Under consideration for publication in Math. Struct. in Comp. Science

Concurrency Can't Be Observed, Asynchronously[†]

Paolo Baldan, Filippo Bonchi, Fabio Gadducci, Giacoma Valentina Monreale

Dipartimento di Matematica Pura e Applicata, Università Padova

email address:*baldan@math.unipd.it*

ENS Lyon, Université de Lyon, LIP (UMR 5668 CNRS ENS Lyon UCBL INRIA)

email address:*filippo.bonchi@ens-lyon.fr*

Dipartimento di Informatica, Università di Pisa

email address:*gadducci@di.unipi.it*

Dipartimento di Informatica, Università di Pisa

email address:*vale@di.unipi.it*

Received 6 September 2012

The paper is devoted to an analysis of the concurrent features of asynchronous systems. A preliminary step is represented by the introduction of a non-interleaving extension of barbed equivalence. This notion is then exploited in order to prove that *concurrency cannot be observed* through asynchronous interactions, i.e., that the interleaving and concurrent versions of a suitable asynchronous weak equivalence actually coincide. The theory is validated on some case studies, related to nominal calculi (π-calculus) and visual specification formalisms (Petri nets). Additionally, we prove that a class of systems which are deemed (output-buffered) asynchronous, according to a characterisation that was previously proposed in the literature, falls into our theory.

Introduction

Since the introduction of process calculi, one of the richest sources of foundational investigations stemmed from the analysis of behavioural equivalences. The rationale is that in any formalism, specifications which are syntactically different may intuitively denote the same system, and it is pivotal to equate specifications at the right level of abstraction.

[†] Supported by the MIUR project SisteR and the University of Padova project AVIAMO.

One of the most influential synthesis on the issue is offered by the taxonomy proposed in the so-called *linear time/branching time spectrum* (van Glabbeek 1990). Since then, a major dichotomy among equivalences was established between *interleaving* and *truly concurrent* semantics, according to the possibility of capturing the parallel composition of two systems by means of a non-deterministic selection. Concretely, adopting a CCS-like syntax, the system represented by the specification $a|b$ either coincides with (interleaving) or differs from (truly concurrent) the system represented by $a.b + b.a$.

Behavioural equivalences for process calculi often rely on *labelled transitions*: each evolution step of a system is tagged by some information aimed at capturing the possible interactions of the system with the environment. Nowadays, though, the tendency is to adopt operational semantics based on *unlabelled transitions*. This is due to the intricacies of the intended behaviour of a system, especially in the presence of topological or transactional features (see, e.g., calculi such as Mobile Ambients (Cardelli and Gordon 2000) or Join (Fournet and Gonthier 1996)).

This paradigmatic shift stimulated the adoption of *barbed congruence* (Milner and Sangiorgi 1992), a behavioural equivalence based on a family of predicates over the states of a system, called *barbs*. Even if they are defined ad hoc for each formalism, in general terms barbs are intended to capture the ability of a system of performing an *interaction* with the environment. For instance, in the calculus of Mobile Ambients (Cardelli and Gordon 2000), ambient names can be used as barbs: for a name n the corresponding barb verifies the occurrence of an ambient named n at top level in the process (Merro and Nardelli 2003); this reveals the possibility for the process of engaging an interaction with another process that aims at opening or entering into an ambient n. In CCS (Milner 1989), channel names can be used as barbs: for a name a the corresponding barb checks if the process may input on a (Milner and Sangiorgi 1992).

Assuming that systems interact with a form of synchronous communication, barbs can be explained by a scenario where a system is just a black box with several buttons, one for each possible interaction with the environment. An observer can push a button only if the system is able to perform the corresponding interaction. In this scenario, barbs check if buttons can be pushed. Similarly, an asynchronous system can be seen as a black box equipped with several bags (unordered buffers) that are used to exchange messages with the environment. At any time the observer can insert a message in a bag or remove one, whenever present. In this case, barbs check the presence of messages inside bags.

Additionally, in order to properly capture the scenario outlined above, internal steps should not be visible to an observer. For this reason we will focus on weak equivalences.

To the best of our knowledge, barbed congruences capturing concurrent features of a system have not been considered so far in the literature, i.e., barbs have not been yet used to abstractly characterise the possibility for a system of performing *simultaneously* more than just one single interaction.

It is intuitively clear, however, that in the synchronous scenario, the possibility of checking concurrent interactions would increase the discriminating power of the observer. Let us consider again the systems specified by $a.b + b.a$ and $a|b$: they are distinguished by an observer that is able to push two buttons at the same time, since only $a|b$ allows for the simultaneous pressing of buttons a and b.

The situation is less clearly-cut for asynchronous systems. Indeed, one of the assumptions of this communication style is that message sending is non-blocking: a system may send a message with no agreement with the receiver, and then continue its execution. Hence, an observer interacting with a system by message exchanges cannot know if or when a message has been received and thus message reception is deemed unobservable. And since message sending is non-blocking, a system that may emit a sequence of messages can also hold them, proceed with internal computation and make them available at once at a later time. So, the simultaneous observation of many sendings seems to add no discriminating power to the observer. Concretely, systems $a.b + b.a$ and $a|b$ should be equated in an asynchronous setting, even if observing concurrent barbs.

Moving from this intuition, we propose a framework where the slogan *concurrency can't be observed, asynchronously* is formalised. We work in a setting where an operator for parallel composition of systems is available and we define a notion of concurrent barbed congruence by assuming that *concurrent barbs* can be constructed from basic ones by using a binary operator $\otimes$. Although, in the general case, we just assume that basic barbs form a set of generators for concurrent barbs, the intuition is that a system will exhibit a concurrent barb $a_1 \otimes a_2$ if it includes two parallel subcomponents exhibiting barbs a_1 and a_2, respectively. We then identify a set of axioms which are intended to capture essential features of asynchronous systems in a barbed setting, showing that for any formalism satisfying them barbed congruence and its concurrent variant coincide.

The appropriateness of the axioms is checked by proving that they are satisfied by several concrete formalisms. Specifically, we consider the asynchronous π-calculus (Honda and Tokoro 1991, Boudol 1992) endowed with two distinct concurrent semantics, differing for the fact that one imposes a bound on the capacity of the channels (disallowing for concurrent communications on the same channel), and open Petri nets (Kindler 1997, Milner 2003, Sassone and Sobociński 2005, Baldan, Corradini, Ehrig and Heckel 2005), a reactive variant of Petri nets. In the latter case the barbed concurrent equivalence is shown to coincide with standard step semantics. We finally consider a class of systems abstractly characterised as (output-buffered) asynchronous in (Selinger 1997), showing that also these fit in our theory.

The impossibility of observing concurrency through asynchronous interactions is no longer true for process calculi with priorities, even though asynchronous, and, more generally, for formalisms where some transitions of a system can be inhibited by a system running in parallel. We show that indeed such formalisms escape our framework, by focusing, as a case study, on an asynchronous CCS with priorities.

This is an extended version of the conference paper (Baldan, Bonchi, Gadducci and Monreale 2010). In particular, here we consider a more general notion of concurrent barbs (technically, in (Baldan et al. 2010) concurrent barbs were defined as multisets of barbs, while here we consider a generic abelian semigroup) which allows to simplify our theory and widen its scope. In particular, as mentioned above, the theory now applies also to calculi with bounded-capacity channels and calculi featuring notions of asynchrony based on buffers which are not just unordered bags, but ordered structures like queues (see e.g. (Bergstra, Klop and Tucker 1984, de Boer, Klop and Palamidessi 1992, Beauxis, Palamidessi and Valencia 2008)).

Synopsis. Section 1 introduces our framework (the notion of concurrent barb and the corresponding behavioural equivalence) and states the unobservability of concurrency through asynchronous interactions. Sections 2 and 3 show how our theory captures asynchronous π-calculus and open Petri nets, respectively. Section 4 proves that systems deemed as (output-buffered) asynchronous in (Selinger 1997) fall into our theory. Section 5 shows that our theory does not apply to asynchronous CCS with priorities, a paradigmatic example of formalism with inhibitory effects between transitions. Finally, Section 6 draws some conclusions, discusses related works and outlines directions for further research.

1. A Theory of Concurrent Barbs and Asynchrony

This section introduces a notion of equivalence based on *concurrent* barbs. It is then argued that, for a reasonable notion of asynchronous system, the possibility of observing concurrent barbs does not add any discriminating power.

1.1. *Transition Systems and Barbs*

In order to develop a general theory, applicable to a range of different examples, we work on (suitably enriched) transition systems rather than focusing on some specific calculus.

Definition 1 (transition systems). A transition system is a pair $\langle \mathcal{P}, \rightarrow \rangle$, where $\mathcal{P}$ is a set of *systems* (ranged over by p, q ...) and $\rightarrow \subseteq \mathcal{P} \times \mathcal{P}$ is a binary relation over $\mathcal{P}$, called *transition relation*. We write $p \rightarrow q$ for $\langle p, q \rangle \in \rightarrow$, and we denote by $\rightarrow^*$ the reflexive and transitive closure of $\rightarrow$.

We work in a fixed transition system $\langle \mathcal{P}, \rightarrow \rangle$, and we additionally assume to have a commutative and associative *parallel composition* operator on systems $| : \mathcal{P} \times \mathcal{P} \rightarrow \mathcal{P}$, satisfying the axiom below

$$\textsf{(P)} \quad \frac{p \rightarrow p'}{p|q \rightarrow p'|q}$$

In other terms, the parallel operator must preserve the transition relation: the requirement concerning its associativity and commutativity, making $\langle \mathcal{P}, | \rangle$ an abelian semigroup, would not be essential for our theory, but it simplifies the presentation.

Definition 2 (barbs). A *barb* is a predicate over the set $\mathcal{P}$. The set of barbs, ranged over by a, b, x, y, ..., is denoted $\mathcal{B}$ and we write $p\downarrow_a$ when the system p *satisfies* the barb a. A system p *weakly satisfies* a, written $p\Downarrow_a$, if $p' \downarrow_a$ for some p' such that $p \rightarrow^* p'$. Moreover, p *permanently satisfies* a, written $p\,\Box\!\downarrow_a$, if $p' \downarrow_a$ for all p' such that $p \rightarrow^* p'$. We write $p\Box\!\Downarrow_a$ if $p'\Downarrow_a$ for all p' such that $p \rightarrow^* p'$.

With these ingredients we can define a behavioural equivalence which equates two systems when they cannot be distinguished by an observer that can add components in parallel and observe the barbs which are exposed. In the paper we focus only on weak equivalences, hence the qualification "weak" is omitted.

	SYNCHRONOUS			ASYNCHRONOUS	
p	$::=$	$m,\ p_1 \vert p_2$	p	$::=$	$m,\ p_1 \vert p_2,\ \bar{a}$
m	$::=$	$\tau.p,\ a.p,\ \bar{a}.p,\ m_1 + m_2,\ \mathbf{0}$	m	$::=$	$\tau.p,\ a.p,\ m_1 + m_2,\ \mathbf{0}$

$$(\text{SYN})\ (a.p + m)|(\bar{a}.q + n) \rightarrow p|q \qquad (\text{ASYN})\ (a.p + m)|\bar{a} \rightarrow p$$

$$(\text{TAU})\ \ \tau.p + m \rightarrow p \qquad (\text{PAR})\ \ \frac{p \rightarrow q}{p|r \rightarrow q|r}$$

$$\begin{array}{ccc} p|q \equiv q|p & p|(q|r) \equiv (p|q)|r & p|\mathbf{0} \equiv p \\ m + n \equiv n + m & m + (n + o) \equiv (m + n) + o & m + \mathbf{0} \equiv m \end{array}$$

Fig. 1. The syntax and the reduction semantics of SCCS and ACCS.

Definition 3 (saturated barbed bisimilarity). A symmetric relation $R \subseteq \mathcal{P} \times \mathcal{P}$ is a *saturated barbed bisimulation* if whenever $p\ R\ q$ then

— $\forall a \in \mathcal{B}$, if $p \Downarrow_a$ then $q \Downarrow_a$
— if $p \rightarrow^* p'$ then $q \rightarrow^* q'$ and $p'\ R\ q'$
— $\forall r \in \mathcal{P}$, $p|r\ R\ q|r$

We say that p and q are *saturated barbed bisimilar*, written $p \sim q$, if there exists a saturated barbed bisimulation relating them.

Note that $\sim$ is, by definition, closed with respect to the parallel composition operator[†]. It differs from *barbed congruence* (Milner and Sangiorgi 1992) since in the latter the observer is allowed to add a parallel component only at the beginning of the computation and not at any step. Hence, in general, barbed congruence is coarser than saturated barbed bisimilarity, although in many cases the two definitions coincide (as e.g. in the asynchronous π-calculus (Fournet and Gonthier 2005)).

The following simple observation will be needed later to prove our main result.

Lemma 1. Let $p, q \in \mathcal{P}$ be systems such that $p \sim q$. Then $p\Box\Downarrow_a$ iff $q\Box\Downarrow_a$.

Proof. Assume that $p\Box\Downarrow_a$, i.e., $p'\Downarrow_a$ for all p' such that $p \rightarrow^* p'$. For all q' such that $q \rightarrow^* q'$, by the fact that $p \sim q$ we deduce that there exists p' such that $p \rightarrow^* p'$ and $p' \sim q'$. And since $p'\Downarrow_a$, necessarily also $q'\Downarrow_a$. This means that also $q\Box\Downarrow_a$. $\Box$

As a running example for illustrating our theory we use the finite, restriction-free fragment of CCS (Milner 1989) and its asynchronous counterpart, with the reduction semantics in (Milner 1999), but our considerations would extend to the full calculus (with some care in the treatment of the restriction operator, as discussed in detail for the π-calculus in Section 2). A set of *names* $\mathcal{N}$ is fixed (ranged over by a, b, ...) with $\tau \notin \mathcal{N}$. The syntax of synchronous CCS (SCCS) processes is defined by the grammar on the left of Figure 1, the one for asynchronous CCS (ACCS) processes by the grammar

[†] Requiring $\sim$ to be closed under all unary contexts (see (Honda and Yoshida 1995, Merro and Nardelli 2003)), would not substantially change our theory, yet it would make its presentation more complex.

on the right. In both cases processes are considered up to structural congruence $\equiv$. The transition relation $\rightarrow$ for the synchronous calculus SCCS is defined by rules SYN, TAU, and PAR. In particular, rule SYN allows a process $a.p + m$ that is ready to receive an input on a to synchronise with a process $\bar{a}.q + n$ ready to send an output on the same channel. For the asynchronous calculus ACCS, rule SYN is replaced by ASYN: the occurrence of an unguarded $\bar{a}$ indicates a message that is available on some communication media named a. The message disappears whenever it is received. Note that output prefixes $\bar{a}.p$ are absent in ACCS, the intuition being that message sending is non-blocking and thus the reception of a message cannot enable a continuation.

1.2. *Witnesses for barbs*

The definition of the "right" class $\mathcal{B}$ of barbs is not a trivial task. For SCCS both input and output barbs are considered (see e.g. (Milner and Sangiorgi 1992)). Intuitively, a process has an input (output) barb on a if it is ready to perform an input (output) on a. Formally, if $\alpha \in \{a, \bar{a}\}$, then $p\downarrow_\alpha$ when $p \equiv \alpha.p_1 + m|p_2$ for processes p_1, p_2, m. Following (Amadio, Castellani and Sangiorgi 1996), for ACCS only output barbs are considered, defined by $p\downarrow_{\bar{a}}$ when $p \equiv \bar{a}|p_1$ for a process p_1. The idea is that, since message sending is non-blocking, an external observer can just send messages without knowing if they will be received or not. Hence inputs are deemed unobservable.

Several works (e.g. (Honda and Yoshida 1995, Rathke, Sassone and Sobociński 2007, Bonchi, Gadducci and Monreale 2010)) have proposed abstract criteria for defining "good" barbs independently from the formalism at hand. Here, inspired by (Rathke et al. 2007), we propose to formalise the intuition that barbs should capture the possibility of exhibiting an observable behaviour by introducing a notion of *test*.

Definition 4 (barbs witnessed by a test). A *concrete test* for a barb $a \in \mathcal{B}$ on a system $p \in \mathcal{P}$ is a pair $\langle t, x\rangle$, denoted t_x for short, where $t \in \mathcal{P}$ and $x \in \mathcal{B}$ such that

$$p\downarrow_a \qquad \text{iff} \qquad p|t \rightarrow p' \text{ and } p' \Box\!\downarrow_x.$$

We say that the concrete test t_x for a on p is *stable* if t_x is a concrete test for any p' such that $p \rightarrow^* p'$ and for any p', p'' such that $p = p'|p''$.

A *test* for a barb $a \in \mathcal{B}$ is a family $T = \{t_x : x \in \mathcal{B}\}$ such that for any $p \in \mathcal{P}$ there exists $x \in \mathcal{B}$ such that t_x is a stable concrete test for a on p. In this case we say that the test *witnesses* the barb a with respect to $\rightarrow$.

Intuitively, a concrete test for a barb a on a system p will be chosen as a system t capable of exposing a barb x, which instead would never be observable in the evolution of p. System t releases a (permanent) barb x only after interacting with a system exposing barb a. Since x can never be generated by p, observing x in the evolution of $p|t$ witnesses that p has exposed the barb a. The stability condition ensures that a concrete test on p can be used also for any reduct and any parallel subsystem of p.

When the transition relation is clear from the context, we will simply say that a test witnesses a barb a. Moreover, abusing the notation, we will often denote by t_x both a concrete test and the underlying system.

$$\frac{p \to p'}{p \rightsquigarrow p'} \qquad\qquad \frac{p \rightsquigarrow p' \quad q \rightsquigarrow q'}{p|q \rightsquigarrow p'|q'}$$

Fig. 2. Parametric rules for a concurrent transition relation.

Hereafter, we assume that any barb $a \in \mathcal{B}$ is witnessed by some test and that we may uniquely choose such a test, referred to as the *canonical test* for a and denoted T^a

(B) For any $a \in \mathcal{B}$ there exists a test T^a witnessing a with respect to $\to$.

The assumption above holds for any calculus endowed with reduction semantics and barbs that we are aware of (see e.g. (Milner 1999, Amadio et al. 1996, Cardelli and Gordon 2000, Fournet and Gonthier 1996)). For instance, in the asynchronous calculus ACCS each output barb $\bar{a}$ is witnessed by the test $T^{\bar{a}} = \{a.\bar{x} : x \in \mathcal{N}\}$. Indeed, for all processes p, a stable concrete test for $\bar{a}$ on p can be $t_x^{\bar{a}} = a.\bar{x}$, for $x \in \mathcal{N}$ a name that does not occur syntactically in p. Note that input barbs cannot be witnessed by any test in ACCS, since there are no output prefixes. In SCCS, instead, for the presence of both input and output prefixes, an input barb a is witnessed by the test $\{\bar{a}.\bar{x} : x \in \mathcal{N}\}$.

The existence of tests witnessing barbs will be pivotal for the results in Section 1.4: the chosen witnesses for (concurrent) barbs will be used in the formulation of our Axiom of Asynchrony (AA), which abstractly characterises a basic feature of asynchronous systems with reduction semantics and barbs.

1.3. *Concurrent Transitions, Concurrent Barbs and Non-Interleaving Semantics*

Most semantics for interactive systems are *interleaving*, meaning that parallelism is reduced to non-determinism, or, in terms of processes, $a.b + b.a \sim a|b$. Here we propose a non-interleaving semantics based on barbs. For this, we first need a *concurrent transition relation* on systems $\rightsquigarrow \subseteq \mathcal{P} \times \mathcal{P}$: the concurrent transition system $\langle \mathcal{P}, \rightsquigarrow \rangle$, built upon the non-concurrent one $\langle \mathcal{P}, \to \rangle$, is assumed to satisfy the axiom

$$\textbf{(C)} \quad \to \subseteq \rightsquigarrow \subseteq \to^*$$

The assumption is quite natural: it just means that (1) each non-concurrent transition can be seen as a special concurrent transition and (2) each concurrent transition $p \rightsquigarrow q$ can be simulated by a sequence of non-concurrent ones $p \to \ldots \to q$.

An immediate consequence of axiom (C) is that reachability with respect to the sequential or the concurrent transition relation coincide.

Lemma 2 (concurrent vs. sequential reachability). The transitive closure of the concurrent and non-concurrent transition relations coincide, i.e., $\rightsquigarrow^* = \to^*$

As an example, for both SCCS and ACCS the concurrent transition relation $\rightsquigarrow$ can be defined by the rules in Figure 2. Note that processes running in parallel can always perform transitions concurrently. Alternative definitions of $\rightsquigarrow$ could be given, in order e.g. to avoid several concurrent communications on the same channel. Still, the theory would be applicable (see Section 2.3) since we abstract from the actual definition of $\rightsquigarrow$ and we only rely on property (C) above.

$$\frac{p\downarrow_a}{p\downarrow^c_a} \qquad\qquad \frac{p\downarrow^c_A \quad q\downarrow^c_B}{p|q\downarrow^c_{A\otimes B}}$$

Fig. 3. Parametric rules for concurrent barbs.

As a second ingredient, we introduce concurrent barbs as an extension of the set of barbs. Given an abelian semigroup $\langle S, \otimes\rangle$, i.e., a set S with an associative and commutative operation $\otimes$, we say that $X \subseteq S$ is a set of generators for S if for any $a \in S$ there exists $x_1, \ldots, x_n \in X$ such that $a = x_1 \otimes \ldots \otimes x_n$.

Definition 5 (concurrent barbs). A set of *concurrent barbs* $\mathcal{CB}$ is a set of predicates on $\mathcal{P}$, endowed with an associative and commutative operation $\otimes$ such that $\mathcal{B} \subseteq \mathcal{CB}$ is a set of generators for $\langle \mathcal{CB}, \otimes\rangle$ and for all barbs $a, a_1, \ldots, a_n \in \mathcal{B}$

1 if $p\downarrow_a$, then $p\downarrow^c_a$
2 if $p\downarrow^c_{a_1\otimes\ldots\otimes a_n}$, then $p\downarrow_{a_i}$ for all $i \in \{1, \ldots, n\}$

where $\downarrow^c_-$ denotes the satisfaction relation for $\mathcal{CB}$.

Concurrent barbs in $\mathcal{CB}$ will be ranged over by A, B, X, Y, ... Weak and permanent satisfaction for concurrent barbs are then defined in the obvious way. A system p *weakly satisfies* A, written $p\Downarrow^c_A$, if $p'\downarrow^c_A$ for some p' such that $p \rightsquigarrow^* p'$. Moreover, p *permanently satisfies* A, written $p\,\Box\!\downarrow^c_A$, if $p'\downarrow^c_A$ for all p' such that $p \rightsquigarrow^* p'$. We also write $p\Box\!\Downarrow^c_A$ if $p'\Downarrow^c_A$ for all p' such that $p \rightsquigarrow^* p'$.

According to the definition above, concurrent barbs are built from basic barbs by using the operator $\otimes$. Condition (1) says that the satisfaction relation for basic barbs remains unchanged when they are seen as concurrent barbs, while condition (2) guarantees that the satisfaction of a concurrent barb implies the satisfaction of its components in $\mathcal{B}$.

For the running examples of SCCS and ACCS, we can take $\mathcal{CB} = \mathcal{B}^\otimes$ (the free commutative monoid over $\mathcal{B}$). The elements of $\mathcal{B}^\otimes$ are multisets of barbs in $\mathcal{B}$ and the operator $\otimes$ is multiset composition. Then the satisfaction relation is defined by the rules in Figure 3: concurrent barbs essentially check the presence of several parallel inputs and outputs.

We remark that the definition of what a concurrent barb is depends on the choice of the concurrent transition relation. The link is established by the fact that, as clarified in the next section, also concurrent barbs must be witnessed by a test.

Definition 6 (concurrent saturated barbed bisimilarity). *Concurrent saturated barbed bisimilarity* $\sim^c$ is obtained by replacing $\rightarrow$ with $\rightsquigarrow$ and $\Downarrow_a$ with $\Downarrow^c_A$ in Definition 3.

The concurrent equivalence may distinguish systems that are identical in the interleaving semantics. For example, in SCCS $a.b + b.a \not\sim^c a|b$ since $a.b + b.a$ does not satisfy $\Downarrow^c_{a\otimes b}$, while $a|b$ does. Instead, it is easy to see that in ACCS, where only output barbs are available, the two processes are equivalent with respect to $\sim^c$. Indeed, it can be shown that in ACCS, $\sim^c = \sim$. In the next section we will argue that this is a general fact that applies to any formalism where systems can only interact asynchronously. Here we show that, by only relying on the definition of concurrent barbs, and on the axioms introduced so far, concurrent saturated barbed equivalence $\sim^c$ refines the non-concurrent one $\sim$.

We first prove a technical lemma which relates concurrent and non-concurrent barbs.

Lemma 3. Let p be a system and $a, a_1, \ldots, a_n \in \mathcal{B}$ barbs. Then

1 if $p{\Downarrow}_a$, then $p{\Downarrow}^c_a$
2 if $p{\Downarrow}^c_{a_1 \otimes \ldots \otimes a_n}$, then $p{\Downarrow}_{a_i}$ for all $i \in \{1, \ldots, n\}$
3 if $p \,\Box{\downarrow}^c_{a_1 \otimes \ldots \otimes a_n}$, then $p \,\Box{\downarrow}_{a_i}$ for all $i \in \{1, \ldots, n\}$

Proof. (1) If $p{\Downarrow}_a$, then $p' \downarrow_a$ for some p' such that $p \to^* p'$. By Lemma 2, $p \rightsquigarrow^* p'$ and by Definition 5 (property 1) $p' \downarrow^c_a$. Hence $p{\Downarrow}^c_a$. (2) Let $A = a_1 \otimes \ldots \otimes a_n$ and assume that $p{\Downarrow}^c_A$. Thus $p' \downarrow^c_A$ for some p' such that $p \rightsquigarrow^* p'$. By Definition 5 (property 2), $p' \downarrow_{a_i}$ for all $i \in \{1, \ldots, n\}$. Moreover, by Lemma 2 we have that $p \to^* p'$ and, by definition of weak barbs, we have that $p{\Downarrow}_{a_i}$ for all $i \in \{1, \ldots, n\}$. (3) Assume that for $A = a_1 \otimes \ldots \otimes a_n$, it holds $p \,\Box{\downarrow}^c_A$. Then $p' \downarrow^c_A$ for all p' such that $p \rightsquigarrow^* p'$. Now, for all p' such that $p \to^* p'$, by Lemma 2, $p \rightsquigarrow^* p'$ and thus $p' \downarrow^c_A$. By Definition 5 (property 2), this in turn implies that $p' \downarrow_{a_i}$ for all $i \in \{1, \ldots, n\}$. Thus we have $p \,\Box{\downarrow}_{a_i}$ for all $i \in \{1, \ldots, n\}$. $\Box$

The first two items are the weak counterparts of the properties holding for concurrent barbs (see Definition 5). The third result is needed in later sections (see Lemma 4).

Now, the desired result follows immediately.

Proposition 1. Concurrent saturated barbed bisimilarity refines saturated barbed bisimilarity, i.e., $\sim^c \subseteq \sim$.

Proof. We prove that $\sim^c$ is a saturated barbed bisimulation according to Definition 3. Let $p, q \in \mathcal{P}$ such that $p \sim^c q$.

— If $p{\Downarrow}_a$, then by Lemma 3(1) $p{\Downarrow}^c_a$ and, since $p \sim^c q$, then $q{\Downarrow}^c_a$. By Lemma 3(2) $q{\Downarrow}_a$.
— If $p \to^* p'$, then by Lemma 2 $p \rightsquigarrow^* p'$ and, since $p \sim^c q$, then $q \rightsquigarrow^* q'$ and $p' \sim^c q'$. Again by Lemma 2 $q \to^* q'$.
— For any r, since $\sim^c$ is a congruence, $p|r \sim^c q|r$.

By the three properties above the proposition holds. $\Box$

1.4. *Concurrency Can't Be Observed, Asynchronously*

This section focuses on the observability of concurrency through asynchronous interactions, arguing that $\sim^c = \sim$ in formalisms with asynchronous communication. Tests are thus needed that witness *concurrent* barbs. For this reason, we require that (B) actually holds for concurrent barbs and we fix a canonical test T^A for each $A \in \mathcal{CB}$.

(CB) For any $A \in \mathcal{CB}$ there exists a test T^A witnessing A with respect to $\rightsquigarrow$.

It is easy to see that axiom (CB) holds for our running examples. In ACCS, every concurrent barb $A = \overline{a_1} \otimes \ldots \otimes \overline{a_n} \in \mathcal{CB}$ is witnessed by the test $T^A = \{a_1.\overline{x_1} | \ldots | a_n.\overline{x_n} : \forall i,\ x_i \in \mathcal{N}\}$. Indeed, for any process p, a stable concrete test for A on p can be $t^A_X = a_1.\overline{x_1} | \ldots | a_n.\overline{x_n}$, where $X = x_1 \otimes \ldots \otimes x_n$ for $x_1, \ldots, x_n \in \mathcal{N}$ different names that do not occur syntactically in p. In SCCS, tests are defined analogously, i.e., given a concurrent barb $A = \alpha_1 \otimes \ldots \otimes \alpha_n$ (where now each α_i can be either an input or an output), a test is given by $T^A = \{\overline{\alpha_1}.\overline{x_1} | \ldots | \overline{\alpha_n}.\overline{x_n} : x_i \in \mathcal{N} \text{ for } i \in \{1, \ldots, n\}\}$.

Next lemma relates tests witnessing concurrent barbs in $\mathcal{CB}$ with barbs in $\mathcal{B}$ and $\rightarrow^*$.

Lemma 4. Let $A \in \mathcal{CB}$ be a concurrent barb, p a system and t_X^A a stable concrete test for A on p with respect to $\rightsquigarrow$ such that $X = x_1 \otimes \ldots \otimes x_m$ for $x_1, \ldots, x_m \in \mathcal{B}$. If $p \Downarrow_A^c$, then there exists p' such that $p|t_X^A \rightarrow^* p'$ and $p' \Box\Downarrow_{x_i}$ for all $i \in \{1, \ldots, m\}$.

Proof. If $p \Downarrow_A^c$ then there exists q such that $p \rightsquigarrow^* q$ and $q \downarrow_A^c$. By stability, t_X^A is also a concrete test on q, hence we have $q|t_X^A \rightsquigarrow p'$ and $p' \Box\downarrow_X^c$. By Lemma 3(3), for all $i \in \{1, \ldots, m\}$ we have that $p' \Box\downarrow_{x_i}$, and thus $p' \Box\Downarrow_{x_i}$. In order to conclude we just need to prove that $p|t_X^A \rightarrow^* p'$. Since $p \rightsquigarrow^* q$ and $q|t_X^A \rightsquigarrow p'$, by Lemma 2, $p \rightarrow^* q$ and $q|t_X^A \rightarrow^* p'$, and thus, by axiom (P), we have that $p|t_X \rightarrow^* q|t_X^A \rightarrow^* p'$. $\square$

A further assumption is now needed, ensuring the inverse of the above lemma. As it is intended to capture an essential feature of asynchronous communication, it is referred to as the *Axiom of Asynchrony*

(AA) Let $A \in \mathcal{CB}$ be a concurrent barb and let T^A be the canonical test for A. Let p be a system and t_X^A a stable concrete test for A on p with $X = x_1 \otimes \cdots \otimes x_m$ for $x_1, \ldots, x_m \in \mathcal{B}$. If $p|t_X^A \rightarrow^* p_1 \rightarrow^* \ldots \rightarrow^* p_n$, with $p_i \downarrow_{x_i}$ for $i \in \{1, \ldots, n\}$, then $p \Downarrow_A^c$.

Informally, the axiom can be explained as follows. We may think of A as a multiset of output messages. The fact that t_X^A is a concrete test for A on p and that $p|t_X^A \rightarrow^* p_1 \downarrow_{x_1} \rightarrow^* \ldots \rightarrow^* p_n \downarrow_{x_n}$ means that p can emit the messages in A one after the other. Then the intuition is that, if the system is asynchronous and thus sending is non-blocking, the messages can be also kept internally and made all available concurrently at the end.

As for our running examples, axiom (AA) holds in ACCS for the previously defined concurrent barbs and canonical tests. Instead, it fails for SCCS. In fact, take the SCCS process $p = \bar{a}.\bar{b}$. A concrete test for the concurrent barb $A = \bar{a} \otimes \bar{b}$ on p can be $t_X^A = a.\overline{x_1}|b.\overline{x_2}$ with $X = \overline{x_1} \otimes \overline{x_2}$. Yet, $p|t_X^A \rightarrow \bar{b}|\overline{x_1}|b.\overline{x_2} \rightarrow \overline{x_1}|\overline{x_2}$, with $\bar{b}|\overline{x_1}|b.\overline{x_2} \downarrow_{\overline{x}_1}$ and $\overline{x_1}|\overline{x_2} \downarrow_{\overline{x}_2}$, but $p \not\Downarrow_A^c$. Without going any further, it might be possible to argue that the axiom would still fail for any choice of a concrete stable test for $A = \bar{a} \otimes \bar{b}$ on p in SCCS and thus (AA) could never be satisfied.

Relying on the Axiom of Asynchrony, we prove that an inverse of Lemma 4 holds.

Lemma 5. Let $A \in \mathcal{CB}$ be a concurrent barb, p a system and t_X^A a stable concrete test for A on p such that $X = x_1 \otimes \ldots \otimes x_m$ for $x_1, \ldots, x_m \in \mathcal{B}$. If there exists p' such that $p|t_X^A \rightarrow^* p'$ and $p' \Box\Downarrow_{x_i}$ for all $i \in \{1, \ldots, m\}$, then $p \Downarrow_A^c$.

Proof. The lemma easily follows from the Axiom of Asynchrony (AA). Indeed, assume that $p|t_X^A \rightarrow^* p'$ and $p' \Box\Downarrow_{x_i}$ for all $i \in \{1, \ldots, m\}$. Then, in particular, $p' \Downarrow_{x_1}$ hence $p' \rightarrow^* p_1$ for some p_1 such that $p_1 \downarrow_{x_1}$ and $p_1 \Box\Downarrow_{x_i}$ for all $i \in \{1, \ldots, n\}$. From the latter we have that $p_1 \rightarrow^* p_2$, with $p_2 \downarrow_{x_2}$ and $p_2 \Box\Downarrow_{x_i}$ for all $i \in \{1, \ldots, n\}$. Iterating this reasoning, we have that

$$p' \rightarrow^* p_1 \rightarrow^* p_2 \rightarrow^* \ldots \rightarrow^* p_n$$

with $p_i \downarrow_{x_i}$ for all $i \in \{1, \ldots, n\}$. Thus by using (AA) we conclude $p \Downarrow_A^c$. $\square$

Lemmata 4 and 5 above are the real key for our main theorem: they state that concurrent barbs add no observational power. With these results, it is easy to prove that if two processes are "sequential" bisimilar then they satisfy the same concurrent barbs.

Proposition 2. If $p \sim q$ then it holds that $p \Downarrow^c_A$ iff $q \Downarrow^c_A$ for all $A \in \mathcal{CB}$.

Proof. Let $A \in \mathcal{CB}$ be a concurrent barb and let t^A_X a concrete test for A on both p and q. Observe that we can find a common concrete test on the two systems by taking a stable concrete test t^A_X for A on $p|q$. Then stability ensures that it is also a concrete test for A on p and q.

Since $p \sim q$ then $p|t^A_X \sim q|t^A_X$. Now suppose that $p \Downarrow^c_A$. Since, according to Definition 5, $\mathcal{B}$ is a set of generators for $\mathcal{CB}$, $X = x_1 \otimes \ldots \otimes x_n$ for some $x_1, \ldots, x_n \in \mathcal{B}$. Hence, by Lemma 4, $p|t^A_X \to^* p'$ and $p' \Box \Downarrow_{x_i}$ for all $i \in \{1, \ldots, m\}$. Since $p|t^A_X \sim q|t^A_X$, then also $q|t^A_X \to^* q'$ with $p' \sim q'$. By Lemma 1, $q' \Box \Downarrow_{x_i}$ for all $i \in \{1, \ldots, m\}$. Now, by Lemma 5 we have that $q \Downarrow^c_A$. $\square$

With the above proposition and Lemma 2, it is easy to prove that saturated barbed bisimilarity $\sim$ is finer than its concurrent version $\sim^c$.

Proposition 3. $\sim \subseteq \sim^c$

Proof. In order to prove that $\sim \subseteq \sim^c$, we show that $\sim$ is a saturated concurrent barbed bisimulation according to Definition 6. Let $p, q \in \mathcal{P}$ such that $p \sim q$.

— If $p \Downarrow^c_A$, then by Proposition 2 also $q \Downarrow^c_A$.
— If $p \rightsquigarrow^* p'$, then by Lemma 2 also $p \to^* p'$. Since $p \sim q$, then $q \to^* q'$ with $p' \sim q'$. Again by Lemma 2, $q \rightsquigarrow^* q'$.
— For any r, since $\sim$ is a congruence, $p|r \sim q|r$.

By the three properties above the proposition holds. $\square$

From Proposition 3 and Proposition 1, our main result immediately follows.

Theorem 1 (concurrency can't be observed, asynchronously). For any formalism satisfying axioms (P), (CB), (C), and (AA), concurrent saturated barbed bisimilarity and saturated barbed bisimilarity coincide, i.e., $\sim = \sim^c$.

2. Asynchronous π-calculus

This section shows that the asynchronous π-calculus fits in the theory of Section 1, and thus saturated barbed bisimilarity (which coincides with barbed congruence (Amadio et al. 1996)) and its concurrent version coincide. We prove the result for two concurrent semantics, which differ on the possibility of performing multiple concurrent communications on the same channel. As a side effect, the behavioural equivalences induced by the two concurrent semantics are thus proved to coincide.

$$p ::= \bar{a}b,\ p_1|p_2,\ (\nu a)p,\ !m,\ m \qquad m ::= \mathbf{0},\ \alpha.p,\ m_1 + m_2 \qquad \alpha ::= a(b),\ \tau$$

$$p|q \equiv q|p \qquad (p|q)|r \equiv p|(q|r) \qquad p|\mathbf{0} \equiv p$$
$$m+n \equiv n+m \qquad (m+n)+o \equiv m+(n+o) \qquad m+\mathbf{0} \equiv m$$
$$(\nu a)(\nu b)p \equiv (\nu b)(\nu a)p \qquad (\nu a)(p|q) \equiv p|(\nu a)q \quad \text{if } a \notin fn(p) \qquad (\nu a)\mathbf{0} \equiv \mathbf{0}$$
$$(\nu a)p \equiv (\nu b)(p\{^b/_a\}) \ \text{ if } b \notin fn(p) \qquad a(b).p \equiv a(c).(p\{^c/_b\}) \ \text{ if } c \notin fn(p) \qquad !p \equiv p|!p$$

$$\bar{a}b|(a(c).p+m) \to p\{^b/_c\} \qquad \tau.p+m \to p \qquad \frac{p \to q}{(\nu a)p \to (\nu a)q} \qquad \frac{p \to q}{p|r \to q|r}$$

Fig. 4. Syntax, structural congruence and reduction relation of the asynchronous π.

2.1. *Asynchronous π-calculus*

The asynchronous π-calculus has been introduced in (Honda and Tokoro 1991) as a model of distributed systems interacting via asynchronous message passing. Its syntax is shown in Figure 4: we assume an infinite set $\mathcal{N}$ of *names*, ranged over by a, b, $\ldots$, with $\tau \notin \mathcal{N}$, and we let p, q, $\ldots$ range over the set $\mathcal{P}_\pi$ of processes. *Free names* of a process p (denoted by $fn(p)$) are defined as usual. Processes are taken up to a *structural congruence*, axiomatised in Figure 4 and denoted by $\equiv$. The *reduction relation*, denoted by $\to$, describes process evolution: it is the least relation $\to \subseteq \mathcal{P}_\pi \times \mathcal{P}_\pi$ closed under $\equiv$ and inductively generated by the axioms and rules in Figure 4.

As for ACCS (Section 1), barbs account only for outputs. More precisely, the set of barbs for the asynchronous π-calculus is $\mathcal{B}_\pi = \{\bar{a} : a \in \mathcal{N}\}$ and a process $p \in \mathcal{P}_\pi$ satisfies the barb $\bar{a}$, in symbols $p\downarrow_{\bar{a}}$, if $p \equiv (\nu c)(\bar{a}b|q)$, where $a \neq c$ (Amadio et al. 1996).

2.2. *Concurrent Semantics with Unbounded Capacity Channels*

A non-interleaving semantics for the calculus can be obtained by introducing a concurrent transition relation $\rightsquigarrow$, as defined by the rules in Figure 2 plus the additional rule below, taking into account the restriction operator

$$\frac{p \rightsquigarrow p'}{(\nu a)p \rightsquigarrow (\nu a)p'}$$

Note that processes running in parallel (possibly under a restriction) can always perform transitions concurrently. This is due to the fact that we assume that channels have no bounded capacity and thus multiple communications over the same channel are allowed, as in the semantics proposed in (Busi and Gorrieri 1995, Montanari and Pistore 1995). Different approaches are conceivable, see e.g. (Lanese 2007): next section shows how they can be accommodated in our theory.

Concurrent barbs are multisets of outputs, and they check for the presence of several parallel outputs. Formally, $\mathcal{CB}_\pi = {\mathcal{B}_\pi}^{\otimes}$ and the satisfaction relation $\downarrow^c$ is defined by the rules in Figure 3, extended with the rules in Figure 5, where $\varnothing$ is the empty multiset and $A \setminus a$ is the multiset obtained from A by removing all the occurrences of a.

$$\frac{}{p\downarrow^c_{\varnothing}} \qquad\qquad \frac{p\downarrow^c_A}{(\nu a)p\downarrow^c_{A\setminus a}}$$

Fig. 5. Additional rules for concurrent barbs in asynchronous π.

The rightmost rule takes into account concurrent processes running under a restriction. Consider for instance the process $(\nu b)(\overline{a}b|\overline{b}c|\overline{c}b)$: it cannot be decomposed into the parallel composition of sub-processes, yet, intuitively, barbs $\bar{a}$ and $\bar{c}$ should be observable, while b should be not. Indeed, since $\overline{a}b|\overline{b}c|\overline{c}b\downarrow^c_{\bar{a}\otimes\bar{b}\otimes\bar{c}}$, using the rightmost rule in Figure 5 we get that $(\nu b)(\overline{a}b|\overline{b}c|\overline{c}b)\downarrow^c_{\bar{a}\otimes\bar{c}}$. The removal of barbs due to restrictions may end up in concurrently observing the empty multiset, as it happens, e.g., for the process $(\nu b)\overline{b}c$, hence all processes should intuitively observe it. This calls for the leftmost rule in Figure 5.

Now, let $\sim_\pi$ denote saturated barbed bisimilarity for the asynchronous π-calculus and let $\sim^c_\pi$ denote the concurrent one. It is worth remarking that $\sim_\pi$ coincides with the standard semantics for the calculus, namely, *asynchronous bisimilarity* (Amadio et al. 1996), as shown in (Fournet and Gonthier 2005). Then we have the following result.

Corollary 1 (concurrency can't be observed in asynchronous π). $\sim_\pi=\sim^c_\pi$.

Proof. The result follows from Theorem 1 as all the needed axioms are satisfied. Indeed, axioms (P) and (C) clearly hold. Concerning axiom (CB), given any concurrent barb $A = \overline{a_1}\otimes\ldots\otimes\overline{a_n}$ a test witnessing A is $T^A = \{t^A_C : C = \overline{c_1}\otimes\ldots\otimes\overline{c_n}\}$ where

$$t^A_C = a_1(b_1).\overline{c_1}c_1|\ldots|a_n(b_n).\overline{c_n}c_n \qquad \text{with } b_i \neq c_i \text{ for all } i.$$

For a processes p, we obtain a stable concrete test t^A_C for A on p by taking $C = \overline{c_1}\otimes\ldots\otimes\overline{c_n}$ containing only names c_i that do not occur syntactically in p.

With the above definition, it is easy to prove that the axiom (AA) holds. In fact, for the sake of simplicity, take $A = a_1\otimes a_2$ (the general case is analogous) and consider the test $t^A_C = a_1(b_1).\overline{c_1}c_1|a_2(b_2).\overline{c_2}c_2$, where c_1 and c_2 do not occur in p. Assume that $p|t^A_C \to^* p_1\downarrow_{\overline{c_1}} \to^* p_2\downarrow_{\overline{c_2}}$. We need to prove that $p\Downarrow^c_A$. Since $p|t^A_C \to^* p_1\downarrow_{\overline{c_1}}$ we have that

$$p \to^* (\nu d_1)(\overline{a_1}e_1|p'), \tag{1}$$

where $a_1 \neq d_1$. With this setup, $p_1 = (\nu d_1)(p'|\overline{c_1}c_1)|a_2(b_2).\overline{c_2}c_2 \equiv (\nu d_1)(p'|\overline{c_1}c_1|a_2(b_2).\overline{c_2}c_2)$, assuming with no loss of generality that $d_1 \neq a_2$ (moreover $d_1 \neq c_2$ by hypothesis).

Now, from $p_1 \to^* p_2\downarrow_{\overline{c_2}}$, an analogous reasoning allows us to conclude that

$$p' \to^* (\nu d_2)(\overline{a_2}e_2|p''), \tag{2}$$

with $a_2 \neq d_2$. Putting together (1) and (2), and assuming that $d_2 \notin \{a_1, e_1\}$ we conclude

$$p \to^* p''' = (\nu d_1)(\overline{a_1}e_1|(\nu d_2)(\overline{a_2}e_2|p'')) \equiv (\nu d_1)(\nu d_2)(\overline{a_1}e_1|\overline{a_2}e_2|p'')$$

and by case analysis on d_1, d_2 it holds $p'''\downarrow^c_{\overline{a_1}\otimes\overline{a_2}}$, i.e., $p'''\downarrow^c_A$. Thus, as desired, $p\Downarrow^c_A$. $\square$

$$\bar{a}b|(a(c).p+m) \to_a p\{^b/_c\} \qquad \tau.p+m \to_\tau p \qquad \frac{p \to_\alpha q}{p|r \to_\alpha q|r} \qquad \frac{p \to_\alpha q \quad a \neq \alpha}{(\nu a)p \to_\alpha (\nu a)q} \qquad \frac{p \to_a q}{(\nu a)p \to_\tau (\nu a)q}$$

$$\frac{p \to_\alpha p'}{p \rightsquigarrow_{\{\alpha\}} p'} \qquad \frac{p \rightsquigarrow_A p' \quad q \rightsquigarrow_B q' \quad A \cap B \subseteq \{\tau\}}{p|q \rightsquigarrow_{A \cup B} p'|q'} \qquad \frac{p \rightsquigarrow_A p' \quad a \notin A}{(\nu a)p \rightsquigarrow_A (\nu a)p'} \qquad \frac{p \rightsquigarrow_A p' \quad a \in A}{(\nu a)p \rightsquigarrow_{(A\setminus\{a\})\cup\{\tau\}} (\nu a)p'}$$

Fig. 6. Semantics for the asynchronous π with 1-bounded channels.

2.3. *Concurrent Semantics with Bounded Capacity Channels*

The concurrent transition relation $\rightsquigarrow$ that we introduced above (as well as the one in Section 1.3) always allows processes running in parallel to perform transitions concurrently. However, since interactions are message exchanges on a channel, it can be reasonable to assume a bound on the number of concurrent communications on the *same* channel (see e.g. (Lanese 2007)). Here we consider 1-bounded channels, i.e., we allow for the transition

$$a(c_1).p|\bar{a}d_1|b(c_2).q|\bar{b}d_2 \rightsquigarrow p\{^{d_1}/_{c_1}\}|q\{^{d_2}/_{c_2}\}$$

but forbid

$$a(c_1).p|\bar{a}d_1|a(c_2).q|\bar{a}d_2 \not\rightsquigarrow p\{^{d_1}/_{c_1}\}|q\{^{d_2}/_{c_2}\}.$$

This choice becomes mandatory when considering systems interacting through ordered buffers (such as queues and stacks). Indeed, only one input (output) at a time can be performed on such buffers (Beauxis et al. 2008).

A concurrent transition relation $\rightsquigarrow$ capturing the above intuition can be defined by keeping track of the channels where synchronisations occur and avoiding two concurrent synchronisations on the same channel. This is formalised by the rules in Figure 6. The (sequential) transition relation $\to$ is labelled either by the name of the channel where a synchronisation occurs or by τ, in case of internal actions or restricted channels (α stands for a generic label). Then the concurrent transition relation $\rightsquigarrow$ is labelled by sets containing channel names (those where synchronisations occur) and τ. Two transitions can happen concurrently only if they are labelled with sets whose intersection includes at most τ. It is important to note that the label on the transition is just a syntactical device that allows for properly defining the relation itself, but it is not considered when defining the corresponding saturated barbed bisimilarity, which is denoted $\sim^c_{\pi a}$.

In this perspective, we also have to change the notion of concurrent barb. Indeed, multisets barbs would not be witnessed by a test with the new definition of $\rightsquigarrow$. As an example, consider the process $p = \bar{a}|\bar{a}|a|a$ (names exchanged along channel a are omitted since they are irrelevant here) and the barb $A = \bar{a} \otimes \bar{a}$. We have that $p{\downarrow_A}$, but there exists no stable concrete test for A on p, essentially because there exists no process that can consume both outputs concurrently. More formally, assume by contradiction that t^A_X is a stable test for A on p. This means that $p|t^A_X \rightsquigarrow p'$ and $p' \,\Box{\downarrow^c_X}$. Since channels are 1-bounded, the reaction can involve at most a single synchronisation on a, hence $p' = \bar{a}|a|p''$, with

$$\bar{a}|a|t^A_X \rightsquigarrow p'' \tag{3}$$

Now, observe that $p' \to p''$ and hence, recalling that $p' \,\Box\!\downarrow^c_X$, it must hold $p'' \,\Box\!\downarrow^c_X$. This, together with (3) and the fact that, by stability, t^A_X is a concrete test also on $\bar{a}|a$, would imply that $\bar{a}|a \downarrow^c_A$, which is false.

The problem above can be solved by defining concurrent barbs just as sets (rather then multisets) of barbs in $\mathcal{B}_\pi$, i.e. by taking $\mathcal{CB}_\pi = \mathbf{2}^{\mathcal{B}_\pi}$, the powerset of $\mathcal{B}_\pi$. The satisfaction relation $\downarrow^c_A$ is still defined by the rules in Figure 3 and Figure 5, where the operator $\otimes$ denotes set union (instead of multiset composition) and the operator $\setminus$ now stands for set difference. For $A = \{\overline{a_1}, \ldots, \overline{a_n}\}$, we have

$$p \downarrow^c_A \text{ if } p \equiv (\nu c_1) \ldots (\nu c_k)(\overline{a_1} b_1 | \ldots | \overline{a_n} b_n | q) \text{ and } a_i \neq c_j \text{ for all } i, j.$$

Note that several outputs might occur on each channel a_i, but their multiplicity is not taken into account as concurrent outputs on the same channel are not allowed.

With the above characterisation it is immediate to see that the properties 1 and 2 of Definition 5 hold. Also note that since concurrent barbs are now sets, $\{\bar{a}\} \otimes \{\bar{a}\}$ is equal to $\{\bar{a}\}$, which clearly can be witnessed by some test. More generally, the concurrent barb $A = \{\overline{a_1}, \ldots, \overline{a_n}\}$ is witnessed by $T^A = \{t^A_C : C = \{\overline{c_1}, \ldots, \overline{c_n}\}\}$ where

$$t^A_C = a_1(b_1).\overline{c_1} c_1 | \ldots | a_n(b_n).\overline{c_n} c_n \qquad \text{with } b_i \neq c_i \text{ for all } i.$$

For any process p, a stable concrete test t^A_C for A on p can be obtained by considering a set C containing only names c_i that do not occur in p. Therefore all concurrent barbs are witnessed, i.e., axiom (CB) holds.

Along the same lines of the unbounded semantics we can obtain, as a corollary of Theorem 1, the following result.

Corollary 2 (concurrency can't be observed in (bounded) asynchronous π). $\sim_\pi = \sim^c_{\pi a}$.

It is also interesting to observe that, as a consequence of Corollaries 1 and 2, we obtain $\sim^c_\pi = \sim^c_{\pi a}$, i.e., allowing or disallowing multiple synchronisations on the same channel in the asynchronous π-calculus does not affect saturated barbed bisimilarity.

3. Open Petri Nets

Open Petri nets (Kindler 1997, Milner 2003, Sassone and Sobociński 2005, Baldan et al. 2005) are a reactive extension of ordinary P/T nets, equipped with a distinguished set of *open places* that represent the interfaces through which the environment interacts with a net. This kind of interactions is inherently asynchronous (see e.g. (Baldan, Bonchi and Gadducci 2009)) and thus it represents an ideal testbed for our theory.

This section shows that indeed the interleaving and concurrent equivalences defined in the literature for open Petri nets (see e.g. (Baldan et al. 2005)) are instances of $\sim$ and $\sim^c$, respectively. Then, since all the axioms of our theory are satisfied, these equivalences coincide.

As in the previous sections, we denote by $X^\otimes$ the free commutative monoid generated by a set X, whose elements are called multisets. Moreover, the symbol 0 denotes the empty multiset, and for any $x_1, x_2 \in X^\otimes$, we write $x_1 \subseteq x_2$ if $x_1 = x_2 \otimes x$ for some $x \in X^\otimes$.

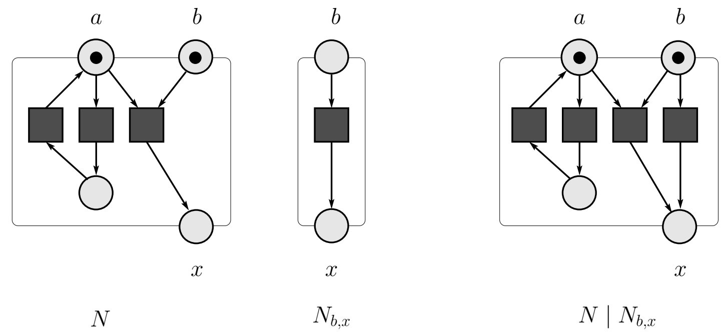


Fig. 7. Marked open nets and their parallel composition.

$$(\text{TR}) \quad \frac{m = {}^\bullet t \otimes m' \quad t \in T}{m \xrightarrow{0} t^\bullet \otimes m'} \qquad (\text{IN}) \quad \frac{s \in O}{m \xrightarrow{s^+} m \otimes s} \qquad (\text{OUT}) \quad \frac{m = m' \otimes s \quad s \in O}{m \xrightarrow{s^-} m'}$$

Fig. 8. Firing semantics for open nets.

Definition 7 (open nets). An *open net* is a tuple $\hat{N} = (S, T, {}^\bullet(.), (.)^\bullet, O)$ for S a set of places, T a set of transitions, ${}^\bullet(.), (.)^\bullet : T \to S^\otimes$ functions mapping each transition to its pre- and post-set, and $O \subseteq S$ a set of *open places*. A *marked (open) net* is a pair $N = \langle \hat{N}, m \rangle$ for $\hat{N}$ an open net and $m \in S^\otimes$ a marking. The *interface* of N is the set of open places O of $\hat{N}$.

Examples of marked nets can be found in Figure 7. As usual, circles represent places and rectangles transitions. Arrows from places to transitions represent function ${}^\bullet(.)$, arrows from transitions to places represent $(.)^\bullet$. An open net is enclosed in a box (representing the interface of the net) and open places are on the border of such a box.

We assume a fixed infinite set $\mathcal{S}$ of place names. The set of *interactions* (ranged over by i) is $\mathcal{I}_\mathcal{S} = \{s^+, s^- : s \in \mathcal{S}\}$. The set of *labels* (ranged over by ℓ) consists in $\{0\} \uplus \mathcal{I}_\mathcal{S}$. The firing (interleaving) semantics of open nets is expressed by the rules in Figure 8, where we write ${}^\bullet t$ and $t^\bullet$ instead of ${}^\bullet(t)$ and $(t)^\bullet$. The rule (TR) is the standard rule of P/T nets (seen as multiset rewriting) modelling internal transitions, which are labelled with 0 for subsequent use. The other two rules model the possible interactions with the environment: at any moment a token can be inserted in (rule (IN)) or removed from (rule (OUT)) an open place.

Weak transitions are defined as usual, i.e., $\stackrel{0}{\Rightarrow}$ denotes the reflexive and transitive closure of $\xrightarrow{0}$ and $\stackrel{i}{\Rightarrow}$ denotes $\stackrel{0}{\Rightarrow}\xrightarrow{i}\stackrel{0}{\Rightarrow}$. We write $N \stackrel{i}{\Rightarrow} N'$ when $N = \langle \hat{N}, m \rangle$, $N' = \langle \hat{N}, m' \rangle$ and $m \stackrel{i}{\Rightarrow} m'$.

Definition 8 (firing bisimilarity). A symmetric relation R over marked nets is a *firing bisimulation* if whenever $N_1 R N_2$ then

— if $N_1 \stackrel{\ell}{\Rightarrow} N_1'$ then $N_2 \stackrel{\ell}{\Rightarrow} N_2'$ and $N_1' R N_2'$.

We say that N_1 and N_2 are *firing bisimilar* (written $N_1 \approx N_2$) if there exists a firing bisimulation R relating them.

$$(\textsc{cfir})\ \frac{m \xrightarrow{\ell} m'}{m \overset{\ell}{\rightsquigarrow} m'} \qquad (\textsc{cstep})\ \frac{m_1 \overset{c_1}{\rightsquigarrow} m_1' \quad m_2 \overset{c_2}{\rightsquigarrow} m_2'}{m_1 \otimes m_2 \overset{c_1 \otimes c_2}{\rightsquigarrow} m_1' \otimes m_2'}$$

Fig. 9. Step semantics for open nets.

In order to ease the intuition, nets can be thought of as black boxes, where only the interfaces are visible. Two nets are bisimilar if they cannot be distinguished by an observer that may only insert and remove tokens in open places.

Steps of open nets ($\rightsquigarrow$) are defined in Figure 9. Step labels (ranged over by $c, c_1, c_2 \ldots$) are multisets of interactions $\mathcal{I}_N$. By rule (CFIR), each firing is also a step and, in particular, the label 0 is interpreted as the empty multiset. Rule (CSTEP) allows to construct concurrent steps. *Weak transitions* are defined as usual: $\overset{0}{\mapsto}$ denotes the reflexive and transitive closure of $\overset{0}{\rightsquigarrow}$ and $\overset{c}{\mapsto}$ denotes $\overset{0}{\mapsto}\overset{c}{\rightsquigarrow}\overset{0}{\mapsto}$. *Step bisimilarity* ($\approx^c$) is defined by replacing $\Rightarrow$ with $\mapsto$ in Definition 8.

We now show that $\approx$ and $\approx^c$ are instances of $\sim$ and $\sim^c$, respectively. The parallel composition of open nets N_1, N_2 is obtained by gluing them on their open places. In order to simplify the definition, given two open nets N_1 and N_2, we will assume, without loss of generality, that $T_1 \cap T_2 = \varnothing$ and $(S_1 - O_1) \cap (S_2 - O_2) = \varnothing$. This is possible as the identity of transitions and non-open places is irrelevant.

Definition 9 (parallel composition). Given two marked open nets N_1 and N_2, their *parallel composition* is the marked open net $N_1|N_2 = (S_1 \cup S_2, T_1 \cup T_2, {}^{\bullet}(.), (.)^{\bullet}, O_1 \cup O_2, m_1 \otimes m_2)$.

In words, $N_1|N_2$ is obtained by taking the disjoint union of the nets, merging open places with the same name and summing the markings. An example of composition is shown in Figure 7.

Transitions $\xrightarrow{0}$ of marked nets correspond to transitions $\rightarrow$ in the theory of Section 1, and $\overset{0}{\rightsquigarrow}$ corresponds to $\rightsquigarrow$. *Barbs* check the presence of tokens in open places. Formally, the set of barbs for open nets is

$$\mathcal{B}_N = \{b : b \in \mathcal{S}\},$$

and the marked net $N = \langle \hat{N}, m \rangle$ satisfies the barb $b \in \mathcal{B}_N$, denoted $N\downarrow_b$, if $b \in O$ (i.e., b is an open place of $\hat{N}$) and $b \subseteq m$. The set of *concurrent barbs* is the free commutative monoid over $\mathcal{B}_N$, i.e.,

$$\mathcal{CB}_N = {\mathcal{B}_N}^{\otimes}$$

A concurrent barb $m' \in \mathcal{CB}_N$ checks for the presence of a multiset of tokens in open places, namely, satisfaction is defined by $N\downarrow^c_{m'}$ if $m' \in O^{\otimes}$ and $m' \subseteq m$.

In order to apply the theory in Section 1, we need to show that the behavioural equivalences considered on open nets, i.e., firing bisimilarity and step bisimilarity, coincide with saturated barbed bisimilarity and its concurrent version, respectively.

Proposition 4. Let N_1, N_2 be two marked nets with the same interface. Then $N_1 \approx N_2$ iff $N_1 \sim N_2$ and $N_1 \approx^c N_2$ iff $N_1 \sim^c N_2$.

Proof. We first show that equivalences $\approx$ and $\sim$ coincide, proving the two inclusions.

($\approx\, \subseteq\, \sim$) In order to prove this inclusion we show that $\approx$ is a saturated barbed bisimulation, according to Definition 3. Let $N_1 \approx N_2$.

— If $N_1 \Downarrow_a$, then $N_1 \to^* N_1'$ and $N_1' \downarrow_a$. This means that $N_1' \xrightarrow{-a}$. Therefore

$$N_1 \stackrel{-a}{\Rightarrow} N_1''$$

and since $N_1 \approx N_2$

$$N_2 \stackrel{-a}{\Rightarrow} N_2''$$

which implies $N_2 \Downarrow_a$.

— If $N_1 \to^* N_1'$, then, since $N_1 \approx N_2$, also $N_2 \to^* N_2'$, with $N_1' \approx N_2'$, as desired.

— For any marked net N, since $\approx$ is a congruence (as proved in (Baldan et al. 2005)), we have that $N_1|N \approx N_2|N$.

($\sim\, \subseteq\, \approx$) In order to prove this inclusion we show that $\sim$ is a firing bisimulation. Instead of using the condition of Definition 8, we will use the following equivalent one

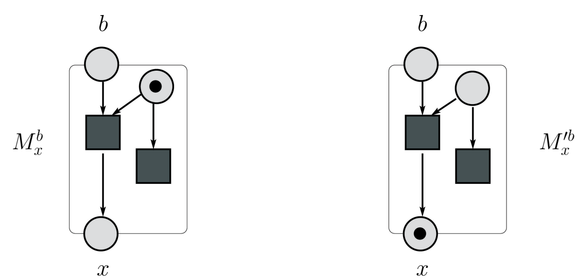
$$\text{if } N_1 \xrightarrow{\ell} N_1',\text{ then } N_2 \stackrel{\ell}{\Rightarrow} N_2' \text{ and } N_1' R N_2'.$$

Suppose that $N_1 \sim N_2$.

— If $N_1 \xrightarrow{0} N_1'$, then $N_1 \to N_1'$ and $N_1 \to^* N_1'$. By definition of $\sim$, $N_2 \to^* N_2'$ (and thus $N_2 \stackrel{0}{\Rightarrow} N_2'$) with $N_1' \sim N_2'$.

— Let $N_1 \xrightarrow{+b} N_1'$. Since b is open in N_1 it must be open also in N_2. Hence $N_2 \xrightarrow{+b} N_2'$, and clearly $N_1' = N_1|N_{+b}$ and $N_2' = N_2|N_{+b}$, where N_{+b} is the net consisting of a single place b, which is open and marked. Since $N_1 \sim N_2$ and $\sim$ is a congruence, we deduce that $N_1' \sim N_2'$.

— Let $N_1 \xrightarrow{-b} N_1'$ and let $x \in \mathcal{S}$ not belonging to the open places of N_1 and N_2. Therefore if M_x^b is the net below on the left



we have that $N_1|M_x^b \to N_1'|M_x'^b \Box\Downarrow_x$. Since $N_1 \sim N_2$ and $\sim$ is a congruence, it holds

$$N_2|M_x^b \to^* N_2''$$

with $N_1'|M_x'^b \sim N_2''$ and thus $N_2'' \Box\Downarrow_x$. It is not difficult to see that the only way in which this can hold is that in N_2'' a token in x has been produced, and thus a token in b has been consumed, i.e., $N_2'' = N_2'|M_x'^b$. Therefore

$$N_2 \stackrel{-b}{\Rightarrow} N_2'.$$

Now, since $N_1'|M_x'^b \sim N_2'|M_x'^b$ and since the transitions in $M_x'^b$ never fire, we have that $N_1' \sim N_2'$.

Concerning the concurrent equivalences, the proof that $\approx^c \subseteq \sim^c$ is obtained from the one of $\approx \subseteq \sim$, simply by replacing barbs with concurrent barbs and firings with steps. The same applies to the converse inclusion, $\sim^c \subseteq \approx^c$. Only note that in the third item, the net M_x^b must be replaced by the parallel composition $M_{x_1}^{b_1}|M_{x_2}^{b_2}|\ldots|M_{x_n}^{b_n}$. □

Note that requiring N_1 and N_2 to have the same interface is needed for having that $\sim \subseteq \approx$. Indeed, for any set $B \subseteq \mathcal{S}$, let N_B be the net consisting of all and only the places in B which are all open, without marking and without transitions. For all nets N, we have that $N \sim N|N_B$, while this is not generally true for $\approx$.

However this requirement is far from being restrictive and it is indeed quite common (see e.g., (Milner 2003, Baldan et al. 2009)). At a more general level, one could argue that all equivalent systems should have the same interface because otherwise they could immediately be distinguished by an external observer.

We can finally apply Theorem 1 in order to prove that firing and step bisimilarity coincide for open nets.

Corollary 3 (concurrency can't be observed in open nets). Firing and step bisimilarity coincide, i.e., $\approx = \approx^c$.

Proof. We prove that all the axioms required by Theorem 1 are satisfied by open nets. This is immediate for (P) and (C). Instead, concerning (CB), first notice that for any barb $b \in \mathcal{B}_N$, a test witnessing b is given by $T^b = \{t_x^b : x \in \mathcal{S}\}$, where $t_x^b = N_{b,x}$ is the net in Figure 7, middle. Then, for a concurrent barb $B = b_1 \otimes \ldots \otimes b_n \in \mathcal{CB}_N$, a test is given by

$$T^B = \{t_X^B \ : \ t_X^B = t_{x_1}^{b_1}|\ldots|t_{x_n}^{b_n} \ \wedge \ X = x_1 \otimes \ldots \otimes x_n\}$$

With this definition of test, the Axiom of Asynchrony (AA) can be easily shown to hold. Therefore, we can apply Theorem 1 to conclude that $\sim$ and $\sim^c$ coincide. Then using Proposition 4, we immediately get the thesis. □

4. On Selinger's Axiomatisation

An axiomatisation of different classes of systems with asynchronous communication has been proposed in (Selinger 1997). Roughly speaking, a system is said to be asynchronous if its observable behaviour is not changed by filtering its input and/or output through a suitable communication medium, which can store messages and release them later on. Different choices of the medium (queues, unordered buffers) are shown to lead to different notions of asynchrony, and suitable sets of axioms are then identified which are shown to precisely capture the various classes of asynchronous systems.

In order to further check the appropriateness of our framework, we prove that the class of systems characterised as asynchronous in (Selinger 1997) satisfy the requirements in Section 1. More precisely, we focus on so-called *out-buffered asynchrony with feedback* (Selinger 1997, Section 3.2), where output is asynchronous, the order of messages is not preserved, and the output of a process can be an input for the process itself (feedback). The corresponding axioms (Selinger 1997, Table 3) are listed in Figure 10.

$$\begin{tikzcd} p \arrow[r, "out\,a"] & p' \arrow[d, "\alpha"'] \\ & q \end{tikzcd} \Rightarrow \begin{tikzcd} p \arrow[r, "out\,a"] \arrow[d, "\alpha"] & p' \arrow[d, "\alpha"'] \\ p'' \arrow[r, "out\,a"] & q \end{tikzcd}$$

output-commutativity (FB1)

$$\begin{tikzcd} p \arrow[r, "out\,a"] \arrow[d, "\alpha"] & p' \\ p'' & \end{tikzcd} \Rightarrow \begin{tikzcd} p \arrow[r, "out\,a"] \arrow[d, "\alpha"] & p' \arrow[d, "\alpha"'] \\ p'' \arrow[r, "out\,a"] & q \end{tikzcd}$$

where $\alpha \neq out\,a$ and $\alpha \neq \tau$
output-confluence (FB2)

$$\begin{tikzcd} p \arrow[r, "out\,a"] \arrow[d, "out\,a"] & p' \\ p'' & \end{tikzcd} \Rightarrow p' = p''$$

output-determinacy (FB3)

$$\begin{tikzcd} p \arrow[r, "out\,a"] & p' \arrow[d, "in\,a"'] \\ & q \end{tikzcd} \Rightarrow \begin{tikzcd} p \arrow[r, "out\,a"] \arrow[dr, "\tau"'] & p' \arrow[d, "in\,a"] \\ & q \end{tikzcd}$$

feedback (FB4)

$$\begin{tikzcd} p \arrow[r, "out\,a"] \arrow[d, "\tau"] & p' \\ p'' & \end{tikzcd} \Rightarrow \begin{tikzcd} p \arrow[r, "out\,a"] \arrow[d, "\tau"] & p' \arrow[d, "\tau"'] \\ p'' \arrow[r, "out\,a"] & q \end{tikzcd} \quad or \quad \begin{tikzcd} p \arrow[r, "out\,a"] \arrow[d, "\tau"'] & p' \arrow[dl, "in\,a"] \\ p'' & \end{tikzcd}$$

output-tau (FB5)

Fig. 10. Axioms for out-buffered agents with feedback.

$$\frac{p \xrightarrow{\alpha} p'}{p|q \xrightarrow{\alpha} p'|q} \qquad \frac{q \xrightarrow{\alpha} q'}{p|q \xrightarrow{\alpha} p|q'} \qquad \frac{p \xrightarrow{out\,a} (\xrightarrow{\tau})^k \xrightarrow{in\,a} r}{p\ (\xrightarrow{\tau})^{k+1}\ r}$$

Fig. 11. Rules for the parallel operator.

They are given for labelled transition systems, with labels $in\,a$, $out\,a$ and τ, denoting input, output and internal transitions, respectively. The names a of inputs and outputs are taken respectively from (possibly equal) sets $\mathcal{X}$ and $\mathcal{Y}$, and α denotes a generic label.

In order to bring the correspondence to a formal level, we must overcome two problems. Firstly, the theory in (Selinger 1997) is developed for a labelled semantics, while we are concerned with barbed reduction semantics, and secondly, the theory in (Selinger 1997) does not consider concurrent transitions, which are pivotal in our setting.

On issue is solved by taking as reductions $p \to p'$ the τ-transitions $p \xrightarrow{\tau} p'$ and letting barbs check if system can perform a transition labelled with an *out* action: the set of barbs is $\mathcal{B} = \{a : a \in \mathcal{Y}\}$ and a system p satisfies the barb a, in symbols $p{\downarrow_a}$, if $p \xrightarrow{out\,a}$.

As a parallel operator for out-buffered agents with feedback, we use the parallel composition with interaction (Selinger 1997, Section 3.1) given by the rules in Figure 11‡.

As far as concurrent barbs are concerned, they are multisets of elements of $\mathcal{Y}$, formally $\mathcal{CB} = \mathcal{B}^{\otimes}$, and we define $p{\downarrow^c_A}$, where $A = \bigotimes_{i=1}^{n} a_i$, whenever $p \xrightarrow{out\,a_1} \ldots \xrightarrow{out\,a_n}$. This is motivated by the fact that, in this situation, by axiom (FB1), the same outputs can be performed by p in *any* order (in particular, $p \xrightarrow{out\,a_i}$ for any $i \in \{1, \ldots, n\}$). In words, although the labelled transition system does not provide any information on concurrency, we assume that outputs which can be observed in any order are generated concurrently.

‡ Actually, this operator is associative and commutative only up-to isomorphism of the underlying transition space of the system, which is implicitly assumed here. Also, we adopted a restrained version of the right-most rule, in order to account for the number of internal transitions performed by a system: it simplifies the presentation, and it is sound since we consider weak semantics anyhow.

Finally, let $\mathcal{L}(p)$ be the set of labels occurring in any transition $p \xrightarrow{\alpha}$ and let $\mathcal{N}(p)$ be the set of names in the labels occurring in any transition $p' \xrightarrow{\alpha}$ for p' reachable from p.

Lemma 6 (existence of tests for barbs). Let $a \in \mathcal{B}$. It is witnessed by the test

$$T^a = \{t^a_x : \mathcal{L}(t^a_x) = \{in\, a\} \wedge (t^a_x \xrightarrow{in\, a} t' \implies \mathcal{L}(t') = \{out\, x\})\}$$

For any p such that $\mathcal{N}(p)$ is finite[§] and $x \notin \mathcal{N}(p)$, t^a_x is a stable concrete test for a on p.

Proof. Let p be a system such that $x \notin \mathcal{N}(p)$. Let us assume that $p|t^a_x \xrightarrow{\tau} p'$ with $p' \Box{\downarrow_x}$: since $x \notin \mathcal{N}(p)$ and $t^a_x \not\downarrow_x$, according to the rules in Figure 11 $p|t^a_x p_1 \xrightarrow{out\, a} \xrightarrow{in\, a} p'$ with t^a_x contributing to the derivation. Since $\mathcal{L}(t^a_x) = \{in\, a\}$, again according to the rules in Figure 11 it must be $p \xrightarrow{out\, a} p''$ and $t^a_x \xrightarrow{in\, a} t'$, hence we conclude $p\downarrow_a$.

Vice versa, let $p\downarrow_a$. Then $p \xrightarrow{out\, a} p'$ and $p|t^a_x \xrightarrow{out\, a} p'|t^a_x \xrightarrow{in\, a} p'|t'$, with $t' \xrightarrow{out\, x}$. Using the rules in Figure 11 we deduce that $p|t^a_x \xrightarrow{\tau} p'|t'$, and $p'|t' \xrightarrow{out\, x}$. Since $x \notin \mathcal{N}(p)$, we deduce that $p'' \xrightarrow{out\, x}$ for any p'' such that $p'|t' \xrightarrow{\tau}{}^{*} p''$. Hence, $p'|t' \Box{\downarrow_x}$. □

Concurrent reductions can now be defined as in Figure 2. With this definition it is not difficult to see that assumptions (P) and (C) hold, and that (CB) is an immediate consequence of (B). In fact, it can be easily proved that a test witnessing a concurrent barb $A = a_1 \otimes \ldots \otimes a_n$ is $T^A = \{t^A_X : X = x_1 \otimes \ldots \otimes x_n\}$ where

$$t^A_X = t^{a_1}_{x_1}| \ldots |t^{a_n}_{x_n} \text{ with } t^{a_i}_{x_i} \in t^{a_i} \text{ for all } i.$$

With this set up we can finally prove that the Axiom of Asynchrony (AA) holds for any out-buffered system p with feedback. Hence the result on unobservability of concurrency as expressed by Theorem 1 applies to systems in this class.

Lemma 7 (validity of (AA)). Let $A \in \mathcal{CB}$ be a concurrent barb, p a system satisfying the axioms in Figure 10, and t^A_X a stable concrete test for A on p such that $X = x_1 \otimes \ldots \otimes x_n$ for $x_1, \ldots, x_n \in \mathcal{B}$. If $p|t^A_X \to^* p_1\downarrow_{x_1} \to^* \ldots \to^* p_n\downarrow_{x_n}$ then $p\Downarrow^c_A$.

Proof. Let $A = a_1 \otimes \ldots \otimes a_n$ for $a_1, \ldots, a_n \in \mathcal{B}$. By hypothesis $p|t^A_X \to^* p_1\downarrow_{x_1}$, thus the action $in\ a_1$ has been consumed by a τ transition, hence $p|t^A_X \to^* q_1 \xrightarrow{out\, a_1} q_2 \xrightarrow{in\ a_1} p_1$. Since t^A_X offer no action labelled $out\, a_1$, then we have that $p \to^* p''_1 \xrightarrow{out\, a_1} p'_1$ and $q_2 = p'_1|t^A_X$. Now, since $p_1\downarrow_{x_1}$, then $t^A_X \xrightarrow{in\ a_1} t_1$, and so $p_1 = p'_1|t_1$.

Now, by applying the same reasoning starting from p_1, we have that $p'_1 \to^* p''_2 \xrightarrow{out\, a_2} p'_2$, $t_1 \xrightarrow{in\ a_2} t_2$ and so $p_2 = p'_2|t_2$. If we continue to apply this reasoning to all agents $p_3, \ldots, p_n$, we obtain that $p \to^* p''_1 \xrightarrow{out\, a_1} p'_1 \to^* p''_2 \xrightarrow{out\ a_2} p'_2 \ldots \to^* p''_n \xrightarrow{out\, a_n} p'_n$. Now, since p satisfies the axiom FB1, then there exists p' such that

$$p \to^* p' \xrightarrow{out\, a_1} \ldots \xrightarrow{out\, a_n}.$$

This implies $p'\downarrow^c_A$ and thus $p\Downarrow^c_A$. □

[§] This requirement is far from restrictive. For instance, it holds in the π-calculus since for all processes p, q such that $p \to^* q$ we have $\mathrm{fn}(q) \subseteq \mathrm{fn}(p)$.

$$p \quad ::= \quad m,\ p_1|p_2,\ \mathbf{0},\ \bar{a} \qquad\qquad m \quad ::= \quad \tau.p,\ a.p \lhd b.q,\ m_1 + m_2$$

$$p|q \equiv q|p \qquad p|(q|r) \equiv (p|q)|r \qquad p|\mathbf{0} \equiv p$$
$$m+n \equiv n+m \quad m+(n+o) \equiv (m+n)+o$$

$$(\text{ASYN})\ a.p \lhd b.q + m|\bar{a} \to p \qquad\qquad (\text{PASYN})\ a.p \lhd b.q + m|\bar{b} \to_{\bar{a}} q$$

$$(\text{TAU})\ \tau.p + m \to p$$

$$(\text{PAR})\ \frac{p \to q}{p|r \to q|r} \qquad\qquad (\text{PPAR})\ \frac{p \to_{\bar{a}} q \quad r \not\downarrow_{\bar{a}}}{p|r \to_{\bar{a}} q|r}$$

Fig. 12. The syntax and reduction semantics of PACCS.

5. Asynchronous CCS with priorities

We have already seen that in those formalisms not satisfying the Axiom of Asynchrony (AA) (like, e.g., SCCS), concurrent saturated bisimilarity ($\sim^c$) is strictly finer than the interleaving one ($\sim$). One might wonder wheter the other axioms of our theory are needed in order to guarantee $\sim^c = \sim$. Axioms (P) and (C) are quite natural, they hold in most of the languages we are aware of, but they may not for languages where a transition of a system can be inhibited by a system running in parallel. This is e.g. the case for CCS with priorities (Phillips 2008), Petri nets with inhibitor arcs (Agerwala and Flynn 1973) and graph rewriting with negative application conditions (Habel, Heckel and Taentzer 1996).

In this section we consider PACCS, a toy calculus that extends ACCS with priorities. It is interesting because it shows that whenever the axioms (P) and (C) do not hold unobservability of concurrency may fail, even if the calculus has an asynchronous flavour.

The syntax of PACCS is presented in Figure 12. The only difference with respect to ACCS consists in the *priority input prefixes*: $a.p \lhd b.q$ can execute either an input on a (and then behave like p) or an input on b (and then behave like q), but the latter can be performed only when the former is not possible (namely, if there are no pending messages on a). Note that the input prefix $a.p$ of ACCS can be implemented in PACCS as $a.p \lhd a.p$.

Barbs are defined as in ACCS, i.e., $p \downarrow_{\bar{a}}$ when $p \equiv \bar{a}|p_1$ for a process p_1. The reduction semantics is defined by the rules and the axioms in Figure 12, where $p \not\downarrow_{\bar{a}}$ means that p does not satisfy the barb $\bar{a}$. Note that the transitions can be either labelled or unlabelled. It is important to remark here that, as in Figure 6, labels are just syntactical devices that allow for defining the transition relation, but they are not considered when defining $\sim$. The label $\bar{a}$ on the transitions generated by the rule (PASYN) denotes that the transition can be executed only when there are no outputs on a. The unlabelled transitions instead can be always executed. This is implemented by the rules (PPAR) and (PAR): a transition $\to_{\bar{a}}$ can be executed only if the parallel process r does not contain outputs on a, while the transition $\to$ can be executed in parallel with any process r. For instance,

$$a.p \lhd b.q + m|\bar{b} \to q \qquad \text{but} \qquad a.p \lhd b.q + m|\bar{b}|\bar{a} \not\to q|\bar{a}$$

(the only possible transition of the second process is $a.p \lhd b.q + m|\bar{b}|\bar{a} \to p|\bar{b}$). The above example also shows that PACCS does not satisfy the axiom (P).

$$\frac{p \to p'}{p \rightsquigarrow_{\varnothing} p'} \qquad \frac{p \to_{\bar{a}} p'}{p \rightsquigarrow_{\{\bar{a}\}} p'} \qquad \frac{p \rightsquigarrow_A p' \quad q \rightsquigarrow_B q' \quad \forall \bar{b} \in B,\ p \not\Downarrow_{\bar{b}} \quad \forall \bar{a} \in A,\ q \not\Downarrow_{\bar{a}}}{p|q \rightsquigarrow_{A \cup B} p'|q'}$$

Fig. 13. Concurrent semantics of PACCS.

The concurrent transition relation of PACCS is defined by the rules in Figure 13: the label A in a concurrent transition $\rightsquigarrow_A$ means that it can be executed only if all the messages $\bar{a} \in A$ are not present in the environment. As for the transition relation $\to$, the labels of $\rightsquigarrow$ are just syntactical devices and do not play any role in the definition of $\sim^c$.

It is easy to see that $\rightsquigarrow$ does not satisfy the axiom (C). Consider the processes

$$p_1 = a_1.\mathbf{0} \lhd b_1.\overline{a_2}, \qquad p_2 = a_2.\mathbf{0} \lhd b_2.\overline{a_1} \qquad \text{and} \qquad p = p_1|\overline{b_1}|p_2|\overline{b_2}.$$

We have that

$$p \rightsquigarrow \overline{a_2}|\overline{a_1} \qquad \text{but} \qquad p \not\to^* \overline{a_2}|\overline{a_1}.$$

Indeed, if p_1 consumes $\overline{b_1}$

$$p \to \overline{a_2}|p_2|\overline{b_2},$$

then p_2 cannot consume $\overline{b_2}$ because $\overline{a_2}$ has higher priority. Similarly, if p_2 consumes $\overline{b_2}$

$$p \to p_1|\overline{b_1}|\overline{a_1},$$

then p_2 cannot consume $\overline{b_1}$ because $\overline{a_1}$ has has higher priority.

Concurrent barbs and the corresponding witnessing tests are defined as for ACCS: the barb $A = \overline{a_1} \otimes \ldots \otimes \overline{a_n}$ is witnessed by the test $T^A = \{a_1.\overline{x_1}| \ldots |a_n.\overline{x_n} : \forall i.\, x_i \in \mathcal{N}\}$ (where $a.p$ is a shorthand for $a.p \lhd a.p$). With these definitions it is easy to see that the Axiom of Asynchrony (AA) holds. However, since (P) and (C) do not hold, our theorem does not apply and, indeed, $\sim \neq \sim^c$. Consider the process

$$q = (a_1.p_2 \lhd b_1.(\overline{a_2}|p_2)) + (a_2.p_1 \lhd b_2.(\overline{a_1}|p_1))|\overline{b_1}|\overline{b_2}.$$

Like the process p above

$$\text{either} \qquad q \to \overline{a_2}|p_2|\overline{b_2} \qquad \text{or} \qquad q \to \overline{a_1}|p_1|\overline{b_1}$$

but, differently from p, $q \not\rightsquigarrow \overline{a_2}|\overline{a_1}$ and, more generally, $q \not\Downarrow^{c}_{\overline{a_1} \otimes \overline{a_2}}$. Thus $p \not\sim^c q$.

We conclude by proving that instead $p \sim q$. First, we observe that p and q exhibit the same weak barbs and perform the same transitions $\to$. Then, we note that for all processes r, a reduction $p|r \to p'$ either is generated by p (i.e., $p \to p_1$ and $p' = p_1|r$) or is generated by r (i.e., $r \to r_1$ and $p' = p|r_1$) or by an interaction between p and r. In the latter case either r consumes the messages $\overline{b_1}$, $\overline{b_2}$ of p or p consumes the messages $\overline{a_1}$, $\overline{a_2}$ of r. The same arguments can be applied to the process q and thus the only case to check is the behaviour of p and q when they consume messages $\overline{a_1}$, $\overline{a_2}$ of r. Take $r = \overline{a_1}|r_1$ for some process r_1. We have that

$$p|\overline{a_1}|r_1 \to \mathbf{0}|\overline{b_1}|p_2|\overline{b_2}|r_1 \qquad \text{and} \qquad q|\overline{a_1}|r_1 \to p_2|\overline{b_1}|\overline{b_2}|r_1.$$

Since an analogous argument applies to the process $r = \overline{a_2}|r_1$, we conclude that $p \sim q$.

6. Conclusions, Related and Future Works

In this paper, building on the notion of concurrent barbs, we introduced a novel non-interleaving observational congruence for systems, and we proved in a rather general and abstract framework that concurrency cannot be observed through asynchronous interactions, i.e., that concurrent barbs add no observational power.

As case studies, we considered open Petri nets and the asynchronous π-calculus, showing that they fall in our framework. In particular, for Petri nets we recovered the ordinary firing and step semantics (as defined in (Baldan et al. 2005)), which are thus proved to coincide. For the π-calculus, to the best of our knowledge, no concurrent barbed equivalence was previously defined. We considered two notions of concurrent barbed equivalence which differ on the possibility of performing multiple concurrent communications on the same channel and proved that both fall into our theory. As a consequence they both coincide with the "interleaving" equivalence (and hence, as a side effect, they are identical). We also prove that systems abstractly characterised as (output-buffered) asynchronous in (Selinger 1997) fall into our theory. As a consequence our result extends to other interesting concrete formalisms as well, such as the Join calculus (Fournet and Gonthier 1996).

The generalisation of the theory with respect to the conference version makes it potentially applicable to calculi with bounded capacity channels (e.g., in Section 2.3 we explicitly treated the case of asynchronous π-calculus with 1-bounded capacity channels), as well as to languages featuring notions of asynchrony based on buffers which are not just unordered bags, but ordered structures like queues (see e.g. (Bergstra et al. 1984, de Boer et al. 1992, Selinger 1997, Beauxis et al. 2008)). A preliminary investigation on the calculi $\pi_{\mathfrak{Q}}$ and $\pi_{\mathfrak{S}}$ in (Beauxis et al. 2008) (where buffers are, respectively, queues and stacks) suggests that the results on the unobservability of concurrency should easily extend also to "ordered asynchrony". In fact, since these ordered buffers should not allow concurrent operations, the situation appears to be similar to that of languages with bounded-capacity channels, where concurrent barbs are sets of barbs.

The non-interleaving equivalence we introduced intuitively corresponds to *step semantics*. This has been shown for open Petri nets, even if it seems hard to raise the correspondence at an abstract level. Some ideas might come from the observation that once a concurrent reduction relation has been defined, steps could arise from the *theory of reactive systems* (Leifer and Milner 2000) when replacing $\rightarrow$ with $\rightsquigarrow$. Since $p \xrightarrow{a} q$ means that $-|\bar{a}$ is the smallest context $c[-]$ such that $c[p] \rightarrow q$, analogously the step $p \xrightarrow{a \otimes b} q$ would mean that $-|\bar{a}|\bar{b}$ is the smallest $c[-]$ such that $c[p] \rightsquigarrow q$. As a side remark, note that one of the compelling arguments against step semantics (i.e., that it is not preserved by *action refinement* (van Glabbeek and Goltz 1989)) loses its strength in the paradigm of reduction semantics and barbed equivalences, since actions (labels) disappear.

As for *ST-equivalences* (van Glabbeek and Vaandrager 1987), it seems conceivable to develop an ST-operational semantics in an asynchronous setting, making production and consumption of messages (tokens) non-instantaneous (see, e.g., (van Glabbeek, Goltz and Schicke 2009) for a net model where token consumption is non-instantaneous and (Busi, Gorrieri and Zavattaro 2000) for a similar study on Linda-like languages) and we conjecture that unobservability of concurrency would hold true also in this setting.

Close to our spirit are also *equivalences with localities* (Boudol, Castellani, Hennessy and Kiehn 1991) that distinguish (interleaving equivalent) processes by observing the locations where interactions occur. We chose of not adopting this kind of equivalence for two main reasons: (1) localities are usually structured as trees, but this does not make much sense either in a calculus featuring joins (e.g. (Fournet and Gonthier 1996)) or in a graphical formalism such as open Petri nets; (2) equivalence with localities have never been defined for reduction semantics and, more importantly, for asynchronous formalisms.

It seems apparent though that equivalences with localities are not comparable with ours. Consider e.g. located bisimilarity for ordinary CCS (Boudol et al. 1991). Processes $(\nu b)a.b.c|e.\bar{b}$ and $(\nu b)a.b|e.\bar{b}.c$ are not located bisimilar (in the former c always occurs in a sub-locality of a), but they are equated by $\sim^c$ (since the both satisfy the concurrent barb $a \otimes e$). On the other hand, $a|c$ and $(\nu b)((a.\bar{b}|b.c) + (c.\bar{b}|b.a))$ are not equated by $\sim^c$ (the former satisfies $a \otimes c$), but they are located bisimilar.

Nevertheless, we conjecture that our slogan "concurrency can't be observed, asynchronously" still holds for equivalences with localities. Indeed, since in the asynchronous case inputs are not observable, also their locations should not be observable. Therefore, only the locations of outputs could be observed, but these are all independent (since outputs have no continuations). A formal study of equivalences with localities for asynchronous systems is left as future work.

Our proposal is far from other non-interleaving semantics, such as those in (Darondeau and Degano 1989, Degano, Nicola and Montanari 1988, van Glabbeek and Goltz 1989): these consider *causal properties* of the systems, either by direct inspection of the state structure or by suitably enriching the labels of the transition steps, thus being of a more extensional nature. For these semantics, the fact that the internals of the systems are directly inspected clearly implies that the unobservability of concurrency will not hold.

The different distinguishing power of concurrent equivalences in the synchronous and asynchronous case could also be inspiring for the development of additional separation results between the two paradigms, along the style of (Palamidessi 2003). In more general terms, integrating our framework with the one proposed in (Gorla 2008) seems to represent a promising direction for future investigations.

So far, few papers (such as e.g. (Boreale and Sangiorgi 1998, Crafa, Varacca and Yoshida 2007, Bruni, Melgratti and Montanari 2006)) tackled the study of the concurrency features of asynchronous systems. And to the best of our knowledge our result, albeit quite intuitive, has never been shown on any specific formalism, let alone for a general framework as in our paper. Indeed, besides the catchy slogan, we do believe that our work unearthed some inherent features of asynchronous systems that should hopefully shed some further light on the issue. That is, it should represent a further step towards a satisfactory characterisation of the synchronous/asynchronous dichotomy.

Acknowledgements. The authors would like to thank Catuscia Palamidessi for the helpful discussions and the pointers to the literature and the anonymous reviewers for their precious comments on the preliminary version of this paper.

References

Agerwala, T. and Flynn, M. J. (1973). Comments on capabilities, limitations and "correctness" of Petri nets, *Computer Architecture News* **4**(2): 81–86.

Amadio, R. M., Castellani, I. and Sangiorgi, D. (1996). On bisimulations for the asynchronous π-calculus, *Proc. of CONCUR'96*, Vol. 1119 of *LNCS*, Springer, pp. 147–162.

Baldan, P., Bonchi, F. and Gadducci, F. (2009). Encoding asynchronous interactions using open Petri nets, *Proc. of CONCUR'09*, Vol. 5710 of *LNCS*, Springer, pp. 99–114.

Baldan, P., Bonchi, F., Gadducci, F. and Monreale, G. V. (2010). Concurrency can't be observed, asynchronously, *Proc. of APLAS'10*, Vol. 6461 of *LNCS*, Springer, pp. 424–438.

Baldan, P., Corradini, A., Ehrig, H. and Heckel, R. (2005). Compositional semantics for open Petri nets based on deterministic processes, *Math. Str. in Comp. Sci.* **15**(1): 1–35.

Beauxis, R., Palamidessi, C. and Valencia, F. D. (2008). On the asynchronous nature of the asynchronous π-calculus, *Concurrency, Graphs and Models*, Vol. 5065 of *LNCS*, Springer, pp. 473–492.

Bergstra, J. A., Klop, J. W. and Tucker, J. V. (1984). Process algebra with asynchronous communication mechanisms, *Seminar on Concurrency*, Vol. 197 of *LNCS*, Springer, pp. 76–95.

Bonchi, F., Gadducci, F. and Monreale, G. V. (2010). On barbs and labels in reactive systems, *Proc. of SOS'09*, Vol. 18 of *EPTCS*, pp. 46–61.

Boreale, M. and Sangiorgi, D. (1998). Some congruence properties for π-calculus bisimilarities, *Theor. Comp. Sci.* **198**(1-2): 159–176.

Boudol, G. (1992). Asynchrony and the π-calculus, *Technical Report 1702*, INRIA.

Boudol, G., Castellani, I., Hennessy, M. and Kiehn, A. (1991). Observing localities, *Proc. of MFCS'91*, Vol. 520 of *LNCS*, Springer, pp. 93–102.

Bruni, R., Melgratti, H. C. and Montanari, U. (2006). Event structure semantics for dynamic graph grammars, *Proc. of PNGT'06*, Vol. 2 of *ECEASST*, EASST.

Busi, N. and Gorrieri, R. (1995). A Petri net semantics for π-calculus, *Proc. of CONCUR'95*, Vol. 962 of *LNCS*, Springer, pp. 145–159.

Busi, N., Gorrieri, R. and Zavattaro, G. (2000). Comparing three semantics for linda-like languages, *Theor. Comp. Sci.* **240**(1): 49–90.

Cardelli, L. and Gordon, A. D. (2000). Mobile ambients, *Theor. Comp. Sci.* **240**(1): 177–213.

Crafa, S., Varacca, D. and Yoshida, N. (2007). Compositional event structure semantics for the internal π-calculus, *Proc. of CONCUR'07*, Vol. 4703 of *LNCS*, Springer, pp. 317–332.

Darondeau, P. and Degano, P. (1989). Causal trees, *Proc. of ICALP'89*, Vol. 372 of *LNCS*, Springer, pp. 234–248.

de Boer, F. S., Klop, J. W. and Palamidessi, C. (1992). Asynchronous communication in process algebra, *Proc. of LICS'92*, IEEE Computer Society, pp. 137–147.

Degano, P., Nicola, R. D. and Montanari, U. (1988). Partial orderings descriptions and observations of nondeterministic concurrent processes, *Proc. of REX Workshop*, Vol. 354 of *LNCS*, Springer, pp. 438–466.

Fournet, C. and Gonthier, G. (1996). The reflexive CHAM and the Join-calculus, *Proc. of POPL'96*, ACM Press, pp. 372–385.

Fournet, C. and Gonthier, G. (2005). A hierarchy of equivalences for asynchronous calculi, *J. Log. Algebr. Program.* **63**(1): 131–173.

Gorla, D. (2008). Towards a unified approach to encodability and separation results for process calculi, *Proc. of CONCUR'08*, Vol. 5201 of *LNCS*, Springer, pp. 492–507.

Habel, A., Heckel, R. and Taentzer, G. (1996). Graph grammars with negative application conditions, *Fundam. Inform.* **26**(3/4): 287–313.

Honda, K. and Tokoro, M. (1991). An object calculus for asynchronous communication, *Proc. of ECOOP'91*, Vol. 512 of *LNCS*, Springer, pp. 133–147.

Honda, K. and Yoshida, N. (1995). On reduction-based process semantics, *Theor. Comp. Sci.* **151**(2): 437–486.

Kindler, E. (1997). A compositional partial order semantics for Petri net components., *Proc. of ATPN'97*, Vol. 1248 of *LNCS*, pp. 235–252.

Lanese, I. (2007). Concurrent and located synchronizations in π-calculus, *Proc. of SOFSEM'07*, Vol. 4362 of *LNCS*, Springer, pp. 388–399.

Leifer, J. J. and Milner, R. (2000). Deriving bisimulation congruences for reactive systems, *Proc. of CONCUR'00*, Vol. 1877 of *LNCS*, Springer, pp. 243–258.

Merro, M. and Nardelli, F. Z. (2003). Bisimulation proof methods for mobile ambients, *Proc. of ICALP'03*, Vol. 2719 of *LNCS*, Springer, pp. 584–598.

Milner, R. (1989). *Communication and Concurrency*, Prentice Hall.

Milner, R. (1999). *Communicating and Mobile Systems: the π-Calculus*, Cambridge University Press.

Milner, R. (2003). Bigraphs for Petri nets, *Lectures on Concurrency and Petri Nets*, Vol. 3098 of *LNCS*, Springer, pp. 686–701.

Milner, R. and Sangiorgi, D. (1992). Barbed bisimulation., *Proc. of ICALP'92*, Vol. 623 of *LNCS*, Springer, pp. 685–695.

Montanari, U. and Pistore, M. (1995). Concurrent semantics for the π-calculus, *Proc. of MFPS'95*, Vol. 1 of *ENTCS*, Elsevier.

Palamidessi, C. (2003). Comparing the expressive power of the synchronous and asynchronous π-calculi, *Math. Str. in Comp. Sci.* **13**(5): 685–719.

Phillips, I. (2008). CCS with priority guards, *J. Log. Algebr. Program.* **75**(1): 139–165.

Rathke, J., Sassone, V. and Sobociński, P. (2007). Semantic barbs and biorthogonality, *Proc. of FoSSaCS'07*, Vol. 4423 of *LNCS*, Springer, pp. 302–316.

Sassone, V. and Sobociński, P. (2005). A congruence for Petri nets, *Proc. of PNGT'04*, Vol. 127 of *ENTCS*, Elsevier, pp. 107–120.

Selinger, P. (1997). First-order axioms for asynchrony, *Proc. of CONCUR'97*, Vol. 1243 of *LNCS*, Springer, pp. 376–390.

van Glabbeek, R. J. (1990). The linear time-branching time spectrum, *Proc. of CONCUR'90*, Vol. 458 of *LNCS*, Springer, pp. 278–297.

van Glabbeek, R. J. and Goltz, U. (1989). Equivalence notions for concurrent systems and refinement of actions, *Proc. of MFCS'89*, Vol. 379 of *LNCS*, Springer, pp. 237–248.

van Glabbeek, R. J. and Vaandrager, F. W. (1987). Petri net models for algebraic theories of concurrency, *Proc. of PARLE'87*, Vol. 259 of *LNCS*, Springer, pp. 224–242.

van Glabbeek, R. J., Goltz, U. and Schicke, J.-W. (2009). Symmetric and asymmetric asynchronous interaction, *Proc. of ICE'08*, Vol. 229(3) of *ENTCS*, Elsevier, pp. 77–95.