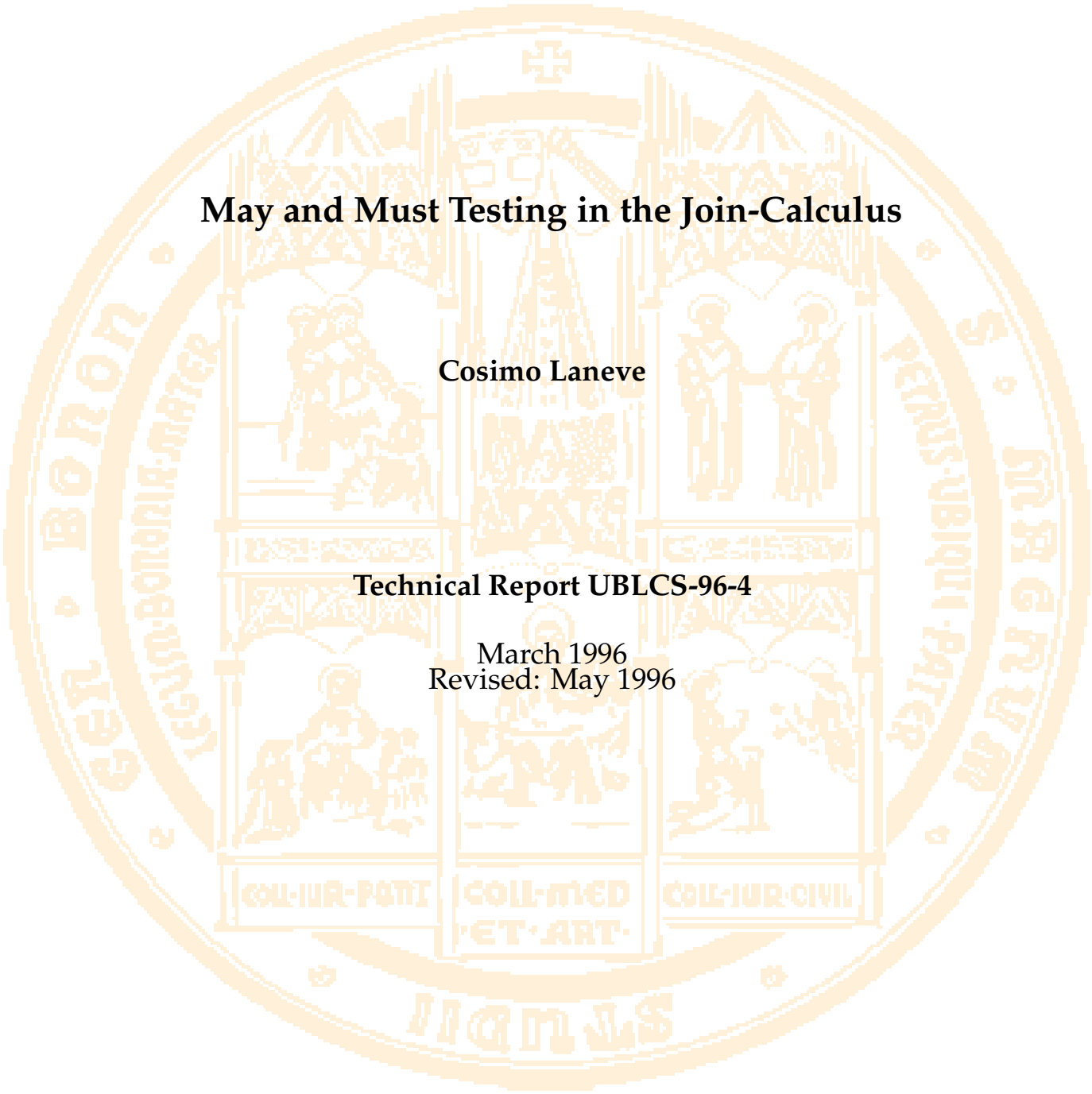

May and Must Testing in the Join-Calculus

Cosimo Laneve

Technical Report UBLCS-96-4

March 1996
Revised: May 1996

Department of Computer Science
University of Bologna
Piazza di Porta S. Donato, 5
40127 Bologna (Italy)

The University of Bologna Department of Computer Science Research Technical Reports are available in gzipped PostScript format via anonymous FTP from the area `ftp.cs.unibo.it:/pub/TR/UBLCS` or via WWW at URL `http://www.cs.unibo.it/`. Plain-text abstracts organized by year are available in the directory `ABSTRACTS`. All local authors can be reached via e-mail at the address *last-name*`@cs.unibo.it`. Questions and comments should be addressed to `tr-admin@cs.unibo.it`.

Recent Titles from the UBLCS Technical Report Series

94-19 *On Group Communication in Large-Scale Distributed Systems*, Ö. Babaoğlu, A. Schiper, July 1994.

94-20 *Dynamic Allocation of Signature Files in Multiple-Disk Systems*, P. Ciaccia, August 1994.

94-21 *Parallel Independent Grid Files Based on a Dynamic Declustering Method Using Multiple Error Correcting Codes*, P. Ciaccia, November 1994.

95-1 *Performance Preorder and Competitive Equivalence*, F. Corradini, R. Gorrieri, M. Roccetti, January 1995 (Revised December 1995).

95-2 *Clepsydra Methodology*, P. Ciaccia, O. Ciancarini, W. Penzo, January 1995.

95-3 *A Unified Framework for the Specification and Run-time Detection of Dynamic Properties in Distributed Computations*, Ö. Babaoğlu, E. Fromentin, M. Raynal, January 1995 (Revised February 1995).

95-4 *Effective Applicative Structures*, A. Asperti, A. Ciabattoni, January 1995.

95-5 *An Open Framework for Cooperative Problem Solving*, M. Gaspari, E. Motta, A. Stutt, February 1995.

95-6 *Considering New Guidelines in Group Interface Design: a Group-Friendly Interface for the CHAOS System*, D. Bottura, C. Maioli, S. Mangiaracina, February 1995.

95-7 *Modelling Interaction in Agent Systems*, A. Dalmonte, M. Gaspari, February 1995.

95-8 *Building Hypermedia for Learning: a Framework Based on the Design of User Interface*, S. Mangiaracina, C. Maioli, February 1995.

95-9 *The Bologna Optimal Higher-Order Machine*, A. Asperti, C. Giovannetti, A. Naletto, March 1995.

95-10 *Synchronization Support and Group-Membership Services for Reliable Distributed Multimedia Applications*, F. Panzieri, M. Roccetti, March 1995 (Revised July 1995).

95-11 *The Inherent Cost of Strong-Partial View-Synchronous Communication*, Ö. Babaoğlu, R. Davoli, L.-A. Giachini, P. Sabattini, April 1995.

95-12 *On the Complexity of Beta-Reduction*, A. Asperti, July 1995.

95-13 *Optimal Multi-Block Read Schedules for Partitioned Signature Files*, P. Ciaccia, August 1995.

95-14 *Integrating Performance and Functional Analysis of Concurrent Systems with EMPA*, M. Bernardo, L. Donatiello, R. Gorrieri, September 1995.

95-15 *On Programming with View Synchrony*, Ö. Babaoğlu, A. Bartoli, G. Dini, September 1995.

95-16 *Generative Communication in Process Algebra*, P. Ciancarini, R. Gorrieri, G. Zavattaro, October 1995.

95-17 *Dynamic Declustering Methods for Parallel Grid Files*, P. Ciaccia, A. Veronesi, November 1995.

95-18 *Failure Detectors, Group Membership and View-Synchronous Communication in Partitionable Asynchronous Systems (Preliminary Version)*, Ö. Babaoğlu, R. Davoli, A. Montresor, November 1995.

96-1 *An Investigation on the Optimal Implementation of Processes*, C. Laneve, January 1996.

96-2 *Expansivity, Permutivity, and Chaos for Cellular Automata*, F. Fagnani, L. Margara, January 1996.

96-3 *Enriched View Synchrony: A Paradigm for Programming Dependable Applications in Partitionable Asynchronous Distributed Systems*, Ö. Babaoğlu, A. Bartoli, G. Dini, February 1996.

96-4 *May and Must Testing in the Join-Calculus*, C. Laneve, March 1996.

96-5 *The Shape of Shade: a Coordination System*, S. Castellani, P. Ciancarini, D. Rossi, March 1996.

96-6 *Engineering Formal Requirements: an Analysis and Testing Method for Z Documents*, P. Ciancarini, S. Cimato, C. Mascolo, March 1996.

96-7 *Using Bayesian Belief Networks for the Automated Assessment of Students' Knowledge of Geometry Problem Solving Procedures*, M. Roccetti, P. Salomoni, March 1996.

96-8 *Virtual Interactions: An Investigation of the Dynamics of Sharing Graphs*, A. Asperti, C. Laneve, April 1996.

96-9 *Towards an Algebra of Actors*, M. Gaspari, April 1996.

96-10 *Mobile Petri Nets*, A. Asperti, N. Busi, May 1996.

May and Must Testing in the Join-Calculus

Cosimo Laneve [1]

Technical Report UBLCS-96-4

March 1996
Revised: May 1996

Abstract

The may and must-semantics are studied for the join-calculus. We provide a complete characterization of may-testing through a restricted set of contexts. The same characterization, up-to the basic observations, is also proved to be complete with respect to the must-testing semantics.

1. Department of Computer Science, University of Bologna, Piazza Porta S. Donato 5, 40127 Bologna, Italy.

1 Introduction

The join-calculus has been recently introduced for describing communicating processes [6]. As in the asynchronous π-calculus [2, 9] messages have no continuation and the scoping discipline is static. However, the join-calculus departs from the asynchronous π-calculus for two features: the presence of elaborate synchronization schemas and the possibility to create computational rules dynamically. Dynamic creation may happen when a pattern of messages matches with a synchronization schema. This triggers a new process which may add new computational rules. These new rules do not concern already defined names (static binding).

The main motivation pervading the definition of join-calculus is pragmatical: the distributed implementation of the π-calculus [11]. We refer to [6] for a detailed discussion of this issue. Apart from that, Fournet and Gonthier have also pursued on a semantic investigation with the worthy purpose of carrying over the join-calculus the metatheory of the π-calculus. To this aim they have considered weak barbed congruence and, up-to this equivalence, they have proved that join-calculus and π-calculus have the same expressive power.

However the semantic theory of the join-calculus deserves a more detailed analysis because the results of [6] cannot reveal properties which are peculiar of the calculus. In this paper we undertake this analysis by considering the testing approach and exploring the corresponding theory in the join-calculus.

Testing semantics [4, 7] is the adaptation of Morris' semantics in the λ-calculus to generic languages. The paradigm is the following:

1. define a notion of "observable" for terms of the language. Usually it is the outcome of a computation or a test whether a computation terminates or not;
2. take the coarsest congruence which is closed under the above notion of observable.

In deterministic calculi (such as call-by-name, lazy or call-by-value λ-calculus) every term has always a unique observable outcome. This is false in nondeterministic calculi. Take for instance the convergence of a computation as observable. Then, given a term P, we have the following outcomes: (a) every computation starting at P converges, (b) there are convergent and divergent computations starting at P, (c) every computation pointed at P diverges. Testing equivalence is the congruence yielded by these three observations. It turns out that this equivalence is indeed the intersection of two coarser semantics: may and must-testing semantics. May-semantics is the congruence which is unsensitive to observations (a) and (b). Must semantics is unsensitive to outcomes (b) and (c).

Checking testing equivalence between processes is usually difficult because one has to verify the coincidence of observables over *all possible* contexts. The expedient to overcome this problem consists in determining a set of canonical tests which is suitable to establish testing equality. The characterization of these tests in the join-calculus is the main contribution of this paper. In particular we prove that the same tests suit both for may and must-testing (and therefore for testing equivalence). This seems odd, if compared to the corresponding property owned by Milner's CCS, for instance [7]. Therefore some insights on our results may be useful.

Let P be a join-calculus process. Each test $C[\,]$ allows to check the *immediate emissions* only, that is the messages emitted by $C[P]$ after a sequence of internal moves. In order to check for successive emissions we have to plug $C[P]$ into other contexts which define free variables of $C[P]$, and so on. It follows that may-tests are sequences of nested definitions. The crucial point is to realize that may-tests can be rid of computational rules replacing patterns of messages with complex processes. Better, the relevant tests for may-testing have the shape:

$$\mathsf{def}\ J_1 \rhd K_1\ \mathsf{in}\ \cdots\ \mathsf{def}\ J_n \rhd K_n\ \mathsf{in}\ [\,] \mid K\ .$$

where J_i, K_i, K are multisets of emissions $x\langle u_1, \cdots, u_h\rangle$. After that, the striking observation is that the same tests allows to check for the alternative presence of a set of emissions. To this purpose it suffices to define separately each variable which is emitted. These latter tests allow in some sense to encode the must-tests of [7], thus justifying our contribution. Finally, thanks to the elaborate synchronization schema of the join-calculus, it is also possible to test for the simultaneous presence of a *multiset* of emissions. In other words, the testing approach in the

join-calculus gives a *multiset testing semantics*.

The structure of the paper is as follows. In Section 2 we present the join-calculus, namely its syntax, the structural equivalence and the reduction relation. Our contribution starts at Section 3 with the definition of may-testing semantics and few simple properties about it. Section 4 is devoted to the proof of the Context Lemma for the may-semantics. Evidence is given to the importance of this result for checking semantic relations. The must-semantics is defined in Section 5. In the same section it is also proved the Context Lemma for must-testing. The Conclusion contains few remarks on related works and some issues for future research.

2 The join-calculus

Let $\mathcal{V}$ be a countable set of variables $\{x, y, z, u, v, \cdots\}$, each of them comes with an arity. Let $\widetilde{v}$ be a tuple of variables in $\mathcal{V}$. We shall write $x\langle\widetilde{v}\rangle$ provided that the length of the tuple $\widetilde{v}$ is equal to the arity of x. This amounts to use a recursive sort discipline and to consider only well-sorted processes. A *process* P, a *definition* D and a *join pattern* J are inductively defined by the following syntax:

$$\begin{array}{rcl} P & ::= & x\langle\widetilde{v}\rangle \mid \mathsf{def}\ D\ \mathsf{in}\ P \mid P \mid P \\ D & ::= & J \rhd P \mid D\ \mathsf{and}\ D \\ J & ::= & x\langle\widetilde{v}\rangle \mid J \mid J \end{array}$$

In an elementary definition $J \rhd P$, where $J = x_1\langle\widetilde{v_1}\rangle \mid \cdots \mid x_k\langle\widetilde{v_k}\rangle$, we shall always assume that variables in $\widetilde{v_i}$ are pairwise distinct; similarly the sets $\{\widetilde{v_i}\}$, $\{\widetilde{v_j}\}$ and $\{x_1, \cdots, x_k\}$ are mutually disjoint, with $i \neq j$. The set $\{\widetilde{v_1}, \cdots, \widetilde{v_k}\}$ is called the set of *received variables* of J (notation $\mathsf{rv}(J)$); $\{x_1, \cdots, x_k\}$ is the set of *defined variables* of J (notation $\mathsf{dv}(J)$), which lifts to definitions as follows:

$$\mathsf{dv}(J_1 \rhd P_1\ \mathsf{and}\ \cdots\ \mathsf{and}\ J_k \rhd P_k) = \bigcup_{1 \leq i \leq k} \mathsf{dv}(J_i)$$

Finally, the set of *free variables* of a definition D (resp. process P), written $\mathsf{fv}(D)$ (resp. $\mathsf{fv}(P)$), is defined inductively as follows:

$$\begin{array}{rcl} \mathsf{fv}(J \rhd P) & = & \mathsf{dv}(J) \cup (\mathsf{fv}(P) \setminus \mathsf{rv}(J)) \\ \mathsf{fv}(D\ \mathsf{and}\ D') & = & \mathsf{fv}(D) \cup \mathsf{fv}(D') \end{array}$$

$$\begin{array}{rcl} \mathsf{fv}(x\langle\widetilde{v}\rangle) & = & \{x, \widetilde{v}\} \\ \mathsf{fv}(\mathsf{def}\ D\ \mathsf{in}\ P) & = & (\mathsf{fv}(P) \cup \mathsf{fv}(D)) \setminus \mathsf{dv}(D) \\ \mathsf{fv}(P \mid Q) & = & \mathsf{fv}(P) \cup \mathsf{fv}(Q) \end{array}$$

The process $x\langle\widetilde{v}\rangle$ is called *atom*. The following ones are syntactical definitions:

Definition 2.1 *A process is* simple *if it is the parallel composition of atoms. A definition is* simple *when it has the shape $J \rhd K$, where K is a simple process.*

Simple definitions will play an important role in the theory we are going to develop.

2.1 The structural equivalence

The *structural equivalence* $\equiv$ is the least congruence over processes, definitions and patterns satisfying the following axioms:

- let P be α-equivalent to Q, then $P = Q$;
- $P \mid Q = Q \mid P, (P \mid Q) \mid R = P \mid (Q \mid R)$;
- if $\mathsf{dv}(D) \cap \mathsf{fv}(P) = \emptyset$ then $P \mid \mathsf{def}\ D\ \mathsf{in}\ Q = \mathsf{def}\ D\ \mathsf{in}\ (P \mid Q)$;
- if $\mathsf{fv}(D) \cap \mathsf{fv}(D') = \emptyset$ then $\mathsf{def}\ D\ \mathsf{in}\ \mathsf{def}\ D'\ \mathsf{in}\ P = \mathsf{def}\ D'\ \mathsf{in}\ \mathsf{def}\ D\ \mathsf{in}\ P$;
- $D\ \mathsf{and}\ D' = D'\ \mathsf{and}\ D$, $D\ \mathsf{and}\ (D'\ \mathsf{and}\ D'') = (D\ \mathsf{and}\ D')\ \mathsf{and}\ D''$, $D\ \mathsf{and}\ D = D$;
- $J \mid J' = J' \mid J$.

It is immediate by induction on the definition of $\equiv$ (including the rules of reflexivity, symmetry, transitivity and congruence) that $P \equiv Q$ entails $\mathsf{fv}(P) = \mathsf{fv}(Q)$ and that $P \equiv \mathsf{def}\ D_1\ \mathsf{in}\ \cdots\ \mathsf{def}\ D_n\ \mathsf{in}\ J$, for some $D_1, \cdots D_n, J$.

2.2 The reduction relation

The reduction relation is the set of τ-transitions of a labelled transition system $\xrightarrow{\delta}$, where δ ranges over $\{J \rhd P\} \cup \{\tau\}$. The transition relation is the smallest relation such that for every $J \rhd P$,

$$J\rho \xrightarrow{J \rhd P} P\rho \qquad \rho \text{ is a renaming of } \mathsf{rv}(J)$$

and for every transition $P \xrightarrow{\delta} P'$,

$$\begin{array}{rll} P \mid Q & \xrightarrow{\delta} & P' \mid Q \\ \mathsf{def}\ D\ \mathsf{in}\ P & \xrightarrow{\delta} & \mathsf{def}\ D\ \mathsf{in}\ P' \qquad \text{if } \mathsf{fv}(D) \cap \mathsf{dv}(\delta) = \emptyset \\ \mathsf{def}\ D\ \mathsf{in}\ P & \xrightarrow{\tau} & \mathsf{def}\ D\ \mathsf{in}\ P' \qquad \text{if } D = \delta \text{ or } D = \delta\ \mathsf{and}\ D' \\ Q & \xrightarrow{\delta} & Q' \qquad \text{if } P \equiv Q \text{ and } Q' \equiv P' \end{array}$$

We give some examples of processes together with a description of their meaning.

(the forwarder) The process

$$\mathsf{def}\ x\langle u\rangle \rhd y\langle u\rangle\ \mathsf{in}\ P$$

behaves as P except that every emission over x is captured by the definition $x\langle u\rangle \rhd y\langle u\rangle$ and transformed into an emission on y. More precisely the above process is equivalent, up-to an internal move, to the process $P[{}^{y}/_{x}]$, where $-[{}^{-}/_{\_}]$ is the substitution operator.

(the chaos) The process

$$\Omega \stackrel{\text{def}}{=} \mathsf{def}\ x\langle\rangle \rhd x\langle\rangle\ \mathsf{in}\ x\langle\rangle$$

is the most undefined process (in our semantics). It performs an infinite sequence of internal moves and it is reminiscent of the term $(\lambda x.\, xx)(\lambda x.\, xx)$ in the λ-calculus.

(the internal and external nondeterminism) The process

$$\mathsf{def}\ x\langle\rangle \rhd P\ \mathsf{and}\ x\langle\rangle \rhd Q\ \mathsf{in}\ x\langle\rangle$$

encodes internal nondeterminism. It may become P or Q by means of an internal move. This choice is by no means affected by the environment. The process

$$\mathsf{def}\ u\langle\rangle \mid c\langle\rangle \rhd P\ \mathsf{and}\ v\langle\rangle \mid c\langle\rangle \rhd Q\ \mathsf{in}\ x\langle u, v\rangle \mid c\langle\rangle$$

encodes external nondeterminism. It may become P or Q according to the environment emits an atom $u\langle\rangle$ or $v\langle\rangle$. In virtue of the static scoping discipline, this is possible provided that a rule involving the atom $x\langle u, v\rangle$ is fired. Indeed, it is the participation of $x\langle u, v\rangle$ into a rule which allows to extrude the variables u and v out of the enclosing definition $u\langle\rangle \mid c\langle\rangle \rhd P\ \mathsf{and}\ v\langle\rangle \mid c\langle\rangle \rhd Q$. The atom $c\langle\rangle$ guarantees that at most one of the processes P and Q is triggered.

The transition system of the join-calculus, restricted to τ-moves, is finite branching.

Proposition 2.2 (Finite branching property) *For every P, $\{[P'] \mid P \xrightarrow{\tau} P'$ and $P' \equiv [P']\}$ is finite.*

Proof: By definition every process may emit a finite number of atoms. Moreover emissions may be enclosed by a finite number of definitions. Now observe that a finite number of emissions may trigger definitions in a finite number of different ways. Henceforth the finiteness of successors of processes. ∎

3 The may-testing semantics

The only way for a process to communicate with the environment is eventually to emit a message on one of its free variables. The emission of a message is formalized by the following predicate $\Downarrow_m$. We first define the *immediate emission* on a particular variable x, in notation $P \downarrow x$, as the least predicate satisfying:

1. $x\langle\widetilde{v}\rangle \downarrow x$;
2. $P \downarrow x$ implies $(P \mid Q) \downarrow x$;
3. $P \downarrow x$ and $x \notin \mathsf{dv}(D)$ imply $(\mathsf{def}\ D\ \mathsf{in}\ P) \downarrow x$.

Remark 3.1 *1. $P \downarrow x$ implies $x \in \mathsf{fv}(P)$;*

2. an immediate emission cannot be destroyed by reductions, namely if $P \downarrow x$ and $P \xrightarrow{\tau} Q$ then $Q \downarrow x$.

Definition 3.2 *The* may-emission *relation is defined by:*

$$P \Downarrow_m x \stackrel{\text{def}}{=} \exists Q.\ P \xrightarrow{\tau}{}^{*} Q \text{ and } Q \downarrow x.$$

We shall write $P \not\Downarrow_m x$ when $P \Downarrow_m x$ is false. When $P \Downarrow_m x$ we say P may-emits *on x, and we shall often omit the prefix "may".*

Remark 3.3 *It is important to note that in the join-calculus we are forced to observe outputs only. This because inputs are hidden in the definitions. In other calculi, the choice of observations is questionable [1, 2, 9].*

Proposition 3.4 *$P \Downarrow_m x$ and $P \equiv Q$ imply $Q \Downarrow_m x$.*

This is an immediate consequence of definitions of may-emission and structural equivalence.

The semantics we are going to define is the *may-testing preorder*. Firstly let us introduce the join-calculus contexts as the set of processes with exactly one hole $[\,]$ in the place where a process may appear. Contexts are ranged over by $C[\,]$.

Definition 3.5 *The* may-testing preorder *is the relation $\sqsubseteq_m$ over processes defined by*

$$P \sqsubseteq_m Q \stackrel{\text{def}}{=} \forall C[\,].\ \forall x.\ C[P] \Downarrow_m x \Rightarrow C[Q] \Downarrow_m x$$

Two processes P and Q are may-testing equivalent, in notation $P \simeq_m Q$, if $P \sqsubseteq_m Q$ and $Q \sqsubseteq_m P$.

Both the may-testing preorder and the may-testing equivalence are pre-congruences, by definition. By Proposition 3.4 we have the following:

Corollary 3.6 *$P \equiv Q$ implies $P \sqsubseteq_m Q$.*

A few examples should clarify the discriminating power of the may-testing preorder. For every process P:

- $\Omega \sqsubseteq_m P$ (recall that $\Omega \stackrel{\text{def}}{=} \mathsf{def}\ x\langle\rangle \rhd x\langle\rangle\ \mathsf{in}\ x\langle\rangle$). This because, for every context $C[\,]$, if $C[\Omega] \Downarrow_m y$ then there must be a derivation σ of $C[\Omega]$ which never uses the definition $x\langle\rangle \rhd x\langle\rangle$. Observe that σ is also a derivation of $C[P]$;
- $P \simeq_m \mathsf{def}\ x\langle\rangle \rhd x\langle\rangle\ \mathsf{and}\ x\langle\rangle \rhd P\ \mathsf{in}\ x\langle\rangle$ (we assume $x \notin \mathsf{fv}(P)$). This because, at each moment, the process $\mathsf{def}\ x\langle\rangle \rhd x\langle\rangle\ \mathsf{and}\ x\langle\rangle \rhd P\ \mathsf{in}\ x\langle\rangle$ may behave as P or perform an internal move and may-testing is unsensitive to internal moves;

- $P[^z/_x] \simeq_m \mathsf{def}\ x\langle\widetilde{u}\rangle \rhd z\langle\widetilde{u}\rangle\ \mathsf{in}\ P$. Because every emission on x by P is replaced by an emission on z in $P[^z/_x]$. This replacement is explicitly defined in $\mathsf{def}\ x\langle\widetilde{u}\rangle \rhd z\langle\widetilde{u}\rangle\ \mathsf{in}\ P$ at the cost of an internal move.

In standard process algebras (CCS, CSP, π-calculus, etc.) may-testing coincides with trace semantics [7]. In join-calculus this coincidence fails because the synchronization schemas allow to test for the presence of multisets of messages (see Remark 3.7 below). So we get some sort of multiset trace semantics. However this is still not so evident because definitions combine in a single place the operation of restriction, reception and replication of π-calculus (the reader may get some evidence by looking at the encoding of the core join-calculus into the π-calculus in [6]). For instance

$$y\langle\rangle \sqsubseteq_m \mathsf{def}\ x\langle\rangle \rhd y\langle\rangle\ \mathsf{and}\ x\langle\rangle \rhd z\langle\rangle\ \mathsf{in}\ x\langle\rangle$$

but the *vice versa* is false because the process in the right may also emit on z. Now take the process

$$P = \mathsf{def}\ x\langle u\rangle \rhd y\langle u\rangle\ \mathsf{in}\ z\langle x\rangle\ .$$

The environment recognizes immediately that P emits on z. However it is also possible to test that P emits on y *after* the emission on z. To this aim take the context

$$C[\,] = \mathsf{def}\ z\langle v\rangle \rhd v\langle w\rangle\ \mathsf{in}\ [\,]$$

and observe that $C[\,]$ distinguishes between P and $z\langle x\rangle$, for instance.

It is also possible to count the number of times a definition is used (this is an expedient we will exploit in the proofs). For example in order to count how many times the definition $J \rhd Q$ is used in $\mathsf{def}\ J \rhd Q\ \mathsf{in}\ Q'$, consider the process $\mathsf{def}\ J \rhd Q \mid d\langle\rangle\ \mathsf{in}\ Q'$ where d is a fresh variable. When the process reaches a stable state (no transition is possible) the foregoing number is given by the amount of emissions on d.

Remark 3.7 *Alternative definitions of may-testing, still equivalent to the foregoing one, may be provided by taking one of the following basic observations:*

1. P may-converges *if and only if there exists x such that $P \Downarrow_m x$;*
2. P may-converges to $\{\!|x_1, \cdots, x_n|\!\}$ *($\{\!|\ |\!\}$ is the multiset constructor) if there exist P' such that $P \xrightarrow{\tau}{}^{*} P'$ and P' emits on the multiset $\{\!|x_1, \cdots, x_n|\!\}$. This last predicate can be formalized generalizing the definition of immediate emission.*

4 The context lemma for the may-testing

Due to the universal quantification over contexts, it is difficult to prove or disprove $P \sqsubseteq_m Q$. We now prove a property, called *the context lemma*, which establishes that, in order to test a process, it is enough to plug it into a restricted set of contexts. These contexts, ranged over by $T[\,]$ and called *simple contexts*, are those given by the grammar:

$$T[\,] \ ::= \ [\,] \ \mid \ [\,] \mid K \ \mid \ \mathsf{def}\ J \rhd K\ \mathsf{in}\ T[\,]$$

where K is simple process. simple processNamely, the generic shape of a simple context is:

$$\mathsf{def}\ J_1 \rhd K_1\ \mathsf{in}\ \cdots\ \mathsf{def}\ J_n \rhd K_n\ \mathsf{in}\ [\,] \mid K\ .$$

Observe that the definitions in the above contexts are always simple. Simple contexts induce the following preorder:

$$P \sqsubseteq_s Q \ \stackrel{\text{def}}{=}\ \forall T[\,].\, \forall x.\, T[P] \Downarrow_m x \Rightarrow T[Q] \Downarrow_m x$$

(the index s stays for "simple"). Clearly $\sqsubseteq_m \subseteq \sqsubseteq_s$ but the *vice versa* is by no means obvious. In order to prove the coincidence of $\sqsubseteq_m$ and $\sqsubseteq_s$ it is enough to prove that $\sqsubseteq_s$ is a pre-congruence. This is the purpose of the following propositions. The proof technique consists in encoding the behaviours induced by generic contexts into behaviours of simple ones.

It turns out that a satisfactory formalization of the following proof must use a labelled variant of the join-calculus. Let Ψ be a countable set of labels and $\alpha \in \Psi$. The labelled join-calculus is similar to the one defined in Section 2 with the following changes:

- definitions are generated by the following grammar:

$$D \ ::= \ J \rhd P \ \mid \ J \rhd^{\alpha} P \ \mid \ D \text{ and } D$$

- the reduction relation is enriched by adding the following rule:
  $P \stackrel{J \rhd Q}{\longrightarrow} P'$ implies

$$\text{def } J \rhd^{\alpha} Q \text{ in } P \stackrel{\alpha}{\longrightarrow} \text{def } J \rhd^{\alpha} Q \text{ in } P'$$
$$\text{def } J \rhd^{\alpha} Q \text{ and } D \text{ in } P \stackrel{\alpha}{\longrightarrow} \text{def } J \rhd^{\alpha} Q \text{ and } D \text{ in } P'$$

It is possible to generalize may-emission in order to take into account transitions labelled in $\Psi \cup \{\tau\}$. Namely

$$P \Downarrow_m^+ x \ \stackrel{\text{def}}{=} \ \exists Q.\ P \longrightarrow^* Q \text{ and } Q \downarrow x.$$

where $\longrightarrow^*$ denotes a sequence of transitions labelled in $\Psi \cup \{\tau\}$.

Proposition 4.1 *1. Let P be a process, syntactically equal to Q, up-to labels of definitions. Then $P \Downarrow_m^+ x$ if and only if $Q \Downarrow_m^+ x$.*

2. Let P be an unlabelled process (definitions miss of labels); let Q be a process syntactically equal to P, up-to labels of definitions. Then $P \Downarrow_m x$ if and only if $Q \Downarrow_m^+ x$.

In virtue of the foregoing proposition we can safely reason on labelled processes and carrying the results over the unlabelled ones. In particular, we shall always consider *well-labelled processes*:

Definition 4.2 *A process is* well-labelled *when every definition D is generated by the following syntax:*

$$D \ ::= \ J \rhd K \ \mid \ J \rhd^{\alpha} P \ \mid \ D_\ell \text{ and } D_\ell \ \ (K \text{ is simple; } P \text{ is not simple})$$

$$D_\ell \ ::= \ J \rhd^{\alpha} Q \ \mid \ D_\ell \text{ and } D_\ell$$

and labels are pairwise different.

Informally, a process is *well-labelled* when every not simple definition is labelled by a unique label.

Proposition 4.3 *$P \sqsubseteq_s Q$ implies $P \mid R \sqsubseteq_s Q \mid R$, for every process R.*

Proof: We assume that R is well labelled. Let $T[\ ]$ be a simple context such that $T[P \mid R] \Downarrow_m^+ x$ and $T[Q \mid R] \not\Downarrow_m^+ x$. Let σ be the derivation $T[P \mid R] \longrightarrow^* P' \downarrow x$, and let n be the number of reductions along σ which are marked by labels in Ψ. By induction on n we prove that, for every R and for every context $T[\ ]$ such that $T[P \mid R] \longrightarrow^* P' \downarrow x$, with n' transitions marked in Ψ, and $T[Q \mid R] \not\Downarrow_m^+ x$ and $n' \leq n$, there exists a simple context $T'[\ ]$ such that $T'[P] \Downarrow_m x$ and $T'[Q] \not\Downarrow_m x$.

Without loss of generality we may assume that $R = \text{def } D_1 \text{ in } \cdots \text{def } D_r \text{ in } J$ and bound variables in R are different from variables in P and Q.

($n = 0$) Then only simple definitions D_i are used along σ. Let D_i' be D_i if D_i is simple and D_i' be $x_1\langle\widetilde{v_1}\rangle \mid \cdots \mid x_h\langle\widetilde{v_h}\rangle \mid a\langle\rangle \rhd a\langle\rangle$, where $\mathsf{dv}(D_i) = \{x_1, \cdots, x_h\}$ and where a is a fresh variable (with respect to those used in P, Q and R), otherwise. The simple context $T'[\ ]$ is defined as follows:

$$T'[\ ] = T[\text{def } D_1' \text{ in } \cdots \text{def } D_r' \text{ in } ([\ ] \mid J)]$$

($n > 0$) Let $\gamma_1 \cdots \gamma_n$ be the sequence of labels in Ψ along σ and let D_i be the first definition in R such that a label in D_i occurs in $\gamma_1 \cdots \gamma_n$. Let

$$T_1[\ ] = \text{def } D_1' \text{ in } \cdots \text{def } D_{i-1}' \text{ in } [\ ]$$

where D_j' is defined as in the basic case (with respect to definitions $D_1, \cdots, D_{i-1}$). There are two subcases:

1. D_i is $J' \rhd^{\alpha} R'$. Let c be the number of occurrences of α in $\gamma_1 \cdots \gamma_n$ and let $\mathsf{fv}(R') = \{x_1, \cdots, x_t\}$. Consider the following context $T_2[\,]$:

$$\begin{array}{rcl} T_2[\,] & = & \mathsf{def}\ x_1\langle \widetilde{u_1}\rangle \mid d_{x_1}\langle v\rangle \rhd v\langle \widetilde{u_1}\rangle \mid d_{x_1}\langle v\rangle\ \mathsf{in} \\ & & \vdots \\ & & \mathsf{def}\ x_t\langle \widetilde{u_t}\rangle \mid d_{x_t}\langle v\rangle \rhd v\langle \widetilde{u_t}\rangle \mid d_{x_t}\langle v\rangle\ \mathsf{in}\ ([\,] \mid d\langle d_{x_1}, \cdots, d_{x_t}\rangle) \end{array}$$

where d and d_{x_i} are fresh variables. Now consider the context $T_3[\,]$:

$$T_3[\,] = \mathsf{def}\ J' \mid d\langle d_{x_1}, \cdots, d_{x_t}\rangle \rhd d_{x_1}\langle x_1\rangle \mid \cdots \mid d_{x_t}\langle x_t\rangle\ \mathsf{in}\ [\,]$$

and let

$$\begin{array}{rcl} T_4[\,] & = & T_1[\,T_3[\,]\,] \\ R'' & = & (\mathsf{def}\ D_{i+1}\ \mathsf{in} \cdots \mathsf{def}\ D_r\ \mathsf{in}\ J) \mid \underbrace{T_2[R'] \mid \cdots \mid T_2[R']}_{c \text{ times}} \end{array}$$

where the labels in the istances of R' in R'' are pairwise different. It is possible to verify that $T[\,T_4[P \mid R'']\,] \Downarrow_m x$ and $T[\,T_4[Q \mid R'']\,] \not\Downarrow_m x$. Moreover $T[\,T_4[P \mid R'']\,]$ shows up x with a number of reductions labelled in Ψ which is strictly less then n.

2. D_i is $J_1 \rhd^{\alpha_1} R_1$ and $\cdots$ and $J_{i'} \rhd^{\alpha_{i'}} R_{i'}$. Let $\beta_1 \cdots \beta_c$ be the subsequence of $\gamma_1 \cdots \gamma_n$ such that $\gamma_j \in \{\alpha_1, \cdots, \alpha_{i'}\}$. We denote with $J_{(k)} \rhd^{\beta_k} R_{(k)}$ the subdefinition of D_i labelled by β_k. Let $J_{(k)} = x_1^k\langle \widetilde{v_1}\rangle \mid \cdots \mid x_{e_k}^k\langle \widetilde{v_{e_k}}\rangle$, $e = \sum_{k=1}^{c} e_k$ and $\mathsf{dv}(D) = \{x_1, \cdots, x_{e'}\}$ (observe that $e \geq e'$). Let

$$\begin{array}{rcl} m_j^k & = & \max\{s \mid J_{(k)} \equiv \underbrace{x_j\langle \widetilde{v_1}\rangle \mid \cdots \mid x_j\langle \widetilde{v_s}\rangle}_{s \text{ times}} \mid J'\} \\ M_j & = & \sum_{k=1}^{c} m_j^k \end{array}$$

Let $\widetilde{a_j} = a_1^j, \cdots, a_{M_j+1}^j$. The sets $\widetilde{a_j}$ and $d_1, \cdots, d_{e'}$ are all fresh variables with respect to those in P, Q, R. Let

$$\begin{array}{rcl} H[\,] & = & \mathsf{def}\ x_1\langle \widetilde{v_1}\rangle \mid d_1\langle a', \widetilde{b_1}\rangle \rhd a'\langle x_1, \widetilde{v_1}\rangle \mid d_1\langle \widetilde{b_1}, a'\rangle\ \mathsf{in} \\ & & \vdots \\ & & \mathsf{def}\ x_{e'}\langle \widetilde{v_{e'}}\rangle \mid d_{e'}\langle a', \widetilde{b_{e'}}\rangle \rhd a'\langle x_{e'}, \widetilde{v_{e'}}\rangle \mid d_{e'}\langle \widetilde{b_{e'}}, a'\rangle\ \mathsf{in}\ [\,] \end{array}$$

and let $A = a_{M_1+1}\langle \widetilde{u_1}\rangle \mid \cdots \mid a_{M_{e'}+1}\langle \widetilde{u_{e'}}\rangle$ in

$$\begin{array}{rcl} C[\,] & = & \mathsf{def}\ A \rhd A\ \mathsf{in} \\ & & \quad \mathsf{def}\ a_{e-e_c}\langle y_1^c, \widetilde{v_1}\rangle \mid \cdots \mid a_e\langle y_{e_c}^c, \widetilde{v_c}\rangle \rhd R_{e_c}\ \mathsf{in} \\ & & \quad \vdots \\ & & \quad \mathsf{def}\ a_1\langle y_1^1, \widetilde{v_1}\rangle \mid \cdots \mid a_{e_1}\langle y_{e_1}^1, \widetilde{v_c}\rangle \rhd R_{e_1}\ \mathsf{in} \\ & & \quad [\,] \mid d_1\langle \widetilde{a_1}\rangle \mid \cdots \mid d_{e'}\langle \widetilde{a_{e'}}\rangle \end{array}$$

where a_l and y_j^k are defined as follows:
• a_l is a_r^s if over the first l-th emissions in the sequence $J_{(1)} \cdots J_{(c)}$ there are exactly r emissions on x_s;
• $y_j^k = x_j^k$ if over the first j emissions in the sequence $J_{(k)}$ (fix an ordering for $J_{(k)}$) there is exactly one emission on x_j^k. Otherwise y_j^k is a fresh variable.
Observe that definitions in $C[\,]$ are not simple in general. Let

$$R'' = H[C[\mathsf{def}\ D_{i+1}\ \mathsf{in}\ \cdots\ \mathsf{def}\ D_r\ \mathsf{in}\ J]]\,.$$

where $R_{e_1}, \cdots, R_{e_c}$ are such that labels are pairwise different. It is possible to prove that there is a derivation of $T[T_1[R'' \mid P]]$ which emits x by using at most n-times not simple definitions in R''. Similarly no derivation of $T[T_1[R'' \mid Q]]$ emits x. Finally remark that it is possible to transform R'' as described in the subcase 1. This allows to apply inductive hypothesis and terminate the proof. □

Proposition 4.3 has an immediate consequence:

Corollary 4.4 *$P \sqsubseteq_s P'$ and $Q \sqsubseteq_s Q'$ imply $P \mid Q \sqsubseteq_s P' \mid Q'$.*

Proposition 4.5 *$P \sqsubseteq_s Q$ implies* def D in $P \sqsubseteq_s$ def D in Q*, for every definition D.*

Proof: The proof is along the lines of the one of Proposition 4.3. Indeed, if def D in $P \not\sqsubseteq_s$ def D in Q, by means of the technique developed in Proposition 4.3, it is possible to define a simple context $T[\,]$ such that $T[P \mid R] \Downarrow_m x$ and $T[Q \mid R] \not\Downarrow_m x$, for some R and x. This means that $P \mid R \not\sqsubseteq_s Q \mid R$ and, by Proposition 4.3, this entails $P \not\sqsubseteq_s Q$. ∎

Proposition 4.6 *$P \sqsubseteq_s Q$ implies* def $J \rhd P$ in $R \sqsubseteq_s$ def $J \rhd Q$ in R *and* def $J \rhd P$ and D in $R \sqsubseteq_s$ def $J \rhd Q$ and D in R*, for every definition D and process R.*

Proof: Let D_P and D_Q be $J \rhd P$ and $J \rhd Q$, respectively, or $J \rhd P$ and D and $J \rhd Q$ and D, respectively. Let $T[\,]$ be a simple context which discriminates between def D_P in R and def D_Q in R and let σ be a derivation of $T[$def D_P in $R]$ showing up x and let n be the number of times the definition $J \rhd P$ is used along σ. We consider the case $D_P = J \rhd P$ and D, the other being simpler.

Let $\mathsf{fv}(P) = \{x_1, \cdots, x_k, y_1, \cdots, y_h\}$ and $\mathsf{fv}(Q) = \{x_1, \cdots, x_k, z_1, \cdots, z_l\}$ with $x_i \neq y_j \neq z_r$. Take

$$
\begin{array}{rcl}
H[\,] & = & \mathsf{def}\ x_1\langle\widetilde{u_1}\rangle \mid d_{x_1}\langle v\rangle \rhd v\langle\widetilde{u_1}\rangle \mid d_{x_1}\langle v\rangle\ \mathsf{in} \\
 & & \qquad\vdots \\
 & & \mathsf{def}\ x_k\langle\widetilde{u_k}\rangle \mid d_{x_k}\langle v\rangle \rhd v\langle\widetilde{u_k}\rangle \mid d_{x_k}\langle v\rangle\ \mathsf{in} \\
 & & \quad \mathsf{def}\ y_1\langle\widetilde{v_1}\rangle \mid d_{y_1}\langle v\rangle \rhd v\langle\widetilde{v_1}\rangle \mid d_{y_1}\langle v\rangle\ \mathsf{in} \\
 & & \qquad\vdots \\
 & & \quad \mathsf{def}\ y_h\langle\widetilde{v_h}\rangle \mid d_{y_h}\langle v\rangle \rhd v\langle\widetilde{v_h}\rangle \mid d_{y_h}\langle v\rangle\ \mathsf{in} \\
 & & \qquad \mathsf{def}\ z_1\langle\widetilde{w_1}\rangle \mid d_{z_1}\langle v\rangle \rhd v\langle\widetilde{w_1}\rangle \mid d_{z_1}\langle v\rangle\ \mathsf{in} \\
 & & \qquad\qquad\vdots \\
 & & \qquad \mathsf{def}\ z_l\langle\widetilde{w_l}\rangle \mid d_{z_l}\langle v\rangle \rhd v\langle\widetilde{w_l}\rangle \mid d_{z_l}\langle v\rangle\ \mathsf{in} \\
 & & \qquad\qquad [\,] \mid d\langle d_{x_1}, \cdots, d_{x_k}, d_{y_1}, \cdots, d_{y_h}, d_{z_1}, \cdots, d_{z_l}\rangle
\end{array}
$$

where d and $d_{x_i}, d_{y_i}, d_{z_i}$ are fresh variables with respect to those in P, Q, R. Now take Δ, $\Delta\langle X\rangle$ and the context $C[\,]$ defined below:

$$\Delta = d_{x_1}, \cdots, d_{x_k}, d_{y_1}, \cdots, d_{y_h}, d_{z_1}, \cdots, d_{z_l}$$

$$\Delta\langle X\rangle = d_{x_1}\langle x_1\rangle \mid \cdots \mid d_{x_k}\langle x_k\rangle \mid d_{y_1}\langle y_1\rangle \mid \cdots \mid d_{y_h}\langle y_h\rangle \mid d_{z_1}\langle z_1\rangle \mid \cdots \mid d_{z_l}\langle z_l\rangle$$

$$C[\,] = \mathsf{def}\ J \mid d\langle\Delta\rangle \rhd \Delta\langle X\rangle\ \mathsf{and}\ D\ \mathsf{in}\ [\,]$$

It is possible to show that

$$T[C[R \mid \underbrace{H[P] \mid \cdots \mid H[P]}_{n \text{ times}}]] \Downarrow_m x \qquad T[C[R \mid \underbrace{H[Q] \mid \cdots \mid H[Q]}_{n \text{ times}}]] \not\Downarrow_m x$$

and, by Proposition 4.5 and Corollary 4.4, this implies $P \not\sqsubseteq_s Q$. ∎

Corollary 4.7 *The relation $\sqsubseteq_s$ is a pre-congruence.*

Lemma 4.8 (The context lemma for the may-testing) *$\sqsubseteq_s$ coincides with $\sqsubseteq_m$.*

Proof: The containment $\sqsubseteq_m \subseteq \sqsubseteq_s$ is obvious by definition. Let us prove $\sqsubseteq_s \subseteq \sqsubseteq_m$. Let $P \sqsubseteq_s Q$ and $C[P] \Downarrow_m \boldsymbol{x}$, where $C[\,]$ is a (generic) context. Then $C[P] \sqsubseteq_s C[Q]$ because $\sqsubseteq_s$ is a pre-congruence. This implies that $C[Q] \Downarrow_m \boldsymbol{x}$. ∎

The context lemma is important for several reasons. It shows that the may-semantics of a process is completely characterized by replacing multiset of emissions on free variables with other multisets of emissions. In a sense, what counts of a process is its reaction in terms of emissions to stimuli which are also emissions. It is useless to look at complex reactions. Moreover, by restricting the universal quantification over contexts, the context lemma allows us to prove more easily some semantic relations. For instance, let us check the following:

$$\forall P.\ \mathsf{fv}(P) = \emptyset \Rightarrow \Omega \simeq_m P\ .$$

It suffices to verify that, for every simple context $T[\,]$ and for every $\boldsymbol{x}$, $T[\Omega] \Downarrow_m \boldsymbol{x}$ if and only if $T[P] \Downarrow_m \boldsymbol{x}$. Since both Ω and P are closed, there exists a Q such that $T[\Omega] \equiv Q \mid \Omega$ and $T[P] \equiv Q \mid P$. Then, by definition of may-emission and the hypothesis $\mathsf{fv}(P) = \emptyset$, $Q \mid \Omega \Downarrow_m \boldsymbol{x}$ if only if $Q \Downarrow_m \boldsymbol{x}$ if only if $Q \mid P \Downarrow_m \boldsymbol{x}$. This entails $\Omega \simeq_m P$.

Remark 4.9 *By the proof of the context lemma it appears that simple contexts are in some sense a minimal set of tests for checking whether two processes are may-testing equivalent or not. The existence of a smaller set of tests is a question that this paper leaves open. Surely, if this is the case, one should provide completely different proofs.*

5 The must-testing semantics

In the may-testing approach the relevant tests for processes were *the ability to respond positively to a test*. Must testing is obtained by changing the relevant tests into *the inability to respond negatively to a test*. Namely a process P is considered less specified than the process Q if, whenever P *must* respond positively to a particular test, Q *must* respond positively, too.

It turns out that a satisfactory formalization of the must-testing semantics must take into account the notion of maximal computation. We say that $P_1 \xrightarrow{\tau} \cdots \xrightarrow{\tau} P_n$ is maximal P_n is *terminal*, namely there is no Q such that $P_n \xrightarrow{\tau} Q$, noted with $P_n \not\rightarrow$.

Definition 5.1 *The* must-emission *relation is defined as follows: $P \Downarrow_M \boldsymbol{x}$ if, for every maximal computation*

$$P_1 \xrightarrow{\tau} \cdots \xrightarrow{\tau} P_n \xrightarrow{\tau} \cdots$$

there exists k such that $P_k \downarrow \boldsymbol{x}$.

We write $P \not\Downarrow_M \boldsymbol{x}$ when $P \Downarrow_M \boldsymbol{x}$ is false. When $P \Downarrow_M \boldsymbol{x}$ we say P must-emits *on $\boldsymbol{x}$, and we shall often omit the prefix "must".*

In the following we note $P \longrightarrow^n P'$ when there is a computation $P = P_1 \xrightarrow{\tau} \cdots \xrightarrow{\tau} P_n = P'$.

Proposition 5.2 *$P \Downarrow_M \boldsymbol{x}$ if and only if there exists n such that the following items hold:*

1. $(P \longrightarrow^m P' \not\rightarrow \wedge\ m < n) \Rightarrow P' \downarrow \boldsymbol{x}$;
2. $(P \longrightarrow^{n'} P' \wedge n' > n) \Rightarrow P' \downarrow \boldsymbol{x}$.

Proof: The *if-direction* is obvious. Let us check the *only-if* one. Assume that, for every n,

$$(P \longrightarrow^m P' \not\rightarrow \wedge\ m < n \wedge P' \not\downarrow \boldsymbol{x}) \text{ or } (P \longrightarrow^{n'} P' \wedge n' > n \wedge P' \not\downarrow \boldsymbol{x})$$

We prove that $P \not\Downarrow_M \boldsymbol{x}$. This is immediate when $P \longrightarrow^m P' \not\rightarrow \wedge\ m < n \wedge P' \not\downarrow \boldsymbol{x}$ because, by Remark 3.1, emission is persistent over transitions.

Let $P \longrightarrow^{n'} P' \wedge n' > n \wedge P' \not\downarrow \boldsymbol{x}$. Let $\mathcal{T} = \lim_n \mathcal{T}_n$, where $\mathcal{T}_n$ is inductively defined as follows:

1. $\mathcal{T}_0$ is the tree with the node $(P, 0)$ and without edges;
2. $\mathcal{T}_{n+1}$ is the tree $\mathcal{T}_n$ where an edge is added to every leaf (P, n) such that $P \not\downarrow x$ and $P \xrightarrow{\tau} P'$. The edge connects (P, n) and $(P', n+1)$.

$\mathcal{T}$ is finite branching, by Proposition 2.2. Moreover, by hypothesis, $\mathcal{T}_{n+1} \neq \mathcal{T}_n$. Then, by the König Lemma, $\mathcal{T}$ has an infinite path. This means there exists a maximal computation

$$P = P_1 \xrightarrow{\tau} P_2 \xrightarrow{\tau} \cdots$$

such that, for every i, $P_i \not\downarrow x$. Hence $P \not\Downarrow_M x$. □

Proposition 5.3 *$P \Downarrow_M x$ and $P \equiv Q$ imply $Q \Downarrow_M x$.*

Now we can define the *must-testing preorder*.

Definition 5.4 *The* must-testing preorder *is the relation $\sqsubseteq_M$ over processes defined by*

$$P \sqsubseteq_M Q \stackrel{\text{def}}{=} \forall C[\,]. \forall x. C[P] \Downarrow_M x \Rightarrow C[Q] \Downarrow_M x$$

Two processes P and Q are must-testing equivalent, in notation $P \simeq_M Q$, if $P \sqsubseteq_M Q$ and $Q \sqsubseteq_M P$.

Must-testing preorder and equivalence are pre-congruences, by definition. An immediate consequence of Proposition 5.3 is:

Corollary 5.5 *$P \equiv Q$ implies $P \sqsubseteq_M Q$.*

We examine the discriminating power of must-testing through some examples which also highlight differences with may-testing. For every process P and Q:

- $\Omega \sqsubseteq_M P$ because, informally, if $C[\Omega] \Downarrow_M y$ then every derivation σ starting at $C[\Omega]$ must converge. This means that the process Ω is never "enabled" along σ. Therefore also P is never enabled along derivations of $C[P]$.
- let $x \notin \mathsf{fv}(P) \cup \mathsf{fv}(Q)$. Then $\Omega \mid \mathsf{def}\ x\langle\rangle \rhd P\ \mathsf{in}\ x\langle\rangle \simeq_M \Omega \mid \mathsf{def}\ x\langle\rangle \rhd Q\ \mathsf{in}\ x\langle\rangle$;
- let $0 = \mathsf{def}\ x\langle\rangle \mid y\langle\rangle \rhd x\langle\rangle\ \mathsf{in}\ x\langle\rangle$. Observe that $0 \not\sqsubseteq_M \Omega$. For example take the context $C[\,] = \mathsf{def}\ z\langle\rangle \rhd a\langle\rangle\ \mathsf{in}\ [\,] \mid z\langle\rangle$. It happens that $C[0] \Downarrow_M a$ while $C[\Omega] \not\Downarrow_M a$ because of the computation
$$C[\Omega] \xrightarrow{\tau} C[\Omega] \xrightarrow{\tau} \cdots$$
and $C[\Omega] \not\downarrow a$.

Notice that the second and third items establish that may and must-testing are uncomparable. The first item decrees that Ω is the least process in the must-testing scenario, too. It is also worth noting that must-testing remains unchanged by taking the "must-adaptations" of the basic observations defined in Remark 3.7. Since we are not primarily concerned with these refinements here, we overlook this issue.

The surprising property (at least for us) of the join-calculus is that, up-to the basic observations, the must-testing preorder holds the same context lemma of the may-testing one. This is false in Milner's CCS, for instance [7]. In order to show that simple contexts suffice, let

$$P \sqsubseteq_S Q \stackrel{\text{def}}{=} \forall T[\,]. \forall x. T[P] \Downarrow_M x \Rightarrow T[Q] \Downarrow_M x$$

where $T[\,]$ are the simple contexts defined in the previous section. The proof for checking the coincidence of $\sqsubseteq_M$ and $\sqsubseteq_S$ carries over along the same lines as the one of Lemma 4.8. Nemely, we (re)consider the labelled join-calculus and the relation $P \Downarrow_M^+ x$ which is similar to the must-emission, except that it forgets labels. We also omit the adaptation of Proposition 4.1 to the case of must-testing. For running the induction in the case of must-testing it is crucial a measure of complexity of well-labelled processes, noted $\natural$, which gives a sequence of integers $a_n \cdots a_0$. Foremost let us define the complexity of a not simple definition labelled α in a well-labelled process P, noted $\flat(P, \alpha)$, by structural induction over P as follows:

1. if $P = P_1 \mid \cdots \mid P_n$ and α occurs in P_i then $\flat(P,\alpha) = \flat(P_i,\alpha)$;
2. if $P = \mathsf{def}\ J_1 \rhd^{\beta_1} Q_1$ and $\cdots$ and $J_k \rhd^{\beta_k} Q_k$ $\mathsf{in}\ R$ and $\alpha \in \{\beta_1,\cdots,\beta_k\}$ then $\flat(P,\alpha) = 1$;
3. if $P = \mathsf{def}\ D\ \mathsf{in}\ R$ and α occurs in R then $\flat(P,\alpha) = \flat(R,\alpha)$;
4. if $P = \mathsf{def}\ J_1 \rhd^{\beta_1} Q_1$ and $\cdots$ and $J_k \rhd^{\beta_k} Q_k$ $\mathsf{in}\ R$ and α occurs in Q_i then $\flat(P,\alpha) = \flat(Q_i,\alpha) + 1$.

The complexity of a label is robust with respect to structural equivalence:

Proposition 5.6 *If $P \equiv Q$ then $\flat(P,\alpha) = \flat(Q,\alpha)$.*

Finally, the complexity of a well-labelled process P, noted $\natural(P)$ is the sequence of integer $a_n \cdots a_0$ such that a_i is the cardinality of the set of labels α such that $\flat(P,\alpha) = i+1$.

Proposition 5.7 *$P \sqsubseteq_S Q$ implies $P \mid R \sqsubseteq_S Q \mid R$, for every process R.*

Proof: For every ξ and for every R such that $\natural(R) = \xi$ and $T[P \mid R] \Downarrow^+_M x$ and $T[Q \mid R] \not\Downarrow^+_M x$, for some $T[\ ]$, we show the existence of a $T'[\ ]$ such that $T'[P] \Downarrow_M x$ and $T'[Q] \not\Downarrow_M x$. We argue by induction on ξ, where sequences are ordered by the lexicographic ordering. Without loss of generality we assume $R = \mathsf{def}\ D_1\ \mathsf{in}\ \cdots\ \mathsf{def}\ D_r\ \mathsf{in}\ J$.

($\xi = 0$) Obvious.

($\xi = a_n \cdots a_0$) Let $T[P \mid R] \Downarrow_M x$ and let σ be a maximal computation of $T[Q \mid R]$ never emitting on x.

If no label in Ψ occurs along σ then let $R' = H_1[\cdots H_r[\ ]\cdots]$ where $H_i[\ ] = \mathsf{def}\ D_i\ \mathsf{in}\ [\ ]$ if D_i is simple and $H_i[\ ] = \mathsf{def}\ x_1\langle\widetilde{v_1}\rangle \rhd x_1\langle\widetilde{v_1}\rangle\ \mathsf{in}\ \cdots\ \mathsf{def}\ x_h\langle\widetilde{v_h}\rangle \rhd x_h\langle\widetilde{v_h}\rangle\ \mathsf{in}\ [\ ]$ if D_i is not simple and $\mathsf{dv}(D_i) = \{x_1,\cdots,x_h\}$. It is easy to prove that $T[P \mid R'] \Downarrow_M x$ and $T[Q \mid R'] \not\Downarrow_M x$. Remark also that by the structural equivalence $T[P \mid R']$ and $T[Q \mid R']$ may be easily transformed into $T'[P]$ and $T'[Q]$, for some simple context $T'[\ ]$.

Otherwise, let D_i be the not simple definition of R whose label (or a label of a component definition) α occurs as first label in Ψ along σ. There are three subcases:

1. D_i is $J \rhd^{\alpha} S$ or D_i is a sub-definition, namely $J_1 \rhd^{\beta_1} R_1$ and $\cdots$ and $J_{i'} \rhd^{\beta_{i'}} R_{i'}$, $\alpha \in \{\beta_1,\cdots,\beta_{i'}\}$ and no β_k is the first label in Ψ occurring along σ.
   Let $R' = H_1[\cdots H_r[\ ]\cdots]$ where $H_j[\ ] = \mathsf{def}\ D_j\ \mathsf{in}\ [\ ]$ if D_j is simple; $H_i[\ ] = \mathsf{def}\ x_1\langle\widetilde{v_1}\rangle \rhd x_1\langle\widetilde{v_1}\rangle\ \mathsf{in}\ \cdots\ \mathsf{def}\ x_h\langle\widetilde{v_h}\rangle \rhd x_h\langle\widetilde{v_h}\rangle\ \mathsf{in}\ [\ ]$ where $\mathsf{dv}(D_i) = \{x_1,\cdots,x_h\}$; $H_j[\ ] = \mathsf{def}\ x_1\langle\widetilde{v_1}\rangle \rhd x\langle\widetilde{v}\rangle\ \mathsf{in}\ \cdots\ \mathsf{def}\ x_h\langle\widetilde{v_h}\rangle \rhd x\langle\widetilde{v}\rangle\ \mathsf{in}\ [\ ]$ if D_j is not simple and $\mathsf{dv}(D_j) = \{x_1,\cdots,x_h\}$. It is easy to show that $T[R' \mid P] \Downarrow_M x$ and $T[R' \mid Q] \not\Downarrow_M x$.
2. D_i is a sub-definition, namely $J_1 \rhd^{\beta_1} R_1$ and $\cdots$ and $J_{i'} \rhd^{\beta_{i'}} R_{i'}$, $\alpha \in \{\beta_1,\cdots,\beta_{i'}\}$ and no derivation of $T[P \mid R]$ has α as first label in Ψ. Let $\alpha = \beta_k$ and let $\mathsf{dv}(D_i) \setminus \mathsf{dv}(J_k) = \{y_1,\cdots,y_{h'}\}$. Remark that, by hypothesis, $\{y_1,\cdots,y_{h'}\}$ is not empty. Now take $R' = H_1[\cdots H_r[\ ]\cdots]$ where $H_j[\ ] = \mathsf{def}\ D_j\ \mathsf{in}\ [\ ]$ if D_j is simple; $H_i[\ ] = \mathsf{def}\ J_k \rhd J_k\ \mathsf{in}\ \mathsf{def}\ y_1\langle\widetilde{v_1}\rangle \rhd x\langle\widetilde{v}\rangle\ \mathsf{in}\ \cdots\ \mathsf{def}\ y_{h'}\langle\widetilde{v_{h'}}\rangle \rhd x\langle\widetilde{v}\rangle\ \mathsf{in}\ ([\ ] \mid x\langle\widetilde{v}\rangle)$; and $H_j[\ ] = \mathsf{def}\ x_1\langle\widetilde{v_1}\rangle \rhd x\langle\widetilde{v}\rangle\ \mathsf{in}\ \cdots\ \mathsf{def}\ x_h\langle\widetilde{v_h}\rangle \rhd x\langle\widetilde{v}\rangle\ \mathsf{in}\ [\ ]$ if $i \neq j$ and D_j is not simple and $\mathsf{dv}(D_j) = \{x_1,\cdots,x_h\}$. It is possible to prove that $T[P \mid R'] \Downarrow_M x$ and $T[Q \mid R'] \not\Downarrow_M x$ and $T[P \mid R']$ and $T[Q \mid R']$ may be easily transformed into $T'[P]$ and $T'[Q]$, for some $T'[\ ]$.
3. There is a derivation of $T[P \mid R]$ having α as first label in Ψ. There are two subcases. (D_i is $J' \rhd^{\alpha} R'$). Let $\mathsf{fv}(R') = \{x_1,\cdots,x_t\}$ and let n be the integer of Proposition 5.2. This means that every derivation of $T[P \mid R]$ emits x with at most n transitions labelled α. Now take the following context $H[\ ]$:

$$
\begin{array}{rcl}
H[\ ] & = & \mathsf{def}\ x_1\langle\widetilde{u_1}\rangle \mid d_{x_1}\langle v\rangle \rhd v\langle\widetilde{u_1}\rangle \mid d_{x_1}\langle v\rangle\ \mathsf{in} \\
 & & \vdots \\
 & & \mathsf{def}\ x_t\langle\widetilde{u_t}\rangle \mid d_{x_t}\langle v\rangle \rhd v\langle\widetilde{u_t}\rangle \mid d_{x_t}\langle v\rangle\ \mathsf{in} \\
 & & [\ ] \mid d\langle d_{x_1},\cdots,d_{x_t}\rangle \mid d'\langle\rangle
\end{array}
$$

where d, d' and d_{x_i} are fresh variables. Let

$$\Delta = \underbrace{d'\langle\rangle \mid \cdots \mid d'\langle\rangle}_{n \text{ times}}$$

and:

$$H'[\,] = \mathsf{def}\ \Delta \rhd \Delta\ \mathsf{in}$$
$$\mathsf{def}\ J \mid d\langle d_{x_1}, \cdots, d_{x_t}\rangle \rhd d_{x_1}\langle x_1\rangle \mid \cdots \mid d_{x_t}\langle x_t\rangle\ \mathsf{in}\ [\,]$$

Finally let

$$R'' = \mathsf{def}\ D_1\ \mathsf{in} \cdots \mathsf{def}\ D_{i-1}\ \mathsf{in}$$
$$H'[\mathsf{def}\ D_{i+1}\ \mathsf{in} \cdots \mathsf{def}\ D_r\ \mathsf{in}\ J \mid \underbrace{H[R'] \mid \cdots \mid H[R']}_{n \text{ times}}]$$

It follows that $T[P \mid R''] \Downarrow_M x$ and $T[P \mid R''] \not\Downarrow_M x$. Furthermore, notice that $\natural(R'') < \natural(R)$ (assuming that occurrences of R' in R'' have different labels). This allows to apply inductive hypothesis.
(D_i is $J_1 \rhd^{\beta_1} R_1$ and $\cdots$ and $J_{i'} \rhd^{\beta_k} R_{i'}$). Let $\alpha = \beta_k$ and let $\mathsf{dv}(D_i) = \{x_1, \cdots, x_t\}$. Take

$$m_j = \max\{m \mid J_k \equiv \underbrace{x_j\langle\widetilde{v_1}\rangle \mid \cdots \mid x_j\langle\widetilde{v_m}\rangle}_{m \text{ times}} \mid J'\}$$
$$M = \max\{m_1, \cdots, m_t\}\,.$$

Moreover let $\widetilde{d_i} = d^i_{x_1}, \cdots, d^i_{x_t}$, where $d^i_{x_j} = d_{x_j}$ if $i \leq m_j$ and $d^i_{x_j} = d'_{x_j}$ otherwise. Observe that $d^{M+1}_{x_j} = d'_{x_j}$ and let $\Delta_k = \widetilde{d_1}, \cdots, \widetilde{d_{M+1}}$. Now consider the operation lshift^{M+1}_i defined as follows:

$$\mathsf{lshift}^{M+1}_i(\Delta_k) = d^1_{x_1}, \cdots, d^2_{x_i}, \cdots, d^1_{x_t}, \cdots, d^M_{x_1}, \cdots, d^{M+1}_{x_i}, \cdots, d^M_{x_t}, \widetilde{d_{M+1}}$$

When $j \neq k$ let $\Delta_j = c^j_{x_1}, c^j_{x_1}, \cdots, c^j_{x_t}, c^j_{x_t}$. We define

$$\Delta = \Delta_1, \cdots, \Delta_t$$

$$\mathsf{lshift}(\Delta) = \mathsf{lshift}^2_1(\Delta_1), \cdots, \mathsf{lshift}^{M+1}_k(\Delta_k), \cdots, \mathsf{lshift}^2_t(\Delta_t)$$

$$H[\,] = \mathsf{def}\ x_1\langle\widetilde{v_1}\rangle \mid d\langle\Delta\rangle \rhd d^1_{x_1}\langle x_1, \widetilde{v_1}\rangle \mid c^1_{x_1}\langle\rangle \mid \cdots \mid c^t_{x_1}\langle\rangle \mid d\langle\mathsf{lshift}(\Delta)\rangle\ \mathsf{in}$$
$$\vdots$$
$$\mathsf{def}\ x_t\langle\widetilde{v_t}\rangle \mid d\langle\Delta\rangle \rhd d^1_{x_t}\langle x_t, \widetilde{v_t}\rangle \mid c^1_{x_t}\langle\rangle \mid \cdots \mid c^t_{x_t}\langle\rangle \mid d\langle\mathsf{lshift}(\Delta)\rangle\ \mathsf{in}$$
$$[\,] \mid d\langle\Delta\rangle$$

Moreover, when $j \neq k$, let J^{ch1}_j be the pattern J_j where each atom $x_i\langle\widetilde{w}\rangle$ has been replaced by $c^j_{x_i}\langle\rangle$; J^{ch1}_k be the pattern J_k each atom $x_i\langle\widetilde{w}\rangle$ has been replaced by $d_{x_i}\langle x_i, \widetilde{w}\rangle$; J^{ch2}_j be the pattern J_j where each atom $x_i\langle\widetilde{w}\rangle$ has been replaced by $d'_{x_i}\langle x_i, \widetilde{w}\rangle$; $D^{d'}_i$ be the definition D_i where each J_j has been replaced by J^{ch2}_j.
Consider

$$H'[\,] = \mathsf{def}\ J^{\mathrm{ch1}}_k \rhd R_k\ \mathsf{in}\ \mathsf{def}\ D^{d'}_i\ \mathsf{in}$$
$$\mathsf{def}\ J^{\mathrm{ch1}}_1 \rhd x\langle\widetilde{u}\rangle\ \mathsf{in}$$
$$\vdots$$
$$\mathsf{def}\ J^{\mathrm{ch1}}_{k-1} \rhd x\langle\widetilde{u}\rangle\ \mathsf{in}\ \mathsf{def}\ J^{\mathrm{ch1}}_{k+1} \rhd x\langle\widetilde{u}\rangle\ \mathsf{in}$$
$$\vdots$$
$$\mathsf{def}\ J^{\mathrm{ch1}}_{i'} \rhd x\langle\widetilde{u}\rangle\ \mathsf{in}\ x\langle\widetilde{u}\rangle\ \mathsf{in}\ [\,]$$

Finally, let

$$R' = \mathsf{def}\ D_1\ \mathsf{in}\ \cdots\ \mathsf{def}\ D_{i-1}\ \mathsf{in}\ H'[H[\mathsf{def}\ D_{i+1}\ \mathsf{in}\ \cdots\ \mathsf{def}\ D_r\ \mathsf{in}\ J]]\,.$$

It is possible to prove that $T[P \mid R'] \Downarrow_M x$ and $T[Q \mid R'] \not\Downarrow_M x$. On $T[P \mid R']$ and $T[Q \mid R']$ we eventually reiterate the above argument (if R_k is a simple process) or use the subcase 2 and simplify the complexity of R'. Remark also that the reiteration is used a finite number of times because, by Proposition 5.2, there exist n such that every computation of $T[P \mid R']$ emits x with at most n transitions. □

The proof of Proposition 5.7 deserves few comments, in particular the proof of the item 3. In this case there is a maximal computation σ of $T[Q \mid R]$ (which never emits on x) whose first label $\alpha \in \Psi$ marks a not simple definition D_i of R. There is also a derivation σ' of $T[P \mid R]$ whose first label in Ψ is α. There are two cases: D_i is $J' \rhd^{\alpha} R'$ or D_i is $J_1 \rhd^{\beta_1} R_1$ and $\cdots$ and $J_{i'} \rhd^{\beta_{i'}} R_{i'}$ ($\alpha = \beta_k$). When D_i is $J' \rhd^{\alpha} R'$, the number of derivations marked α before σ' emits on x is bound by n, where n is given by Proposition 5.2. This means that R' may be instantiated at most n times. The proof states that this instantiation may be performed once at all without altering the must emission on x. The reader is invited to check that every maximal computation of $T[P \mid R'']$ eventually emits on x. The case when D_i is $J_1 \rhd^{\beta_1} R_1$ and $\cdots$ and $J_{i'} \rhd^{\beta_{i'}} R_{i'}$ ($\alpha = \beta_k$) is more subtle. Indeed R' must be transformed into a "simpler" process R'' by keeping $T[P \mid R''] \Downarrow_M x$ and $T[Q \mid R''] \not\Downarrow_M x$. To this aim the idea is to take the sequence of labels in Ψ along σ and define R'' in such a way that every computation showing up a different sequence of labels in Ψ emits on x. This is the purpose of the J_j^{ch1}. The argument by recurrence allows to reiterate the technique to transitions of σ after the first labelled α.

Corollary 5.8 *$P \sqsubseteq_S P'$ and $Q \sqsubseteq_S Q'$ imply $P \mid Q \sqsubseteq_S P' \mid Q'$.*

With an argument similar to the one used for Proposition 5.7 it is possible to prove:

Proposition 5.9 *$P \sqsubseteq_S Q$ implies* def D in $P \sqsubseteq_S$ def D in Q*, for every definition D.*

Proposition 5.10 *$P \sqsubseteq_M Q$ implies* def $J \rhd P$ in $R \sqsubseteq_M$ def $J \rhd Q$ in R *and* def $J \rhd P$ and D in $R \sqsubseteq_M$ def $J \rhd Q$ and D in R*, for every definition D and process R.*

Proof: Let $T[\,]$ be a simple context which discriminates between def D_P in R and def D_Q in R, where D_P and D_Q may be $J \rhd P$ and $J \rhd Q$, respectively, or $J \rhd P$ and D' and $J \rhd Q$ and D'. Let n be the integer which follows from Proposition 5.2 for $T[\text{def } D_P \text{ in } R] \Downarrow_M x$. Let $H[\,]$ be the simple context defined in the proof of Proposition 4.6. Now take

$$R_P = R \mid \underbrace{H[P] \mid \cdots \mid H[P]}_{n \text{ times}} \qquad R_Q = R \mid \underbrace{H[Q] \mid \cdots \mid H[Q]}_{n \text{ times}}$$

By Corollary 5.8, $R_P \not\sqsubseteq_S R_Q$ implies $P \not\sqsubseteq_S Q$. Let us focus on the case when D_P be $J \rhd P$ and D' and D_Q be $J \rhd Q$ and D', the other one being simpler.

Consider the following context $C[\,]$ where $a = k + h + l$ (see Proposition 4.6 for the definitions of k, h, l):

$$\begin{aligned} C[\,] = \text{def } & c_{n+1}\langle \widetilde{u} \rangle \rhd c_{n+1}\langle \widetilde{u} \rangle \text{ in} \\ & \text{def } c_n\langle d_1, \cdots, d_a, u_1, \cdots, u_a \rangle \rhd d_1\langle u_1 \rangle \mid \cdots \mid d_a\langle u_a \rangle \text{ in} \\ & \qquad \vdots \\ & \text{def } c_1\langle d_1, \cdots, d_a, u_1, \cdots, u_a \rangle \rhd d_1\langle u_1 \rangle \mid \cdots \mid d_a\langle u_a \rangle \text{ in} \\ & \quad \text{def } J \mid d\langle \widetilde{d} \rangle \mid c\langle c', \widetilde{c} \rangle \rhd c'\langle \widetilde{d}, \widetilde{x} \rangle \mid c\langle \widetilde{c}, c' \rangle \text{ and } D' \text{ in} \\ & \qquad [\,] \mid c\langle c_1, \cdots, c_{n+1} \rangle \end{aligned}$$

If $T[\text{def } D_P \text{ in } R] \Downarrow_M x$ and $T[\text{def } D_Q \text{ in } R] \not\Downarrow_M x$ then, by construction, $T[C[R_P]] \Downarrow_M x$ and $T[C[R_Q]] \not\Downarrow_M x$. This means that $C[R_P] \not\sqsubseteq_S C[R_Q]$ and, by Proposition 5.9, $R_P \not\sqsubseteq_S R_Q$. □

Corollary 5.11 *The relation $\sqsubseteq_S$ is a pre-congruence.*

Lemma 5.12 (The context lemma for the must-testing) $\sqsubseteq_S$ *coincides with* $\sqsubseteq_M$.

Proof: The containment $\sqsubseteq_M \subseteq \sqsubseteq_S$ is obvious by definition. Let us prove $\sqsubseteq_S \subseteq \sqsubseteq_M$. Let $P \sqsubseteq_S Q$ and $C[P] \Downarrow_M \boldsymbol{x}$, where $C[\,]$ is a (generic) context. Then $C[P] \sqsubseteq_S C[Q]$ because $\sqsubseteq_S$ is a pre-congruence. This implies that $C[Q] \Downarrow_M \boldsymbol{x}$. □

Because of the above lemma it is possible to define the may and must-semantics (and their intersection, the so-called *testing semantics* [4]) in a uniform way by means of simple contexts. Moreover, as in the may-testing, Lemma 5.12 turns out useful for proving easily semantic relations. The reader is invited to check that $\forall P.\ \Omega \sqsubseteq_M P; \forall P.\ \Omega \mid P \sqsubseteq_M P; \forall P, Q.\ \Omega \mid P \simeq_M \Omega \mid Q$.

Lemmas 4.8 and 5.12 entail that much of the strength of the join-calculus must be recognized to the elaborate synchronization schema it provides. For instance through join patterns it is possible to test the alternative presence of a set of actions or the simultaneous presence of a multiset of actions. Precisely, it turns out that elaborate synchronization schema are the unique mechanism for setting may and must preorders. One should wonder whether even simpler contexts could be used. In [6] it has been proved that the language yielded by stripping away join patterns including more than two messages is as much expressive as the full join-calculus. Unfortunately the encoding therein introduces infinite sequences of internal reductions and this invalidates the must-testing preorder, which is sensible to divergence.

6 Conclusion

To conclude let us briefly comment on our results with respect to related works and hint to future research.

6.1 Related works

The asynchronous version of mobile process algebras has been investigated only recently [2, 9]. In particular Honda and Tokoro studied asynchronous bisimulations, whilst Boudol defines may-testing with the purpose of fixing some adequacy results about Milner's encoding of lazy λ-calculus into π-calculus [10]. May-testing has been also used in [3] for determining the discriminating power of π-calculus contexts with respect to the encoding of lazy λ-calculus. However both [2] and [3] miss of any characterization of "canonical" contexts for may-testing, even if some conclusions may be drawn (see below).

Switching to synchronous mobile calculi, testing semantics has been studied thoroughly by Boreale and De Nicola in [1] where a proof system and a term model have been provided for π-calculus. We also recall Hennessy's detailed study of may-testing for an higher order process calculus with *dynamic binding* [8]. Also in [1] and in [8] the problem of determining canonical tests has not been addressed. Hennessy is investigating a model for the testing semantics of the π-calculus. Details of this work are unknown to us.

6.2 The asynchronous π-calculus

Fournet and Gonthier give equivalence result between the core join-calculus and the asynchronous π-calculus. Therefore it is fair to ask whether the latter also holds a context lemma for testing semantics. To this aim we foremost observe that there are two kinds of actions that may be observed in asynchronous π-calculus: inputs and outputs. It turns out that observing inputs suffices, since it is always possible to find a context which firstly consumes all the outputs and then performs an input on a fresh variable. This fits with the may-semantics defined in [2]. The results in [3] seem to imply that canonical contexts for may-testing are quite involved. In particular every operator of asynchronous π-calculus is used (even if we feel that we may rid of replication). This consideration follows by the Separation Lemma, which provides contexts discriminating λ-terms with different lazy-trees, and the full-abstraction result (Theorem 7.5). A precise characterization of these contexts is still not known.

6.3 The trinity

In the terminology of [7], the trinity of a language consists of an operational semantics, a denotational one and a proof system, all defining the same congruence over processes. This paper gives only one vertex of the trinity: finding the other twos is an interesting direction of research. To this purpose, one soon realizes that the source of problems is the unique, not standard operator: the definition. Let us start with a simple case. The process

$$P = \mathsf{def}\ x\langle\widetilde{u}\rangle \rhd R\ \mathsf{in}\ Q$$

may be modelled in the very same way as

$$\mathsf{let\ rec}\ x(\widetilde{u}) = R\ \mathsf{in}\ Q$$

in the usual functional languages. Namely the denotation of P is the least upper bound of the chain of processes $Q[^{R^{(n)}}/_{x}]$, where $R^{(n)}$ is the n-th approximant of the fixpoint of $\lambda x\widetilde{u}.\, R$.

The problems begin when patterns have at least two messages. In this case the above approach is wrong because in the chain $Q[^{R^{(n)}}/_{x}]$ the patterns which are replaced are those produced by R or those produced by Q. No chance to replace patterns produced by both R and Q. For instance in the process

$$\mathsf{def}\ x\langle u\rangle \mid y\langle v\rangle \rhd x\langle v\rangle\ \mathsf{in}\ x\langle a\rangle \mid y\langle b\rangle \mid y\langle c\rangle$$

the second move is caused by a message from R and a message from Q.

This issue needs surely further investigation. Perhaps recent progresses on denotational semantics of π-calculus may help [5, 12].

References

[1] M. Boreale and R. De Nicola. Testing equivalence for mobile processes. *Information and Computation*, 120:279 – 303, 1995.

[2] G. Boudol. Asynchrony and the π-calculus. Technical Report 1702, INRIA Sophia-Antipolis, May 1992.

[3] G. Boudol and C. Laneve. Lambda-calculus, multiplicities and the π-calculus. Technical Report 2581, INRIA Sophia-Antipolis, March 1995.

[4] R. De Nicola and M. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83 – 133, 1984.

[5] M. Fiore, E. Moggi, and D. Sangiorgi. A fully-abstract model for the π-calculus. In *Proceedings* 11^{th} *Annual Symposium on Logic in Computer Science,* New Brunswick, New Jersey, USA, 1996.

[6] C. Fournet and G. Gonthier. The reflexive CHAM and the join-calculus. In *Proceedings* 23^{th} *ACM Symposium on Principles of Programmining Languages*, 1996.

[7] M. Hennessy. *Algebraic Theory of Processes*. MIT Press, 1988.

[8] M. Hennessy. A fully abstract denotational model for higher-order processes. *Information and Computation*, 112:55 – 95, 1994.

[9] K. Honda and M. Tokoro. On asynchronous communication semantics. In *Proceeding of ECOOP'91*, volume 612 of *Lecture Notes in Computer Science*, pages 21 – 51. Springer-Verlag, 1992.

[10] R. Milner. Functions as processes. *Mathematical Structures in Computer Science*, 2:119 – 141, 1992.

[11] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes I and II. *Information and Computation*, 100:1 – 41, 42 – 78, 1992.

[12] I. Stark. A fully-abstract domain model for the π-calculus. In *Proceedings* 11^{th} *Annual Symposium on Logic in Computer Science,* New Brunswick, New Jersey, USA, 1996.