_2[\![d?(x).c?(x).\omega]\!]$. Since $1 \in \mathcal{O}(\Gamma_N \rhd N \between \mathcal{T})$ and this is the unique transition from $\Gamma_N \rhd N \between \mathcal{T}$ it follows that $1 \in \mathcal{O}(\Gamma_N \rhd \Theta'_1 \between \mathcal{T}')$. Again, this would require each network in $\lceil \Gamma_N \rhd \Theta'_1 \rceil$ to be able to broadcast a value to node o_2. However, this is not possible, as the deadlocked network $\Gamma_N \rhd m[\![\mathbf{0}]\!] \mid n[\![\mathbf{0}]\!]$ is included in $\lceil \Gamma_N \rhd \Theta'_1 \rceil$.

Thus we have provided the contradiction, by showing $\Gamma_M \rhd M_1 \sqsubseteq_{\text{may}} \Gamma_N \rhd \Theta_1$ is not possible. □

6. Application: probabilistic routing

In this Section we provide an application of our theory; we decided to focus on a simple routing model to show how networks can be related via the may testing preorder by providing a simulation between them. We first define a specification (or model) for routing in terms of a network $\mathcal{M}$; then we consider a more complicated network $\mathcal{N}$ and we show that it is may testing related to our model by exhibiting a simple simulation between $\mathcal{M}$ and $\mathcal{N}$; by Theorem 5.6, it follows that $\mathcal{M} \sqsubseteq_{\text{may}} \mathcal{N}$. Finally, we generalize our result by focusing on a network $\mathcal{L}$ which is only partially defined; again, we prove that $\mathcal{M}$ and $\mathcal{L}$ are testing related by showing that $\mathcal{M} \lhd^s \mathcal{L}$.

6.1. **The specification.** Routing is the central task that has to be accomplished in the network layer of (wireless) network protocols [19]. The goal of a routing protocol is that of guaranteeing that a message, generated by a node of the network and intended for a second, flows through the nodes of the network to eventually reach the desired destination. The design of routing protocols relies on the assumption that the communication between two nodes is perfect; in practice, this task is accomplished by the Datalink and MAC layers, while in our calculus this is guaranteed by the intensional semantics that define network transitions.

Here we propose a basic network to model routing; as we will see, the way it is defined ensures it enjoys the following features:

- Two extenal nodes o_1, o_2, are used as the endpoints of a communication; a message generated by node o_1 has to be forwarded to o_2, and vice-versa. The constraint that o_1, o_2 are external nodes guarantees that messages are generated non-deterministically,
- The network detects the messages generated by the external nodes only along a single channel; all the messages generated via other channels are ignored,

- Routing is sequential; that is, only one packet at time can be routed.

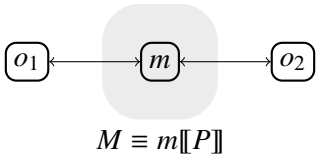


Figure 15: A model for routing

Our model consists in the network $\mathcal{M} = \Gamma \rhd M$ depicted in Figure 15. We define P to be the process $c?(x).c!\langle x\rangle.P$. The role of the internal node is that of repeatedly listening for incoming messages (either from o_1 or o_2); once a message has been received, it will forward it to the destination by performing a broadcast. In this case the message is heard by both the external nodes o_1 and o_2, thus we ensure that it will reach the destination node. This is needed because it is impossible for node m, upon receiving a message, to detect if it was originally sent by o_1 or o_2. Further, it is easy to check that the network $\mathcal{M}$ satisfies the constraints above. Finally, in order to ensure that the pLTS generated by network $\mathcal{M}$ is finitary, we assume that the sets of both channels and values are finite.

The pLTS induced by network $\mathcal{M}$ is depicted in Figure 16. Below we summarise the actions that network $\mathcal{M}$ can perform:

(1) $\mathcal{M} \xrightarrow{d.o_i?v} \overline{\mathcal{M}}$, provided channel d is different from c, $i = 1, 2$,

(2) $\mathcal{M} \xrightarrow{c.o_i?v} \overline{\mathcal{M}'_v}$, $i = 1, 2$, where $\mathcal{M}'_v = \Gamma \rhd m[\![c!\langle v\rangle.P]\!]$.

We also list the possible actions that can be performed by the derivative of $\mathcal{M}$, $\mathcal{M}'_v$.

(1) $\mathcal{M}'_v \xrightarrow{d.o_i?v} \overline{\mathcal{M}'}$, where d is an arbitrary channel (including c),

(2) $\mathcal{M}'_v \xrightarrow{c!v\rhd\eta} \overline{\mathcal{M}}$, where $\eta = \{o_1, o_2\}$.

6.2. **A simple implementation.** Now that we have provided a specification for routing, let us look at a possible implementation. In our framework, for implementation of $\mathcal{M}$ we mean a network $\mathcal{N} = \Gamma_N \rhd N$ such that $\mathcal{M} \sqsubseteq_{\text{may}} \mathcal{N}$. The main idea here is to build $\mathcal{N}$ by replacing node m in $\mathcal{M}$ with a rather simple network, consisting of different nodes. The main goal we want to achieve for $\mathcal{N}$ consists in routing a message generated from o_1 to o_2, and vice versa. To this end, we design $\mathcal{N}$ to have at least one computation in which the constraints imposed for $\mathcal{M}$ are satisfied, thereby ensuring that $\mathcal{M} \sqsubseteq_{\text{may}} \mathcal{N}$ will be true.

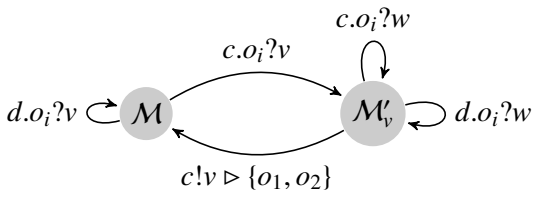


Figure 16: the pLTS induced by our routing model. Here d is an arbitrary channel different from c, v and w are arbitrary messages and i ranges over $1, 2$.

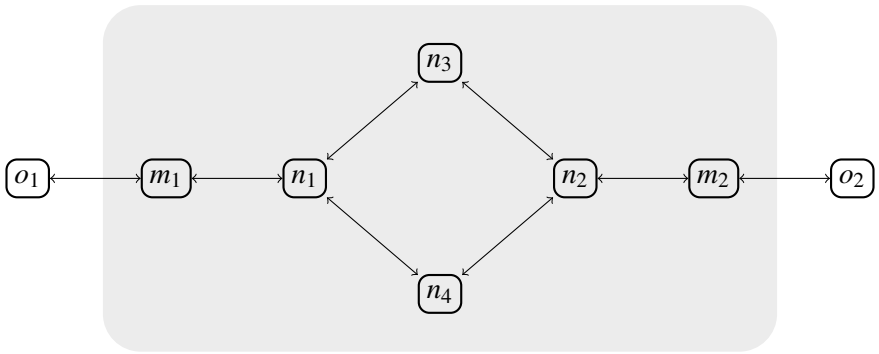


Figure 17: A simple implementation of $\mathcal{M}$

The network $\mathcal{N}$ we consider is depicted in Figure 17. Here N is defined to be

$$m_1[\![P_m]\!] \mid m_2[\![P_m]\!] \mid \prod_{i=1}^{4} n_i[\![P_i]\!]$$

where

$$\begin{array}{rcll} P_m & = & c?(x).c!\langle x\rangle.P_m + c?(x).P_m & \\ P_i & = & c?(x).[(c_3!\langle x\rangle.P_i \,{}_{\frac{1}{2}}\!\oplus c_4!\langle x\rangle.P_i)] + c?(x).P_i & \\ & + & c_i?(x).[(c_3!\langle x\rangle.P_i \,{}_{\frac{1}{2}}\!\oplus c_4!\langle x\rangle.P_i)] + c_i?(x).c!\langle x\rangle.P_i, \quad i = 1,2 & (6.1) \\ P_i & = & c_i?(x).(c_1!\langle x\rangle \,{}_{\frac{1}{2}}\!\oplus c_2!\langle x\rangle) \quad i = 3,4 & \end{array}$$

Let us discuss the intuitive behaviour of each node in network $\mathcal{N}$. The idea is that of implementing probabilistic routing; once a message is received by a node in the network, it will perform a probabilistic choice to select a node, among its neighbours, to which the message will be forwarded. For this purpose, each internal node $n_i, i = 1, \cdots, 4$, has a channel c_i associated to it; the code of network $\mathcal{N}$ is designed so that each of these internal nodes n_i waits for a message to be received along its associated channel c_i. Further, it is the only node in the network which can receive messages along this channel; this ensures that whenever a message is broadcast along channel c_i only node n_i is able to actually receive it.

The behaviour of an internal nodes n_3 or n_4 is straightforward; if it receives a message, which may come from either of its neighbours, it selects according to a fair probabilistic choice one of these neighbours to whom the message is forwarded.

The behaviour of n_1, n_2 is more complicated. We describe that of n_1; the behaviour of n_2 is symmetric. If it receives a message along channel c_1, its associated channel, we know that it must come from one of its internal neighbours n_3 or n_4. Non-deterministically, the message is either

- forwarded to the externally connected node m_1 using the channel c
- or using a fair probabilistic choice it is rebroadcast back to one of its internal neighbours, along their associated channels c_j.

But n_1 can also listen to messages broadcast along channel c; this allows it to receive messages from the node m_1, which in turn is connected to the interface. When such a message is received nondeterministically it is either

- ignored

- or forwarded to one if the internal nodes n_3, n_4, using their associated channels c_j; the destination node is selcted randamly, using a fair probabilistic choice.

The nodes m_1, m_2, being connected to the interface, are responsible for routing messages between the external nodes o_1, o_2 and the internal ones n_1, n_2 respectively. This is achieved by m_i forwarding every message it receives along channel c; we have already seen that its neighbour n_i is both broadcasting and listening on c. However, node m_1 can also decide non-deterministically to discard any of these messages.

Note that the behaviour of network $\mathcal{N}$ is decidedly more complicated of the routing model $\mathcal{M}$. For example, it is possible in $\mathcal{N}$ that the task of routing a message fails before being completed; as nodes m_i, n_i, $i = 1, 2$ can non-deterministically ignore messages received along channel c, a message can be lost at one of these nodes before the routing activity is completed.

Despite having a more complicated behaviour, network $\mathcal{N}$ simulates the routing model $\mathcal{M}$. However this simulation is far from being trivial. For example in $\mathcal{M}$ when a message is received from o_1 it is broadast simultaneously to both nodes in the interface o_1 and o_2. But in $\mathcal{N}$ there is no node directly connected to both o_1 and o_2 and so this behaviour can not be replicated. Moreover all communication between the individual nodes in $\mathcal{N}$ is probabilistic, so that the multi-cast which simulates this broadcast is only achieved in the *probabilistic limit.*

We show that $\mathcal{M} \lhd^s \mathcal{N}$. From Theorem 5.10 it will follow that that $\mathcal{M} \lhd_{sim} \mathcal{N}$, and therefore by Soundness, Theorem 5.6 we will have $\mathcal{M} \sqsubseteq_{\text{may}} \mathcal{N}$.

In order to show that $\mathcal{M} \lhd^s \mathcal{N}$, we have to exhibit a simulation between them; this is facilitated by introducing some suitable notation. We define the system term $N_{m_1} = m_2[\![P_m]\!] \mid \prod_{i=1}^{4} n_i[\![P_i]\!]$. That is, N_{m_1} is the term obtained by removing from N the code from node m_1. Similar definitions apply for each node in nodes(N). For any message v, let $\mathcal{N}_v^1 = \Gamma_N \rhd m_1[\![c!\langle v\rangle.P_m]\!] \mid N_{m_1}$, $\mathcal{N}_v^2 = \Gamma_N \rhd m_2[\![c!\langle v\rangle.P_m]\!] \mid N_{m_2}$.

Now we show that the relation

$$\mathcal{S} = \{(\mathcal{M}, \overline{\mathcal{N}})\} \cup \{(\mathcal{M}'_v, \overline{\mathcal{N}_v^1}), (\mathcal{M}'_v, \overline{\mathcal{N}_v^2}) \mid v \in \mathcal{V}\}$$

satisfies the requirements of Definition 5.9.

Let us first look at the pair $(\mathcal{M}, \overline{\mathcal{N}})$. Recall that network $\mathcal{M}$ has only four possible actions (up to the choice of a message v):

- $\mathcal{M} \xrightarrow{d.o_1?v} \overline{\mathcal{M}}$, where $d \neq c$. We need to match this action with a derivation of the form $\overline{\mathcal{N}} \stackrel{d.o_1?v}{\Longrightarrow} \Theta$ for some Θ such that $(\overline{\mathcal{M}}, \Theta) \in \overline{\mathcal{S}}$. It is not difficult to note that $\overline{\mathcal{N}} \xrightarrow{d.o_1?v} \overline{\mathcal{N}}$, as none of the nodes m_1 and m_2 (which are the only one which can detect messages broadcast from external nodes) is waiting to receive a value on channel d. By Definition 2.2 we have $(\overline{\mathcal{M}}, \overline{\mathcal{N}}) \in \overline{\mathcal{S}}$.
- $\mathcal{M} \xrightarrow{d.o_2?v} \overline{\mathcal{M}}$, where $d \neq c$. This case is analogous to the one above
- $\mathcal{M} \xrightarrow{c.o_1?v} \overline{\mathcal{M}'_v}$. In this case it is easy to show that $\overline{\mathcal{N}} \xrightarrow{c.o_1?v} \overline{\mathcal{N}_v^1}$, and $(\overline{\mathcal{M}'_v}, \overline{\mathcal{N}_v^1}) \in \overline{\mathcal{S}}$.
- $\mathcal{M} \xrightarrow{c.o_2?v} \overline{\mathcal{M}'_v}$. As above, one can check that $\overline{\mathcal{N}} \xrightarrow{c.o_1?v} \overline{\mathcal{N}_v^2}$, and $(\overline{\mathcal{M}'_v}, \mathcal{N}_v^2) \in \overline{\mathcal{S}}$.

It remains to check the pairs of the form $(\mathcal{M}'_v, \overline{\mathcal{N}_v^1})$ and $(\mathcal{M}'_v, \overline{\mathcal{N}_v^2})$. We only supply the details for the former case, as the latter one is analogous.

- $\mathcal{M}'_v \xrightarrow{d.o_1?v} \overline{\mathcal{M}'_v}$, where d is an arbitrary channel, including c. Note that in $\mathcal{N}_v^1$ the node m_1 is not waiting to receive a message along any channel. That is, we have $\mathcal{N}_v^1 \xrightarrow{d.o_1?v} \overline{\mathcal{N}_v^1}$, and $(\overline{\mathcal{M}_v^1}, \overline{\mathcal{N}_v^1}) \in \overline{\mathcal{S}}$

- $\mathcal{M}'_v \xrightarrow{d.o_2?v} \overline{\mathcal{M}'_v}$, where d is an arbitrary channel, included c. If $d \neq c$, then this case is similar to the above one. If $d = c$, note that node m_2 is waiting to receive a message along channel c. However, we already remarked that node m_2 can non-deterministically choose to ignore messages broadcast along channel c, so that it is easy to derive $\mathcal{N}^1_v \xrightarrow{c.o_2?v} \overline{\mathcal{N}^1_v}$. Now it suffices to note that $(\overline{\mathcal{M}^1_v}, \overline{\mathcal{N}^1_v}) \in \mathcal{S}$.
- $\mathcal{M}'_v \xrightarrow{c!v\triangleright\{o_1,o_2\}} \overline{\mathcal{M}}$. This is the most interesting case. In fact, it is not possible to match the strong extensional output performed by $\mathcal{M}'_v$ directly. Rather, we exhibit a weak derivation of the form $\mathcal{N}^1_v \overset{c!v\triangleright\{o_1,o_2\}}{\Longrightarrow} \overline{\mathcal{N}}$. This is obtained by exploiting the non-standard definition of weak extensional outputs, given in Definition 5.3(3). Specifically, we show that that there exist Δ_1 and Δ_2 such that

$$\mathcal{N}^1_v \xrightarrow{c!v\triangleright\{o_1\}} \Delta_1 \overset{\tau}{\Longrightarrow} \Delta_2 \xrightarrow{\tau} \overline{\mathcal{N}^2_v} \xrightarrow{c!v\triangleright\{o_2\}} \overline{\mathcal{N}}. \tag{6.2}$$

At this point, since $(\overline{\mathcal{M}}, \overline{\mathcal{N}}) \in \overline{\mathcal{S}}$ the proof is finished.

In order to provide the sequence of derivations (6.2) above, we use

$$\begin{aligned}\Delta_1 &= \frac{1}{2}\cdot\overline{\Gamma_N \triangleright n_1[\![c_3!\langle v\rangle.P_1]\!] \mid N_{n_1}} + \frac{1}{2}\cdot\overline{\Gamma_N \triangleright n_1[\![c_4!\langle v\rangle.P_1]\!] \mid N_{n_1}},\\ \Delta_2 &= \overline{\Gamma_N \triangleright n_2[\![c!\langle v\rangle.P_2]\!] \mid N_{n_2}}\end{aligned}$$

The transitions $\mathcal{N}^1_v \xrightarrow{c!v\triangleright\{o_1\}} \Delta_1$, $\Delta_2 \xrightarrow{\tau} \overline{\mathcal{N}^2_v}$ and $\mathcal{N}^2_v \xrightarrow{c!v\triangleright\{o_2\}} \overline{\mathcal{N}}$ are easy to derive. In the latter, to obtain the pointed distribution $\overline{\mathcal{N}}$ as the result of the transition, we exploited the ability of node n_2 to ignore messages received along channel c. The only difficulty lies in exhibiting the hyper-derivation $\Delta_1 \overset{\tau}{\Longrightarrow} \Delta_2$.

First, note that each network in the support of Δ_1 can perform a τ-action. Specifically, we have $\Gamma_N \triangleright n_1[\![c_3!\langle v\rangle.P_1]\!] \mid N_{n_1} \xrightarrow{\tau} \Gamma_N \triangleright \Delta_3$, where

$$\Delta_3 = \frac{1}{2}\cdot\overline{\Gamma_N \triangleright n_3[\![c_1!\langle v\rangle.P_3]\!] \mid N_{n_3}} + \frac{1}{2}\cdot\overline{\Gamma_N \triangleright n_1[\![c_2!\langle v\rangle.P_3]\!] \mid N_{n_3}}$$

and $\Gamma_N \triangleright n_1[\![c_4!\langle v\rangle.P_1]\!] \mid N_{n_1} \xrightarrow{\tau} \Gamma_N \triangleright \Delta_4$, where

$$\Delta_4 = \frac{1}{2}\overline{\Gamma_N \triangleright n_4[\![c_1!\langle v\rangle.P_4]\!] \mid N_{n_4}} + \frac{1}{2}\overline{\Gamma_N \triangleright n_4[\![c_2!\langle v\rangle.P_4]\!] \mid N_{n_4}}.$$

The last two derivations ensure that $\Delta_1 \xrightarrow{\tau} \frac{1}{2}\cdot\Delta_3 + \frac{1}{2}\cdot\Delta_4$. In a similar way, we can derive the following τ transitions for Δ_3 and Δ_4:

(1) $\Delta_3 \xrightarrow{\tau} \frac{1}{2}\cdot\Delta_1 + \frac{1}{2}\cdot\Delta_2$,

(2) $\Delta_4 \xrightarrow{\tau} \frac{1}{2}\cdot\Delta_1 + \frac{1}{2}\cdot\Delta_2$.

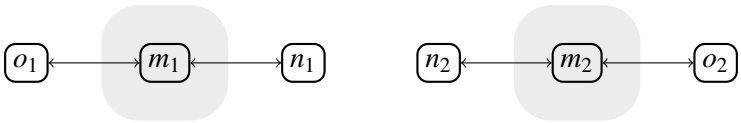


Figure 18: The network $\mathcal{L}' = \Gamma'_L \triangleright L'$

Putting together these derivations, we obtain the hyper-derivation

$$\begin{array}{lcll}
\Delta_1 & \xrightarrow{\tau} & \frac{1}{2}\cdot\Delta_3 + \frac{1}{2}\cdot\Delta_4 & + \quad \varepsilon \\
\frac{1}{2}\cdot\Delta_3 + \frac{1}{2}\cdot\Delta_4 & \xrightarrow{\tau} & \frac{1}{2}\cdot\Delta_1 & + \quad \frac{1}{2}\cdot\Delta_2 \\
\frac{1}{2}\cdot\Delta_1 & \xrightarrow{\tau} & \frac{1}{4}\cdot\Delta_3 + \frac{1}{4}\cdot\Delta_4 & + \quad \varepsilon \\
\frac{1}{4}\cdot\Delta_3 + \frac{1}{4}\cdot\Delta_4 & \xrightarrow{\tau} & \frac{1}{4}\cdot\Delta_1 & + \quad \frac{1}{4}\cdot\Delta_2 \\
\vdots & & \vdots & \vdots \\
\frac{1}{2^n}\cdot\Delta_1 & \xrightarrow{\tau} & \frac{1}{2^{n+1}}\cdot\Delta_3 + \frac{1}{2^{n+1}}\cdot\Delta_4 & + \quad \varepsilon \\
\frac{1}{2^{n+1}}\cdot\Delta_3 + \frac{1}{2^{n+1}}\cdot\Delta_4 & \xrightarrow{\tau} & \frac{1}{2^{n+1}}\cdot\Delta_1 & + \quad \frac{1}{2^{n+1}}\cdot\Delta_2
\end{array}$$

where we recall that ε is the empty sub-distribution. Thus we have $\Delta_1 \stackrel{\tau}{\Longrightarrow} \sum_{i=1}^{\infty} \frac{1}{2^i}\cdot\Delta_2$, which is exactly Δ_2. This concludes the proof that $\mathcal{N}_v^1 \stackrel{c!v\triangleright\{o_1,o_2\}}{\Longrightarrow} \overline{\mathcal{N}}$.

6.3. **Implementation using paramaterised networks.** In this Section we provide another example of network $\mathcal{L}$ which implements the routing model $\mathcal{M}$. In this case rather than a single instance of an implementation, we outline a set of properties of networks, and show that any network satisfying these properties implements the routing model $\mathcal{M}$. The code for the various nodes will be fixed and so the properties all concern the connectivity allowed between them.

Formally, we split $\mathcal{L}$ in two sub-networks, $\mathcal{L}'$ and C, such that $\mathcal{L} = \mathcal{L}' \between C$. Network $\mathcal{L}'$ is completely defined, and its representation is given in Figure 18. Here the process P_m is the same used for nodes m_1, m_2 in network $\mathcal{N}$, defined in Section 6.2. In contrast, network C is specified only in terms of a list properties which we assume it satisfies. These are as follows.

(1) $n_1, n_2 \in \text{nodes}(\Gamma_C \triangleright C)$. For the sake of simplicity, we also assume that $\text{nodes}(\Gamma_C \triangleright C) = (\Gamma_C)_V = \{n_1, \cdots, n_k\}$ for some $k > 2$.
(2) The connectivity graph Γ_C contains a single connected component.
(3) Every node n_i, $i = 1, \cdots, k$ is associated with a channel c_i and a probability distribution $\Lambda_i : \{1, \cdots, k\} \to [0, 1]$. The latter are defined so that $\lceil \Lambda_i \rceil = \{j \mid \Gamma_C \vdash n_i \leftrightarrow n_j\}$, for any $i = 1, \cdots, k$.

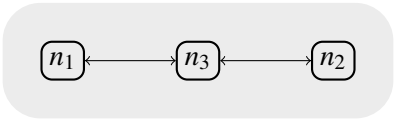
(4) $C = \prod_{i\in I} n_i[\![P_i]\!]$, where

$$\begin{aligned} P_i &= c?(x).\left[\bigoplus_{j=1}^{k} \Lambda_i(j)\cdot c_j!\langle x\rangle.P_i\right] + c?(x).P_i \\ &+ c_i?(x).\left[\bigoplus_{j=1}^{k} \Lambda_i(j)\cdot c_j!\langle x\rangle.P_i\right] + c_i?(x).c!\langle x\rangle.P_n, \quad i = 1,2 \\ P_i &= c_i?(x).(\bigoplus_{j=1}^{k} \Lambda_i(j)\cdot c_j!\langle x\rangle.P_i), \qquad i > 2 \end{aligned}$$

Here the construct $\bigoplus_{i\in I} p_i\cdot P_i$ is interpreted as the probability distribution $[\![\bigoplus_{i\in I} p_i\cdot P_i]\!] = \sum_{i\in I} p_i\cdot [\![P_i]\!]$, and is defined only whenever $\sum_{i\in I} p_i = 1$.

Let us comment on these requirements. Requirement (2) is needed to ensure that, in $\mathcal{L}' \Vdash C$, inputs received by node m_1 from node o_1 can be routed to the external node o_2, and vice-versa. For requirement (3), note that only node n_i can listen to a message broadcast along channel c_i. As we already explained in Section 6.2, this allows a node n_i to select one of its neighbour n_j, $j = 1,\cdots,k$, as the next hop in a routing path by simply forwarding a message along channel c_i. This choice, by node n_i uses the probability distribution Δ_i. Intuitively, the value $\Lambda_i(j)$ corresponds to the probability for node n_i to select n_j as the next hop in a routing path.

Finally, requirement (4) simply defines the structure of the system term C. Note that, with these requirements, the network C is determined completely by the connectivity graph Γ_C and by the set of probability distributions $\{\Lambda_i : 1 \le i \le k\}$.



$\Gamma'_C \rhd C'$

Figure 19: A network C'

Remark 6.1. We require that $\lceil\Lambda_i\rceil = \{j : \Gamma_C \vdash n_i \leftrightarrow n_j\}$, that is every neighbour of node n_i has some non-zero probability of being selected as the next hop. This is needed to ensure that inputs received from node o_1 in $\mathcal{L}' \Vdash C$ can be routed until they eventually reach node the external node o_2, and vice-versa.

In fact, suppose we drop the requirement above from those defined for network C; consider the network C' of Figure 19. Here we assume that the system term C' is defined according to Requirement (4) above and by letting $\Lambda_1 = \overline{3}$, $\Lambda_3 = \overline{1}$ and $\Lambda_2(c) = \overline{3}$. It is easy to note that network C' satisfies the constraints listed above. However, notice that in network $\mathcal{L}' \Vdash C'$, when an input is fired from node o_1, it cannot flow to the external node o_2. This is because the message will never reach node n_2, as node n_3 always selects n_1 as the next hop in a routing path.

Thus $\mathcal{L}' \Vdash C'$ can not be an implementation of the routing model $\mathcal{M}$. □

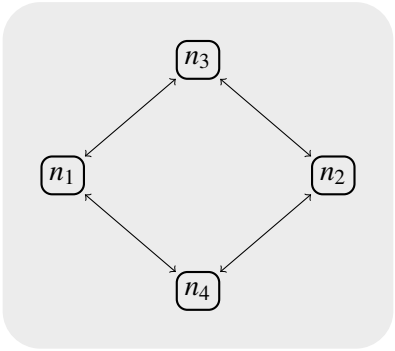


Figure 20: A network $\mathcal{N}'$

Remark 6.2. The network $\mathcal{N}$, defined in Section 6.2, can be obtained as the network $\mathcal{L}' \Vdash \mathcal{N}'$, where $\mathcal{N}'$ is defined in Figure 20, by letting

$$\begin{aligned} \Lambda_i &= \frac{1}{2} \cdot \overline{3} + \frac{1}{2} \cdot \overline{4}, \quad i = 1, 2 \\ \Lambda_i &= \frac{1}{2} \cdot \overline{1} + \frac{1}{2} \cdot \overline{2}, \quad i = 3, 4. \end{aligned}$$

It is easy to note that $\mathcal{N}'$ satisfies the constraints required by the network C, so that it is actually an instantiation of the paramaterised network $\mathcal{L}$ we are considering. Indeed many similar instances can be generated by changing the probabilities used in the code in (6.1) of Section 6.2 to arbitrary non-zero values. □

Now we show that the specification for routing $\mathcal{M}$ and any network $\mathcal{L}$ satisfying the above constraints are simulation related. For the sake of clarity let $L_{m_1} = m_2[\![P_m]\!] | \prod_{i=1}^{k} n_i[\![P_i]\!]$. That is, L_{m_1} is the system term obtained by deleting node m_1 from L. Similar definitions apply for every node in nodes($\mathcal{L}$).

Let $\mathcal{L}_v^1 = \Gamma_L \rhd m_2[\![c!\langle v\rangle.P_m]\!] | L_{m_1}$, $\mathcal{L}_v^2 = \Gamma_L \rhd m_2[\![c!\langle v\rangle.P_m]\!] | L_{m_2}$. We show that the relation

$$\mathcal{S} = \{(\mathcal{M}, \overline{\mathcal{L}})\} \cup \{(\mathcal{M}'_v, \overline{\mathcal{L}_v^1}) \mid v \in \mathrm{Val}\} \cup \{(\mathcal{M}'_v, \overline{\mathcal{L}_v^2}) \mid v \in \mathrm{Val}\}$$

is a simple simulation.

Let us first look at the pair $(\mathcal{M}, \overline{\mathcal{L}})$; recall that network $\mathcal{M}$ has only four possible actions, for a given message v.

- $\mathcal{M} \xrightarrow{d.o_1?v} \overline{\mathcal{M}}$, where $d \neq c$. We need to match this action with a derivation of the form $\overline{\mathcal{L}} \stackrel{d.o_1?v}{\Longrightarrow} \Theta$ for some Θ such that $(\mathcal{M}, \Theta) \in \overline{\mathcal{S}}$. It is not difficult to note that $\overline{\mathcal{L}} \xrightarrow{d.o_1?v} \overline{\mathcal{L}}$, as none of the nodes m_1 and m_2 (which are the only one which can detect messages broadcast from external nodes) is waiting to receive a value on channel d. Thus we have $(\overline{\mathcal{M}}, \overline{\mathcal{L}}) \in \overline{\mathcal{S}}$
- $\mathcal{M} \xrightarrow{d.o_2?v} \overline{\mathcal{M}}$, where $d \neq c$. This case is analogous to the one above
- $\mathcal{M} \xrightarrow{c.o_1?v} \overline{\mathcal{M}'_v}$. In this case it is easy to show that $\overline{\mathcal{L}} \xrightarrow{c.o_1?v} \overline{\mathcal{L}_v^1}$, and $(\overline{\mathcal{M}'_v}, \overline{\mathcal{L}_v^1}) \in \overline{\mathcal{S}}$
- $\mathcal{M} \xrightarrow{c.o_1?v} \overline{\mathcal{M}'_v}$. Again this is straightforward.

It remains to check the pairs of the form $\{(\mathcal{M}'_v, \mathcal{L}_v^1)\}$ and $\{(\mathcal{M}'_v, \mathcal{L}_v^2)\}$. We only supply the details for the former case, as the latter is analogous.

- $\mathcal{M}'_v \xrightarrow{d.o_1?v} \overline{\mathcal{M}'_v}$, where d is an arbitrary channel, including c. Note that, in $\mathcal{L}^1_v$, node m_1 is not waiting to receive a message along any channel. That is, we have $\mathcal{L}^1_v \xrightarrow{d.o_1?v} \overline{\mathcal{L}^1_v}$, and $(\overline{\mathcal{M}^1_v}, \overline{\mathcal{L}^1_v}) \in \overline{S}$
- $\mathcal{M}'_v \xrightarrow{d.o_2?v} \overline{\mathcal{M}'_v}$, where d is an arbitrary channel, including c. If $d \neq c$, then this case is similar to the above one. However, if $d = c$, note that node m_2 is waiting to receive a message along channel c. We already remarked that node m_2 can non-deterministically choose to ignore messages broadcast along channel c, so that it is easy to derive $\mathcal{L}^1_v \xrightarrow{c.o_2?v} \overline{\mathcal{L}^1_v}$. Now it suffices to note that $(\overline{\mathcal{M}^1_v}, \overline{\mathcal{L}^1_v}) \in \mathcal{S}$
- $\mathcal{M}'_v \xrightarrow{c!v\triangleright\{o_1,o_2\}} \overline{\mathcal{M}}$. This is the most interesting case. Here we exhibit a weak derivation of the form $\mathcal{L}^1_v \overset{c!v\triangleright\{o_1,o_2\}}{\Longrightarrow} \overline{\mathcal{L}}$. This is obtained by exploiting the non-standard definition of weak extensional outputs, given in Definition 5.3(3). Specifically, we show that that there exists a distribution Δ_1 such that

$$\mathcal{L}^1_v \xrightarrow{c!v\triangleright\{o_1\}} \Delta_1 \overset{\tau}{\Longrightarrow} \overline{\mathcal{L}^2_v} \xrightarrow{c!v\triangleright\{o_2\}} \overline{\mathcal{L}}. \tag{6.3}$$

here we have that

$$\Delta_1 = \sum_{j=1}^{k} \Lambda_i(j) \cdot \overline{\Gamma_L \triangleright n_i[\![c_j!\langle v\rangle.P_i]\!] \mid L_{n_i}}$$

The result will follow because $(\overline{\mathcal{M}}, \overline{\mathcal{L}}) \in \overline{\mathcal{S}}$.

The only difficult derivation to prove in Equation 6.3 is the hyper-derivation $\Delta_1 \overset{\tau}{\Longrightarrow} \overline{\mathcal{L}^2_v}$. All the other cases, in fact, are analogous to those analysed in Section 6.2.

Here the main idea is that to reduce the state-based network $\mathcal{L}$ and its derivatives, in which the code at nodes n_i, m_i ($i = 1, 2$) is non-deterministic, to a deterministic one in which the computation stops when the value v being routed is delivered at node m_2. For deterministic systems, in fact, we can rely on the useful result stated below.

Lemma 6.3. Let $\langle S, \mathrm{Act}, \tau, \omega\rangle$ be a deterministic pLTS, that is $\Delta' = \Delta''$ whenever $\Delta \xrightarrow{\tau} \Delta'$ and $\Delta \xrightarrow{\tau} \Delta''$. Then, whenever $\Delta \xrightarrow{\tau} \Delta'$ and $\Delta \Longrightarrow \Theta$ it follows that $\Delta' \Longrightarrow \Theta$. □

The resolution of our network to a deterministic one is very easy to define. Let $P'_2 = c_2?(x).c!\langle v\rangle.\mathbf{0}$, $P'_m = c?(x).\mathbf{0}$; the network $(\mathcal{L}_v{}^2)'$ is defined by replacing, in any of the states of the pLTS generated by $\mathcal{L}$, the code at nodes n_i, $i = 1, 2$ with P'_2 and the code at nodes m_i, $i = 1, 2$ with P'_m. That is, we establish that once that a message is received at node n_2, it will broadcast to node m_2, after which the computation of the network stops. This transformation can be applied to the network distribution Δ_1 defined above, leading to another network distribution Δ'_1.

It is trivial to note that if we prove that $\Delta'_1 \Longrightarrow \overline{(\mathcal{L}^2_v)'}$, then we also have that $\Delta_1 \Longrightarrow \overline{\mathcal{L}^2_v}$. This is because $\Delta'_1, (\mathcal{L}^2_v)'$ have been defined by removing the non-deterministic choices from $\Delta_1, \mathcal{L}^2_v$ respectively, and by imposing that the computation stops once a message is received at the node m_2.

In practice, we show that $\Delta'_1 \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$, which by definition gives the required $\Delta'_1 \Longrightarrow \overline{(\mathcal{L}^2_v)'}$. In the following, given a node $m \in \mathrm{nodes}(\mathcal{L})$ we use L'_m for the system term obtained from L_m by resolving the non-deterministic choices as described above, and by requiring that nodes n_2 and m_2

do not perform any activity after n_2 has broadcast a value along channel c. Further, we let

$$\begin{aligned}\Delta'_i &= \sum_{j=1}^{k} \Lambda_i(j) \cdot \overline{\Gamma_L \rhd n_i[\![c_j!\langle v\rangle.P_i]\!] \mid L'_{n_i}}, \quad \text{if } i > 2\\ \Delta'_2 &= \overline{\Gamma_L \rhd n_2[\![c!\langle v\rangle.\mathbf{0}]\!] \mid L'_{n_2}}\end{aligned}$$

be probability distributions. We prove that, for any $i = 1, \cdots, k$, we have the exterme derivative $\Delta'_i \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$; to this end, we show that

(i) $\Delta'_2 \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$ and

(ii) For any two indexes i, j such that $\Gamma_L \vdash n_i \leftrightarrow n_j$, if $\Delta'_i \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$ then $\Delta'_j \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$.

Then it remains to note that the connectivity graph Γ_L has a single connected component to infer that $\Delta'_i \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$ for any i ranging over $1, \cdots, k$.

The proof of the first point is straigthforward. We have that $\Delta'_2 = \overline{\Gamma_L \rhd n_2[\![c!\langle v\rangle.\mathbf{0}]\!] \mid L'_{n_2}} \xrightarrow{\tau} \overline{(\mathcal{L}^2_v)'}$', and since no node can perform a transition in $(\mathcal{L}^2_v)'$ (recall that we changed the code at nodes m_2 so that, once it has received the value broadcast from node n_2 the network deadlocks), this transition can be easily transformed in the extreme derivation $\Delta'_2 \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$.

For the second statement, consider now a distribution Δ'_i, where $i = 1, \cdots, k$. This distribution in deterministic, and therefore it has a unique τ-transition. It is not difficult to show that, for $i \neq 2$, then $\Delta'_i \xrightarrow{\tau} \sum_{i=1}^{k} \Lambda_i(j) \cdot \Delta'_j$, where $\Lambda_i(j) > 0$ if and only if $\Gamma_L \vdash n_i \leftrightarrow n_j$. This is because, any state $(\mathcal{L}^j_i)' = \Gamma_L \rhd n_i[\![c_j!\langle v\rangle.P_i]\!] \mid L'_{n_i}$ has the unique transition $(\mathcal{L}^j_i)' \xrightarrow{\tau} \Delta'_j$.

Now consider any $j \neq 2$, and suppose that $\Gamma_L \vdash n_i \leftrightarrow n_j$ for some index i (possibly equal to 2). Also, suppose that $\Delta'_i \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$. We have already proved that $\Delta'_i \xrightarrow{\tau} p \cdot \Delta'_j + (1-p) \cdot \Theta$ for some distribution Θ and $p \in [0, 1]$ such that $p > 0$. By Lemma 6.3 it follows that $(p \cdot \Delta'_j + (1-p) \cdot \Theta) \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$; since $p > 0$ this leads to the required $\Delta'_j \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$.

As we already observed the graph Γ_L has a single connected component. Thus (i) and (ii) allows us to infer that $\Delta'_i \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$ for any $i = 1, \cdots, k$; in particular, $\Delta'_1 \Longrightarrow\!\!\!\!\!\rightarrow \overline{(\mathcal{L}^2_v)'}$.

7. Conclusions

In this paper we developed a calculus for wireless systems, which enjoys both probabilistic behaviour and local broadcast communication. We developed a theory based on the probabilistic may-testing, and provided a proof method for finitary networks to prove that they can be related via our behavioural preorder.

We believe that this is the first work that consider testing theories for wireless systems. Also, it is the first one in which the problem of composing wireless networks is addressed in detail.

However, in the past the development of formal tools for wireless networks has focused either on other forms of behavioural theories (such as variants of weak bisimulation) and the analysis of protocols. Here we give a brief review of the main works which have inspired our calculus.

To the best of our knowledge, the first process calculus that takes into account broadcast transmission is CBS from Prasad [14]. Here a communication between a sender and a receiver is modeled so that an observer will detect it as a message being sent; this principle enables multiple receivers to

detect a message sent by a single process. CBS has deeply influenced the style of other calculi (included ours) focusing on broadcasting systems, and behavioural theories for it have been analysed [4].

In [10] local broadcast communication and the concept of locations (or node names) have been introduced; in the same style of our calculus, a node name is associated with a process, representing the code it runs. However, here the topology of a network is modeled to be probabilistic, in the sense that whenever a broadcast is issued by a node, other nodes have the capability of detecting such a message with a given probability. Tools for developing the static analysis of routing protocols are then developed for this calculus, which the author refers to as Extended CBS.

The same authors have proposed a second variant of CBS, referred to as CBS# [11], in which the communication topology is defined by a connectivity graph, and the transition relation is parametric in a connectivity graph G; this is very similar to the style we have developed the intensional semantics in our language. For this language, a notion of T-bisimilarity is developed; intuitively, T-bisimilarity is defined by considering standard bisimilarity for two networks defined over a specific network connectivity graph, and then quantifying over all network connectivities.

In [8] a different attempt to formalize wireless networks is made; the authors develop a calculus CWS, where the concepts of node names and location are differentiated; thus, a process is associated both with a node name and a location. Also, every process has a positive real value associated to it, denoting the radius of transmission. A metric distance between locations is assumed; this function, together with the mapping from processes and radius of transmissions, define the network connectivity. The authors propose both a reduction semantics and a labelled transmission semantics, proving the corresponding Harmony Theorem to relate them. It is worth mentioning that in this calculus the communication between nodes consists of two phases, one to start it and one to end it. The authors also model the possibility of a message whose transmission has started to be corrupted by another transmission, thus modeling collisions.

In [7], a timed calculus for wireless systems (TCWS) is presented; in this case, the authors address the problem of representing collisions in wireless networks, suggesting that formal tools for dealing with interferences in wireless networks can aid in the development of MAC level protocols. Here time is assumed to be discrete, in particular an action σ is defined to represent the passage of time. The topology of the wireless networks here is described by associating every node a semantic tag representing its set of neighbours. The authors propose a compositional theory for wireless networks based on the notion of reduction barbed congruence; further, they develop a sound proof methodology based on bisimulations over an extensional lts. It is worth enough to notice that the set of extensional actions they propose (and the activities that can be detected by the external environment) is exactly the same we use in this paper.

In the future our research will concentrate on the development of a characterisation of the must testing preorder for our broadcast language; at the current state of the art, we believe that much of the theory defined here can be used to provide a characterisation of the must-testing preorder in terms of a non-standard version of the failure simulation preorder [2], defined along the same lines of the simulation preorder contained in this paper.

Further, we are currently developing simpler calculi which will enjoy the same features of our simple broadcast language, namely probabilistic behaviour and local broadcast. Testing theory will be defined and analysed for those calculi, in order to provide an evidence that the need of a non-standard theory of simulations depends on these features, rather than on our calculus itself. We are in fact confident that the same theory we developed can be applied to a wide range of calculi for wireless systems.

Appendix A. Decomposition and composition results

To prove propositions 5.17 and 5.18, we first need to prove the following statements for actions which can be derived in the intensional semantics:

Proposition A.1 (Weakening)**.** *Let $\Gamma_1 \rhd M$ be a network, and let Γ_2 such that whenever $\Gamma_2 \vdash m \leftrightarrow n$ with $m \in nodes(M)$ and $n \notin nodes(M)$, then $\Gamma_1 \vdash m \leftrightarrow n$. Then*

$$\Gamma_1 \rhd M \xrightarrow{\alpha} \Delta \text{ implies } (\Gamma_1 \cup \Gamma_2) \rhd M \xrightarrow{\alpha} \Delta$$

where α ranges over the actions $m.\tau, c.m!v, c.m?v$.

Proof. The two implications are proved separately; the only if case is proved by structural induction on the proof of the derivation $\Gamma_1 \rhd M \xrightarrow{\alpha} \Delta$, while the if implication is proved by structural induction on the proof of the derivation $(\Gamma_1 \cup \Gamma_2) \xrightarrow{\alpha} \Delta$; in both cases, the conditions required for the structure of Γ_2 are vital. □

Proposition A.2 (Strengthening)**.** *Let $\Gamma_1 \rhd M$ be a network, and let Γ_2 such that whenever $\Gamma_2 \vdash m \leftrightarrow n$ with $m \in nodes(M)$ and $n \notin nodes(M)$, then $\Gamma_1 \vdash m \leftrightarrow n$. Then*

$$(\Gamma_1 \cup \Gamma_2) \rhd M \xrightarrow{\alpha} \Delta \text{ implies } \Gamma_1 \rhd M \xrightarrow{\alpha} \Delta$$

where α ranges over the actions $m.\tau, c.m!v, c.m?v$.

Proposition A.3 (Node identification)**.** *Let $\Gamma_M \rhd M$ be a network such that*

(1) $\Gamma_M \rhd M \xrightarrow{m.\tau} \Delta$*; then*
- $M = m[\![s]\!] \mid M'$,
- $s \xrightarrow{\tau} P$, *for some* P,
- $\Delta = m[\![\Delta']\!] \mid \overline{M'}$, *with* $\Delta' = [\![P]\!]$.

(2) $\Gamma_M \rhd M \xrightarrow{c.m!v} \Delta$*, then*
- $M = m[\![s]\!] \mid M'$,
- $s \xrightarrow{c!v} P$ *for some* P,
- $\Gamma_M \rhd M' \xrightarrow{c.m?v} \Theta$ *for some* Θ,
- $\Delta = m[\![\Delta']\!] \mid \Theta$, *with* $\Delta' = [\![P]\!]$.

All the equivalences above are defined modulo structural equivalence.

Proof. Both cases are proved by structural induction on the proof of the derivation $\Gamma_M \rhd M \xrightarrow{\alpha} \Delta$, using Proposition A.1 and with α ranging over $m.\tau, c.m!v$. □

Proof of Proposition 5.17. We only prove the first statements; details for the other statements are similar.

Suppose $(\Gamma_M \rhd M) \mid\mid\mid_{\mathrm{m}} (\Gamma_n \rhd n[\![s]\!]) \xrightarrow{\tau} \Delta$; First we rewrite the network in the left hand side of the transition as $(\Gamma_M \cup \Gamma_n) \rhd M \mid n[\![s]\!]$. By definition of extensional actions, there are two possible cases:

(1) $(\Gamma_M \cup \Gamma_n) \rhd M \mid n[\![s]\!] \xrightarrow{m.\tau} \Delta$; in this case we can apply Proposition A.3 (1) to derive $M \mid n[\![s]\!] = m[\![s']\!] \mid M'$. Here again we have two possible cases:
- $m = n$; then, by Proposition A.3 (1) $M = M'$, $s = s'$, $s \xrightarrow{n.\tau} P_n$, with $[\![P_n]\!] = \Delta_n$ and $\Delta = n[\![\Delta_n]\!] \mid \overline{M}$. Now we can derive $\Gamma_n \rhd n[\![s]\!] \xrightarrow{n.\tau} n[\![\Delta_n]\!]$, and therefore $\Gamma_n \rhd n[\![s]\!] \xrightarrow{\tau} \Gamma_n \rhd n[\![\Delta_n]\!]$.

- $m \neq n$; in this case $m \in \text{nodes}(M)$. For $M = m[\![s']\!] \mid M'$, by Proposition A.3 (1) it holds $s' \xrightarrow{\tau} P'$ and $\Delta = m[\![\Delta_m]\!] \mid \overline{M'}$, where $\Delta_m = [\![P]\!]$. Let now $\Delta_M = m[\![\Delta_m]\!] \mid \overline{M'}$. It is straightforward to show that $\Gamma_M \cup \Gamma_n \rhd M \xrightarrow{m.\tau} \Delta_M$. Note also that, whenever $\Gamma_n \vdash l \leftrightarrow k$ for some $l \in \text{nodes}(M)$, then $k = n$, and $n \notin \text{nodes}(M)$, Since $\Gamma_M \rhd M \;|||_{\text{m}}\; \Gamma_n \rhd n[\![s]\!]$ is defined, it follows that $\Gamma_M \vdash l \leftrightarrow k$. Therefore, we can apply Strengthening, Proposition A.2, to the derivation $(\Gamma_M \cup \Gamma_n) \rhd M \xrightarrow{m.\tau} \Delta_M$ to infer $\Gamma_M \rhd M \xrightarrow{m.\tau} \Delta_M$, from which we obtain the extensional action $\Gamma_M \rhd M \xrightarrow{\tau} \Delta_M$.

(2) $\Gamma_M \cup \Gamma_n \rhd M \mid n[\![s]\!] \xrightarrow{c.m!v} \Delta$, with $\{l \mid (\Gamma_M \cup \Gamma_n) \vdash m \leftrightarrow l\} \subseteq \text{nodes}(M \mid n[\![s]\!])$. Denote the set in the left hand side of the inclusion above as η_m. Here, by Proposition A.3 (2) there are two different possible cases

- $m = n$. First, note that, as $\eta_n \subseteq \text{nodes}(M)$, whenever $\Gamma_n \vdash m \leftrightarrow n$ then $m \in \text{nodes}(M)$. Now we apply Proposition A.3 (2) to obtain $s \xrightarrow{c.n!v} P_n$ and $(\Gamma_M \cup \Gamma_n) \rhd M \xrightarrow{c.n?v} \Delta_M$, with $\Delta = \Delta_M \mid n[\![\Delta_n]\!]$ for $\Delta_n = [\![P_n]\!]$. By Strengthening, Proposition A.2, it follows that $\Gamma_n \rhd n[\![s]\!] \xrightarrow{c.n!v} \Gamma_n \rhd \Delta_n[\![s]\!]$; by definition of extentional actions, we obtain $\Gamma_n \rhd n[\![s]\!] \xrightarrow{c!v \rhd \eta_n} \Gamma_n \rhd \Delta_n[\![s]\!]$. However, we have already noticed that every node in the set η_n is a node in nodes(M), that is $\eta_n \subseteq \text{nodes}(M)$. It remains to prove $\Gamma_M \rhd M \xrightarrow{c.n?v} \Gamma_M \rhd \Delta_M$. We have already shown that $(\Gamma_1 \cup \Gamma_2) \rhd M \xrightarrow{c.n?v} \Delta_M$, and by Proposition A.2 we obtain that $\Gamma_M \rhd M \xrightarrow{c.n?v} \Delta_M$. By definition of weak extensional action, it remains to show that $n \in \mathsf{Int}(\Gamma_M \rhd M)$. However, this is ensured, since $\mathsf{Int}(\Gamma_n \rhd n[\![s]\!]) = \eta \subseteq \text{nodes}(M)$ $n \in \mathsf{Int}(\Gamma_M \rhd M)$. Since η is non-empty, there exists at least a node $m \in \text{nodes}(M)$ such that $\Gamma_n \vdash m \leftrightarrow n$, and by the definition of $|||_{\text{m}}$ we obtain $\Gamma_M \vdash mm \leftrightarrow n$.
- $n \neq m$; this case is similar to the one above, noting that in this case we have $\eta = \{n\}$.

□

Proof of Proposition 5.18. We prove only (i) and (ii); the other cases ore similar. For (i), suppose that $(\Gamma_M \rhd \Delta) \stackrel{\tau}{\Longrightarrow} (\Gamma_M \rhd \Delta_M), \Gamma_n \rhd n[\![\Theta]\!] \stackrel{\tau}{\Longrightarrow} \Gamma_n \rhd n[\![\Theta_n]\!]$, and $(\Gamma_M \rhd \Delta) \;|||_{\text{m}}\; (\Gamma_n \rhd n[\![\Theta]\!])$ is well defined

Note that Proposition A.1, which has been proved for state-based networks, also holds for node-stable network distributions. The proof of this statement is trivial, and it can be performed by looking and the individual transitions of the state-based networks in the support of a distribution. Thus, whenever for a distribution Δ_0 such that $\text{nodes}(\Delta_0) \subseteq \text{nodes}(\Delta)$ we can derive a (strong) action $\Gamma_M \rhd \Delta_0 \xrightarrow{\tau} \Gamma_M \rhd \Delta_1$, we can apply Propositions A.1 and A.3 to obtain $(\Gamma_M \cup \Gamma_n)\Delta_0 \mid s[\![\Theta]\!] \xrightarrow{\tau} (\Gamma_M \cup \Gamma_n) \rhd \Delta_1 \mid n[\![\Theta]\!]$. It is now easy to show that we can infer the hyper-derivation $(\Gamma_M \cup \Gamma_n) \rhd \Delta \stackrel{\tau}{\Longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd \Delta_M \mid n[\![\Theta]\!]$. A similar argument can be used to show that $(\Gamma_M \cup \Gamma_n) \rhd \Delta_M \mid n[\![\Theta]\!] \stackrel{\tau}{\Longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd \Delta_M \mid n[\![\Theta_n]\!]$; the result follows now from the transitivity of $\Longrightarrow$, Theorem 2.4 (1).

For (ii), suppose that $(\Gamma_M \rhd \Delta) \stackrel{c!v \rhd \eta}{\Longrightarrow} (\Gamma_M \rhd \Delta_M)$, with $n \notin \eta$. We prove that $(\Gamma_M \cup \Gamma_N) \rhd \Delta | n[\![\Theta]\!] \stackrel{c!v \rhd \eta}{\Longrightarrow}$ $(\Gamma_M \cup \Gamma_N) \rhd \Delta_M \mid n[\![\Theta]\!]$ by performing an induction on the derivation $(\Gamma_M \rhd \Delta) \stackrel{c!v \rhd \eta}{\Longrightarrow} (\Gamma_M \rhd \Delta_M)$. FOr the moment, suppose that $\Theta = \overline{s}$.

The base case is given by $(\Gamma_M \rhd \Delta) \stackrel{\tau}{\Longrightarrow} (\Gamma_M \rhd \Delta_1) \stackrel{c!v\rhd\eta}{\longrightarrow} (\Gamma_M \rhd \Delta_2) \stackrel{\tau}{\Longrightarrow} (\Gamma_M \rhd \Delta_M)$. By (i) we have that $(\Gamma_M \cup \Gamma_n) \rhd \Delta \,|\, n[\![\Theta]\!] \stackrel{\tau}{\Longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd \Delta_1 \,|\, n[\![\Theta]\!]$, and $(\Gamma_M \cup \Gamma_n) \rhd \Delta_2 \,|\, n[\![\Theta]\!] \stackrel{\tau}{\Longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd \Delta_M \,|\, n[\![\Theta]\!]$, so that it remains to show that $(\Gamma_M \cup \Gamma_n) \rhd \Delta_1 \mid n[\![\Theta]\!] \stackrel{c!v\rhd\eta}{\longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd \Delta_2 \mid n[\![\Theta]\!]$.

In this case we can rewrite Δ_1 as

$$\Delta_1 = \sum_{i \in I} p_i \cdot M_i$$

Since $\Gamma_M \rhd \Delta_1 \stackrel{c!v\rhd\eta}{\longrightarrow} \Gamma_M \rhd \Delta_2$, it follows that, for all $i \in I$, $\Gamma_M \rhd M_i \stackrel{c!v\rhd\eta}{\longrightarrow} \Gamma_M \rhd \Delta_i'$ and $\Delta_2 = \sum_{i \in I} p_i \cdot \Delta_i'$. It is sufficient to show that, for every $i \in I$, $\Gamma_M \rhd M_i \;|||_{\mathrm{m}}\; \Gamma_n \rhd n[\![\Theta]\!] \stackrel{c!v\rhd\eta}{\longrightarrow} \Gamma_M \rhd \Delta_i' \;|||_{\mathrm{m}}\; \Gamma_n \rhd n[\![\Theta]\!]$, thus proving

$$(\Gamma_M \cup \Gamma_n) \rhd (\Delta_1 \mid n[\![\Theta]\!]) \stackrel{c!v\rhd\eta}{\longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd (\Delta_2 \mid n[\![\Theta]\!])$$

The proof of this statement is straigthforward. Let $i \in I$; By (i) and by propositions A.1 and A.3[2], it is easy to derive the transition above. Hence we have shown that

$$\begin{aligned}(\Gamma_M \cup \Gamma_n) \rhd (\Delta \mid n[\![\Theta]\!]) &\stackrel{\tau}{\Longrightarrow} (\Gamma_1 \cup \Gamma_n) \rhd (\Delta_1 \mid n[\![\Theta]\!]) \\ &\stackrel{c!v\rhd\eta}{\longrightarrow} (\Gamma_2 \cup \Gamma_n) \rhd (\Delta_2 \mid n[\![\Theta]\!]) \\ &\stackrel{\tau}{\Longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd (\Delta_M \mid n[\![\Theta]\!])\end{aligned}$$

from which it follows that $(\Gamma_M \cup \Gamma_n) \rhd (\Delta \mid n[\![\Theta]\!]) \stackrel{c!v\rhd\eta}{\Longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd (\Delta_M \mid n[\![\Theta]\!])$.

For the inductive case, we have that $\Gamma_M \rhd \Delta \stackrel{c!v\rhd\eta_1}{\Longrightarrow} \Gamma_M \rhd \Delta' \stackrel{c!v\rhd\eta_2}{\Longrightarrow} \Gamma_M \rhd \Delta_M$, with $\eta = \eta_1 \cup \eta_2$, and $\eta_1 \cap \eta_2 = \emptyset$; in this case it is sufficient to note that $n \notin \eta_1, n \notin \eta_2$ to apply the inductive hypothesis to the individual transitions $\Gamma_M \rhd \Delta' \stackrel{c!v\rhd\eta_2}{\Longrightarrow} \Gamma_M \rhd \Delta_M$ and $\Gamma_M \rhd \Delta_1 \stackrel{c!v\rhd\eta_1}{\Longrightarrow} \Gamma_M \rhd \Delta'$, to obtain that

$$\begin{aligned}(\Gamma_M \cup \Gamma_n) \rhd (\Delta \mid n[\![\Theta]\!]) &\stackrel{c!v\rhd\eta_1}{\Longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd (\Delta' \mid n[\![\Theta]\!]) \\ (\Gamma_M \cup \Gamma_n) \rhd (\Delta' \mid n[\![\Theta]\!]) &\stackrel{c!v\rhd\eta_1}{\Longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd (\Delta_M \mid n[\![\Theta]\!])\end{aligned}$$

The non-standard definition of extensional output actions, Definition 5.3, and the condition $\eta_1 \cap \eta_2 = \emptyset$ ensure that

$$(\Gamma_M \cup \Gamma_n) \rhd (\Delta \mid n[\![\Theta]\!]) \stackrel{c!v\rhd\eta}{\Longrightarrow} (\Gamma_M \cup \Gamma_n) \rhd (\Delta_m \mid n[\![\Theta]\!])$$

where we recall that $\eta = \eta_1 \cup \eta_2$.

References

[1] Eoin Curran and Jim Dowling. Sample: Statistical network link modelling in an on-demand probabilistic routing protocol for ad hoc networks. In *WONS*, pages 200–205. IEEE Computer Society, 2005.

[2] Yuxin Deng, Rob van Glabbeek, Matthew Hennessy, and Carroll Morgan. Testing finitary probabilistic processes. In *Proceedings of the 20th International Conference on Concurrency Theory*, volume 5710 of *Lecture Notes in Computer Science*, pages 274–288. Springer, 2009. Full-version available from http://www.scss.tcd.ie/Matthew.Hennessy/onlinepubs.html.

[3] Yuxin Deng, Rob van Glabbeek, Matthew Hennessy, and Carroll Morgan. Testing finitary probabilistic processes (extended abstract). In *Proceedings of the 20th International Conference on Concurrency Theory*, volume 5710 of *Lecture Notes in Computer Science*, pages 274–288. Springer, 2009.

[2]Note that it is first necessary to convert an extensional action in an intensional one, in order to apply these theorems

[4] Hennessy and Rathke. Bisimulations for a calculus of broadcasting systems. *TCS: Theoretical Computer Science*, 200, 1998.
[5] Alan Jeffrey and Julian Rathke. Java jr. : Fully abstract trace semantics for a core java language. In *Proc. European Symposium on Programming ESOP*, volume 3444 of *Lecture Notes in Computer Science*, pages 423–438. Springer-Verlag, 2005.
[6] Raja Jurdak, Cristina Videira Lopes, and Pierre Baldi. A survey, classification and comparative analysis of medium access control protocols for ad hoc networks. *IEEE Communications Surveys and Tutorials*, 6(1-4):2–16, 2004.
[7] Massimo Merro and Eleonora Sibilio. A timed calculus for wireless systems. In Farhad Arbab and Marjan Sirjani, editors, *FSEN*, volume 5961 of *Lecture Notes in Computer Science*, pages 228–243. Springer, 2009.
[8] Nicola Mezzetti and Davide Sangiorgi. Towards a calculus for wireless systems. *Electr. Notes Theor. Comput. Sci*, 158:331–353, 2006.
[9] R. Milner. A calculus of communicating systems. *LNCS*, 92, 1980.
[10] Sebastian Nanz and Chris Hankin. Static analysis of routing protocols for ad-hoc networks, March 25 2004.
[11] Sebastian Nanz and Chris Hankin. A framework for security analysis of mobile wireless networks. *TCS: Theoretical Computer Science*, 367, 2006.
[12] Rocco De Nicola and Matthew Hennessy. Testing equivalences for processes. *Theor. Comput. Sci*, 34:83–133, 1984.
[13] Sotiris Nikoletseas and Paul G. Spirakis. Probabilistic data propagation in wireless sensor networks. In Sotiris Nikoletseas and Jos D.P. Rolim, editors, *Theoretical Aspects of Distributed Computing in Sensor Networks*, Monographs in Theoretical Computer Science. An EATCS Series, pages 353–380. Springer Berlin Heidelberg, 2011.
[14] Prasad. A calculus of broadcasting systems. *SCIPROG: Science of Computer Programming*, 25, 1995.
[15] S. Rahamatkar, A. Agarwal, and N. Kumar. Analysis and comparative study of clock synchronization schemes in wireless sensor networks. *Analysys*, 2(3):536–541, 2010.
[16] D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.
[17] Yoav Sasson, David Cavin, and André Schiper. Probabilistic broadcast for flooding in wireless mobile ad hoc networks. In *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC 2003)*, March 2003.
[18] Roberto Segala and Nancy A. Lynch. Probabilistic simulations for probabilistic processes. *Nord. J. Comput*, 2(2):250–273, 1995.
[19] Andrew S. Tanenbaum. *Computer networks*. P T R Prentice-Hall, pub-PHPTR:adr, fourth edition, 2003.