Observing Success in the Pi-Calculus

David Sabel and Manfred Schmidt-Schauß

Goethe-University, Frankfurt am Main
{sabel,schauss}@ki.cs.uni-frankfurt.de

Abstract

A contextual semantics – defined in terms of successful termination and may- and should-convergence – is analyzed in the synchronous pi-calculus with replication and a constant Stop to denote success. The contextual ordering is analyzed, some nontrivial process equivalences are proved, and proof tools for showing contextual equivalences are provided. Among them are a context lemma and new notions of sound applicative similarities for may- and should-convergence. A further result is that contextual equivalence in the pi-calculus with Stop conservatively extends barbed testing equivalence in the (Stop-free) pi-calculus and thus results on contextual equivalence can be transferred to the (Stop-free) pi-calculus with barbed testing equivalence.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages, D.3.1 Formal Definitions and Theory F.4.2 Grammars and Other Rewriting Systems,

Keywords and phrases concurrency, process calculi, pi-calculus, rewriting, semantics

Digital Object Identifier 10.4230/OASIcs.WPTE.2015.31

1 Introduction

The π-calculus [9, 8, 20] is a well-known model for concurrent processes with message passing. Its minimalistic syntax includes parallel process-composition, named channels, and input/output-capabilities on the channels. The data flow in the π-calculus is programmed by communication between processes. There is a lot of research on several bisimulations of π-processes (see e.g. [20, 10]). They equate processes if testing the processes (using reduction) exhibits that they have the same input and output capabilities and that they reach equivalent states. Bisimulations occur in *strong* variants, where bisimilar processes must have an identical reduction behavior for every single reduction step, and there are *weak* bisimulations, where the numbers of internal reduction need not coincide, but equivalent states w.r.t. the reflexive-transitive closure of reduction must be reached.

While proving processes bisimilar is often easy and elegant, the bisimilarities (in weak and strong variants) are very fine grained notions, and thus may not allow to equate processes even if they can be seen as semantically equal. We are interested in coarser notions of process equivalences as the *semantic base* and view bisimulations (provided that they are sound w.r.t. the semantics) as very helpful proof tools for investigating the (contextual) semantics.

For program calculi based on the lambda-calculus the usual approach to program equivalence is Morris style-contextual equivalence [11, 15] which can be used in a uniform way for a lot of those calculi. For deterministic languages, contextual equivalence is based on the notion of a terminated (or successful) program and it equates programs, if the ability to terminate (i.e. so-called *may-convergence*) is indistinguishable when exchanging one program by the other in any surrounding program context. For non-deterministic languages this equivalence is too coarse, but it can be strengthened by additionally observing whether the program successfully terminates on all execution paths, i.e. whether the program *must-converges*, or as a slightly different approach, whether the program *should-converges*, which holds, if

2nd International Workshop on Rewriting Techniques for Program Transformations and Evaluation (WPTE'15).
Editors: Yuki Chiba, Santiago Escobar, Naoki Nishida, David Sabel, and Manfred Schmidt-Schauß; pp. 31–46

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

the property of being *may-convergent* holds on all execution paths. In contrast to must-convergence, should-convergence (see e.g. [12, 2, 16, 21]) has some kind of fairness built-in: the predicate does not change even if instead of all reduction sequences only *fair* ones are taken into account, where fair reduction sequences ensure that every reducible expression is reduced after finitely many reduction steps.

In this paper we analyze contextual equivalence in the synchronous π-calculus with replication and – for simplicity – without sums. Since the π-calculus has no notion of successful termination, we use the same approach as e.g. [6, 13] and add a syntactic construct (the constant Stop) which indicates successful termination. We call the extended calculus Π_{Stop}. We develop two proof tools for showing program equivalences: We prove that a context lemma holds, which restricts the required class of contexts to show contextual equivalence. We introduce notions of applicative similarities for the may- and the should-convergence. There are soundness results on bisimilarities and barbed may- and should-testing for the asynchronous π-calculus in [5], but to the best of our knowledge our notion of an applicative similarity for should-convergence is new. We prove soundness of these similarities w.r.t. the contextual preorders and thus they can be used for co-inductive proofs to show contextual equivalence. Even though the test for may-convergence is subsumed by testing should-convergence, our reasoning tools require also reasoning about may-convergence, and thus we will consider both predicates. Equipped with these tools we show that process interaction is correct, if it is deterministic, prove some further process equations, and investigate the contextual ordering. We show that a contextually least element does not exist in Π_{Stop}, but a largest element exists – the constant Stop.

Our notion of contextual equivalence seems to be close to testing equivalences (see e.g. [3] for CCS, [1] for a restricted variant of the π-calculus, [5] for the asynchronous π-calculus, and [7] for the join-calculus), which are defined analogously to contextual equivalence, but instead of observing successful termination, other observations are relevant. For *barbed testing equivalences* [5] the capability of receiving (or emitting) a name on an (open) input (or output) channel is observed (i.e. the process has an input or output *barb*). There is also some work where testing is a combination of may- and must- or should-testing (which is sometimes called fair must-testing, e.g. [5]). Roughly speaking, these predicates require an input or output capability on every execution path. We prove a strong connection between contextual equivalence and barbed may- and should-testing: On Stop-free processes, barbed testing equivalence coincides with contextual equivalence. This connection enables us to transfer several of our results to the classic π-calculus without Stop.

Outline. In Sect. 2 we introduce the synchronous π-calculus with Stop. The context lemma and soundness of applicative similarity is proven in Sect. 3. We analyze the contextual ordering and prove correctness of deterministic process interaction in Sect. 4. In Sect. 5 we analyze the connection between contextual equivalence in Π_{Stop} and barbed testing equivalence in the π-calculus without Stop and transfer our results obtained in Π_{Stop} to the Stop-free calculus. Finally, we conclude in Section 6.

2 The π-Calculus Π_{Stop} with Stop

We consider the synchronous π-calculus Π_{Stop} with replication and a constant Stop. For simplicity, we neither include sums nor name matching. Let $\mathcal{N}$ be a countably infinite set of *names*. *Processes* $Proc_{\mathsf{Stop}}$ and *action prefixes* π are defined as follows, where $x, y \in \mathcal{N}$:

$$P, Q, R \in Proc_{\mathsf{Stop}} \quad ::= \quad \pi.P \mid P_1 \mid P_2 \mid !P \mid \mathbf{0} \mid \nu x.P \mid \mathsf{Stop} \qquad \pi \quad ::= \quad x(y) \mid \overline{x}y$$

$P_1 \mid P_2$ is the *parallel composition* of processes P_1 and P_2. A process $x(y).P$ has the capability to *receive* some name z along the channel x and then behaves like $P[z/y]$ where $[z/y]$ is the capture free substitution of y by z. A process $\overline{x}y.P$ can *send* the name y along the channel x and thereafter it behaves like P. $\mathbf{0}$ is the *silent process* and Stop is the successful process. A *restriction* $\nu z.P$ restricts the scope of the name z to process P. The *replication* $!\,P$ represents arbitrary many parallel copies of process P. The constructs $\nu x.P$ and $y(x).P$ bind the name x with scope P which induces a notion of α-renaming and α-equivalence $=_\alpha$ as usual. We use $\mathsf{fn}(P)$ for the set of *free names* of P and adopt the distinct name convention, and assume that free names are distinct from bound names and bound names are pairwise distinct. We also use name substitutions $\sigma : \mathcal{N} \to \mathcal{N}$. With Σ we denote the set of all name substitutions.

In the remainder of the paper, we use several binary relations on processes. Given a relation $\mathcal{R} \subseteq (Proc_{\mathsf{Stop}} \times Proc_{\mathsf{Stop}})$, we write $\mathcal{R}^{-1}$ for the relation $\{(Q,P) \mid (P,Q) \in \mathcal{R}\}$ and $\mathcal{R}^\sigma$ is defined as: $(P,Q) \in \mathcal{R}^\sigma$ iff $(\sigma(P),\sigma(Q)) \in \mathcal{R}$ for all $\sigma \in \Sigma$.

A *context* $C \in \mathcal{C}_{\mathsf{Stop}}$ is a process with a *hole* $[\cdot]$. Replacing the hole of C by process P is written as $C[P]$. *Structural congruence* $\equiv$ is the smallest congruence satisfying the axioms:

$$
\begin{array}{ccc}
P \equiv Q, \text{ if } P =_\alpha Q & P \mid \mathbf{0} \equiv P & \nu x.\mathsf{Stop} \equiv \mathsf{Stop} \\
P_1 \mid (P_2 \mid P_3) \equiv (P_1 \mid P_2) \mid P_3 & P \mid Q \equiv Q \mid P & \nu z.\nu w.P \equiv \nu w.\nu z.P \\
\nu z.(P_1 \mid P_2) \equiv P_1 \mid \nu z.P_2, \text{ if } z \notin \mathsf{fn}(P_1) & \nu z.\mathbf{0} \equiv \mathbf{0} & !\,P \equiv P \mid !\,P
\end{array}
$$

Processes $x(y).\mathbf{0}$ and $\overline{x}y.\mathbf{0}$ are abbreviated as $x(y)$ and $\overline{x}y$. Instead of $\nu x_1.\nu x_2.\ldots.\nu x_n.P$ we write $\nu x_1,\ldots,x_n.P$, or also $\nu\mathcal{X}.P$ if the concrete names $x_1,\ldots,x_n$ and the number $n \geq 0$ are not of interest. We also use set-notation for $\mathcal{X}$ and e.g. write $x_i \in \mathcal{X}$ with its obvious meaning. With $\mathtt{choice}(P,Q)$ we abbreviate the internal choice of two processes $\mathtt{choice}(P,Q) := \nu x,y.(x(y_1).P \mid x(y_2).Q \mid \overline{x}y)$ (where $x,y,y_1,y_2 \notin \mathsf{fn}(P \mid Q)$).

The main reduction rule of Stop expresses a synchronous communication between two process, i.e. the reduction rule $x(y).P \mid \overline{x}v.Q \xrightarrow{ia} P[v/y] \mid Q$ performs *interaction* between two processes. The standard reduction of Stop is the closure of the rule $\xrightarrow{ia}$ w.r.t. structural congruence and reduction contexts. We prefer to use reduction contexts instead of closing reduction by congruence rules for the reduction as done e.g. in [20]. However, this leads to the same notion of a standard reduction. Note that there are also approaches [19] to make even the structural transformations of $\equiv$ more deterministic (by using reduction rules), however, for our goals in this paper this does not seem to be helpful.

▶ **Definition 2.1.** *Reduction contexts* $\mathcal{D}$ are $D \in \mathcal{D} ::= [\cdot] \mid D \mid P \mid P \mid D \mid \nu x.D$. A *standard reduction* $\xrightarrow{sr}$ applies an $\xrightarrow{ia}$-reduction in a reduction context (modulo structural congruence):

$$\frac{P \equiv D[P'], P' \xrightarrow{ia} Q', D[Q'] \equiv Q, \text{ and } D \in \mathcal{D}}{P \xrightarrow{sr} Q}$$

The *redex* of an $\xrightarrow{sr}$-reduction is the subprocess $x(y).P \mid \overline{x}v.Q$ which is replaced by $P[v/y] \mid Q$. We define $\xrightarrow{sr,*} := \bigcup_{i\geq 0} \xrightarrow{sr,i}$ and $\xrightarrow{sr,+} := \bigcup_{i>0} \xrightarrow{sr,i}$ where for $P,Q \in Proc_{\mathsf{Stop}}$: $P \xrightarrow{sr,0} P$ and $P \xrightarrow{sr,i} Q$ if there exists $P' \in Proc_{\mathsf{Stop}}$ s.t. $P \xrightarrow{sr} P'$ and $P' \xrightarrow{sr,i-1} Q$.

We define contextual equivalence by observing whether a process may- or should become successful, i.e. may-observation means that the process can be reduced to a successful process, and should-observation means that the process never looses the ability to become successful.

▶ **Definition 2.2.** A process P is *successful* (denoted by $\mathsf{stop}(P)$) if $P \equiv \mathsf{Stop} \mid P'$ for some process P'. *May-convergence* $\downarrow$ is defined as $P\!\downarrow$ iff $\exists P' : P \xrightarrow{sr,*} P' \wedge \mathsf{stop}(P')$ and

should-convergence $\Downarrow$ is defined as: $P\Downarrow$ iff $\forall P' : P \xrightarrow{sr,*} P' \implies P'\downarrow$. We write $P\uparrow$ (P is *may-divergent*) iff $P\Downarrow$ does not hold, and $P\Uparrow$ (P is *must-divergent*) iff $P\downarrow$ does not hold.

For $\xi \in \{\downarrow, \Downarrow, \uparrow, \Uparrow\}$, the *preservation preorders* $\leq_\xi$ are defined as $P \leq_\xi Q$ iff $P\xi \implies Q\xi$. *Contextual preorder* $\leq_c$ is defined as $\leq_c := \leq_{c,\downarrow} \cap \leq_{c,\Downarrow}$ where for $\xi \in \{\downarrow, \Downarrow, \uparrow, \Uparrow\}$: $P \leq_{c,\xi} Q$ iff $\forall C \in \mathcal{C}_{\mathsf{Stop}} : C[P] \leq_\xi C[Q]$. *Contextual equivalence* $\sim_c$ is defined as $\sim_c := \leq_c \cap \geq_c$.

For $\xi \in \{\downarrow, \Downarrow, \uparrow, \Uparrow\}$ we write $\geq_{c,\xi}$ for $(\leq_{c,\xi})^{-1}$ and $\sim_{c,\xi}$ for the intersection $\leq_{c,\xi} \cap \geq_{c,\xi}$.

Note that $P\uparrow$ is equivalent to $\exists P' : P \xrightarrow{sr,*} P' \wedge P'\Uparrow$ and note also that for any successful process P, any contractum P' is also successful, i.e. $\mathsf{stop}(P) \wedge P \xrightarrow{sr,*} P' \implies \mathsf{stop}(P')$.

Since reduction includes transforming processes using structural congruence, structural congruent processes are contextually equivalent:

▶ **Proposition 2.3.** *If $P \equiv Q$, then for $\xi \in \{\downarrow, \Downarrow, \uparrow, \Uparrow\}$: $P \leq_\xi Q$, $P \leq_{c,\xi} Q$, $Q \leq_\xi P$, $Q \leq_{c,\xi} P$ and thus in particular $P \sim_c Q$.*

Some easier properties are:

▶ **Lemma 2.4.** ***1.*** *If $P \xrightarrow{sr} Q$ then $\nu x.P \xrightarrow{sr} \nu x.Q$.*
2. *If $\nu x.P \xrightarrow{sr} Q$ then $P \xrightarrow{sr} Q'$ such that either $Q \equiv \nu x.Q'$ or $Q \equiv Q'$.*
3. *For $\xi \in \{\downarrow, \Downarrow, \uparrow, \Uparrow\}$: $P \leq_\xi \nu x.P$ and $\nu x.P \leq_\xi P$.*

3 Proof Methods for Contextual Equivalence

For disproving an equation $P \sim_c Q$, it suffices to find a distinguishing context. Proving an equation $P \sim_c Q$ is in general harder, since all contexts must be considered. Hence, we develop proof tools supporting those proofs. In Sect. 3.1 we show that a context lemma holds, which restricts the set of contexts that need to be taken into account for proving $P \sim_c Q$. In contrast to the co-inductive proofs given in [14, 20] for a context lemma for barbed congruence in the π-calculus, our context lemma is proved inductively and also for should-convergence. In Sect. 3.2 we show soundness of applicative similarities which permit co-inductive proofs by analyzing the input or output possibilities on open channels of P and Q. The applicative similarities are related to the "weak early bisimilarities" in the π-calculus, but there are some differences which are discussed after the definitions.

3.1 A Context Lemma for May- and Should-Convergence

As a preparation we show two extension lemmas for $\leq_\downarrow$ and $\leq_\uparrow$ w.r.t. contexts of hole depth 1, which are the contexts $[\cdot] \mid R$, $R \mid [\cdot]$, $x(y).[\cdot]$, $\overline{x}y.[\cdot]$, $\nu x.[\cdot]$, and $!\,[\cdot]$. To ease reading the proofs are given in the appendix.

▶ **Lemma 3.1.** *Let $P, Q \in Proc_{\mathsf{Stop}}$. If $\sigma(P) \mid R \leq_\downarrow \sigma(Q) \mid R$ for all $\sigma \in \Sigma$ and $R \in Proc_{\mathsf{Stop}}$, then $\sigma(C[P]) \mid R \leq_\downarrow \sigma(C[Q]) \mid R$ for all $C \in \mathcal{C}_{\mathsf{Stop}}$ of hole depth 1, all $\sigma \in \Sigma$ and $R \in Proc_{\mathsf{Stop}}$.*

▶ **Lemma 3.2.** *Let $P, Q \in Proc_{\mathsf{Stop}}$. Assume that $\sigma(C[Q]) \mid R \leq_\downarrow \sigma(C[P]) \mid R$ for all $\sigma \in \Sigma$, all $C \in \mathcal{C}_{\mathsf{Stop}}$, and all $R \in Proc_{\mathsf{Stop}}$. If $\sigma(P) \mid R \leq_\uparrow \sigma(Q) \mid R$ for all $\sigma \in \Sigma$ and $R \in Proc_{\mathsf{Stop}}$, then $\sigma(C[P]) \mid R \leq_\uparrow \sigma(C[Q]) \mid R$ for all $C \in \mathcal{C}_{\mathsf{Stop}}$ of hole depth 1, all $\sigma \in \Sigma$ and $R \in Proc_{\mathsf{Stop}}$.*

▶ **Theorem 3.3** (Context Lemma)**.** *For all processes P, Q:*
- *If for all σ, R: $\sigma(P) \mid R \leq_\downarrow \sigma(Q) \mid R$, then $P \leq_{c,\downarrow} Q$.*
- *If for all σ, R: $\sigma(P) \mid R \leq_\downarrow \sigma(Q) \mid R \wedge \sigma(P) \mid R \leq_\Downarrow \sigma(Q) \mid R$, then $P \leq_c Q$.*

Proof. For the first part it suffices to show that $\sigma(C[P]) \mid R \leq_{\downarrow} \sigma(C[Q]) \mid R$ for all $C \in \mathcal{C}_{\mathsf{Stop}}$, $\sigma \in \Sigma$, and $R \in Proc_{\mathsf{Stop}}$, which follows from Lemma 3.1 by induction on the depth of the hole of the context C. For the second part we use the fact that $\sigma(P) \mid R \leq_{\downarrow} \sigma(Q) \mid R$ for all σ, R implies $\sigma(C[P]) \mid R \leq_{\downarrow} \sigma(C[Q]) \mid R$ for all σ, R, C. By induction on the depth of the hole of the context C, the fact that $\sigma(P) \mid R \leq_{\Downarrow} \sigma(Q) \mid R$ is equivalent to $\sigma(Q) \mid R \leq_{\uparrow} \sigma(P) \mid R$, and by Lemma 3.2 it follows $\sigma(C[Q]) \mid R \leq_{\uparrow} \sigma(C[P]) \mid R$ for all C, σ and thus $P \leq_{c,\Downarrow} Q$. ◀

▶ Remark. The condition for all σ, R: $\sigma(P) \mid R \leq_{\Downarrow} \sigma(Q) \mid R$ is in general not sufficient for $P \leq_{c,\Downarrow} Q$. Let $P := \nu x.\overline{x}y \mid x(y).\mathsf{Stop} \mid x(y)$ and $Q := \mathbf{0}$. Then $\sigma(P) \mid R \leq_{\Downarrow} \sigma(Q) \mid R$ for all σ and R, but $P \not\leq_{c,\Downarrow} Q$, since the context $!\,[\cdot]$ distinguishes P and Q: $!\,P\Downarrow$ while $!\,Q\Uparrow$.

3.2 Applicative Similarities

We first provide co-inductive definitions of $\leq_{\downarrow}$ and $\leq_{\uparrow}$, which will ease some of our proofs:

▶ **Definition 3.4.** We define the operators $F_{\downarrow}$ and $F_{\uparrow}$ on binary relation on processes:

- For $\eta \subseteq (Proc_{\mathsf{Stop}} \times Proc_{\mathsf{Stop}})$, $P\ F_{\downarrow}(\eta)\ Q$ holds iff
  1. If P is successful (i.e. $\mathsf{stop}(P)$), then $Q{\downarrow}$.
  2. If $P \xrightarrow{sr} P'$, then there exists Q' such that $Q \xrightarrow{sr,*} Q'$ and $P'\ \eta\ Q'$.
- For $\eta \subseteq (Proc_{\mathsf{Stop}} \times Proc_{\mathsf{Stop}})$, $P\ F_{\uparrow}(\eta)\ Q$ holds iff
  1. If $P\Uparrow$, then $Q\Uparrow$.
  2. If $P \xrightarrow{sr} P'$, then there exists Q' such that $Q \xrightarrow{sr,*} Q'$ and $P'\ \eta\ Q'$.

The relation $\precsim_{\downarrow}$ ($\precsim_{\uparrow}$, resp.) is the greatest fixpoint of the operator $F_{\downarrow}$ ($F_{\uparrow}$, resp.).

For an operator F on binary relations, a relation η is *F-dense*, iff $\eta \subseteq F(\eta)$. The co-induction principle is that an F-dense relation η is contained in the greatest fixpoint of F.

Since$\leq_{\downarrow}$ is $F_{\downarrow}$-dense and $\leq_{\uparrow}$ is $F_{\uparrow}$-dense, the following lemma holds.

▶ **Lemma 3.5.** *$\leq_{\downarrow} = \precsim_{\downarrow}$ and $\leq_{\uparrow} = \precsim_{\uparrow}$.*

Before defining "applicative similarities" for may- and should-convergence, we define the property of a relation to preserve the input and output capabilities of one process w.r.t. another process. This definition is analogous to preserving actions in labeled bisimilarities. We prefer this definition here, since we can omit the definition of a labeled transition system.

▶ **Definition 3.6.** For processes $P, Q \in Proc_{\mathsf{Stop}}$ and a binary relation $\eta \subseteq (Proc_{\mathsf{Stop}} \times Proc_{\mathsf{Stop}})$, we say *$\eta$ preserves the input/output capabilities of P w.r.t. Q* iff:

- *Open input:* If $P \equiv \nu\mathcal{X}.(x(y).P_1 \mid P_2)$ with $x \notin \mathcal{X}$, then for every name $z \in \mathcal{N}$ there exists a process $Q' \in Proc_{\mathsf{Stop}}$ s.t. $Q \xrightarrow{sr,*} Q'$, $Q' \equiv \nu\mathcal{Y}.(x(y).Q_1 \mid Q_2)$ with $x \notin \mathcal{Y}$, and $(\nu\mathcal{X}.(P_1[z/y] \mid P_2))\ \eta\ (\nu\mathcal{Y}.(Q_1[z/y] \mid Q_2))$.
- *Open output:* If $P \equiv \nu\mathcal{X}.(\overline{x}y.P_1 \mid P_2)$ with $x, y \notin \mathcal{X}$, then there exists a process Q' s.t. $Q \xrightarrow{sr,*} Q'$, $Q' \equiv \nu\mathcal{Y}.(\overline{x}y.Q_1 \mid Q_2)$ with $x, y \notin \mathcal{Y}$, and $(\nu\mathcal{X}.(P_1 \mid P_2))\ \eta\ (\nu\mathcal{Y}.(Q_1 \mid Q_2))$.
- *Bound output:* If $P \equiv \nu\mathcal{X}, \nu y.(\overline{x}y.P_1 \mid P_2)$ with $x \notin \mathcal{X}$, then there exists Q' s.t. $Q \xrightarrow{sr,*} Q'$, $Q' \equiv \nu\mathcal{Y}, \nu y.(\overline{x}y.Q_1 \mid Q_2)$ with $x \notin \mathcal{Y}$, and $(\nu\mathcal{X}.(P_1 \mid P_2))\ \eta\ (\nu\mathcal{Y}.(Q_1 \mid Q_2))$.

▶ **Definition 3.7** ((Full) Applicative Similarities)**.** We define the operators $F_{b,\downarrow}$ and $F_{b,\uparrow}$ on binary relations on processes, where *applicative $\downarrow$-similarity* $\precsim_{b,\downarrow}$ is the greatest fixpoint of $F_{b,\downarrow}$ and *applicative $\uparrow$-similarity* $\precsim_{b,\uparrow}$ is the greatest fixpoint of the operator $F_{b,\uparrow}$.

- For $\eta \subseteq (Proc_{\mathsf{Stop}} \times Proc_{\mathsf{Stop}})$, $P\ F_{b,\downarrow}(\eta)\ Q$ holds iff
  1. If P is successful (i.e. $\mathsf{stop}(P)$), then $Q{\downarrow}$.
  2. If $P \xrightarrow{sr} P'$, then $\exists Q'$ with $Q \xrightarrow{sr,*} Q'$ and $P'\ \eta\ Q'$.
  3. If P is not successful, then η preserves the input/output capabilities of P w.r.t. Q.

- For $\eta \subseteq (Proc_{\mathsf{Stop}} \times Proc_{\mathsf{Stop}})$, $P\ F_{b,\uparrow}(\eta)\ Q$ holds iff
  1. If $P\Uparrow$, then $Q\uparrow$.
  2. If $P \xrightarrow{sr} P'$, then $\exists Q'$ with $Q \xrightarrow{sr,*} Q'$ and $P'\ \eta\ Q'$.
  3. If $\neg P\Uparrow$, then η preserves the input/output capabilities of P w.r.t. Q.
  4. $Q \precsim_{b,\downarrow} P$.

Full applicative $\downarrow$-similarity $\precsim^{\sigma}_{b,\downarrow}$ and *full applicative $\uparrow$-similarity* $\precsim^{\sigma}_{b,\uparrow}$ are defined as $P \precsim^{\sigma}_{b,\downarrow} Q$ ($P \precsim^{\sigma}_{b,\uparrow} Q$, resp.) iff $\sigma(P) \precsim_{b,\downarrow} \sigma(Q)$ ($\sigma(P) \precsim_{b,\uparrow} \sigma(Q)$, resp.) for all $\sigma \in \Sigma$. *Full applicative similarity* $\precsim_b$ is defined as the intersection $\precsim_b := \precsim^{\sigma}_{b,\downarrow} \cap (\precsim^{\sigma}_{b,\uparrow})^{-1}$. *Mutual full $\Downarrow$-applicative similarity* $\simeq_{b,\Downarrow}$ is the intersection $\precsim^{\sigma}_{b,\uparrow} \cap (\precsim^{\sigma}_{b,\uparrow})^{-1}$ and *mutual full applicative similarity* $\simeq_b$ is the intersection $\simeq_b := \precsim_b \cap (\precsim_b)^{-1}$.

We discuss our definitions of applicative similarity. We first consider $\precsim_{b,\downarrow}$. Its definition is related to early labeled bisimilarity for the π-calculus [20], but adapted to the successfulness-test. However, there is a difference whether a similarity or a bisimilarity is used. Applicative $\downarrow$-bisimilarity would be defined as the largest relation $\mathcal{R}$ such that $\mathcal{R}$ and $\mathcal{R}^{-1}$ are $F_{b,\downarrow}$-dense. The relation $\precsim_{b,\downarrow} \cap (\precsim_{b,\downarrow})^{-1}$ is much coarser than applicative $\downarrow$-bisimilarity. For instance, the processes $P_{a,bc} := \mathtt{choice}(a(u_1), \mathtt{choice}(b(u_2), c(u_3)))$ and $P_{ab,c} := \mathtt{choice}(\mathtt{choice}(a(u_1), b(u_2)), c(u_3))$ are not applicative $\downarrow$-bisimilar, since after reducing $P_{a,bc} \xrightarrow{sr} P_0 \equiv \nu x, y.(x(y_1).a(u_1) \mid \mathtt{choice}(b(u_2), c(u_3)))$ there is no process P_1 with $P_{ab,c} \xrightarrow{sr,*} P_1$ s.t. P_0 and P' are applicative $\downarrow$-bisimilar. However, $P_{a,bc} \precsim_{b,\downarrow} P_{ab,c}$ and $P_{ab,c} \precsim_{b,\downarrow} P_{a,bc}$. The following example (adapted from an example in [20]) shows that even $\precsim_{b,\downarrow}$ is more discriminating than contextual may preorder:

▶ **Proposition 3.8.** *Let $S_{xy} := x(z).\overline{y}z$ and $S_{yx} := y(z).\overline{x}z$. For $P := \overline{a}x \mid\!\mid S_{xy} \mid\!\mid S_{yx}$ and $Q := \overline{a}y \mid\!\mid S_{xy} \mid\!\mid S_{yx}$, it holds: $\neg(P \precsim_{b,\downarrow} Q)$ (and thus also $\neg(P \precsim^{\sigma}_{b,\downarrow} Q)$), but $P \leq_{c,\downarrow} Q$.*

Proof. $P \precsim_{b,\downarrow} Q$ does not hold, since the output on channel a is different. $P \leq_{c,\downarrow} Q$ is proved in the appendix in Lemma A.1. ◀

The definition of applicative $\uparrow$-similarity includes the property $Q \precsim_{b,\downarrow} P$, i.e.:

▶ **Proposition 3.9.** $P \precsim_{b,\uparrow} Q \implies Q \precsim_{b,\downarrow} P$.

Thus – like the discussion before on bisimilarity – this requirement makes the relation $\precsim_{b,\uparrow}$ very fine-grained: the processes $P_{a,bc}$ and $P_{ab,c}$ are not applicative $\uparrow$-similar, although the processes are contextually equivalent. The reason for our choice of this definition is that we did not find a coarser $\uparrow$-similarity which is sound for contextual should-preorder. Properties that must hold for such a definition are that it preserves may-divergence w.r.t. Stop, i.e. $\uparrow$, but also (due to Theorem 5.5, see below) that it preserves the may-divergence w.r.t. barbs. The second condition holds for $\precsim_{b,\uparrow}$, since we added $Q \precsim_{b,\downarrow} P$ in Definition 3.7. Obviously, $\precsim_{b,\downarrow}$ preserves may-convergence and $\precsim_{b,\uparrow}$ preserves may-divergence:

▶ **Lemma 3.10.** $\precsim_{b,\downarrow} \subseteq \precsim_{\downarrow}$ *and* $\precsim_{b,\uparrow} \subseteq \precsim_{\uparrow}$.

We now show soundness of our applicative similarities.

▶ **Proposition 3.11.** *For all $P, Q, R \in Proc_{\mathsf{Stop}}$ and all $\mathcal{X}$, the following implications hold:*
1. $(P \precsim_{b,\downarrow} Q) \implies \nu\mathcal{X}.(P \mid R) \precsim_{\downarrow} \nu\mathcal{X}.(Q \mid R)$.
2. $(P \precsim_{b,\uparrow} Q) \implies \nu\mathcal{X}.(P \mid R) \precsim_{\uparrow} \nu\mathcal{X}.(Q \mid R)$.

Proof. The relation $\precsim_{\downarrow} \cup \{(\nu\mathcal{X}.(P \mid R), \nu\mathcal{X}.(Q \mid R)) \mid P \precsim_{b,\downarrow} Q, \text{ for any } \mathcal{X}, R\}$ is $F_{\downarrow}$-dense (proved in the appendix, Lemma A.2) and thus the first part holds. The second part holds, since the relation $\precsim_{\uparrow} \cup \{(\nu\mathcal{X}.(P \mid R), \nu\mathcal{X}.(Q \mid R)) \mid P \precsim_{b,\uparrow} Q, \text{ for any } \mathcal{X}, R\}$ is $F_{\uparrow}$-dense, which is proved in the appendix, Lemma A.3. ◀

▶ **Theorem 3.12** (Soundness of Full Applicative Similarities). *The following inclusions hold:*

1. $\precsim^{\sigma}_{b,\downarrow} \subseteq \leq_{c,\downarrow}$,
2. $\precsim_{b} \subseteq \leq_{c}$, *and*
3. $\simeq_{b,\Downarrow} = \simeq_{b} \subseteq \sim_{c}$.

Proof. For the first part Proposition 3.11 part (1) shows that $\sigma(P) \precsim_{b,\downarrow} \sigma(Q)$ implies $\sigma(P) \mid R \precsim_{\downarrow} \sigma(Q) \mid R$ for all σ, R. Thus, $P \precsim^{\sigma}_{b,\downarrow} Q$ implies $\sigma'(P) \mid R \precsim_{\downarrow} \sigma'(Q) \mid R$ for all σ', R. Since $\precsim_{\downarrow} = \leq_{\downarrow}$ (Lemma 3.5) the context lemma (Theorem 3.3) shows $P \leq_{c,\downarrow} Q$.

For the second part we apply both parts of Proposition 3.11 which shows that $P \precsim^{\sigma}_{b,\downarrow} Q$ and $Q \precsim^{\sigma}_{b,\uparrow} P$ imply that $\sigma'(P) \mid R \precsim_{\downarrow} \sigma'(Q) \mid R$ and $\sigma'(Q) \mid R \precsim_{\uparrow} \sigma'(P) \mid R$ for all σ', R. Since $\precsim_{\downarrow} = \leq_{\downarrow}$ and $\precsim_{\uparrow} = \geq_{\Downarrow}$ (Lemma 3.5), Theorem 3.3 shows $P \leq_{c} Q$.

The equation of the last part follows from Proposition 3.9. The inclusion of the last part follows from the second part and the definitions of $\simeq_b$ and $\sim_c$. ◀

4 Equivalences and the Contextual Ordering

In this section we analyze the contextual ordering and also show some contextual equivalences.

4.1 Correctness of Deterministic Interaction

We demonstrate our developed techniques for an exemplary program optimization and apply Theorem 3.12 to show correctness of a restricted variant of the *ia*-reduction that ensures determinism. Moreover, the result can be used to show a completeness result w.r.t. the tests in the context lemma (Corollary 4.3).

▶ **Theorem 4.1** (Correctness of Deterministic Interaction). *For all processes P, Q the equation $\nu x.(x(y).P \mid \overline{x}z.Q)) \sim_c \nu x.(P[z/y] \mid Q)$ holds.*

Proof. We use Theorem 3.12 and show that $\nu x.(x(y).P \mid \overline{x}z.Q)) \simeq_{b,\Downarrow} \nu x.(P[z/y] \mid Q)$.

Let $\mathcal{S} := \{(\sigma(\nu x.(x(y).P \mid \overline{x}z.Q)), \sigma(\nu x.(P[z/y] \mid Q))) \mid \text{for all } x, y, z, P, Q, \sigma\} \cup \equiv$. We show that $\mathcal{S}$ and $\mathcal{S}^{-1}$ are $F_{b,\downarrow}$-dense and $F_{b,\uparrow}$-dense.

Let $(R_1, R_2) = (\sigma(\nu x.(x(y).P \mid \overline{x}z.Q)), \sigma(\nu x.(P[z/y] \mid Q)))$. Then $(R_1, R_2) \in F_{b,\downarrow}(\mathcal{S})$:

1. R_1 is not successful, so there is nothing to show.
2. If $R_1 \xrightarrow{sr} R_1'$, then $R_1' \equiv R_2$ and $(R_2, R_2) \in \mathcal{S}$.
3. R_1 does not have an open input or output, thus there is nothing to show.

Also $(R_2, R_1) \in F_{b,\downarrow}(\mathcal{S}^{-1})$:

1. If R_2 is successful, then $R_1\downarrow$, since $R_1 \xrightarrow{sr} R_2$.
2. If $R_2 \xrightarrow{sr} R_2'$, then $R_1 \xrightarrow{sr,2} R_2'$ and $(R_2', R_2') \in \mathcal{S}^{-1}$.
3. If R_2 has an open input or output, then $R_1 \xrightarrow{sr} R_2$ and the condition of $F_{b,\downarrow}$ can be fulfilled.

Thus $\mathcal{S}$ and $\mathcal{S}^{-1}$ are $F_{b,\downarrow}$-dense, and thus $R_1 \precsim_{b,\downarrow} R_2$ and $R_2 \precsim_{b,\downarrow} R_1$ for any $(R_1, R_2) \in \mathcal{S}$.

Now we show that $(R_1, R_2) \in F_{b,\uparrow}(\mathcal{S})$:

1. If $R_1\Uparrow$ then $R_2\Uparrow$, since $R_1 \xrightarrow{sr} R_2$.
2. If $R_1 \xrightarrow{sr} R_1'$, then $R_1' \equiv R_2$ (since there is only one reduction possibility for R_1) and $(R_1', R_1') \in \mathcal{S}$.

3. R_1 does not have an open input or output, thus there is nothing to show.
4. $R_2 \precsim_{b,\downarrow} R_1$ is already proved.

Finally, also $(R_2, R_1) \in F_{b,\uparrow}(\mathcal{S}^{-1})$:

1. If $R_2\Uparrow$ then clearly $R_1\uparrow$.
2. If $R_2 \xrightarrow{sr} R_2'$, then $R_1 \xrightarrow{sr,2} R_2'$ and $(R_2', R_2') \in \mathcal{S}^{-1}$.
3. If R_2 has an open input or output, then $R_1 \xrightarrow{sr} R_2$ and the condition of $F_{b,\uparrow}$ can be fulfilled.
4. $R_1 \precsim_{b,\downarrow} R_2$ is already proved.

Thus $\mathcal{S}$ and $\mathcal{S}^{-1}$ are $F_{b,\uparrow}$-dense and $R_1 \precsim_{b,\uparrow} R_2$ and $R_2 \precsim_{b,\uparrow} R_1$ for all $(R_1, R_2) \in \mathcal{S}$ and thus Theorem 3.12 shows the claim. ◀

Contextual preorder does not change, if we additionally consider all name substitutions:

▶ **Lemma 4.2.** *For $\xi \in \{\downarrow, \Downarrow\}$: $P \leq_{c,\xi} Q$ iff $\forall C \in \mathcal{C}_{\mathsf{Stop}}$, $\sigma \in \Sigma$: $C[\sigma(P)] \leq_\xi C[\sigma(Q)]$.*

Proof. "$\Leftarrow$" is trivial. For "$\Rightarrow$" we define for $\sigma = \{x_1 \mapsto y_1, \ldots, x_n \mapsto y_n\}$ the context $C_\sigma := \nu\mathcal{W}.(w_1(x_1).w_2(x_2).\ldots.w_n(x_n).[\cdot] \mid \overline{w_1}y_1 \mid \ldots \mid \overline{w_n}y_n)$ where $\mathcal{W} = \{w_1, \ldots, w_n\}$ and $\mathcal{W} \cap (\mathsf{fn}(P) \cup \mathsf{fn}(Q)) = \emptyset$. The reductions $C_\sigma[P] \xrightarrow{ia,*} \sigma(P)$ and $C_\sigma[Q] \xrightarrow{ia,*} \sigma(Q)$ are valid, where all *ia*-steps are deterministic and thus by Theorem 4.1 $C_\sigma[P] \sim_c \sigma(P)$ and $C_\sigma[Q] \sim_c \sigma(Q)$. Now let $P \leq_{c,\xi} Q$ and let C, σ s.t. $C[\sigma(P)]\xi$. Since $\sigma(P) \sim_c C_\sigma[P]$ also $C[C_\sigma(P)]\xi$ which in turn implies $C[C_\sigma(Q)]\xi$. Since $C_\sigma[Q] \sim_c \sigma(Q)$, this shows $C[\sigma(Q)]\xi$. Since C, σ were chosen arbitrarily, $C[\sigma(P)] \leq_\xi C[\sigma(Q)]$ holds for all $C \in \mathcal{C}_{\mathsf{Stop}}$ and $\sigma \in \Sigma$. ◀

Thus, the tests of the context lemma (Theorem 3.3) are complete w.r.t. $\leq_c$:

▶ **Corollary 4.3.** *For all $P, Q \in Proc_{\mathsf{Stop}}$:*

- *$P \leq_{c,\downarrow} Q$ iff for all $\sigma \in \Sigma, R \in Proc_{\mathsf{Stop}}$: $\sigma(P) \mid R \leq_\downarrow \sigma(Q) \mid R$.*
- *$P \leq_c Q$ iff for all $\sigma \in \Sigma, R \in Proc_{\mathsf{Stop}}, \xi \in \{\downarrow, \Downarrow\}$: $\sigma(P) \mid R \leq_\xi \sigma(Q) \mid R$.*

4.2 Results on the Contextual Ordering

We show several properties on the contextual ordering and equivalence. All successful processes are in the same equivalence class. More surprisingly, all may-convergent processes are equal w.r.t. contextual may-convergence, which is a strong motivation to also consider should-convergence. Further results are that Stop is the largest element in the contextual ordering, and there is no least element:

▶ **Theorem 4.4.**

1. *If P, Q are two successful processes, then $P \sim_c Q$.*
2. *If P, Q are two processes with $P\downarrow, Q\downarrow$, then $P \sim_{c,\downarrow} Q$.*
3. *There are may-convergent processes P, Q with $P \not\sim_c Q$.*
4. *Stop is the greatest process w.r.t. $\leq_c$.*
5. *$\mathbf{0}$ is the smallest process w.r.t. $\leq_{c,\downarrow}$.*
6. *There is no smallest process w.r.t. $\leq_c$.*

Proof. For (1) let P and Q be successful. Then for any $\sigma \in \Sigma$ and any $R \in Proc_{\mathsf{Stop}}$ also $\sigma(P) \mid R$ and $\sigma(Q) \mid R$ are successful. This implies $\sigma(P) \mid R\downarrow$, $\sigma(Q) \mid R\downarrow$, $\sigma(P) \mid R\Downarrow$, and $\sigma(Q) \mid R\Downarrow$ for all $R \in Proc_{\mathsf{Stop}}$ and $\sigma \in \Sigma$ and thus Theorem 3.3 shows $P \sim_c Q$.

Since $P\downarrow \implies \sigma(P) \mid R\downarrow$ for any process P, R and $\sigma \in \Sigma$, Theorem 3.3 shows part (2).

For (3) the empty context distinguishes $\mathtt{choice}(\mathsf{Stop}, \mathbf{0})$ and Stop: $\mathtt{choice}(\mathsf{Stop}, \mathbf{0})\downarrow$, and $\mathtt{choice}(\mathsf{Stop}, \mathbf{0})\uparrow$, while Stop$\Downarrow$, hence $\mathtt{choice}(\mathsf{Stop}, \mathbf{0}) \not\sim_c \mathsf{Stop}$.

For part (4) clearly $\mathsf{Stop} \mid R\Downarrow$ for all R. Since $\mathsf{Stop} \mid R\Downarrow \implies \mathsf{Stop} \mid R\downarrow$, we have $\sigma(P) \mid R \leq_{\downarrow} \mathsf{Stop} \mid R$ and $\sigma(P) \mid R \leq_{\Downarrow} \mathsf{Stop} \mid R$ for any P, σ, and R. Thus Theorem 3.3 shows $P \leq_c \mathsf{Stop}$ for any process P.

Part (5) follows from Theorem 3.12, since $\{(\mathbf{0}, P) \mid P \in Proc_{\mathsf{Stop}}\}$ is $F_{b,\downarrow}$-dense.

For (6) assume that there is a process P_0 that is the smallest one, i.e. $P_0 \leq_c P$ for all processes P. Then $P_0\Uparrow$, since $P_0 \leq_c \mathbf{0}$. Let $P_0 \xrightarrow{*} P_1$, such that $P_1 = D[x(y).P_3]$, and where x is free. With $D_1 = \overline{x}y.\mathsf{Stop}$ we obtain $D_1[P_1]\downarrow$, but $D_1[\mathbf{0}] \equiv \overline{x}y.\mathsf{Stop}\Uparrow$. We argue similarly for outputs. Thus the reducts of P_0 do not have open outputs. Now let $P = x(y).\mathbf{0}$, where by our assumption $P_0 \leq_{c,\Downarrow} P$ holds. Let $D = [\cdot] \mid \overline{x}y.\mathbf{0} \mid x(y).\mathsf{Stop}$. Then $D[P_0]\Downarrow$, since there is no communication between the reducts of P_0 and D, but D can always be reduced to a successful process. Now consider $D[P]$. It is $D[P] \to \mathbf{0} \mid x(y).\mathsf{Stop}$, which is must-divergent, hence we have reached the contradiction $P_0 \not\leq_{c,\Downarrow} P$. ◀

We show that it is suffices to test should-convergence in all contexts, since all tests for may-convergence can be encoded:

▶ **Theorem 4.5.**

1. $\leq_{c,\Downarrow} = \leq_c$,
2. $\leq_c \neq \leq_{c,\downarrow}$,
3. *and* $\leq_{c,\Downarrow} \not\subseteq \sim_{c,\downarrow}$.

Proof. For part (1), we show that $\leq_{c,\Downarrow} \subseteq \leq_{c,\downarrow}$: let $C_{x,y,\mathcal{X}} := !\,\nu x, y, \nu\mathcal{X}.[\cdot]$. For any process P with $x, y \notin \mathsf{fn}(P)$ and $\mathcal{X} \supseteq \mathsf{fn}(P)$ one can verify that $P\downarrow$ iff $C_{x,y,\mathcal{X}}[P]\Downarrow$: If $P\downarrow$, then $P' := \nu x, y.\nu\mathcal{X}.P\downarrow$ by Lemma 2.4 and for $!\,P'$ we can generate a parallel copy of P', and thus $C_{x,y,\mathcal{X}}[P]\Downarrow$. If $C_{x,y,\mathcal{X}}[P]\Downarrow$, then $\nu x, y, \mathcal{X}.P\downarrow$, since parallel copies of $\nu x, y, \mathcal{X}.P$ cannot communicate due to the name restriction. Lemma 2.4 shows $P\downarrow$. Now let $P \leq_{c,\Downarrow} Q$, $C[P]\downarrow$, but $C[Q]\Uparrow$. With fresh names x, y, $\mathcal{X} = \mathsf{fn}(P) \cup \mathsf{fn}(Q)$: $C_{x,y}[C[P]]\Downarrow$ but $C_{x,y}[C[Q]]\uparrow$ which contradicts $P \leq_{c,\Downarrow} Q$.

The inequality of part (2) follows from Theorem 4.4 items (5), (6).

For part (3), clearly, $\mathbf{0} \leq_c \mathsf{Stop}$, since Stop is a largest element of $\leq_c$, but $\mathbf{0}\Uparrow$ while $\mathsf{Stop}\downarrow$, and thus in Π_{Stop} contextual should-preorder does not imply contextual may-equivalence. ◀

We conclude this subsection, by analyzing several equations, including the ones from [4].

▶ **Theorem 4.6.** *For all processes P, Q, the following equivalences hold:*

1. $!\,P \sim_c !\,!\,P$.
2. $!\,P \mid !\,P \sim_c !\,P$.
3. $!\,(P \mid Q) \sim_c !\,P \mid !\,Q$.
4. $!\,\mathbf{0} \sim_c \mathbf{0}$.
5. $!\,\mathsf{Stop} \sim_c \mathsf{Stop}$.
6. $!\,(P \mid Q) \sim_c !\,(P \mid Q) \mid P$.
7. $x(y).\nu z.P \sim_c \nu z.x(y).P$ *if* $z \notin \{x, y\}$.
8. $\overline{x}y.\nu z.P \sim_c \nu z.\overline{x}y.P$ *if* $z \notin \{x, y\}$.

Proof. This holds, since $\mathcal{S}_i \cup \precsim_{b,\uparrow}$ and $\mathcal{S}_i^{-1} \cup \precsim_{b,\uparrow}$ are $F_{b,\uparrow}$-dense, where $\mathcal{S}_i := \{(R \mid l_i, R \mid r_i) \mid$ for all $R\}$, and l_i, r_i are the left and right hand side of the i^{th} equation. ◀

5 Results for the Stop-free Calculus

In this section we consider the π-calculus Π without the constant Stop but with barbed may- and should-testing as notion of process equivalence. We will show a strong connection between Π_{Stop} and Π which makes a lot of results transferable.

▶ **Definition 5.1.** Let Π be the subcalculus of Π_{Stop} that does not have the constant Stop as a syntactic construct. Processes, contexts, reduction, structural congruences are accordingly adapted for Π. We write *Proc* for the set of processes of Π and $\mathcal{C}$ for the set of contexts of Π.

We define the notion of a barb, i.e. that a process can receive a name on an open channel[1]. Barbed may- and should-testing is defined analogously to contextual equivalence, where the observation of success is replaced by observing barbs:

▶ **Definition 5.2.** Let $P \in \mathit{Proc}$ and $x \in \mathcal{N}$. A process P *has a barb on input* x (written as $P\!\uparrow^x$) iff $P \equiv \nu\mathcal{X}.(x(y).P' \mid P'')$ where $x \notin \mathcal{X}$. We write $P\downarrow_x$ iff there exists P' s.t. $P \xrightarrow{sr,*} P'$ and $P'\!\uparrow^x$. We write $P \Downarrow_x$ iff for all P' with $P \xrightarrow{sr,*} P'$ also $P'\downarrow_x$ holds. We write $P \Uparrow_x$ iff $P\downarrow_x$ does not hold, and we write $P\uparrow_x$ iff $P \Downarrow_x$ does not hold.

For a name $x \in \mathcal{N}$, *barbed may- and should-testing preorder* $\leq_{c,barb}$ and *barbed may- and should-testing equivalence* $\sim_{c,barb}$ are defined as $\leq_{c,barb} := \leq_{c,\downarrow_x} \cap \leq_{c,\Downarrow_x}$ and $\sim_{c,barb} := \leq_{c,barb} \cap (\leq_{c,barb})^{-1}$ where for $\xi \in \{\downarrow_x, \Downarrow_x, \uparrow_x, \Uparrow_x\}$ and $P, Q \in \mathit{Proc}$ the inequality $P \leq_{c,\xi} Q$ holds iff for all contexts $C \in \mathcal{C} : C[P]\xi \implies C[Q]\xi$.

In difference to observing success, the barb behavior is not stable under reduction, e.g. for the process $P = x(z) \mid \overline{x}y$, $P\!\uparrow^x$ holds, but $P \xrightarrow{sr} \mathbf{0}$ and $\mathbf{0}\Uparrow_x$. We show that in $\sim_{c,barb}$ the concrete name x is irrelevant:

▶ **Proposition 5.3.** *For all* $x, y \in \mathcal{N}$: $\leq_{c,\downarrow_x} = \leq_{c,\downarrow_y}$ *and* $\leq_{c,\Downarrow_x} = \leq_{c,\Downarrow_y}$.

Proof. First assume $P \leq_{c,\downarrow_x} Q$, $C[P]\!\downarrow_y$, but $C[Q]\Uparrow_y$. Let $C' = \overline{y}w.x(w').\mathbf{0} \mid \nu\mathcal{X}.([\cdot])$ where $\mathcal{X} = (\mathsf{fn}(C[P]) \cup \mathsf{fn}(C[Q])) \setminus \{y\}$. From $C[P]\!\downarrow_y$ also $C'[C[P]]\!\downarrow_x$ follows. Hence $C'[C[Q]]\!\downarrow_x$ holds, too. But, the structure of C' shows that this is only possible if $C[Q] \xrightarrow{sr,*} Q'$ with $Q'\!\uparrow^y$ and thus $C[Q]\Uparrow_y$ cannot hold. Now assume $P \leq_{c,\Downarrow_x} Q$ and $C[P]\!\Downarrow_y$, but $C[Q]\!\uparrow_y$. Clearly, $C'[C[P]]\!\Downarrow_x$ holds. The assumption $C[Q]\!\uparrow_y$ implies that $C[Q] \xrightarrow{sr,*} Q'$ with $Q'\Uparrow_y$. Then also $C'[Q']\Uparrow_x$, and since $C'[C[Q]] \xrightarrow{sr,*} C'[Q']$ we also have $C'[C[Q]]\!\uparrow_x$. This is a contradiction, since $P \leq_{c,\Downarrow_x} Q$ implies $C'[C[P]]\!\Downarrow_x \implies C'[C[Q]]\!\Downarrow_x$. ◀

▶ **Corollary 5.4.** $P \leq_{c,barb} Q$ *iff* $\forall x \in \mathcal{N} : P \leq_{c,\downarrow_x} Q \wedge P \leq_{c,\Downarrow_x} Q$.

We show that contextual equivalence of Π_{Stop} conservatively extends barbed testing equivalence of the π-calculus: $P \sim_{c,barb} Q \implies P \sim_c Q$ for all stop-free P, Q. Moreover, on Stop-free processes[2] P, Q, contextual equivalence is also complete for barbed testing, i.e. $P \sim_c Q \implies P \sim_{c,barb} Q$. Thus the identity translation (see e.g. [22] for properties of translations) from Π into Π_{Stop} is fully-abstract w.r.t. $\sim_{c,barb}$ in Π and $\sim_c$ in Π_{Stop}.

▶ **Theorem 5.5.** *For all processes* $P, Q \in \mathit{Proc}$: $P \leq_{c,barb} Q \iff P \leq_c Q$, *and hence also* $P \sim_{c,barb} Q \iff P \sim_c Q$.

---

[1] We only consider an input capability here, since the barbed may- and should-testing equivalence does not change if also output capabilities are observed.

[2] Stop-*free* means without occurrences of Stop.

Proof. Let P, Q be Stop-free processes. It suffices to show that $P \leq_{c,\downarrow_x} Q$ iff $P \leq_{c,\downarrow} Q$ and $P \leq_{c,\Downarrow_x} Q$ iff $P \leq_{c,\Downarrow} Q$. In the remainder of the proof let $\psi_{u,v}(R)$ be the process (or context) R with every occurrence of Stop replaced by $u(v)$ and let $\psi^{-1}_{u,v}(R)$ be the process (or context) R with every occurrence of the subprocess $u(v)$ be replaced by Stop.

- $P \leq_{c,\downarrow_x} Q \implies P \leq_{c,\downarrow} Q$: Let $P \leq_{c,\downarrow_x} Q$ and $C \in \mathcal{C}$ with $C[P]\downarrow$, i.e. $C[P] = P_0 \xrightarrow{sr} P_1 \ldots \xrightarrow{sr} P_n$ s.t. $\mathsf{stop}(P_n)$. Let u, v be fresh names. Then $\psi_{u,v}(C)[P] = \psi_{u,v}(P_0) \xrightarrow{sr} \psi_{u,v}(P_1) \xrightarrow{sr} \ldots \xrightarrow{sr} \psi_{u,v}(P_n)$, since *ia*-reductions do not use Stop and the axiom $\nu z.\mathsf{Stop} \equiv \mathsf{Stop}$ can be replaced by $\nu z.u(v) \equiv u(v)$ (since u is fresh). Since $P_n \equiv \mathsf{Stop} \mid R$, we have $\psi_{u,v}(P_n) \equiv (u(v) \mid \psi_{u,v}(R))$ and thus $\psi_{u,v}(P_n)\!\Uparrow^u$ and $\psi_{u,v}(C)[P]\!\downarrow_u$. By Proposition 5.3 and $P \leq_{c,\downarrow_x} Q$ we have $P \leq_{c,\downarrow_u} Q$ and thus $\psi_{u,v}(C)[Q]\!\downarrow_u$, i.e. $\psi_{u,v}(C)[Q] \xrightarrow{sr} Q_1 \ldots \xrightarrow{sr} Q_m$ where $Q_m\!\Uparrow^u$, i.e. $Q_m \equiv \nu\mathcal{X}.(u(v).R_1 \mid R_2)$. The reduction cannot perform an *ia*-reduction using the prefix $u(v)$, since u is fresh, and thus $R_1 = \mathbf{0}$. Thus the reduction $C[Q] \xrightarrow{sr} \psi^{-1}_{u,v}(Q_1) \ldots \xrightarrow{sr} \psi^{-1}_{u,v}(Q_m)$ exists, and $\psi^{-1}_{u,v}(Q_m) \equiv \nu x_1, \ldots x_n.(\mathsf{Stop} \mid \psi^{-1}_{u,v}(R_2))$ and thus $C[Q]\downarrow$.
- $P \leq_{c,\downarrow} Q \implies P \leq_{c,\downarrow_x} Q$: Let $P \leq_{c,\downarrow} Q$, C be a Stop-free context, and $C[P]\!\downarrow_x$. Then $C[P] \xrightarrow{sr} P_1 \ldots \xrightarrow{sr} P_n \equiv \nu\mathcal{X}.(x(y).P' \mid P'')$, and for $C_1 := ([\cdot] \mid \overline{x}y.\mathsf{Stop})$ we have $C_1[C[P]]\downarrow$, since the reduction for $C[P]$ can be used and results in $C_1[P_n]$ which reduces to a successful process. $P \leq_{c,\downarrow} Q$ also implies $C_1[C[Q]]\downarrow$ and the corresponding reduction $C_1[C[Q]] \xrightarrow{sr,*} Q_m$ with $\mathsf{stop}(Q_m)$ must include an *ia*-reduction with a redex of the form $x(z).R \mid \overline{x}y.\mathsf{Stop}$. Let $Q_i \xrightarrow{sr,*} Q_{i+1}$ be this step in $C_1[C[Q]] \xrightarrow{sr,*} Q_m$. The prefix $C_1[C[Q]] \xrightarrow{sr,i} Q_i$ can be used to construct a reduction $C[Q] \xrightarrow{sr,i} Q'_i$ where $Q'_i\!\Uparrow^x$ and thus $C[Q]\!\downarrow_x$.
- $P \leq_{c,\Downarrow} Q \implies P \leq_{c,\Downarrow_x} Q$: Let $P \leq_{c,\Downarrow} Q$. For any Stop-free context $C \in \mathcal{C}$ we have to show: $C[Q]\!\Downarrow_x \implies C[P]\!\Downarrow_x$. Let C be a Stop-free context with $C[Q]\!\Downarrow_x$, i.e. $C[Q] \xrightarrow{sr,*} Q'$ and $\neg(Q'\!\downarrow_x)$. Then also $C_1[C[Q]]\!\uparrow$ with $C_1 = [\cdot] \mid \overline{x}y.\mathsf{Stop}$, since $C_1[C[Q]] \xrightarrow{sr,*} C_1[Q']$ and $C_1[Q']$ cannot become successful (otherwise $Q'\!\downarrow_x$ would hold). $P \leq_{c,\Downarrow} Q$ implies $C_1[C[P]]\!\uparrow$, i.e. $C_1[C[P]] \xrightarrow{sr,*} P'$ and $P'\!\Uparrow$. The reduction $C_1[C[P]] \xrightarrow{sr,*} P'$ can never reduce $\overline{x}y.\mathsf{Stop}$, since otherwise $P'\!\Uparrow$ cannot hold, and thus we can assume that $P' \equiv C_1[P'']$ and $C[P] \xrightarrow{sr,*} P''$. Again $P''\!\downarrow_{\Uparrow^x}$ cannot hold (otherwise $C_1[P'']\!\Uparrow$ would not hold) and thus $C[P]\!\Downarrow_x$.
- $P \leq_{c,\Downarrow_x} Q \implies P \leq_{c,\Downarrow} Q$. Let $P \leq_{c,\Downarrow_x} Q$ and C be a context with $C[Q]\!\uparrow$, i.e. $C[Q] = Q_0 \xrightarrow{sr} \ldots \xrightarrow{sr} Q_n$ and $Q_n\!\Uparrow$. Let u, v be fresh names. Then $\neg(\psi_{u,v}(Q_n)\!\downarrow_u)$ and also $\psi_{u,v}(Q_i) \xrightarrow{sr} \psi_{u,v}(Q_{i+1})$ and thus $\psi_{u,v}(C)[Q]\!\Downarrow_u$. From $P \leq_{c,\Downarrow_x} Q$ (using Proposition 5.3) we have $P \leq_{c,\Downarrow_u} Q$ and thus $\psi_{u,v}(C)[P]\!\Downarrow_u$, i.e. $C'[P] \xrightarrow{sr,*} P_m$ and $\neg(P_m\!\downarrow_u)$. Then $\psi^{-1}_{u,v}(P_n)\!\Uparrow$ (otherwise $P_n\!\downarrow_u$ would hold). Also $\psi^{-1}_{u,v}(P_i) \xrightarrow{sr} \psi^{-1}_{u,v}(P_{i+1})$, since $P_i \xrightarrow{sr} P_{i+1}$ cannot reduce any occurrence of $u(v)$. This shows $C[P]\!\uparrow$.

◀

Theorem 5.5 enables us to transfer some of the results for Π_{Stop} into Π.

▶ **Corollary 5.6.** *All equations in Theorem 4.6 (except for equation 5) also hold in* Π *for* Stop*-free processes, and for barbed testing equivalence* $\sim_{c,barb}$*. Deterministic interaction (see Theorem 4.1) is correct in* Π *for* $\sim_{c,barb}$*.*

Also Theorem 4.5 can be transferred to Π by applying Theorem 5.5, which shows that barbed should-testing preorder implies barbed may-testing equivalence (which does not hold for Π_{Stop} and contextual preorders, see Theorem 4.5 (3)):

▶ **Corollary 5.7.** *For all* Stop*-free processes* $P, Q \in Proc$*:* $P \leq_{c,\Downarrow_x} Q$ *implies* $P \sim_{c,\downarrow_x} Q$*.*

Proof. The inclusion $\leq_{c,\Downarrow_x} \subseteq \leq_{c,\downarrow_x}$ follows from Theorems 4.5 and 5.5. Before proving the remaining part, we show that the equivalence $P\downarrow_x \iff C_1[P]\Uparrow_y$ holds, where $C_1 := R \mid [\cdot]$ with $R = \nu z.(\overline{z}y \mid z(w).w(w') \mid \overline{x}x'.z(z'))$ and P is any process with $\mathsf{fn}(P) \cap \{w, w', z, z', x'\} = \emptyset$ We have to show two implications:

1. $P\downarrow_x \implies C_1[P]\uparrow_y$: If $P\downarrow_x$, then $C_1[P]$ can be reduced to $P'' := \nu z.(z(w).w(w')) \mid P'$ where P' is the contractum of P after receiving x' along x. Clearly, P'' cannot barb on y (i.e. $P'' \not\Downarrow_y$) and thus $C_1[P]\uparrow_y$.
2. $C_1[P]\uparrow_y \implies P\downarrow_x$: We show its contrapositive $P \not\Downarrow_x \implies C_1[P] \Downarrow_y$. If $P \not\Downarrow_x$, then in any reduction of $C_1[P]$ the process P cannot interact with the process R, and since $R \Downarrow_y$, also $C_1[P] \Downarrow_y$ holds.

We show $\leq_{c,\Downarrow_x} \subseteq (\leq_{c,\downarrow_x})^{-1}$: Let $P \leq_{c,\Downarrow_x} Q$, $C[Q]\downarrow_x$, but $C[P] \not\Downarrow_x$. Proposition 5.3 and $P \leq_{c,\Downarrow_x} Q$ imply $Q \leq_{c,\downarrow_y} P$. But $C_1[C[Q]]\Uparrow_y$ while $C_1[C[P]]\Downarrow_y$ which is a contradiction. ◀

Finally, we show that there is no surjective encoding from Π_{Stop} into Π which preserves the ordering of processes w.r.t. contextual preorder in Π_{Stop} and barbed testing preorder in Π.

▶ **Theorem 5.8.** *There is no surjective translation $\psi : \Pi_{\mathsf{Stop}} \to \Pi$ s.t. for all $P, Q \in Proc_{\mathsf{Stop}}$: $P \leq_c Q \implies \psi(P) \leq_{c,barb} \psi(Q)$.*

Proof. This holds since Stop is a largest element of Π_{Stop} w.r.t. $\leq_c$, but in Π there is no largest element w.r.t. $\leq_{c,barb}$: Assume the claim is false, and P is a largest element w.r.t. $\leq_{c,barb}$. Let $\mathcal{X} = \mathsf{fn}(P)$ and $x \notin \mathcal{X}$. Then $x(z) \not\leq_{c,barb} P$, since $\nu\mathcal{X}.x(z)\downarrow_x$ but $\nu\mathcal{X}.P \not\Downarrow_x$. ◀

6 Conclusion

We analyzed contextual equivalence w.r.t. may- and should-convergence in a π-calculus with Stop. We proved a context lemma and showed soundness of an applicative similarity. Since Π_{Stop} with contextual equivalence conservatively extends the π-calculus without Stop and barbed testing equivalence, this also provides a method to show barbed testing equivalences.

Future work may investigate extensions or variants of the calculus Π_{Stop}, e.g. with (guarded) sums, or with recursion. The results of this paper may also open easier possibilities to define and analyze embeddings of Π_{Stop} into other concurrent program calculi (e.g., the CHF-calculus [17, 18]) which also use a contextual semantics.

References

1 M. Boreale and R. De Nicola. Testing equivalence for mobile processes. *Inform. and Comput.*, 120(2):279–303, 1995.

2 A. Carayol, D. Hirschkoff, and D. Sangiorgi. On the representation of McCarthy's amb in the pi-calculus. *Theoret. Comput. Sci.*, 330(3):439–473, 2005.

3 R. De Nicola and M. Hennessy. Testing equivalences for processes. *Theoret. Comput. Sci.*, 34:83–133, 1984.

4 J. Engelfriet and T. Gelsema. A new natural structural congruence in the pi-calculus with replication. *Acta Inf.*, 40(6-7):385–430, 2004.

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

6 D. Gorla. Towards a unified approach to encodability and separation results for process calculi. *Inf. Comput.*, 208(9):1031–1053, 2010.

7 C. Laneve. On testing equivalence: May and must testing in the join-calculus. Technical Report Technical Report UBLCS 96-04, University of Bologna, 1996.

8 R. Milner. *Communicating and Mobile Systems: the π-calculus*. CUP, 1999.
9 R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, i & ii. *Inform. and Comput.*, 100(1):1–77, 1992.
10 R. Milner and D. Sangiorgi. Barbed bisimulation. In *Proc. ICALP 1992*, *LNCS* 623, pp. 685–695. Springer, 1992.
11 J.H. Morris. *Lambda-Calculus Models of Programming Languages*. PhD thesis, MIT, 1968.
12 V. Natarajan and R. Cleaveland. Divergence and fair testing. In Proc. *ICALP 1995*, *LNCS* 944, pp. 648–659. Springer, 1995.
13 K. Peters, T. Yonova-Karbe, and U. Nestmann. Matching in the pi-calculus. In *Proc. EXPRESS/SOS 2014*, *EPTCS* 160, pp. 16–29. Open Publishing Association, 2014.
14 B. C. Pierce and D. Sangiorgi. Typing and subtyping for mobile processes. *Math. Structures Comput. Sci.*, 6(5):409–453, 1996.
15 Gordon D. Plotkin. Call-by-name, call-by-value, and the lambda-calculus. *Theoret. Comput. Sci.*, 1:125–159, 1975.
16 A. Rensink and W. Vogler. Fair testing. *Inform. and Comput.*, 205(2):125–198, 2007.
17 D. Sabel and M. Schmidt-Schauß. A contextual semantics for Concurrent Haskell with futures. In *Proc. PPDP 2011*, pp. 101–112. ACM, 2011.
18 D. Sabel and M. Schmidt-Schauß. Conservative concurrency in Haskell. In *Proc. LICS 2012*, pp. 561–570. IEEE, 2012.
19 D. Sabel. Structural rewriting in the pi-calculus. In *Proc. WPTE 2014*, volume 40 of *OASIcs*, pages 51–62. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2014.
20 D. Sangiorgi and D. Walker. *The π-calculus: a theory of mobile processes*. CUP, 2001.
21 M. Schmidt-Schauß and D. Sabel. Closures of may-, should- and must-convergences for contextual equivalence. *Inform. Process. Lett.*, 110(6):232 – 235, 2010.
22 M. Schmidt-Schauß, D. Sabel, J. Niehren, and J. Schwinghammer. Observational program calculi and the correctness of translations. *Theoret. Comput. Sci.*, 577:98–124, 2015.

A Proofs for the Calculus Π_{Stop}

Proofs of Lemmas 3.1 and 3.2. For Lemma 3.1, we analyze all contexts of hole depth 1:

1. $C = [\cdot] \mid S$: Then $\sigma(C[P]) \mid R \equiv (\sigma(P) \mid R')$ and $\sigma(C[Q]) \mid R \equiv \sigma(Q) \mid R'$ with $R' = \sigma(S) \mid R$. The precondition of the claim implies that $\sigma(P) \mid R' \leq_{\downarrow} \sigma(Q) \mid R'$ and thus Proposition 2.3 shows $\sigma(C[P]) \mid R \leq_{\downarrow} \sigma(C[Q]) \mid R$.
2. $C = S \mid [\cdot]$: The claim follows from the previous item and Proposition 2.3.
3. $C = \nu x.[\cdot]$: Since $\sigma(P) \mid R \leq_{\downarrow} \sigma(Q) \mid R$ holds by the precondition of the claim, Lemma 2.4 shows the claim.
4. $C = x(y).[\cdot]$: Let $\sigma(C[P]) \mid R \xrightarrow{sr,n} P_n$ where P_n is successful. We use induction on n. The base case $n = 0$ holds, since in this case R must be successful, and thus $\sigma(C[Q]) \mid R$ is successful, too. For the induction step assume $\sigma(x) = x_1$ and w.l.o.g. $\sigma(y) = y$. Let $x_1(y).\sigma(P) \mid R \xrightarrow{sr} \nu\mathcal{X}.\sigma(P)[z/y] \mid R'$ be the first reduction step of the reduction sequence, where $\mathcal{X} \subseteq \{z\}$. The same reduction step for $\sigma(x(y).Q) \mid R$ results in $\nu\mathcal{X}.\sigma'(Q)[z/y] \mid R'$. By induction assumption, the lemma holds for the pair $\sigma(P)[z/y]$ and $\sigma(Q)[z/y]$, and by item (3) also for extending it with ν.
5. $C = \overline{x}y.[\cdot]$: This case is similar to the previous item.
6. $C = !\,[\cdot]$. Let $\sigma(!\,P) \mid R \xrightarrow{sr,n} P_n$ where $\mathsf{stop}(P_n)$. We show $\sigma(!\,Q) \mid R\!\downarrow$ by induction on n. If $n = 0$, then R or P is successful. Thus $\mathsf{stop}(\sigma(P) \mid R)$ holds and the precondition of the lemma shows $(\sigma(Q) \mid R)\!\downarrow$, which implies $\sigma(!\,Q) \mid R\!\downarrow$. For $n > 0$, let $\sigma(!\,P) \mid R \xrightarrow{sr} P_1$ be the first reduction of $\sigma(!\,P) \mid R \xrightarrow{sr,n} P_n$: If the redex is inside R, then the same reduction can be performed for $\sigma(!\,Q \mid R)$ and then

the induction hypothesis shows the claim. If the redex uses one instance of $\sigma(P)$ and parts of R, i.e. $P_1 \equiv \sigma(!\,P) \mid R_P$, s.t. $R \mid \sigma(P) \xrightarrow{sr} R_P$, then we apply the induction hypothesis to P_1 and have $(\sigma(!\,Q) \mid R_P)\downarrow$. This implies $\sigma(!\,Q) \mid \sigma(P) \mid R\downarrow$, since $\sigma(!\,Q) \mid \sigma(P) \mid R \xrightarrow{sr} \sigma(!\,Q) \mid R_P$. By the precondition of the lemma we have $\sigma(!\,Q) \mid \sigma(P) \mid R \equiv \sigma(P) \mid (\sigma(!\,Q) \mid R) \leq_\downarrow \sigma(Q) \mid (\sigma(!\,Q) \mid R) \equiv \sigma(!\,Q) \mid R$, and thus we have $\sigma(!\,Q) \mid R\downarrow$. If the redex uses two instances of $\sigma(P)$, i.e. $P_1 \equiv \sigma(!\,P) \mid R \mid P'$, s.t. $\sigma(P) \mid \sigma(P) \xrightarrow{sr} P'$, then the induction hypothesis for P_1 shows $\sigma(!\,Q) \mid R \mid P'\downarrow$. Since $\sigma(P) \mid \sigma(P) \xrightarrow{sr} P'$, we have $\sigma(!\,Q) \mid R \mid \sigma(P) \mid \sigma(P)\downarrow$. We apply the precondition twice: $\sigma(!\,Q) \mid R \mid \sigma(P) \mid \sigma(P) \equiv \sigma(P) \mid (\sigma(P) \mid (\sigma(!\,Q) \mid R)) \leq_\downarrow \sigma(Q) \mid (\sigma(P) \mid (\sigma(!\,Q) \mid R)) \equiv \sigma(P) \mid (\sigma(!\,Q) \mid R) \leq_\downarrow \sigma(Q) \mid (\sigma(!\,Q) \mid R) \equiv \sigma(!\,Q) \mid R$ and thus $\sigma(!\,Q) \mid R\downarrow$.

The proof of Lemma 3.2 is analogous to Lemma 3.1 by replacing $\leq_\downarrow$ with $\leq_\uparrow$, and replacing the base cases "if $\sigma(C[P]) \mid R$ is successful, then $\sigma(C[Q]) \mid R\downarrow$" with "if $(\sigma(C[P]) \mid R)\Uparrow$, then $(\sigma(C[Q]) \mid R)\uparrow$" which holds, since $\sigma(C[Q]) \mid R \leq_\downarrow \sigma(C[P]) \mid R$. ◀

▶ **Lemma A.1.** *For P, Q, S_{xy}, S_{yx} as defined in Proposition 3.8: $P \leq_{c,\downarrow} Q$.*

Proof. Let $\mathcal{S} := \mathcal{S}_1 \cup \mathcal{S}_2 \cup \precsim_\downarrow$ where $\mathcal{S}_2 := \{(\sigma(P) \mid R, \sigma(Q) \mid R) \mid \text{ for any } R \text{ and } \sigma\}$ and $\mathcal{S}_1 := \{((!\,S_{xy} \mid !\,S_{yx} \mid R[x/w] \mid \overline{y}u_1 \mid \ldots \mid \overline{y}u_n), (!\,S_{xy} \mid !\,S_{yx} \mid R[y/w] \mid \overline{x}u_1 \mid \ldots \mid \overline{x}u_n)) \mid \text{ for any } R, \text{ any } x, y, w, u_i, \text{ and any } n \geq 0\}$

For proving $P \leq_{c,\downarrow} Q$, it suffices to show that the relation $\mathcal{S}$ is $F_\downarrow$-dense: This implies $\sigma(P) \mid R \leq_\downarrow \sigma(Q) \mid R$ for all R, σ and thus the context lemma (Theorem 3.3) shows $P \leq_{c,\downarrow} Q$.

First let $(P_1, P_2) \in \mathcal{S}_1$. If P_1 is successful, then clearly also P_2 is successful and thus $P_2\downarrow$. If $P_1 \xrightarrow{sr} P_1'$, then there are following cases:

- If the redex is inside $R[x/w]$, then either the same reduction can also be performed for P_2, then $P_2 \xrightarrow{sr} P_2'$ and $(P_1', P_2') \in \mathcal{S}$, or the name x occurs in R. We consider two cases, where we use the abbreviations $L_x := \overline{x}u_1 \mid \ldots \mid \overline{x}u_n$ and $L_y := \overline{y}u_1 \mid \ldots \mid \overline{y}u_n$:
  1. If $R = \nu\mathcal{W}.(w(z').R_1 \mid \overline{x}v.R_2 \mid R_3)$ and $P_1' = !\,S_{xy} \mid !\,S_{yx} \mid \nu\mathcal{W}.(R_1[v/z'] \mid R_2 \mid R_3)[x/w] \mid L_y$, then $P_2 \xrightarrow{sr} P_2' \xrightarrow{sr} P_2''$ with $P_2 = !\,S_{xy} \mid !\,S_{yx} \mid L_x \mid \nu\mathcal{W}.(w(z').R_1 \mid \overline{x}v.R_2 \mid R_3)[y/w]$ and $P_2'' = !\,S_{xy} \mid !\,S_{yx} \mid L_x \mid \nu\mathcal{W}.(R_1[v/z'] \mid R_2 \mid R_3))[y/w]$, since $\overline{x}v.R_2 \mid S_{xy} \xrightarrow{sr} R_2 \mid \overline{y}v$. Since $(P_1', P_2'') \in \mathcal{S}$, we are finished.
  2. If $R = \nu\mathcal{W}.(x(z').R_1 \mid \overline{w}v.R_2 \mid R_3)$ and for P_1' we have $P_1' = !\,S_{xy} \mid !\,S_{yx} \mid \nu\mathcal{W}.(R_1[v/z'] \mid R_2 \mid R_3)[x/w] \mid L_y$, then there exists the reduction $P_2 \xrightarrow{sr} P_2' \xrightarrow{sr} P_2''$ with $P_2 = !\,S_{xy} \mid !\,S_{yx} \mid L_x \mid \nu\mathcal{W}.(x(z').R_1 \mid \overline{w}v.R_2 \mid R_3)[y/w]$ and $P_2'' = !\,S_{xy} \mid !\,S_{yx} \mid L_x \mid \nu\mathcal{W}.(R_1[v/z'] \mid R_2 \mid R_3))[y/w]$, since $\overline{y}v.R_2 \mid S_{yx} \xrightarrow{sr} R_2 \mid \overline{x}v$ and thus $(P_1', P_2'') \in \mathcal{S}$.
- The redex is $S_{yx} \mid \overline{y}u_i$, i.e. with the abbreviation $L_y = \overline{y}u_1 \mid \ldots \overline{y}u_{i-1} \mid \overline{y}u_{i+1} \mid \ldots \overline{y}u_n$, the reduction is $P_1 = !\,S_{xy} \mid !\,S_{yx} \mid R[x/w] \mid \overline{y}u_i \mid L_y \xrightarrow{sr} !\,S_{xy} \mid !\,S_{yx} \mid R[x/w] \mid \overline{x}u_i \mid L_y \equiv !\,S_{xy} \mid !\,S_{yx} \mid (R \mid \overline{w}u_i)[x/w] \mid L_y = P_1'$. Then for $L_x := \overline{x}u_1 \mid \ldots \overline{x}u_{i-1} \mid \overline{x}u_{i+1} \mid \ldots \overline{x}u_n$, there is the following reduction for process P_2: $P_2 = !\,S_{xy} \mid !\,S_{yx} \mid R[y/w] \mid \overline{x}u_i \mid L_x \xrightarrow{sr} !\,S_{xy} \mid !\,S_{yx} \mid R[y/w] \mid \overline{y}u_i \mid L_x \equiv !\,S_{xy} \mid !\,S_{yx} \mid (R \mid \overline{w}u_i)[y/w] \mid L_x = P_2'$ and $(P_1', P_2') \in \mathcal{S}$.
- The redex is $S_{xy} \mid R[x/w]$, i.e. $R = \overline{w}v.R'$ and for $L_y := \overline{y}u_1 \mid \ldots \mid \overline{y}u_n$ we have $P_1 = !\,S_{xy} \mid !\,S_{yx} \mid \overline{x}v.R'[x/w] \mid L_y \xrightarrow{sr} !\,S_{xy} \mid !\,S_{yx} \mid R'[x/w] \mid \overline{y}v \mid L_y = P_1'$. Then for $L_x := \overline{x}u_1 \mid \ldots \mid \overline{x}u_n$ the reduction $P_2 = !\,S_{xy} \mid !\,S_{yx} \mid \overline{y}v.R'[y/w] \mid L_x \xrightarrow{sr} P_2'$ exists, where $P_2' := !\,S_{xy} \mid !\,S_{yx} \mid R'[y/w] \mid \overline{x}v \mid L_x = P_2'$ and thus $(P_1', P_2') \in \mathcal{S}$.

Now let $(P_1, P_2) \in \mathcal{S}_2$ and let $a' = \sigma(a), x' = \sigma(x), y' = \sigma(y), z' = \sigma(z)$. If P_1 is successful, then P_2 is successful. If $P_1 \xrightarrow{sr} P_1'$, then there are the cases:

- If the redex is inside R, then $P_2 \xrightarrow{sr} P_2'$ and $(P_1', P_2') \in \mathcal{S}$.

- If $R = \nu\mathcal{W}.(a'(w).R' \mid R'')$ and $P_1 \xrightarrow{sr} !\sigma(S_{xy}) \mid !\sigma(S_{yx}) \mid \nu\mathcal{W}.(R'[x/w] \mid R'') := P_1'$. Then $P_1' \equiv !\sigma(S_{xy}) \mid !\sigma(S_{yx}) \mid \nu\mathcal{W}.(R' \mid R'')[x/w]$, since we may assume that w was renamed fresh for R''. Then $P_2 \xrightarrow{sr} P_2'$ with $P_2' := !\sigma(S_{xy}) \mid !\sigma(S_{yx}) \mid \nu\mathcal{W}.(R'[y/w] \mid R'')$. Since $P_2' \equiv !\sigma(S_{xy}) \mid !\sigma(S_{yx}) \mid \nu\mathcal{W}.(R' \mid R'')[y/w] = P_2'$, this shows $(P_1', P_2') \in \mathcal{S}$.
- $R = \nu\mathcal{W}.(\overline{x'}u.R' \mid R'')$ and $P_1 \xrightarrow{sr} \overline{a'}x' \mid !\sigma(S_{xy}) \mid !\sigma(S_{yx}) \mid (R' \mid R) \mid \overline{y'}u = P_1'$. Then $P_2 \xrightarrow{sr} \overline{a'}y' \mid !\sigma(S_{xy}) \mid !\sigma(S_{yx}) \mid (R' \mid R) \mid \overline{y'}u = P_2'$. and $(P_1', P_2') \in \mathcal{S}$.
- $R = \nu\mathcal{W}.(\overline{y'}u.R' \mid R'')$ and $P_1 \xrightarrow{sr} \overline{a'}x' \mid !\sigma(S_{xy}) \mid !\sigma(S_{yx}) \mid (R' \mid R) \mid \overline{x'}u = P_1'$. Then $P_2 \xrightarrow{sr} \overline{a'}y' \mid !\sigma(S_{xy}) \mid !\sigma(S_{yx}) \mid (R' \mid R) \mid \overline{x'}u = P_2'$ and $(P_1', P_2') \in \mathcal{S}$. ◀

▶ **Lemma A.2.** *The relation* $\mathcal{S} := \{(\nu\mathcal{X}.(P \mid R), \nu\mathcal{X}.(Q \mid R)) \mid P \precsim_{b,\downarrow} Q,\ \textit{for any } \mathcal{X},\ R\} \cup \precsim_{\downarrow}$ *is* $F_{\downarrow}$*-dense.*

Proof. Let $(\nu\mathcal{X}.(P \mid R), \nu\mathcal{X}.(Q \mid R)) \in \mathcal{S}$. We have to show $(\nu\mathcal{X}.(P \mid R), \nu\mathcal{X}.(Q \mid R)) \in F_{\downarrow}(\mathcal{S})$. If $\nu\mathcal{X}.(P \mid R)$ is successful, then P or R is successful too, and thus either $Q\downarrow$ and so does $\nu\mathcal{X}.Q \mid R$ or $\nu\mathcal{X}.(Q \mid R)$ is already successful. For $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$ we show that $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr,*} Q_1$, s.t. $(P_1, Q_1) \in \mathcal{S}$: If the redex of $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$ is inside P, i.e. $P_1 = \nu\mathcal{X}.(P' \mid R)$, then by $P \precsim_{b,\downarrow} Q$ there exists Q' with $Q \xrightarrow{sr,*} Q'$, $P' \precsim_{b,\downarrow} Q'$. Since also $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr,*} \nu\mathcal{X}.(Q' \mid R)$ and thus $(\nu\mathcal{X}.(P' \mid R), \nu\mathcal{X}.(Q' \mid R)) \in \mathcal{S}$, this case is finished. If the redex of $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$ is inside R, i.e. $P_1 = \nu\mathcal{X}.(P \mid R')$ then also $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr} \nu\mathcal{X}.(Q \mid R')$ and $(\nu\mathcal{X}.(P \mid R'), \nu\mathcal{X}.(Q \mid R')) \in \mathcal{S}$.

The remaining cases are that the redex uses parts of P and parts of R.

- If $P \equiv \nu\mathcal{X}_1.(x(y).P' \mid P'')$, $R \equiv \nu\mathcal{X}_2.(\overline{x}z.R' \mid R'')$ with $z \notin \mathcal{X}_2$ and $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} \nu\mathcal{X}.(\nu\mathcal{X}_1.(P'[z/y] \mid P'') \mid \nu\mathcal{X}_2.R' \mid R'') = P_1$, then by $P \precsim_{b,\downarrow} Q$ there exists Q_0 s.t. $Q \xrightarrow{sr,*} Q_0 = \nu\mathcal{Y}_1.(x(y).Q' \mid Q'')$ and $\mathcal{X}_1.(P'[z/y] \mid P'') \precsim_{b,\downarrow} \nu\mathcal{Y}_1.(Q'[z/y] \mid Q'')$. Since $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr,*} \nu\mathcal{X}.(Q_0 \mid R) \xrightarrow{sr} \nu\mathcal{Y}_1.(Q'[z/y] \mid Q'') \mid \nu\mathcal{X}_2.R' \mid R'') = Q_1$, $(P_1, Q_1) \in \mathcal{S}$.
- If $P \equiv \nu\mathcal{X}_1.(x(y).P' \mid P'')$, $R \equiv \nu z, \mathcal{X}_2.(\overline{x}z.R' \mid R'')$ and $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$ with $P_1 := \nu\mathcal{X}.(\nu z.(\nu\mathcal{X}_1.P'[z/y] \mid P'' \mid \nu\mathcal{X}_2.(R' \mid R'')))$, then by $P \precsim_{b,\downarrow} Q$ there exists a process Q_0 s.t. $Q \xrightarrow{sr,*} Q_0$, $Q_0 = \nu\mathcal{Y}_1.(x(y).Q' \mid Q'')$ and $\nu\mathcal{X}_1.(P'[z/y] \mid P'') \precsim_{b,\downarrow} \nu\mathcal{Y}_1.(Q'[z/y] \mid Q'')$. Since $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr,*} \nu\mathcal{X}.(Q_0 \mid R) \xrightarrow{sr} \nu\mathcal{X}.(\nu z.(\nu\mathcal{Y}_1.(Q'[z/y] \mid Q'') \mid \nu\mathcal{X}_2.(R' \mid R''))) = Q_1$ we have $(P_1, Q_1) \in \mathcal{S}$.
- If $P \equiv \nu\mathcal{X}_1.(\overline{x}y.P' \mid P'')$ and $R \equiv \nu\mathcal{X}_2.(x(z).R' \mid R'')$ with $y \notin \mathcal{X}_1$ and $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} \nu\mathcal{X}.(\nu\mathcal{X}_1.P' \mid P'') \mid \nu\mathcal{X}_2.(R'[y/z] \mid R'') = P_1$, then by $P \precsim_{b,\downarrow} Q$ there exists Q_0 with $Q \xrightarrow{sr,*} Q_0$, $Q_0 = \nu\mathcal{Y}_1.(\overline{x}y.Q' \mid Q'')$ where $y \notin \mathcal{Y}_1$ s.t. $\nu\mathcal{X}_1.(P' \mid P'') \precsim_{b,\downarrow} \mathcal{Y}_1.(Q' \mid Q'')$. Since also $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr,*} \nu\mathcal{X}.(Q_0 \mid R) \xrightarrow{sr} \nu\mathcal{X}.(\nu\mathcal{Y}_1.(Q' \mid Q'') \mid \nu\mathcal{X}_2.(R'[y/z] \mid R'')) = Q_1$, we have $(P_1, Q_1) \in \mathcal{S}$.
- If $P \equiv \nu y.\nu\mathcal{X}_1.(\overline{x}y.P' \mid P'')$, $R \equiv \nu\mathcal{X}_2.(x(z).R' \mid R'')$, and $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$ where $P_1 := \nu\mathcal{X}.\nu y.(\nu\mathcal{X}_1.(P' \mid P'') \mid \nu\mathcal{X}_2.(R'[y/z] \mid R''))$, then by $P \precsim_{b,\downarrow} Q$ there exists Q_0 with $Q \xrightarrow{sr,*} Q_0$, $Q_0 = \nu y.\nu\mathcal{Y}_1.(\overline{x}y.Q' \mid Q'')$ s.t. $\nu\mathcal{X}_1.(P' \mid P'') \precsim_{b,\downarrow} \nu\mathcal{Y}_1.(Q' \mid Q'')$. Since also $\nu\mathcal{X}.\nu y.(Q \mid R) \xrightarrow{sr,*} \nu\mathcal{X}.\nu y.(Q_0 \mid R) \xrightarrow{sr} \nu\mathcal{X}.\nu y.(\nu\mathcal{Y}_1.(Q' \mid Q'') \mid \nu\mathcal{X}_2.(R'[y/z] \mid R'')) = Q_1$, we have $(P_1, Q_1) \in \mathcal{S}$. ◀

▶ **Lemma A.3.** *The relation* $\mathcal{S} := \{(\nu\mathcal{X}.(P \mid R), \nu\mathcal{X}.(Q \mid R)) \mid P \precsim_{b,\uparrow} Q,\ \textit{for any } \mathcal{X},\ R\} \cup \precsim_{\uparrow}$ *is* $F_{\uparrow}$*-dense.*

Proof. Note that if $P \precsim_{b,\uparrow} Q$, then $Q \precsim_{b,\downarrow} P$. Let $(\nu\mathcal{X}.(P \mid R), \nu\mathcal{X}.(Q \mid R)) \in \mathcal{S}$. We have to show that $(\nu\mathcal{X}.(P \mid R), \nu\mathcal{X}.(Q \mid R)) \in F_{\uparrow}(\mathcal{S})$. If $\nu\mathcal{X}.(P \mid R) \Uparrow$, then $Q \precsim_{b,\downarrow} P$ and Proposition 3.11 show that $\nu\mathcal{X}.(Q \mid R) \precsim_{\downarrow} \nu\mathcal{X}.(P \mid R)$ which implies that $\nu\mathcal{X}.(P \mid R) \leq_{\Uparrow} \nu\mathcal{X}.(Q \mid R)$ and thus $\nu\mathcal{X}.(Q \mid R) \downarrow$. If $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$, then we have to show that $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr,*} Q_1$, s.t. $(P_1, Q_1) \in \mathcal{S}$. If the redex of $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$ is inside P,

i.e. $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} \nu\mathcal{X}.(P' \mid R)$ then $P \precsim_{b,\uparrow} Q$ shows that $Q \xrightarrow{sr,*} Q'$ s.t. $P' \precsim_{b,\uparrow} Q'$. Since $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr,*} \nu\mathcal{X}.(Q' \mid R)$ and thus $(\nu\mathcal{X}.(P' \mid R), \nu\mathcal{X}.(Q' \mid R)) \in \mathcal{S}$ in this case. If the redex of $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$ is inside R, i.e. $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} \nu\mathcal{X}.(P \mid R')$ then also $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr} \nu\mathcal{X}.(Q \mid R')$ and thus $(\nu\mathcal{X}.(P \mid R'), \nu\mathcal{X}.(Q \mid R')) \in \mathcal{S}$ in this case.

It remains to consider the cases where the redex uses parts of P and parts of R.

- If $P \equiv \nu\mathcal{X}_1.(x(y).P' \mid P'')$, $R \equiv \nu\mathcal{X}_2.(\overline{x}z.R' \mid R'')$ with $z \notin \mathcal{X}_2$ and $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} \nu\mathcal{X}.(\nu\mathcal{X}_1.(P'[z/y] \mid P'') \mid \nu\mathcal{X}_2.R' \mid R'') = P_1$, then by $P \precsim_{b,\uparrow} Q$ there exists Q_0 with $Q \xrightarrow{sr,*} Q_0 = \nu\mathcal{Y}_1.(x(y).Q' \mid Q'')$ s.t. $\mathcal{X}_1.(P'[z/y] \mid P'') \precsim_{b,\uparrow} \nu\mathcal{Y}_1.(Q'[z/y] \mid Q'')$. Since $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr,*} \nu\mathcal{X}.(Q_0 \mid R) \xrightarrow{sr} \nu\mathcal{Y}_1.(Q'[z/y] \mid Q'') \mid \nu\mathcal{X}_2.R' \mid R'') = Q_1$ this shows $(P_1, Q_1) \in \mathcal{S}$.
- If $P \equiv \nu\mathcal{X}_1.(x(y).P' \mid P'')$, $R \equiv \nu z, \mathcal{X}_2.(\overline{x}z.R' \mid R'')$, and $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$ where $P_1 := \nu\mathcal{X}.(\nu z.(\nu\mathcal{X}_1.P'[z/y] \mid P'' \mid \nu\mathcal{X}_2.(R' \mid R'')))$, then $P \precsim_{b,\uparrow} Q$ shows that there exists Q_0 with $Q \xrightarrow{sr,*} Q_0 = \nu\mathcal{Y}_1.(x(y).Q' \mid Q'')$ s.t. $\nu\mathcal{X}_1.(P'[z/y] \mid P'') \precsim_{b,\uparrow} \nu\mathcal{Y}_1.(Q'[y/z] \mid Q'')$. Since $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr,*} \nu\mathcal{X}.(Q_0 \mid R) \xrightarrow{sr} \nu\mathcal{X}.(\nu z.(\nu\mathcal{Y}_1.(Q'[z/y] \mid Q'') \mid \nu\mathcal{X}_2.(R' \mid R''))) = Q_1$ we have $(P_1, Q_1) \in \mathcal{S}$.
- If $P \equiv \nu\mathcal{X}_1.(\overline{x}y.P' \mid P'')$, $R \equiv \nu\mathcal{X}_2.(x(z).R' \mid R'')$ with $y \notin \mathcal{X}_1$, and $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$ with $P_1 := \nu\mathcal{X}.(\nu\mathcal{X}_1.P' \mid P'') \mid \nu\mathcal{X}_2.(R'[y/z] \mid R'')$, then $P \precsim_{b,\uparrow} Q$ shows that $Q \xrightarrow{sr,*} Q_0$ with $Q_0 := \nu\mathcal{Y}_1.(y.Q' \mid Q'')$ where $y \notin \mathcal{Y}_1$ s.t. $\nu\mathcal{X}_1.(P' \mid P'') \precsim_{b,\uparrow} \mathcal{Y}_1.(Q' \mid Q'')$. Since $\nu\mathcal{X}.(Q \mid R) \xrightarrow{sr,*} \nu\mathcal{X}.(Q_0 \mid R) \xrightarrow{sr} \nu\mathcal{X}.(\nu\mathcal{Y}_1.(Q' \mid Q'') \mid \nu\mathcal{X}_2.(R'[y/z] \mid R'')) = Q_1$, we have $(P_1, Q_1) \in \mathcal{S}$.
- If $P \equiv \nu y, \mathcal{X}_1.(\overline{x}y.P' \mid P'')$, $R \equiv \nu\mathcal{X}_2.(x(z).R' \mid R'')$, and $\nu\mathcal{X}.(P \mid R) \xrightarrow{sr} P_1$ with $P_1 := \nu\mathcal{X}.\nu y.(\nu\mathcal{X}_1.(P' \mid P'') \mid \nu\mathcal{X}_2.(R'[y/z] \mid R''))$, then $P \precsim_{b,\uparrow} Q$ shows that $Q \xrightarrow{sr,*} Q_0$ with $Q_0 = \nu y.\nu\mathcal{Y}_1.(y.Q' \mid Q'')$ s.t. $\nu\mathcal{X}_1.(P' \mid P'') \precsim_{b,\uparrow} \nu\mathcal{Y}_1.(Q' \mid Q'')$. Since also the reduction $\nu\mathcal{X}.\nu y.(Q \mid R) \xrightarrow{sr,*} \nu\mathcal{X}.\nu y.(Q_0 \mid R) \xrightarrow{sr} Q_1$ exists, where the process Q_1 is $Q_1 := \nu\mathcal{X}.\nu y.(\nu\mathcal{Y}_1.(Q' \mid Q'') \mid \nu\mathcal{X}_2.(R'[y/z] \mid R''))$, this shows$(P_1, Q_1) \in \mathcal{S}$. ◀