A Partition Refinement Algorithm for the π-Calculus*

Marco Pistore[1] and Davide Sangiorgi[2]

[1] Dipartimento di Informatica, Università di Pisa, Italy
[2] INRIA – Sophia Antipolis, France

Abstract

The *partition refinement algorithm* [11, 7] is the basis for most of the tools for checking bisimulation equivalences and for computing minimal realisations of CCS-like finite state processes.

In this paper, we present a partition refinement algorithm for the π-calculus [8], a development of CCS where channel names can be communicated. It can be used to check bisimilarity and to compute minimal realisations of finite control processes — the π-calculus counterpart of CCS finite state processes. The algorithm is developed for *strong open bisimulation*, and can be adapted to *late* and *early* bisimulations, as well as to *weak* bisimulations. To arrive at the algorithm, a few laws, proof techniques, and four characterizations of open bisimulation are proved.

*Pistore's research was supported by CNR project "Strumenti per la Verifica di Proprietà Critiche di Sistemi Concorrenti e Distribuiti"; Sangiorgi's research by CNET project "Modélisation de Systéme Mobiles".

1 Introduction

Bisimulation is widely used for defining behavioural equivalences on terms of process description languages. It has been extensively studied in CCS, where efficient algorithms and tools for bisimulation checking have been devised. A prominent role among these algorithms is occupied by the *partition refinement algorithm* [11, 7]. The algorithm works in two phases: In the first, the state spaces of the processes to be checked (i.e., the set of their derivatives under arbitrary sequences of actions) are separately generated; in the second, a partition refinement procedure is applied to the union of the state spaces, which terminates when the equivalence classes of bisimulation are found. Most important, the same algorithm can be used to obtain a *minimal realisation* of a process P, i.e., a process which has the minimum number of states and transitions among all those bisimilar with P. In the case of CCS, algorithms for bisimilarity generally apply only to *finite-state* processes, syntactically described by disallowing parallel composition within recursive definitions.

In this paper, we study bisimulation checking in the π-calculus, a development of CCS where channel names can be communicated. Name-passing increases the expressiveness of the calculus, but it also dramatically affects the theory — above all the definition of bisimulation and its associated algorithms. In the π-calculus, the syntactic counterpart of CCS finite state processes are the *finite control* processes [4]. Due to the creation of new names, finite control processes can exhibit an infinite-state behaviour.

Three definitions of bisimulation, called *late* [8], *early* [8] and *open* [13], have been proposed for the π-calculus, and vary in the way name instantiations are handled. Here, we focus on open bisimulation, for these main reasons: First, by contrast with the other two, open bisimulation is a full congruence; this can be used, for instance, for compositional minimisations of processes. Secondly, the average complexity of checking open bisimulation is expected to be substantially lower than those of late and early bisimulations [2, 13, 15]. Thirdly, a partition refinement algorithm for open bisimulation presents more difficulties, hence one expects to extract algorithms for late and early bisimulations from it. These points are developed further in the concluding section.

A possible drawback of open bisimulation is that it is tailored for a process language that, as the original π-calculus language [8], does not have *mismatching*, that is testing for inequality of names. However, we are not aware of compelling examples of applications of π-calculus languages with mismatching.

Open bisimulation. Differently from early and late bisimulations, where free names of processes are viewed as constants (hence cannot be identified), in open bisimulation free names are viewed as free variables. However, permanent inequalities on names can be imposed by means of *distinctions*, i.e., irreflexive and symmetric relations on names. Distinctions allow us, for instance, to create constants by declaring certain names different from all other names; they are also useful to handle name extrusions — see below. We shall write $\sim_D$ to denote open bisimilarity under distinction D.

We review some aspects of the symbolic characterisation of open bisimulation [13]. Symbolic transitions are of the form $P \xrightarrow{(M,\alpha)} P'$ where, intuitively, M represents the least condition, in the form of a conjunction of equalities between names, under which action α can occur. A condition M determines an equivalence relation on names; chosen a representative for each equivalence class, M also determines a name substitution σ_M which maps names to their representatives. We write $\emptyset$ for the "true" condition, that is the empty conjunction. Among the possible forms of action α, there is the *bound output* $\overline{a}(b)$, which denotes the emission at a of the *private* name b. Approximately, the clause of bisimilarity on bound outputs says that if $P \sim_D Q$, and $\mathrm{fn}(P,Q)$ is the set of free names in P and Q, then:

$$\begin{array}{l} \text{whenever } P \xrightarrow{(M,\overline{a}(b))} P' \text{ with } b \text{ not free in } Q, \text{ there are } N \text{ and } Q' \text{ such that} \\ Q \xrightarrow{(N,\overline{a}(b))} Q', \quad M \text{ implies } N, \quad \text{and} \\ P' \ \sim_{D'} \ Q'\sigma_M \quad \text{ for } D' \stackrel{\text{def}}{=} \big(D \cup \{b\} \times \mathrm{fn}(P,Q)\big)\sigma_M . \end{array} \tag{1}$$

Derivative Q' and distinction D are updated according to the name equalities imposed by condition M, and distinction $\{b\} \times \mathrm{fn}(P,Q)$ is added to record the creation of the new name b, which cannot be identified with the previously known names.

Main problems for a partition algorithm. In (1), three forms of dependencies between P and Q affect the transitions and the derivatives to examine:

Dep.1: The name emitted by P cannot occur free in Q;

Dep.2: The condition M in the transition of P determines a substitution σ_M which is applied onto the derivative of Q;

Dep.3: There is a global distinction, which is updated using informations (the free names) from both processes.

These dependencies prevent us from generating the state spaces of P and Q separately from each other, as requested in the first phase of the partition refinement algorithm. Each dependency introduces a separate problem, which we now discuss in more detail.

Dep.1 is imposed to ensure that the bound name chosen by P can also be chosen by Q, and is necessary because bisimilar processes may have different sets of free names, like

$$P_1 \stackrel{\text{def}}{=} P_2 + [a=e]\overline{c}(b).\, a(d).\, 0 \qquad \text{and} \qquad P_2 \stackrel{\text{def}}{=} \overline{c}(b).\, e(d).\, 0 \tag{2}$$

(a matching $[a=e]$ means "if $a = e$ then"). The choice of the bound name could be made locally if *active names* [9], instead of free names, were used. A name x is *active* for a process P if x does affect the behaviour of P, i.e., P and $(\boldsymbol{\nu} x)P$ are not bisimilar ($(\boldsymbol{\nu}_)$ is the restriction construct). Active names would eliminate Dep.1 because bisimilar processes have the same sets of active names. Unfortunately, computing active names is as difficult as computing bisimilarity. For instance, x is active in $a(y).\,([y=x]P+Q)$ iff $P\{{}^{y}/_{x}\}$ and $Q\{{}^{y}/_{x}\}$ are bisimilar.

Dep.2 arises because conditions M and N on the transitions from P and Q may be logically non-equivalent. This situation happens, for instance, when comparing processes P_1 and P_2 in (2), where transition $P_1 \xrightarrow{([a=e],\overline{c}(b))} a(d).\, 0$ is matched by transition $P_2 \xrightarrow{(\emptyset,\overline{c}(b))} e(d).\, 0$, for under the condition $[a=e]$ the two derivatives are bisimilar (in this case, they are syntactically equal). However, this transition of P_1 does not add anything interesting to its behaviour, since covered by transition $P_1 \xrightarrow{(\emptyset,\overline{c}(b))} e(d).\, 0$, which has a logically weaker condition. The first transition can therefore be regarded as *redundant*. Indeed, if redundant transitions were ignored in the bisimulation clauses, then Dep.2 could be removed. But again, determining whether a transition is non-redundant is as difficult as computing bisimilarity.

To avoid Dep.3, distinctions should be made local to processes, and should be locally updated. But the update of the local distinction of a process P has to depend on its free names, otherwise the update might not be sound. As a consequence, since bisimilar processes may have different free names, one must be able to compare processes with different local distinctions. This makes it hard to obtain a *transitive* bisimulation relation and to recover open bisimulation.

Our approach. In general, computing bisimilarity, active names or non-redundant transitions is of equal difficulty, since, unfortunately, they all depend on each other. Our algorithm will hence compute bisimilarity, active names and non-redundant transitions *at the same time*. Since bisimulation is defined as a greatest fixed-point, whereas active names and non-redundant transitions are defined as smallest fixed-points, the algorithm approximates bisimulation from above, and active names and non-redundant transitions from below. At the beginning, all processes are assumed bisimilar, no name is assumed active and no transition is assumed non-redundant. In each approximation step, the appropriate transitions and derivatives of processes are selected according to the current estimation of active names and non-redundant transitions, and the standard partition refinement algorithm is applied. At the end, active names and non-redundant transitions are updated. In this way, at each step the assumed set of bisimilar processes decreases, whereas the assumed sets of active names and non-redundant transitions increase. This procedure is repeated until a fixed-point is reached.

The update of the distinction is local to a process P and uses its free names. The bisimulation relation computed by the algorithm, and defined on processes with a local distinction, is not transitive; but it enjoys a weak form of transitivity which is enough to prove a characterisation theorem w.r.t. open bisimulation and to apply the partition refinement algorithm (in which transitivity is important).

Our algorithm can be used on finite-control processes to check open bisimulation and to compute minimal realisations of processes.

Related work. The closest work to ours is [9], where a partition refinement algorithm for early and late bisimulation is obtained which works for finite-control processes without the matching operator. This work has inspired our own, and has provided us with useful insights. However, our technical development is quite different, because none of the problems in Dep.1–3 arise in [9]. Dep.2 and Dep.3 do not arise because they are specific to open bisimulation; Dep.1 does not pose a serious obstacle because, in late and early bisimulations without matching, active names of processes are trivial to compute: Roughly, a name x is active in a process P if x appears as a free name in a label of a computation of P. Hence active names can be computed by a standard transitive closure procedure.

The *Mobility Workbench* [14] is a tool for mechanically checking open bisimulation on finite control processes. It adopts an *on the fly* [5] approach, as opposed to the partitioning approach (in on-the-fly, the state spaces of processes compared are created at the same time as the candidate bisimulation relation). A disadvantage of on-the-fly is that it cannot be used to give the minimal realisation of a process. Moreover, due to the need of backtracking, in general on-the-fly is less efficient than partitioning both in time and in space (especially in the case of weak bisimulations). However, on-the-fly can be superior on processes which exhibit a limited degree of non-determinism; and it may return an answer even on non-finite-control processes, if not bisimilar.

In [4] decidability of early and late bisimilarity for finite control processes is proved. In particular, it is shown that for every pair of finite control processes only a finite number of names is sufficient for checking bisimilarity; once the number of names is fixed, the state space of both agents can be built and a partition refinement algorithm can be applied. However, this approach can be expensive and, since the number of names can be fixed only for pairs of processes, it does not provide us with minimal realisations. Open bisimulation, and hence Dep.2 and Dep.3, are not considered in [4].

Structure of the paper. In Section 2 we report some notations and necessary background on the π-calculus. In Section 3 we introduce new proof techniques for open bisimulation. In Sections 4, 5 and 6 we propose three different characterizations of open bisimulations, which incrementally refine the ordinary definition, avoiding respectively the dependencies Dep.3, Dep.2 and Dep.1. In Section 7, using the characterization of Section 6, we define an iterative method that computes open bisimilarity through a sequence of approximating partitions. In Section 8 we describe an algorithm that is based on this partition-refinement approach; it can be used to check open bisimilarities and to obtain minimal realisations on the class of finite control processes. Finally, in Section 9 we report some concluding remarks.

Our proofs of the main results of the paper are rather long. The most important of them are developed in the in appendices; we have left short proof sketches in the main text. The missing proofs can be found in a forthcoming Technical Report.

There is an initial appendix with some preliminary useful results; and then one appendix for each of the Sections 3-7.

2 Background

The π-calculus. We review the syntax and operational semantics of the π-calculus; more details can be found in [13]. Letters $a, b, \ldots, x, y, \ldots$ range over the infinite ordered set $\mathcal{N}$ of *names*, and K over the set of *process identifiers*. The class of *processes* is built from the operators of inaction, prefixing, matching, parallel composition, restriction, sum, and recursion; a prefix can be a silent prefix, an input, a free output, or a bound output:

$$\begin{array}{rcl} P & := & 0 \;\mid\; \alpha.P \;\mid\; [a=b]P \;\mid\; P_1 \mid P_2 \;\mid\; (\nu a)P \;\mid\; P_1 + P_2 \;\mid\; K\langle \tilde{a} \rangle \\ \alpha & := & \tau \;\mid\; a(b) \;\mid\; \overline{a}b \;\mid\; \overline{a}(b). \end{array}$$

Processes are ranged over by P, Q, R, prefixes (and actions) by α, β, γ, process identifiers by K. Each identifier K has an associated arity and a definition of the form $K \stackrel{\text{def}}{=} (\tilde{a})P$; in $K\langle\tilde{a}\rangle$ and $(\tilde{a})P$, tuple $\tilde{a}$ is of the same length as the arity of K; moreover, the free names of P are contained in $\tilde{a}$.

We give sum and parallel composition the lowest syntactic precedence among the operators. In a process $\alpha.\,0$, we often omit the trailing 0. In $a(b).\,P$, $(\boldsymbol{\nu} b)P$, and $\overline{a}(b).\,P$, all free occurrences of name b in P are bound; the same holds in $(\tilde{b})P$ for names $\tilde{b}$. The *free names*, *bound names* and *extruded names* of a prefix α, respectively written $\mathrm{fn}(\alpha)$, $\mathrm{bn}(\alpha)$ and $\mathrm{en}(\alpha)$, are defined as follows:

α	$\mathrm{fn}(\alpha)$	$\mathrm{bn}(\alpha)$	$\mathrm{en}(\alpha)$
τ	$\emptyset$	$\emptyset$	$\emptyset$
$a(b)$	$\{a\}$	$\{b\}$	$\emptyset$
$\overline{a}b$	$\{a, b\}$	$\emptyset$	$\emptyset$
$\overline{a}(b)$	$\{a\}$	$\{b\}$	$\{b\}$

Moreover, the *names* of α, written $\mathrm{n}(\alpha)$, are $\mathrm{fn}(\alpha) \cup \mathrm{bn}(\alpha)$. Free names, bound names and names of processes are defined as expected. We use alpha-conversion implicitly as necessary to satisfy side conditions about distinctness of bound and free names. We sometimes write $\mathrm{fn}(X, Y, \dots)$ for $\mathrm{fn}(X) \cup \mathrm{fn}(Y) \cup \cdots$; similarly for names and bound names.

Substitutions, ranged over by σ and ρ, are functions form names to names. A name a is *neutral* for a substitution σ if σ does not affect a, i.e., for all $b \in \mathcal{N}$, $\sigma(b) = a$ iff $b = a$. A set of names V is *neutral* for a substitution σ if all names in V are neutral for σ. The composition of two substitutions σ and ρ, where σ is performed first, is written $\sigma\rho$. Application of a substitution σ to an expression X is written $X\sigma$. Alpha-conversion is applied silently, as necessary, to avoid capture; therefore, for instance, when we write $\alpha\sigma$ we assume that the bound name of α, if it exists, is neutral for σ. We represent with $\{^{x_1}/_{y_1} \cdots {}^{x_n}/_{y_n}\}$ the substitution which maps name y_i into name x_i, for $i = 1..n$, and is the identity otherwise.

Conditions are finite conjunctions of matching, like $[a=b][c=d]$. We use M and N to range over conditions, and $\mathrm{n}(M)$ for the names which appear in M. We write $M \rhd N$ if M implies N, i.e., the equalities in M imply those in N; $M \lhd\rhd N$ is a shortcut for "$M \rhd N$ and $N \rhd M$" (that is, $M \lhd\rhd N$ means that M a and N are logically equivalent), whereas $M \not\lhd\rhd N$ is a shortcut for "$M \rhd N$ but not $N \rhd M$". Notice that, since names are ordered, there are canonical forms for conditions, which are unique up to logical equivalence. Given a matching M, we denote by σ_M the substitution which selects the minimal representative out of each equivalence class on names induced by M, i.e.,

$$\sigma_M(a) = \min\{b \mid M \rhd [a=b]\}.$$

In the paper we often prove properties "by induction on a condition M": this means that we prove the property for M assuming that it holds for all N such that $M \not\lhd\rhd N$; this approach is correct, since there are no infinite chains $M \not\lhd\rhd N_0 \not\lhd\rhd N_1 \cdots$ starting from a condition M.

The symbolic transition system for open bisimulation [13] is reported, in Table 1 (the symmetric of `sum`, `par`, `com` and `close` have been omitted). Transitions have the form $P \stackrel{(M,\alpha)}{\rightsquigarrow} P'$, where M represents the minimal condition on names required by P to perform that action. P and P' are called the *source* and *target* of the transition, respectively. It is convenient to use a "canonical" transition system $P \xrightarrow{(M,\alpha)} P'$ in which M is in canonical form and substitution σ_M has already been applied to action α and derivative P':

$$\frac{P \stackrel{(N,\alpha)}{\rightsquigarrow} P' \qquad M \text{ is the canonical form for } N}{P \xrightarrow{(M,\alpha\sigma_M)} P'\sigma_M} \text{ if } \mathrm{bn}(\alpha) \cap \mathrm{fn}(P) = \emptyset$$

A *distinction* is a finite symmetric irreflexive relation on the names $\mathcal{N}$, which expresses permanent inequalities on $\mathcal{N}$, i.e., if (a, b) is in the distinction, then a must be kept separate from b. We use D and E to range over distinctions and write $\mathrm{n}(D)$ for the names which are mentioned in D. A substitution σ *respects* a distinction D if $(a, b) \in D$ implies $\sigma(a) \neq \sigma(b)$; in this case we write $D\sigma$ for the distinction $\{(\sigma(a), \sigma(b)) \mid (a, b) \in D\}$. Similarly, a matching M *respects* a distinction D if σ_M respects D. Sometimes, in the expressions defining distinctions we shall avoid to give all symmetric pairs; for instance, we might

$$\texttt{alpha:}\quad \frac{P' \overset{\mu}{\rightsquigarrow} P'' \qquad P \text{ and } P' \ \alpha\text{-convertible}}{P \overset{\mu}{\rightsquigarrow} P''}$$

$$\texttt{pre:}\quad \alpha.P \overset{(\emptyset,\alpha)}{\rightsquigarrow} P \qquad\qquad \texttt{con:}\quad \frac{P\{\tilde{a}/\tilde{b}\} \overset{\mu}{\rightsquigarrow} P'}{K\langle\tilde{a}\rangle \overset{\mu}{\rightsquigarrow} P'} \text{ if } K \stackrel{\text{def}}{=} (\tilde{b})P$$

$$\texttt{sum:}\quad \frac{P \overset{\mu}{\rightsquigarrow} P'}{P+Q \overset{\mu}{\rightsquigarrow} P'} \qquad\qquad \texttt{par:}\quad \frac{P \overset{\mu}{\rightsquigarrow} P'}{P \mid Q \overset{\mu}{\rightsquigarrow} P' \mid Q} \text{ if } \mathrm{bn}(\mu) \cap \mathrm{fn}(Q) = \emptyset$$

$$\texttt{com:}\quad \frac{P \overset{(M,\overline{a}y)}{\rightsquigarrow} P' \qquad Q \overset{(N,b(x))}{\rightsquigarrow} Q'}{P \mid Q \overset{(L,\tau)}{\rightsquigarrow} P' \mid Q'\{y/x\}} \text{ where } L \stackrel{\text{def}}{=} MN[a=b]$$

$$\texttt{close:}\quad \frac{P \overset{(M,\overline{a}(x))}{\rightsquigarrow} P' \qquad Q \overset{(N,b(x))}{\rightsquigarrow} Q'}{P \mid Q \overset{(L,\tau)}{\rightsquigarrow} (\boldsymbol{\nu} x)(P' \mid Q')} \text{ where } L \stackrel{\text{def}}{=} MN[a=b]$$

$$\texttt{match:}\quad \frac{P \overset{(M,\alpha)}{\rightsquigarrow} P'}{[a=b]P \overset{(N,\alpha)}{\rightsquigarrow} P'} \text{ where } N \stackrel{\text{def}}{=} M[a=b]$$

$$\texttt{res:}\quad \frac{P \overset{\mu}{\rightsquigarrow} P'}{(\boldsymbol{\nu} b)P \overset{\mu}{\rightsquigarrow} (\boldsymbol{\nu} b)P'} \text{ if } b \notin \mathrm{n}(\mu) \qquad \texttt{open:}\quad \frac{P \overset{(M,\overline{a}b)}{\rightsquigarrow} P'}{(\boldsymbol{\nu} b)P \overset{(M,\overline{a}(b))}{\rightsquigarrow} P'} \text{ if } b \notin \mathrm{n}(M) \cup \{a\}$$

Table 1: The transition system specialised for open bisimulation.

---

define $D = \{(a,b)\}$ without recalling that also $(b,a) \in D$. If N is a set of name, then $D - N$ and $D \cap N$ are, respectively, the distinctions

$$\{(a,b) \in D \ : \ a,b \notin N\} \quad \text{and} \quad \{(a,b) \in D \ : \ a,b \in N\}.$$

We sometimes abbreviate a distinction $\{(a,b)\}$ as (a,b) to avoid double parenthesis.

Open bisimulation. We report the symbolic definition of open bisimulation [13], written using transitions in canonical form. An *indexed relation* $\mathcal{R}$ is a set $\{\mathcal{R}_D\}_D$ of relations $\mathcal{R}_D$ on π-calculus processes, where D ranges over all possible distinctions. Inclusion, union and intersection between two indexed relations $\mathcal{R}$ and $\mathcal{S}$ are defined index-wise; for instance, their union is defined on a generic distinction D as $(\mathcal{R} \cup \mathcal{S})_D \stackrel{\text{def}}{=} \mathcal{R}_D \cup \mathcal{S}_D$.

Definition 2.1 (open bisimulation) *An indexed relation $\mathcal{R}$ is an* open bisimulation *(or a $\sim$-*bisimulation*) if $P \ \mathcal{R}_D \ Q$ implies:*

- *Whenever $P \xrightarrow{(M,\alpha)} P'$, with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(Q,D) = \emptyset$ and M respects D, then there is some $Q \xrightarrow{(N,\beta)} Q'$ such that:*
  - *$M \rhd N$,*
  - *$\alpha = \beta\sigma_M$, and*
  - *$P' \ \mathcal{R}_{D'} \ Q'\sigma_M$ for $D' \stackrel{\text{def}}{=} \big(D \cup (\mathrm{en}(\alpha) \times \mathrm{fn}(P,Q))\big)\sigma_M$.*
- *The converse, with the role of P and Q exchanged.*

Two processes P and Q are open bisimilar w.r.t. distinction D, *written $P \sim_D Q$, if $P \ \mathcal{R}_D \ Q$ for some open bisimulation $\mathcal{R}$.*

According to the previous definition, we have

$$\overline{a}b \mid c(d) \ \sim_\emptyset \ \overline{a}b.\, c(d) + c(d).\, \overline{a}b + [a=c]\tau.$$

Component $[a=c]\tau$ in the second process is required, since we have to take into account the possibility that components $\overline{a}b$ and $c(d)$ of the first process communicate (by exploiting rule com), in case that names a and c coincide. We also have

$$\overline{a}b \mid c(d) \ \sim_{(a,c)} \ \overline{a}b.\, c(d) + c(d).\, \overline{a}b.$$

In this case, component $[a=c]\tau$ is not required, since names a and c are declared non-equal by the distinction.

In Definition 2.1, the distinction D for the source processes P and Q is updated to the distinction D' for the target processes P' and $Q'\sigma_M$ by declaring that the extruded name of the action (if any) is different from all the names which where present in the source processes and by applying the substitution σ_M corresponding to the condition of the transitions. In the remainder, we often use similar updates of the distinctions. We therefore introduce an abbreviation for them:

$$D^M_{(P_1,\ldots,P_n),\alpha} \stackrel{\text{def}}{=} \big(D \cup (\mathrm{en}(\alpha) \times \mathrm{fn}(P_1,\ldots,P_n))\big)\sigma_M.$$

If $n=1$ we simply write $D^M_{P,\alpha}$. Distinction D' in Definition 2.1 can then be written as $D^M_{(P,Q),\alpha}$.

Proposition 2.2 *The indexed relation $\sim$ is an open bisimulation; hence $\sim$ is the largest open bisimulation.*

We conclude this section with an example of two processes that are open bisimilar. They are similar to processes P_1 and P_2 used in the Introduction. However, to make the example non-trivial, the processes have "cyclic" behaviours. We shall use these processes as a running example for the various concepts that we will introduce in the paper.

Example 2.3 *Consider the processes $K_1\langle a,c,e\rangle$ and $K_2\langle c,e\rangle$, where constants K_1 and K_2 are so defined:*

$$K_1 \stackrel{\text{def}}{=} (a,c,e)\big(K_2\langle c,e\rangle + [a=e]\overline{c}(b).\, a(d).\, K_1\langle a,b,d\rangle\big) \qquad \textit{and} \qquad K_2 \stackrel{\text{def}}{=} (c,e)\big(\overline{c}(b).\, e(d).\, K_2\langle b,d\rangle\big).$$

The two processes are open bisimilar (w.r.t. the empty distinction). Obviously $K_1\langle a,c,e\rangle$ can match all actions of $K_2\langle c,e\rangle$. For the converse, note that summand $[a=e]\overline{c}(b).\, a(d).\, K_1\langle a,b,d\rangle$ can only act under the condition $a=e$. The resulting actions $\overline{c}(b)$ and $a(d)$ are matched by actions $\overline{c}(b)$ and $e(d)$ of $K_2\langle c,e\rangle$, yielding the pair of processes $(K_1\langle a,b,d\rangle, K_2\langle b,d\rangle)$ and the distinction $D=\{(a,b),(b,c)\}$, in which the extruded name b is declared different from the names a and c (therefore also from e, since $a=e$ holds). Given that $K_1\langle a,c,e\rangle$ and $K_2\langle c,e\rangle$ can match each other actions w.r.t. the empty distinction, also $K_1\langle a,b,d\rangle$ and $K_2\langle b,d\rangle$ can w.r.t. D since, intuitively, the distinction is larger and the latter pair can be obtained from the former by applying a substitution (both these operations are sound for open bisimulation, see Corollary 3.9).

The above is a simple informal *argument for the bisimilarity between $K_1\langle a,c,e\rangle$ and $K_2\langle c,e\rangle$. A formal proof, following Definition 2.1, is tedious. A bisimulation relation $\mathcal{R}$ such that $\mathcal{R}_\emptyset$ includes the pair $\big(K_1\langle a,c,e\rangle, K_2\langle c,e\rangle\big)$ is a rather large relation, because it must have at least these inclusions, for all n and m with $0 \le m \le n$:*

$$\big(K_1\langle a,b_n,d_n\rangle \ , \ K_2\langle b_n,d_n\rangle\big) \in \mathcal{R}_{\{(a,b_i),(b_{i-1},b_i)\,:\,1\le i\le n\}} \tag{i}$$

$$\big(a(d).\, K_1\langle a,b_{n+1},d\rangle \ , \ a(d).\, K_2\langle b_{n+1},d\rangle\big) \in \mathcal{R}_{\{(a,b_i),(b_{i-1},b_i)\,:\,1\le i\le n+1\}} \tag{ii}$$

$$\big(K_2\langle b_n,d_n\rangle \ , \ K_2\langle b_n,d_n\rangle\big) \in \mathcal{R}_{\{(a,b_i),(b_{i-1},b_i),(d_{j-1},b_j),(b_{j-1},b_j)\,:\,1\le i\le m\le j\le n\}} \tag{iii}$$

$$\big(d_n(d).\, K_2\langle b_{n+1},d\rangle \ , \ d_n(d).\, K_2\langle b_{n+1},d\rangle\big) \in \mathcal{R}_{\{(a,b_i),(b_{i-1},b_i),(d_{j-1},b_j),(b_{j-1},b_j)\,:\,1\le i\le m+1\le j\le n+1\}} \tag{iv}$$

where we assume that $b_0 \stackrel{\text{def}}{=} c$ and $d_0 \stackrel{\text{def}}{=} e$, and that names a, b_i, d_i are all distinct. The initial processes $K_1\langle a,c,e\rangle$ and $K_2\langle c,e\rangle$ falls in (i)*, taking $n=0$.*

To see why all above pairs are necessary, consider, for instance, two processes of a pair in case (i)*. Process $K_1\langle a,b_n,d_n\rangle$ may perform the transition*

$$K_1\langle a,b_n,d_n\rangle \ \xrightarrow{([a=d_n],\overline{b_n}(b_{n+1}))} \ a(d).\, K_1\langle a,b_{n+1},d\rangle. \tag{3}$$

Since all hypothesis of the bisimulation clause of Definition 2.1 are satisfied, a matching transition by process $K_2\langle b_n,d_n\rangle$ is required. Such transition is:

$$K_2\langle b_n,d_n\rangle \ \xrightarrow{(\emptyset,\overline{b_n}(b_{n+1}))} \ d_n(d).\, K_2\langle b_{n+1},d\rangle.$$

Now, applying substitution $\{^a/_{d_n}\}$ *to the target process of the second transition, and updating the distinction, we obtain a pair of processes that falls in case* (ii). *Process* $K_1\langle a, b_n, d_n\rangle$ *may also perform the transition*

$$K_1\langle a, b_n, d_n\rangle \xrightarrow{(\emptyset, \overline{b_n}(b_{n+1}))} d_n(d).\, K_2\langle b_{n+1}, d\rangle, \tag{4}$$

that $K_2\langle b_n, d_n\rangle$ *matches with the transition*

$$K_2\langle b_n, d_n\rangle \xrightarrow{(\emptyset, \overline{b_n}(b_{n+1}))} d_n(d).\, K_2\langle b_{n+1}, d\rangle,$$

and the two target processes fall in case (iv).

Notice that, both in (3) *and in* (4), *the bound name* b_{n+1} *must be syntactically different from names* a, b_n *and* d_n, *which appear free in the source agent, due to the definition of canonical transitions. Moreover, since the bisimulation clause of Definition 2.1 requires the extruded names not to be in the given distinction, name* b_{n+1} *must be also different from all the names* b_i, *for* $i \leq n$.

Relation $\mathcal{R}$ *is so large because the indexing distinctions grow unboundedly during the computation. The techniques introduced in the next sections will allow us to shrink a lot the relation.*

3 Proof techniques for open bisimulation

In this section we develop some proof techniques for open bisimulation. In some later proofs, these techniques will dramatically reduce the size of the relations to exhibit and the work needed to establish bisimilarity results.

We wish to be able to compose proof techniques into more powerful ones. For this, we follow the schema in [12] and we roughly proceed as follows. We first define a notion of *progression* from a process relation $\mathcal{R}$ to a (possibly different) process relation $\mathcal{S}$, which holds if pairs of processes in $\mathcal{R}$ can match each other's actions and their derivatives are in $\mathcal{S}$. We then consider functions $\mathcal{F}$ on process relations. We are interested in the *sound* functions, namely those with the property that if a relation $\mathcal{R}$ progresses to $\mathcal{F}(\mathcal{R})$ then $\mathcal{R}$ only includes pairs of bisimilar processes; each of such functions yields a proof technique for bisimilarity. Finally, we isolate a few simple functions and function constructors and show that any function definable from them is sound. In this way, we can define some powerful functions, and we can use them as proof techniques for bisimilarity. The theory developed in [12] does not apply to our case, because our transition system is symbolic and because the notion of bisimilarity is different. We shall therefore produce an extension of (part of) that theory.

Definition 3.1 (progression) *Given two indexed relations* $\mathcal{R}$ *and* $\mathcal{S}$, *we say that* $\mathcal{R}$ progresses to $\mathcal{S}$, *written* $\mathcal{R} \rightarrowtail \mathcal{S}$, *if* $P\ \mathcal{R}_D\ Q$ *implies:*

- *Whenever* $P \xrightarrow{(M,\alpha)} P'$, *with* $\mathrm{bn}(\alpha) \cap \mathrm{fn}(Q, D) = \emptyset$ *and* M *respects* D, *then there is some* $Q \xrightarrow{(N,\beta)} Q'$ *such that*
  - $M \rhd N$,
  - $\alpha = \beta\sigma_M$, *and*
  - $P'\ \mathcal{S}_{D'}\ Q'\sigma_M$ *for* $D' \stackrel{\text{def}}{=} D^M_{(P,Q),\alpha}$.
- *The converse, with the role of* P *and* Q *exchanged.*

Therefore $\mathcal{R}$ is an open bisimulation if it progresses to itself, i.e., $\mathcal{R} \rightarrowtail \mathcal{R}$ holds.

Definition 3.2 *A* function on indexed relations *is a function from indexed relations to indexed relations. Such a function, say* $\mathcal{F}$, *is* sound *if* $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ *implies* $\mathcal{R} \subseteq \sim$, *for all indexed relations* $\mathcal{R}$.

Definition 3.3 *We define some basic function constructors:* union ($\cup$), composition ($\circ$) *and* iteration (*). *They take functions on indexed relations as arguments and return another function on indexed relations as a result.*

$$\begin{aligned}
(\textstyle\bigcup_{i\in I}\mathcal{F}_i)(\mathcal{R}) &\stackrel{\text{def}}{=} \textstyle\bigcup_{i\in I}\mathcal{F}_i(\mathcal{R}),\\
(\mathcal{F}\circ\mathcal{G})(\mathcal{R}) &\stackrel{\text{def}}{=} \mathcal{F}(\mathcal{G}(\mathcal{R})),\\
\mathcal{F}^n(\mathcal{R}) &\stackrel{\text{def}}{=} \mathcal{F}(\ldots(\mathcal{F}(\mathcal{R}))\ldots) \quad n \textit{ times},\\
\mathcal{F}^\star(\mathcal{R}) &\stackrel{\text{def}}{=} (\textstyle\bigcup_n \mathcal{F}^n)(\mathcal{R}).
\end{aligned}$$

where I is a countable indexing set.

We often abbreviate $\bigcup_{i\in\{1,2\}}\mathcal{R}_i$ as $\mathcal{R}_1\cup\mathcal{R}_2$. We now define some basic functions on indexed relations: Function $\mathcal{S}ub$ gives the closure of a relation under name substitutions; $\mathcal{S}q$ returns the "square" of a relation, that is, the composition of the relation with itself; $\mathcal{S}ym$ returns the symmetric of a relation; and $\mathcal{W}$ and $\mathcal{C}$ return the closure under weakening and strengthening of distinctions, respectively. In a bisimulation proof, using weakening on the derivatives of two matching transitions means adding pairs of names to the distinction; using strengthening means removing from the distinction pairs of names that do not appear free in the processes.

Definition 3.4 *Let $\mathcal{R}$ be an indexed relation; then the indexed relations $\mathcal{S}ub(\mathcal{R})$, $\mathcal{W}(\mathcal{R})$, $\mathcal{C}(\mathcal{R})$ and $\mathcal{S}q(\mathcal{R})$ are defined thus:*

- $(P,Q)\in(\mathcal{S}ub(\mathcal{R}))_D$ *if there are P',Q',D' and σ such that $(P',Q')\in\mathcal{R}_{D'}$, σ respects D', $P=P'\sigma$, $Q=Q'\sigma$ and $D=D'\sigma$.*
- $(P,Q)\in(\mathcal{W}(\mathcal{R}))_D$ *if $(P,Q)\in\mathcal{R}_{D'}$ for some $D'\subseteq D$.*
- $(P,Q)\in(\mathcal{C}(\mathcal{R}))_D$ *if $(P,Q)\in\mathcal{R}_{D\cup D'}$ for some D' such that $D'\cap\mathrm{fn}(P,Q)=\emptyset$.*
- $(P,Q)\in(\mathcal{S}q(\mathcal{R}))_D$ *if there is R such that $(P,R)\in\mathcal{R}_D$ and $(R,Q)\in\mathcal{R}_D$.*
- $(P,Q)\in(\mathcal{S}ym(\mathcal{R}))_D$ *if $(Q,P)\in\mathcal{R}_D$.*

The function that returns the transitive closure of a relation is definable from squaring and iteration as $\mathcal{S}q^\star$. The following theorem expresses the main result of the section.

Theorem 3.5 *Let $\mathcal{CF}$ be the smallest set of functions on indexed relations that contains functions $\mathcal{S}ub$, $\mathcal{W}$, $\mathcal{C}$, $\mathcal{S}q$ and $\mathcal{S}ym$ and that is closed under application of the function constructors $\cup$, $\circ$ and *. Then any function in $\mathcal{CF}$ is sound.*

In [12], the soundness of a set of functions definable from some basic functions and function constructors is proved in simple manner. The inductive definition of the set is exploited to prove that all functions enjoy a few properties which imply soundness. Among these properties, the most important is that of respecting progressions; this holds of a function $\mathcal{F}$ if whenever $\mathcal{R}\rightarrowtail\mathcal{S}$, then also $\mathcal{F}(\mathcal{R})\rightarrowtail\mathcal{F}(\mathcal{S})$.

In our case, where the transition system is symbolic and the bisimilarity is in the open style, we cannot follow the same proof schema because some of our basic functions do not respect progressions. However, something weaker holds: For each basic function $\mathcal{F}$ there is another function $\mathcal{G}$ in the set such that $\mathcal{R}\rightarrowtail\mathcal{S}$ implies $\mathcal{F}(\mathcal{R})\rightarrowtail\mathcal{G}(\mathcal{S})$. This is the content of Lemma 3.6, which is essential for the proof of Theorem 3.5.

Lemma 3.6 *Let $\mathcal{R}$ and $\mathcal{S}$ be two indexed relations. If $\mathcal{R}\rightarrowtail\mathcal{S}$ then:*

1. $\mathcal{S}ub(\mathcal{R})\rightarrowtail\mathcal{S}ub(\mathcal{S})$.
2. $\mathcal{W}(\mathcal{R})\rightarrowtail\mathcal{W}(\mathcal{S})$.
3. $\mathcal{C}(\mathcal{R})\rightarrowtail\mathcal{S}ub(\mathcal{C}(\mathcal{S}))$.

4. $\mathcal{S}q(\mathcal{R}) \rightarrowtail \mathcal{C}(\mathcal{S}q(\mathcal{W}(\mathcal{S}ub(\mathcal{S}))))$.

5. $\mathcal{S}ym(\mathcal{R}) \rightarrowtail \mathcal{S}ym(\mathcal{S})$.

Corollary 3.7 *Let $\mathcal{R}$ and $\mathcal{S}$ be two indexed relations. If $\mathcal{R} \rightarrowtail \mathcal{S}$ then*

$$(\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R}) \rightarrowtail (\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{S}).$$

Corollary 3.8 *Function $(\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star$ is sound.*

Since for each function $\mathcal{F} \in \mathcal{CF}$ and relation $\mathcal{R}$ it holds that $\mathcal{F}(R) \subseteq (\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R})$, function $(\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star$ is the largest function among those in $\mathcal{CF}$. Therefore the soundness of this function implies that of all functions in $\mathcal{CF}$ and proves Theorem 3.5.

In the remainder of the paper, we often prove open bisimilarities by exhibiting relations $\mathcal{R}$ that progress to $(\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R})$. The use of function $(\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star$ allows us to prove open bisimilarities working up to the closure of a relation under arbitrary combinations of name substitution, weakening, strengthening, transitivity and symmetry.

Corollary 3.9 shows that open bisimilarity is closed under name substitution, weakening, strengthening and squaring. These results, already known [13], are straightforward to establish using the the proof techniques outlined above.

Corollary 3.9

1. *$P \sim_D Q$ and σ respects D imply $P\sigma \sim_{D\sigma} Q\sigma$.*

2. *$P \sim_D Q$ and $D' \supseteq D$ imply $P \sim_{D'} Q$.*

3. *$P \sim_{D \cup D'} Q$ and $D' \cap \mathrm{fn}(P,Q) = \emptyset$ imply $P \sim_D Q$.*

4. *$P \sim_D R \sim_D Q$ implies $P \sim_D Q$.*

5. *$P \sim_D Q$ implies $Q \sim_D P$.*

Example 3.10 *Consider the processes $K_1\langle a,c,e\rangle$ and $K_2\langle c,e\rangle$ of Example 2.3. We can exploit the proof techniques in this section and prove that $K_1\langle a,c,e\rangle$ and $K_2\langle c,e\rangle$ are bisimilar, using this indexed relation $\mathcal{S}$:*

$$\begin{array}{rcll} \mathcal{S}_\emptyset & \stackrel{\mathrm{def}}{=} & \big\{\big(K_1\langle a,c,e\rangle, K_2\langle c,e\rangle\big)\ ,\ \big(a(d).\,K_1\langle a,b,d\rangle, a(d).\,K_2\langle b,d\rangle\big)\big\} & \\ \mathcal{S}_D & \stackrel{\mathrm{def}}{=} & \emptyset & \text{if } D \neq \emptyset \end{array} \tag{5}$$

Relation $\mathcal{S}$ is an open bisimulation up to $(\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star$. There are only two pairs in $\mathcal{S}$ — compare $\mathcal{S}$ with relation $\mathcal{R}$ of Example 2.3. Also note that any pair in $\mathcal{R}$ either can be obtained from one of the pairs in $\mathcal{S}$ by applying injective substitutions and weakening, or it is contained in the identity relation. The proof that $\mathcal{S} \rightarrowtail (\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{S})$ is straightforward.

3.1 The "strong" strengthening law

Corollary 3.9(3) shows that, if $P \sim_D Q$, then also $P \sim_E Q$ with $E \stackrel{\mathrm{def}}{=} D \cap \mathrm{fn}(P,Q)$, and represents the "expected" strengthening law for open bisimilarity. Now we want to prove a stronger law, which we call *strong strengthening*, and where $E \stackrel{\mathrm{def}}{=} (D \cap \mathrm{fn}(P)) \cup (D \cap \mathrm{fn}(Q))$; this law will be essential in what follows.

To see that strong strengthening is not an obviously-sound law, consider two processes P and Q such that $P \sim_{(a,b)} Q$, $\mathrm{fn}(P) = \{a,c\}$ and $\mathrm{fn}(Q) = \{b,c\}$. Suppose now that the transition $P \xrightarrow{([a=c],\tau)} P'$ is matched by the transition $Q \xrightarrow{(\emptyset,\tau)} Q'$; that is, $P' \sim_{(a,b)} Q'\{a/c\}$ holds. The "chasing-diagram" among these two pairs of processes and these two transitions may break if we apply the strong strengthening law: The strong strengthening of $P \sim_{(a,b)} Q$ is $P \sim_\emptyset Q$; then, by definition of $\sim$, the match between transitions $P \xrightarrow{([a=c],\tau)} P'$ and $Q \xrightarrow{(\emptyset,\tau)} Q'$ requires the pair $P' \sim_\emptyset Q'\{a/c\}$, which may not be the strong strengthening of $P' \sim_{(a,b)} Q'\{a/c\}$ (since it may be that $a,b \in \mathrm{fn}(Q'\{a/c\})$). The possibility of breaking chasing-diagrams, which does not happen with the simpler strengthening law of Corollary 3.9(3), is what makes it difficult the prove of soundness of the law.

Theorem 3.11 (strong strengthening) *Suppose $P \sim_D Q$ and let $E \stackrel{\text{def}}{=} \big(D \cap \text{fn}(P)\big) \cup \big(D \cap \text{fn}(Q)\big)$. Then also $P \sim_E Q$.*

Sketch of the proof: We take the indexed relation $\mathcal{R}$ that on a distinction E is

$$\mathcal{R}_E \stackrel{\text{def}}{=} \{(P,Q) \ : \ \exists D \text{ such that } P \sim_D Q \text{ and } E \supseteq \big(D \cap \text{fn}(P)\big) \cup \big(D \cap \text{fn}(Q)\big)\}.$$

Then we prove that the progression $\mathcal{R} \rightarrowtail (\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R})$ holds. Since, by Corollary 3.8, function $(\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star$ is sound, this progression implies $\mathcal{R} \subseteq \sim$. □

4 Making distinctions local to processes

The goal of this section is to make the indexing distinctions of open bisimulation local to processes. That is, rather than a bisimilarity indexed by constraints (the distinctions), we wish to have a bisimilarity on processes with a local constraint.

A *constrained process* is a pair $\langle P, D\rangle$, where P is a process and D is a distinction with $\text{n}(D) \subseteq \text{fn}(P)$. The requirement on names is used to keep distinctions "small" and to reduce the number of free names (in the algorithm of Section 8, it will allow for a better re-use of names).

The set $\mathcal{CP}$ of constrained processes is ranged over by A, B and C. If $A \stackrel{\text{def}}{=} \langle P, D\rangle$, then the *free* and *bound names* of A, written $\text{fn}(A)$ and $\text{bn}(A)$, are, respectively, the free and bound names of P. Moreover, $(\boldsymbol{\nu} a)A$ abbreviates $\langle (\boldsymbol{\nu} a)P, D - \{a\}\rangle$, and $A\sigma$ abbreviates $\langle P\sigma, D\sigma\rangle$. When clear from the context, we may call constrained processes simply *processes*. We call the relations on constrained processes *$\mathcal{CP}$-relations*; these are ranged over by $\mathcal{D}$ and $\mathcal{E}$. Transitions for constrained processes are defined from those for processes:

$$\frac{P \stackrel{(M,\alpha)}{\longrightarrow} P' \qquad M \text{ respects } D}{\langle P, D\rangle \stackrel{(M,\alpha)}{\longrightarrow} \big\langle P', D^M_{P,\alpha} \cap \text{fn}(P')\big\rangle}$$

It makes sense to compare only constrained processes whose distinctions are *compatible*, that is, identical on common names. This means that the hypothesis on names, under which the two processes are being compared, are consistent with each other.

Definition 4.1 *Two constrained processes $\langle P, D\rangle$ and $\langle Q, E\rangle$ are* compatible, *written $\langle P, D\rangle \Downarrow \langle Q, E\rangle$, if $D \cap \text{fn}(Q) = E \cap \text{fn}(P)$. A $\mathcal{CP}$-relation $\mathcal{D}$ is* compatible *if $A\ \mathcal{D}\ B$ implies $A \Downarrow B$.*

4.1 Bisimulation on constrained processes

We define bisimilarity on constrained processes and, then, we relate it to open bisimilarity of plain processes. Following the schema in Section 3, the new bisimilarity is given in terms of a notion of progression. We shall use progressions to define proof techniques for the bisimilarity.

Definition 4.2 *Let $\mathcal{D}$ and $\mathcal{E}$ be relations on constrained processes. We say that $\mathcal{D}$* progresses to *$\mathcal{E}$, written $\mathcal{D} \rightarrowtail \mathcal{E}$, if $\langle P, D\rangle\ \mathcal{D}\ \langle Q, E\rangle$ implies:*

- *Whenever $\langle P, D\rangle \stackrel{(M,\alpha)}{\longrightarrow} \langle P', D'\rangle$, with $\text{bn}(\alpha) \cap \text{fn}(Q) = \emptyset$, then there is some $\langle Q, E\rangle \stackrel{(N,\beta)}{\longrightarrow} \langle Q', E'\rangle$ such that:*
  - *$M \rhd N$,*
  - *$\alpha = \beta\sigma_M$, and*
  - *$\langle P', D'\rangle\ \mathcal{E}\ \Big\langle Q'\sigma_M, (D \cup E)^M_{(P,Q),\alpha} \cap \text{fn}(Q'\sigma_M)\Big\rangle$.*
- *The converse, with the roles of $\langle P, D\rangle$ and $\langle Q, E\rangle$ exchanged.*

A relation $\mathcal{D}$ on constrained processes is a $\frown$-bisimulation *if $\mathcal{D} \rightarrowtail \mathcal{D}$. We write $A \frown B$ if $A \Downarrow B$ and there exists a $\frown$-bisimulation $\mathcal{D}$ such that $(A, B) \in \mathcal{D}$.*

A $\frown$-bisimulation is not required to be compatible, whereas $\frown$ is so. Adding the requirement of compatibility into the definition of $\frown$-bisimulation would not change the resulting relation $\frown$, as shown by this lemma:

Lemma 4.3 *$A \frown B$ iff these exists a compatible $\frown$-bisimulation $\mathcal{D}$ such that $A\,\mathcal{D}\,B$.*

Corollary 4.4 *$\frown$ is the largest compatible $\frown$-bisimulation.*

The following theorem is the main result of this section and shows that relation $\sim$ on plain processes and relation $\frown$ on constrained processes are essentially two different representations of the same relation. In the light of this, in the remainder, with some abuse of terminology, we call $\frown$ *open bisimilarity on constrained processes* or, when not ambiguous, simply *open bisimilarity*.

Theorem 4.5 (characterisation of open bisimulation in terms of $\frown$)

1. *$P \sim_D Q$ iff $\langle P, D \cap \mathrm{fn}(P)\rangle \frown \langle Q, D \cap \mathrm{fn}(Q)\rangle$.*
2. *$\langle P, D\rangle \frown \langle Q, E\rangle$ iff $P \sim_{D\cup E} Q$.*

Sketch of the proof: To prove that $P \sim_D Q$ implies $\langle P, D \cap \mathrm{fn}(P)\rangle \frown \langle Q, D \cap \mathrm{fn}(Q)\rangle$ we show that the $\mathcal{CP}$-relation

$$\{(\langle P, D \cap \mathrm{fn}(P)\rangle, \langle Q, D \cap \mathrm{fn}(Q)\rangle) \;:\; P \sim_D Q\}$$

is a $\frown$-bisimulation (Lemma C.7 in Appendix C.1). To prove that $\langle P, D\rangle \frown \langle Q, E\rangle$ implies $P \sim_{D\cup E} Q$ we take the indexed relation $\mathcal{R}$ defined thus:

$$\mathcal{R}_F \stackrel{\mathrm{def}}{=} \{(P, Q) \;:\; \langle P, D\rangle \frown \langle Q, E\rangle \text{ and } F = D \cup E\}$$

and show that $\mathcal{R}$ is an $\sim$-bisimulation up to weakening, i.e., $\mathcal{R} \rightarrowtail \mathcal{W}(\mathcal{R})$ (Lemma C.8 in Appendix C.1); by Theorem 3.5 this implies $\mathcal{R} \subseteq \sim$. The other two implications are easy consequences of these ones. □

Relation $\frown$ is not transitive nor is it preserved by name substitutions. As a counterexample to transitivity, consider the processes:

$$A = \langle [a = b]\tau + \tau, (a, b)\rangle \qquad B = \langle \tau, \emptyset\rangle \qquad C = \langle [a = b]\tau + \tau, \emptyset\rangle;$$

it holds $A \frown B \frown C$, but $A \not\frown C$, since A and C are not compatible. These failures are not entirely surprising because the closures under transitivity and substitution ignore the compatibility between the local distinctions of related processes. By taking compatibility into account, we gain the following weak forms of closure (in the first assertion, compatibility is guaranteed by the requirement "σ respects D"):

Corollary 4.6

1. *If $\langle P, D \cap \mathrm{fn}(P)\rangle \frown \langle Q, D \cap \mathrm{fn}(Q)\rangle$ and σ respects D then $\langle P\sigma, D\sigma \cap \mathrm{fn}(P\sigma)\rangle \frown \langle Q\sigma, D\sigma \cap \mathrm{fn}(P\sigma)\rangle$.*
2. *If $A \frown C \frown B$ and $A \Downarrow B$ then $A \frown B$.*

This corollary is immediate to prove using the proof techniques for open bisimulation on constrained processes in the next subsection.

Example 4.7 *Consider the processes $K_1\langle a, c, e\rangle$ and $K_2\langle c, e\rangle$ of Example 2.3 and let*

$$A_1 \stackrel{\mathrm{def}}{=} \langle K_1\langle a, c, e\rangle, \emptyset\rangle \qquad \text{and} \qquad A_2 \stackrel{\mathrm{def}}{=} \langle K_2\langle c, e\rangle, \emptyset\rangle.$$

By Theorem 4.5, we can deduce $K_1\langle a, c, e\rangle \sim_\emptyset K_2\langle c, e\rangle$ if we can prove $A_1 \frown A_2$. Here is a $\frown$-bisimulation $\mathcal{D}$ that includes this pair of constrained processes. $\mathcal{D}$ has the following pairs:

$$\big(A_1 \;,\; A_2\big) \tag{i}$$

$$\textit{for all } b, d \qquad \big(\langle K_1\langle a, b, d\rangle, (a, b)\rangle \;,\; \langle K_2\langle b, d\rangle, \emptyset\rangle\big) \tag{i-bis}$$

$$\textit{for all } b, d \qquad \big(\langle a(d).\, K_1\langle a, b, d\rangle, (a, b)\rangle \;,\; \langle a(d).\, K_2\langle b, d\rangle, (a, b)\rangle\big) \tag{ii}$$

$$\textit{for all } b, d \qquad \big(\langle K_2\langle b, d\rangle, \emptyset\rangle \;,\; \langle K_2\langle b, d\rangle, \emptyset\rangle\big) \tag{iii}$$

$$\textit{for all } b, d \qquad \big(\langle d(d').\, K_2\langle b, d'\rangle, (b, d)\rangle \;,\; \langle d(d').\, K_2\langle b, d'\rangle, (b, d)\rangle\big) \tag{iv}$$

We only show that $\mathcal{D}$ is closed under the transitions of the initial pair (A_1, A_2); recall that this pair is $\big(\langle K_1\langle a,c,e\rangle,\emptyset\rangle\ ,\ \langle K_2\langle c,e\rangle,\emptyset\rangle\big)$. If $\langle K_1\langle a,c,e\rangle,\emptyset\rangle$ performs transition

$$\langle K_1\langle a,c,e\rangle,\emptyset\rangle \xrightarrow{([a=e],\overline{c}(b))} \langle a(d).\, K_1\langle a,b,d\rangle,(a,b)\rangle$$

then $\langle K_2\langle c,e\rangle,\emptyset\rangle$ can match it with

$$\langle K_2\langle c,e\rangle,\emptyset\rangle \xrightarrow{(\emptyset,\overline{c}(b))} \langle e(d).\, K_2\langle b,d\rangle,(b,e)\rangle.$$

Applying substitution $\{a/e\}$ to the derivative, we obtain $\langle a(d).\, K_2\langle b,d\rangle,(a,b)\rangle$. This process, and the derivative of the first transition, yield a pair in case (ii).

On the other hand, if $\langle K_1\langle a,c,e\rangle,\emptyset\rangle$ performs transition

$$\langle K_1\langle a,c,e\rangle,\emptyset\rangle \xrightarrow{(\emptyset,\overline{c}(b))} \langle e(d).\, K_2\langle b,d\rangle,(b,e)\rangle$$

then process $\langle K_2\langle c,e\rangle,\emptyset\rangle$ can match it with

$$\langle K_2\langle c,e\rangle,\emptyset\rangle \xrightarrow{(\emptyset,\overline{c}(b))} \langle e(d).\, K_2\langle b,d\rangle,(b,e)\rangle,$$

since the derivatives are related according to case (iv).

Let us compare relation $\mathcal{D}$ above with the indexed relation $\mathcal{R}$ of Example 2.3: Both relations are infinite; however, the infinity of $\mathcal{D}$ is only due to the choice of fresh names, whereas the infinity of $\mathcal{R}$ is due to the choice of fresh names and also to the growing of the distinctions.

4.2 Proof techniques

In Section 3 we have introduced proof techniques for open bisimulation on plain processes. Here we want to introduce similar techniques for open bisimulation on constrained processes. In this case, the functions for weakening and strengthening distinctions are less important, because the transition relation on constrained processes is defined so that the local distinctions are maintained as small as possible. Moreover, we follow the weak forms of substitution closure and squaring in Corollary 4.6 — calling $\mathcal{S}ub_W$ and $\mathcal{S}q_W$ the resulting functions. The definition of symmetry — function $\mathcal{S}ym$ — is as expected. Apart from this, the assertion and the proof of the main result in this section (Theorem 4.10) closely follow those of Theorem 3.5 in Section 3.

Definition 4.8 *A function $\mathcal{F}$, from $\mathcal{CP}$-relations to $\mathcal{CP}$-relations, is* sound *if $\mathcal{D} \rightarrowtail \mathcal{F}(\mathcal{D})$ implies $\mathcal{D} \subseteq \frown$, for all compatible $\mathcal{CP}$-relation $\mathcal{D}$.*

Definition 4.9 *Let $\mathcal{D}$ be a $\mathcal{CP}$-relation; then the $\mathcal{CP}$-relations $\mathcal{S}ub_W(\mathcal{R})$, $\mathcal{S}q_W(\mathcal{R})$ and $\mathcal{S}ym(\mathcal{R})$ are defined thus:*

1. *$(A,B) \in \mathcal{S}ub_W(\mathcal{D})$ if there exist P, Q, D, σ such that $\langle P, D \cap \mathrm{fn}(P)\rangle\ \mathcal{D}\ \langle Q, D \cap \mathrm{fn}(Q)\rangle$, σ respects D, $A = \langle P\sigma, D\sigma \cap \mathrm{fn}(P\sigma)\rangle$ and $B = \langle Q\sigma, D\sigma \cap \mathrm{fn}(Q\sigma)\rangle$.*
2. *$(A,C) \in \mathcal{S}q_W(\mathcal{D})$ if $A\ \mathcal{D}\ B\ \mathcal{D}\ C$ and $A \Downarrow C$.*
3. *$(A,B) \in \mathcal{S}ym(\mathcal{D})$ if $B\ \mathcal{D}\ A$.*

The function constructors *union* ($\cup$), *composition* ($\circ$) and *iteration* (*) are defined on functions on $\mathcal{CP}$-relations in the expected way (see also Section 3).

Theorem 4.10 *Let $\mathcal{CF}_W$ be the smallest set of functions on $\mathcal{CP}$-relations that contains functions $\mathcal{S}ub_W$, $\mathcal{S}q_W$ and $\mathcal{S}ym$, and that is closed under application of the function constructors $\cup$, $\circ$ and *. Any function in $\mathcal{CF}_W$ is sound.*

The crux of the proof of Theorem 4.10 is Lemma 4.11 below (similar to Lemma 3.6 of Section 3), from which Corollaries 4.12 and 4.13 are easy to derive.

Lemma 4.11 *Let $\mathcal{D}$ and $\mathcal{E}$ be two compatible $\mathcal{CP}$-relations. If $\mathcal{D} \rightarrowtail \mathcal{E}$ then:*

1. $\mathcal{S}ub_W(\mathcal{D}) \rightarrowtail (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})$.
2. $\mathcal{S}q_W(\mathcal{D}) \rightarrowtail (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})$.
3. $\mathcal{S}ym(\mathcal{D}) \rightarrowtail \mathcal{S}ym(\mathcal{E})$.

Corollary 4.12 *Let $\mathcal{D}$ and $\mathcal{E}$ be two compatible $\mathcal{CP}$-relations. If $\mathcal{D} \rightarrowtail \mathcal{E}$ then*

$$(\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \rightarrowtail (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}).$$

Corollary 4.13 *Function $(\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star$ is sound.*

In the following, to prove bisimilarity of a pair of constrained processes, we often use progressions of the form $\mathcal{D} \rightarrowtail (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D})$. Employing function $(\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star$ allows us to work up to (any combination of) name substitution, transitivity and symmetry.

Example 4.14 *We can use the proof techniques in this section to get a simpler proof of the $\frown$-bisimilarity of the constrained processes A_1 and A_2 of Example 4.7. Take relation $\mathcal{E}$ composed of the pairs:*

$$\big(A_1 \ , \ A_2\big), \tag{6}$$

$$\big(\langle a(d).\, K_1\langle a,b,d\rangle, (a,b)\rangle \ , \ \langle a(d).\, K_2\langle b,d\rangle, (a,b)\rangle\big), \tag{7}$$

$$\big(\langle K_1\langle a,b,d\rangle, (a,b)\rangle \ , \ \langle K_2\langle b,d\rangle, \emptyset\rangle\big). \tag{8}$$

It is easy to check that $\mathcal{E}$ is a $\frown$-bisimulation up to $(\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star$. By contrast with relation $\mathcal{D}$ of Example 4.7, relation $\mathcal{E}$ is finite. This is possible because, using the proof techniques in this section, it is not necessary to consider different choices for the bound name of an action — a single name suffices.

5 Non-redundant transitions

The goal of this section is to remove the dependency called Dep.2 in the Introduction from the bisimilarity $\frown$ obtained in the previous section. This dependency is determined by the different conditions in the labels of two matching actions, which may happen when one of the transitions is redundant.

Roughly, a transition $A \xrightarrow{(M,\alpha)} A'$ is redundant for open bisimulation if there is another transition $A \xrightarrow{(N,\beta)} A''$ which has a strictly weaker condition N and such that, when M holds, α is equal to β and A' is bisimilar to A''. As such the redundant transition can be eliminated without affecting the behaviour of A. Redundant transitions can be defined for an arbitrary relation $\mathcal{D}$ in place of open bisimulation.

Definition 5.1 (non-redundant transitions) *A transition $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ is* redundant *for a relation $\mathcal{D}$ on constrained processes if there is a transition $\langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', D''\rangle$ such that:*

- $M \not\triangleleft\triangleright N$,
- $\alpha = \beta\sigma_M$, *and*
- $\langle P', D'\rangle \ \mathcal{D} \ \big\langle P''\sigma_M, D''^{M}_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\big\rangle$.

A transition $A \xrightarrow{(M,\alpha)} A'$ is non-redundant *for relation $\mathcal{D}$, written $A \xrightarrow{(M,\alpha)} A' \in \mathrm{nr}(\mathcal{D})$, if it is not redundant for $\mathcal{D}$.*

The two results below show important properties of non-redundant transitions and motivate our interest for them. The first result shows that non-redundant transitions are sufficient to describe the behaviour of a process: For any $\mathcal{CP}$-relations $\mathcal{D}$ and transition from a constrained processes A, one can always find a transition from A that is non-redundant for $\mathcal{D}$ and that "covers" the first transition, in the sense that

the first can be seen as a "specialisation" of the second. The second lemma shows that, when comparing the transitions of two bisimilar processes, non-redundant transitions do not introduce dependencies on the labels: A transition from one of the processes that is non-redundant for open bisimilarity can be matched by a transition from the other process that has exactly the same label.

Lemma 5.2 *For each relation $\mathcal{D}$ and each transition $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ there is a transition $\langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', D''\rangle \in \mathrm{nr}(\mathcal{D})$ such that $M \rhd N$, $\alpha = \beta\sigma_M$ and*

$$\langle P', D'\rangle \, (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \, \big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\big\rangle.$$

Lemma 5.3 *If $A \frown B$ and $A \xrightarrow{(M,\alpha)} A' \in \mathrm{nr}(\frown)$ with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(B) = \emptyset$, then there is B' such that $B \xrightarrow{(M,\alpha)} B' \in \mathrm{nr}(\frown)$ and $A' \frown B'$.*

Example 5.4 *Consider the constrained processes A_1 and the relation $\mathcal{D}$ defined in Example 4.7. Transition*

$$A_1 \xrightarrow{([a=e],\overline{c}(b))} \langle a(d).\, K_1\langle a, b, d\rangle, (a, b)\rangle$$

is redundant w.r.t. relation $\mathcal{D}$: it is "covered" by the other transition

$$A_1 \xrightarrow{(\emptyset,\overline{c}(b))} \langle e(d).\, K_2\langle b, d\rangle, (e, b)\rangle$$

because

$$\langle a(d).\, K_1\langle a, b, d\rangle, (a, b)\rangle \; \mathcal{D} \; \langle a(d).\, K_2\langle b, d\rangle, (a, b)\rangle.$$

5.1 Non-redundant bisimulation

We formulate a characterization of $\frown$ that exploits the idea of non-redundant transitions. This characterization will not be used in the remainder of the paper, where all results directly rely on Lemmas 5.2 and 5.3. Nevertheless, we chose to present the characterisation because it is not hard to derive from the lemmas and it is of independent interest.

Definition 5.5 (non-redundant bisimulation) *A relation $\mathcal{D}$ is a* $\frown_{\mathrm{nr}}$-bisimulation *if $A \, \mathcal{D} \, B$ implies:*

- *Whenever $A \xrightarrow{(M,\alpha)} A' \in \mathrm{nr}(\mathcal{D})$ with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(B) = \emptyset$, then there is some $B \xrightarrow{(M,\alpha)} B' \in \mathrm{nr}(\mathcal{D})$ such that $A' \, \mathcal{D} \, B'$.*
- *The converse, with the roles of A and B exchanged.*

We write $A \frown_{\mathrm{nr}} B$ if $A \Downarrow B$ and there exists a $\frown_{\mathrm{nr}}$-bisimulation $\mathcal{D}$ such that $(A, B) \in \mathcal{D}$.

As for the definition of open bisimulation on constrained processes (Definition 4.2), so in the definition above we could require that a $\frown_{\mathrm{nr}}$-bisimulation only relates compatible processes (i.e., we could add condition $\mathcal{D} \subseteq \Downarrow$), without affecting relation $\frown_{\mathrm{nr}}$:

Lemma 5.6 *$A \frown_{\mathrm{nr}} B$ iff there exists some compatible $\frown_{\mathrm{nr}}$-bisimulation $\mathcal{D}$ such that $A \, \mathcal{D} \, B$.*

Moreover, $\frown_{\mathrm{nr}}$ is the largest *compatible* $\frown_{\mathrm{nr}}$-bisimulation (it is not the largest $\frown_{\mathrm{nr}}$-bisimulation because a $\frown_{\mathrm{nr}}$-bisimulation may include pairs of incompatible processes).

Corollary 5.7 *$\frown_{\mathrm{nr}}$ is the largest compatible $\frown_{\mathrm{nr}}$-bisimulation.*

Theorem 5.8 (characterisation of open bisimulation in terms of $\frown_{\mathrm{nr}}$) *Relations $\frown$ and $\frown_{\mathrm{nr}}$ coincide.*

Sketch of the proof: From Lemma 5.3, we derive that $\frown$ is an $\frown_{\mathrm{nr}}$-bisimulation, hence the inclusion $\frown \subseteq \frown_{\mathrm{nr}}$. For the opposite inclusion, by Lemma 4.3 it is sufficient to prove that $\mathcal{D} \subseteq \frown$, for each compatible $\frown_{\mathrm{nr}}$-bisimulation $\mathcal{D}$. For this, we use Lemma 5.2 to show that $\mathcal{D} \rightarrowtail (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D})$; then we can infer $\frown_{\mathrm{nr}} \subseteq \frown$ from Corollary 4.13. □

By contrast with $\frown$-bisimulation, the functional $\psi : 2^{\mathcal{CP}\times\mathcal{CP}} \to 2^{\mathcal{CP}\times\mathcal{CP}}$ associated to the definition of $\frown_{\rm nr}$-bisimulation is not monotone. The functional is so defined:

$A\ \psi(\mathcal{D})\ B$ if:

- Whenever $A \xrightarrow{(M,\alpha)} A' \in \mathrm{nr}(\mathcal{D})$ with $\mathrm{bn}(\alpha)\cap\mathrm{fn}(B) = \emptyset$, then there is some $B \xrightarrow{(M,\alpha)} B' \in \mathrm{nr}(\mathcal{D})$ such that $A'\ \mathcal{D}\ B'$.
- The converse, with the roles of A and B exchanged.

As a counterexample to monotonicity, take two processes A and B that can perform the transitions

$$\begin{array}{ll} A \xrightarrow{(\emptyset,\tau)} A', & A \xrightarrow{([a=b],\tau)} A'' \\ B \xrightarrow{(\emptyset,\tau)} B', & B \xrightarrow{([a=b],\tau)} B'' \end{array}$$

If we consider relations $\mathcal{D} \stackrel{\rm def}{=} \{(A',B'),(A'',B'')\}$ and $\mathcal{E} \stackrel{\rm def}{=} \mathcal{D} \cup \{(B'\{a/b\},B'')\}$ (we omit the symmetric pairs), then $\mathcal{D} \subseteq \mathcal{E}$ holds. However it holds that $\psi(\mathcal{D}) \not\subseteq \psi(\mathcal{E})$, since $(A,B) \in \psi(\mathcal{D})$, but $(A,B) \notin \psi(\mathcal{E})$. This happens since transition $A \xrightarrow{([a=b],\tau)} A''$ is non-redundant for both $\mathcal{D}$ and $\mathcal{E}$, whereas $B \xrightarrow{([a=b],\tau)} B''$ is non-redundant for $\mathcal{D}$ but is redundant for $\mathcal{E}$; hence the latter transition cannot be used to match $A \xrightarrow{([a=b],\tau)} A''$ in $\mathcal{E}$.

Non-redundant transitions allow us to characterize open bisimulation in a way which is more similar to ordinary bisimilarities of process calculi, because they avoid a dependency on labels of transitions. However, the non-monotonicity of functional ψ causes some difficulties when one tries to define an iterative method to compute open bisimilarity, as we shall do in Section 7, since standard fixed-point theorems cannot be used to prove the convergence of the method.

6 Active names

We now make the choice of bound names of matching transitions between bisimilar processes local to processes, i.e., we remove the dependency called Dep.1 in the Introduction. A local choice cannot be based on the free names, because bisimilar processes may have different sets of free names. In place of the free names, we shall use the *active names*, which are the same in bisimilar processes. The active names of a process are the smallest subset of free names which affect its behaviour. For instance, a is active in $a(b).\,0$ (a communication can occur at a) and in $[a=b]\overline{b}c.\,0$ (the output can occur only if $a=b$); but a is inactive in $(\nu b)(\overline{b}a.\,0)$ (the output $\overline{b}a$ is not observable) and in $[a=b]\overline{a}c.\,0 + \overline{b}c.\,0$ (component $[a=b]\overline{a}c.\,0$ yields a transition which is redundant, since covered by that yielded by the component $\overline{b}c.\,0$, and therefore could be ignored).

As we did for non-redundant transitions, we first define active names on a generic relation and then consider active names w.r.t. open bisimulation as a particular case.

Definition 6.1 *For a $\mathcal{CP}$-relation $\mathcal{D}$, the function $\mathrm{an}_{\mathcal{D}} : \mathcal{CP} \to 2^{\mathcal{N}}$, mapping a constrained process onto the set of its* active names w.r.t. $\mathcal{D}$*, is the smallest fixed-point of the monotone function $\psi : (\mathcal{CP} \to 2^{\mathcal{N}}) \to (\mathcal{CP} \to 2^{\mathcal{N}})$ defined thus:*

$$\psi(f)(A) = \bigcup_{\{M,\alpha,A' \mid A \xrightarrow{(M,\alpha)} A' \in \mathrm{nr}(\mathcal{D})\}} \mathrm{fn}(M,\alpha) \cup (f(A') - \mathrm{bn}(\alpha)).$$

If $x \in \mathrm{an}_{\mathcal{D}}(A)$, then we say that name x is active *w.r.t. $\mathcal{D}$ in process A; otherwise we say that x is* inactive *w.r.t. $\mathcal{D}$ in A.*

That is, $\mathrm{an}_{\mathcal{D}}$ is the least function which satisfies the condition:

Whenever $A \xrightarrow{(M,\alpha)} A' \in \mathrm{nr}(\mathcal{D})$, it holds that $\mathrm{fn}(M,\alpha) \cup (\mathrm{an}_{\mathcal{D}}(A') - \mathrm{bn}(\alpha)) \subseteq \mathrm{an}_{\mathcal{D}}(A)$.

The active names of a process are intended to be those which determine its behaviour. Therefore, we expect that behaviourally-equivalent processes have the same sets of active names and that, equivalently, restricting on an active name modifies the behaviour of a process. This is the content of the next two results.

Lemma 6.2 *If* $A \frown B$ *then* $\mathrm{an}_\frown(A) = \mathrm{an}_\frown(B)$.

Lemma 6.3 $x \in \mathrm{an}_\frown(A)$ *iff* $A \not\frown (\boldsymbol{\nu} x)A$.

The idea of using active names is not new. A similar idea has been used in [6] for CCS with value passing. Active names has also been used in [9] for the early and late semantics of the π-calculus. However, our definition of active names is different from that in [9]; there, a name x is defined active for process P if $(\boldsymbol{\nu} x)P$ is not bisimilar to P (which is similar to the content of Lemma 6.3). In the setting of [9], where the π-calculus language does not contain the matching operator and bisimilarity is given in the early and late styles, such a definition is sufficient, because the active names of a process are easy to compute (they are, roughly, the names which appear free in the transitions of P and of its derivatives). By contrast, in our case, as explained in the Introduction, computing active names is as difficult as computing bisimilarity itself. This explains our more sophisticated definition of active names as a fixed-point.

Example 6.4 *Consider the constrained process* A_1 *and the relation* $\mathcal{D}$ *defined in Example 4.7. Names* c *and* e *are active w.r.t.* $\mathcal{D}$ *in process* A_1 *because they appear free in the actions of the two non-redundant transitions*

$$A_1 \xrightarrow{(\emptyset,\overline{c}(b))} \xrightarrow{(\emptyset,e(d))} \langle K_2\langle b,d\rangle,\emptyset\rangle.$$

By contrast, there is no sequence of non-redundant transitions from A_1 *in whose actions name* a *occurs free. For instance, transition*

$$A_1 \xrightarrow{([a=e],\overline{c}(b))} \langle a(d).\, K_1\langle a,b,d\rangle,(a,b)\rangle$$

is redundant w.r.t. $\mathcal{D}$. *Consequently, name* a *is inactive w.r.t.* $\mathcal{D}$ *in* A_1.

6.1 Active-names bisimulation

In the *normalised transitions* below, the bound name in a transition of a process A is imposed to be the first inactive name in A; since the first inactive name may also occur free, rule `Norm2` redenominates its free occurrences to avoid accidental identifications. Moreover, normalised transitions are defined on top of the *non-redundant* transitions only.

Definition 6.5 *The* normalised transitions *for a relation* $\mathcal{D}$, *which have the form* $A \overset{(M,\alpha)}{\longmapsto}_{\mathcal{D}} A'$, *are defined from the two following inference rules, where* $v \stackrel{\mathrm{def}}{=} \min\{\mathcal{N} - \mathrm{an}_{\mathcal{D}}(A)\}$ *and* $y \stackrel{\mathrm{def}}{=} \min\{\mathcal{N} - \mathrm{fn}(A)\}$:

$$\texttt{Norm1}\quad \frac{A \xrightarrow{(M,\alpha)} A' \in \mathrm{nr}(\mathcal{D}) \qquad \alpha = \tau \quad or \quad \alpha = \overline{a}b}{A \overset{(M,\alpha)}{\longmapsto}_{\mathcal{D}} A'}$$

$$\texttt{Norm2}\quad \frac{A \xrightarrow{(M,\alpha)} A' \in \mathrm{nr}(\mathcal{D}) \qquad \alpha = a(y) \quad or \quad \alpha = \overline{a}(y)}{A \overset{(M,\beta)}{\longmapsto}_{\mathcal{D}} A'\{v/y\, y/v\}} \text{ where } \beta \stackrel{\mathrm{def}}{=} \begin{cases} a(v) & \text{if } \alpha = a(y) \\ \overline{a}(v) & \text{if } \alpha = \overline{a}(y) \end{cases}$$

Now we present a characterization of open bisimulation which uses the normalised transitions introduced above.

Definition 6.6 (active-names bisimulation) *A relation* $\mathcal{D}$ *is a* $\frown_{\mathrm{an}}$-bisimulation *if* $A\ \mathcal{D}\ B$ *implies:*

- *Whenever* $A \overset{(M,\alpha)}{\longmapsto}_{\mathcal{D}} A'$ *then there is some* $B \overset{(M,\alpha)}{\longmapsto}_{\mathcal{D}} B'$ *such that* $A'\ \mathcal{D}\ B'$;
- *The converse, with the role of* A *and* B *exchanged.*

We write $A \frown_{\mathrm{an}} B$ *if* $A \Downarrow B$ *and there exists a* $\frown_{\mathrm{an}}$*-bisimulation* $\mathcal{D}$ *such that* $(A,B) \in \mathcal{D}$.

The definition of $\frown_{\mathrm{an}}$ is given in terms of the compatibility relation ($\Downarrow$) and of a notion of bisimulation, in a way similar to the definitions of $\frown$ and $\frown_{\mathrm{nr}}$ in the two previous sections. In the cases of $\frown$ and $\frown_{\mathrm{nr}}$, we saw that the requirement of compatibility can be pushed inside the definition of bisimulation. Indeed, both $\frown$ and $\frown_{\mathrm{nr}}$ can also be defined as the largest compatible $\frown$-bisimulation and $\frown_{\mathrm{nr}}$-bisimulation, respectively (Corollaries 4.4 and 5.7).

By contrast, the analogous property fails for $\frown_{\mathrm{an}}$-bisimulation: Relation $\frown_{\mathrm{an}}$ changes if the compatibility requirement is pushed inside the definition of bisimulation. That is, $\frown_{\mathrm{an}}$ is *not* a $\frown_{\mathrm{an}}$-bisimulation. To see this point, consider the processes

$$A \stackrel{\mathrm{def}}{=} \langle P, (a, c)\rangle \qquad B \stackrel{\mathrm{def}}{=} \langle Q, \emptyset\rangle$$

for

$$P \stackrel{\mathrm{def}}{=} ([a=c]\tau.\,0 + \tau.\,0) \mid c(x).\,0 \qquad Q \stackrel{\mathrm{def}}{=} ([b=c]\tau.\,0 + \tau.\,0) \mid c(x).\,0.$$

Processes A and B are in relation $\frown_{\mathrm{an}}$: They are compatible and there are $\frown_{\mathrm{an}}$-bisimulations including them as a pair. Suppose that $a = \min\{\mathcal{N} - \mathrm{an}_{\frown}(A)\} = \min\{\mathcal{N} - \mathrm{an}_{\frown}(B)\} = \min\{\mathcal{N} - \mathrm{fn}(B)\}$, and that $b = \min\{\mathcal{N} - \mathrm{fn}(A)\}$. Consider now the transition

$$A \stackrel{(\emptyset, c(a))}{\longmapsto}_{\frown} \langle([b=c]\tau + \tau) \mid 0, (b, c)\rangle \stackrel{\mathrm{def}}{=} A'.$$

The only normalised transition from B with the same label is

$$B \stackrel{(\emptyset, c(a))}{\longmapsto}_{\frown} \langle([b=c]\tau + \tau) \mid 0, \emptyset\rangle \stackrel{\mathrm{def}}{=} B'.$$

But A' and B' are not in relation $\frown_{\mathrm{an}}$, because they are not compatible.

Despite this important difference, we can prove that $\frown_{\mathrm{an}}$ coincides with $\frown$ and, hence, with $\frown_{\mathrm{nr}}$. (These facts are not in contradiction, because $\frown_{\mathrm{an}}$ is defined on a different transition system.)

Theorem 6.7 (characterization of open bisimulation in terms of $\frown_{\mathrm{an}}$) *Relation $\frown$ and $\frown_{\mathrm{an}}$ coincide.*

Sketch of the proof: The proof that $\frown \subseteq \frown_{\mathrm{an}}$ is presented in Appendix F; we prove it after deriving some of the results in the next section (where we define an iterative method for computing $\frown_{\mathrm{an}}$). Intuitively, the proof consists in: Defining a functional Φ such that the limit of the iterated applications of Φ starting from the universal relation $\mathcal{CP} \times \mathcal{CP}$ is a $\frown_{\mathrm{an}}$-bisimulation; proving, by transfinite induction, that $\frown$ is contained in all relations obtained from these iteration steps.
The proof that $\frown_{\mathrm{an}} \subseteq \frown$ is reported in Lemma E.8 of the appendices. Given a $\frown_{\mathrm{an}}$-bisimulation $\mathcal{D}$, we define a new relation $\mathcal{E}$ that contains all compatible pairs of $\mathcal{D}$ and satisfies certain additional properties, and prove a progression of the form $\mathcal{E} \rightarrowtail (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})$. By Corollary 4.13, this implies $\mathcal{E} \subseteq \frown$ and proves the thesis.
Unfortunately, we cannot just prove that $\mathcal{D}$ is an open bisimulation (even up to $(\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star$), because this is not true. This is caused by the difference between the two transition systems on which the definitions of $\frown$-bisimulation and $\frown_{\mathrm{an}}$-bisimulation are based. More precisely, it is caused by the fact that the latter transition system has rules, like `Norm2`, which employ substitutions that are defined using the free names of the source process. As a consequence, since equivalent processes may have different sets of free names, two equivalent processes may have matching transitions that involve two different substitutions. □

7 The iterative approach

The characterisation of open bisimilarity as $\frown_{\mathrm{an}}$ in the previous section avoids all dependencies Dep.1–3: In the Definition 6.6 of $\frown_{\mathrm{an}}$ the labels of two matching transitions are exactly the same, no side conditions on the bound names appear, and all distinctions are local. Using this characterisation, we now define an iterative method that computes open bisimilarity through a sequence of approximations that progressively refine a partition of the set of all constrained processes. There are two levels of iteration: The first is given by a class of functions $\Psi_{\mathcal{D}}$, which refine the partition w.r.t. the normalised transitions corresponding to $\mathcal{D}$. The second level is given by a function Φ that computes the greatest fixed-point of $\Psi_{\mathcal{D}}$.

Definition 7.1 *Let $\mathcal{D}$ be a relation on constrained processes. Function $\Psi_{\mathcal{D}} : 2^{\mathcal{CP}\times\mathcal{CP}} \to 2^{\mathcal{CP}\times\mathcal{CP}}$ is defined as follows: $(A,B) \in \Psi_{\mathcal{D}}(\mathcal{E})$ if:*

- *$A\ \mathcal{D}\ B$.*
- *Whenever $A \stackrel{(M,\alpha)}{\longmapsto}_{\mathcal{D}} A' \in \mathrm{nr}(\mathcal{D})$, then $B \stackrel{(M,\alpha)}{\longmapsto}_{\mathcal{D}} B' \in \mathrm{nr}(\mathcal{D})$ and $A'\ \mathcal{E}\ B'$.*
- *The converse, with the role of A and B exchanged.*

Function $\Phi : 2^{\mathcal{CP}\times\mathcal{CP}} \to 2^{\mathcal{CP}\times\mathcal{CP}}$ is defined as follows: $\Phi(\mathcal{D}) = \mathrm{gfp}(\Psi_{\mathcal{D}})$.

Function Φ is well-defined because, for each relation $\mathcal{D}$, function $\Psi_{\mathcal{D}}$ is monotone.

By definition of greatest fixed-point, $\Phi(\mathcal{D})$ is the largest relation $\mathcal{E}$ such that $A\ \mathcal{E}\ B$ implies:

- $A\ \mathcal{D}\ B$.
- Whenever $A \stackrel{(M,\alpha)}{\longmapsto}_{\mathcal{D}} A' \in \mathrm{nr}(\mathcal{D})$, then $B \stackrel{(M,\alpha)}{\longmapsto}_{\mathcal{D}} B' \in \mathrm{nr}(\mathcal{D})$ and $A'\ \mathcal{E}\ B'$.
- The converse, with the role of A and B exchanged.

Moreover, by the definitions of Φ and of $\frown_{\mathrm{an}}$-bisimulation, it follows that

Proposition 7.2 *A $\mathcal{CP}$-relation $\mathcal{D}$ is a fixed-point of Φ if and only if $\mathcal{D}$ is a $\frown_{\mathrm{an}}$-bisimulation.*

Function Φ appears suited to extracting a partition refinement algorithm because derivatives and transitions of processes are computed locally.

Unfortunately, Φ is *not* monotone. (As a counterexample, one can reuse the processes and the relations exhibited in Subsection 5.1 for a counterexample to the monotony of the functional for non-redundant bisimulations.) Therefore we do not know whether Φ has a greatest fixed-point and, even if it existed, we could not use Tarsky's theorem to compute it. As a consequence, to obtain an algorithmic characterization of open bisimulation in terms of Φ, we have to provide a specific proof that the iterated application of Φ on the universal relation $\mathcal{CP}\times\mathcal{CP}$ is convergent, and that the limit contains open bisimulation. We define

$$\begin{array}{rcll} \Phi^0 & \stackrel{\mathrm{def}}{=} & \mathcal{CP}\times\mathcal{CP} & \\ \Phi^{i+1} & \stackrel{\mathrm{def}}{=} & \Phi(\Phi^i) & \\ \Phi^i & \stackrel{\mathrm{def}}{=} & \bigcap_{j<i} \Phi^j & \text{if } i \text{ is an limit ordinal.} \end{array}$$

Our algorithm will calculate the limit of the sequence $\{\Phi^i\}_i$.

Definition 7.3 *We define the $\mathcal{CP}$-relation $\simeq$ as $\simeq \stackrel{\mathrm{def}}{=} \lim_i \Phi^i$.*

The limit exists since, by definition of Φ, the chain $\{\Phi^i\}_i$ is non-increasing. Moreover, as a consequence of Proposition 7.2, $\simeq$ is an $\frown_{\mathrm{an}}$-bisimulation:

Corollary 7.4 *Relation $\simeq$ is an $\frown_{\mathrm{an}}$-bisimulation.*

If the pairs of incompatible processes are discarded from the fixed-point $\simeq$, the resulting relation is precisely open bisimilarity:

Theorem 7.5 (Characterization of open bisimulation in terms of $\simeq$) *Relations $\frown$ and $\simeq \cap \Downarrow$ coincide.*

In the definition of Φ, we cannot demand compatibility between related processes, because the compatibility relation $\Downarrow$ is not transitive, as witnessed by processes

$$A \stackrel{\mathrm{def}}{=} \langle (\boldsymbol{\nu} x)(\overline{x}a.\,\overline{a}b.\,0), \emptyset\rangle \qquad B \stackrel{\mathrm{def}}{=} \langle 0, \emptyset\rangle \qquad C \stackrel{\mathrm{def}}{=} \langle (\boldsymbol{\nu} x)(\overline{x}a.\,\overline{a}b.\,0), (a,b)\rangle$$

where $A \Downarrow B$ and $B \Downarrow C$, but $A \Downarrow C$ does not hold. If we added clause $A \Downarrow B$, then function Φ would not preserve transitivity, and hence we could not use Φ to define a partition refinement algorithm.

The proof of Theorem 7.5 is based on two important properties of function Φ. The first property says that, if relation $\mathcal{D}$ includes $\frown$ and satisfies some other conditions, then $\frown \subseteq \Phi(\mathcal{D}) \subseteq \mathcal{D}$.

1 Generate the saturated state graphs (S_P, T_P) and (S_Q, T_Q) for the processes $\langle P, D \cap \mathrm{fn}(P)\rangle$ and $\langle Q, D \cap \mathrm{fn}(Q)\rangle$.

2 Initialize $\mathcal{P}$ to be the singleton partition on $S_P \cup S_Q$ (i.e., all processes in the same class).

3 Repeat the following steps until partition $\mathcal{P}$ becomes stable:

- 3.1 Set *NonRed* to be the subset of transitions in $T_P \cup T_Q$ which are non-redundant for $\mathcal{P}$.
- 3.2 Compute the active names w.r.t. $\mathcal{P}$, for each process in $S_P \cup S_Q$.
- 3.3 If necessary, refine the partition $\mathcal{P}$ so that processes in the same class have the same set of active names.
- 3.4 Compute the normalised transitions for $\mathcal{P}$ generated by the transitions in *NonRed*.
- 3.5 Apply the Paige and Tarjan refinement algorithm [11] on partition $\mathcal{P}$ using, as transitions, the normalised ones computed in Step 3.4. Redefine $\mathcal{P}$ to be the resulting partition.

4 Check if $\langle P, D \cap \mathrm{fn}(P)\rangle$ and $\langle Q, D \cap \mathrm{fn}(Q)\rangle$ are in the same class.

Table 2: Schema of the algorithm to check $P \sim_D Q$.

Definition 7.6 *Let $\mathcal{D}$ and $\mathcal{E}$ be relations. We say that $\mathcal{D}$* is closed for $\mathrm{an}_{\mathcal{E}}$-injective substitutions*, if whenever $A\ \mathcal{D}\ B$ and σ_A, σ_B are injective substitutions which coincide on $\mathrm{an}_{\mathcal{E}}(A) \cap \mathrm{an}_{\mathcal{E}}(B)$ then $A\sigma_A\ \mathcal{D}\ B\sigma_B$.*

Lemma 7.7 *Let $\mathcal{D}$ be an equivalence closed for $\mathrm{an}_{\mathcal{D}}$-injective substitutions. If $\frown \subseteq \mathcal{D}$ then $\frown \subseteq \Phi(\mathcal{D}) \subseteq \mathcal{D}$.*

The second property shows that function Φ preserves the properties of a $\mathcal{CP}$-relations in Lemma 7.7.

Lemma 7.8 *If $\mathcal{D}$ is an equivalence closed for $\mathrm{an}_{\mathcal{D}}$-injective substitutions, then also $\Phi(\mathcal{D})$ is an equivalence closed for $\mathrm{an}_{\Phi(\mathcal{D})}$-injective substitutions.*

The two propositions guarantee that all relations Φ^i contain $\frown$. This must also be true for the fixed-point $\simeq$, hence:

Corollary 7.9 $\frown \subseteq \simeq \cap \Downarrow$.

Moreover, since $\simeq$ is an $\frown_{\mathrm{an}}$-bisimulation (Proposition 7.2), by Theorem 6.7 also the converse inclusion holds. This, plus Corollary 7.9, proves Theorem 7.5.

Thanks to Theorem 4.5, the iterative method described in this section can be used to check open bisimulation also for plain processes:

Corollary 7.10 *$P \sim_D Q$ if and only if $\langle P, D \cap \mathrm{fn}(P)\rangle \simeq \langle Q, D \cap \mathrm{fn}(Q)\rangle$.*

In next section we describe an algorithm for checking open bisimilarity. Each iteration of the algorithm precisely corresponds to an application of function Φ.

8 The algorithm

The algorithm we propose for open bisimulation is based on the characterisation in Corollary 7.10. The main steps are sketched in Table 2. A partition on processes can be viewed as a relation in which all processes in the same class are related; in this way, we can talk of non-redundant transitions, active names and normalised transitions w.r.t. a partition. Some comments on the steps in the table:

Step 1: The *saturated state graph* of a process A_0 is the pair (S,T) where S and T are, respectively, the minimal set of constrained processes and of transitions between processes in S such that $A_0 \in S$ and such that S and T are closed under the following operations:

Sat-trans If $A \in S$ and $A \xrightarrow{(M,\alpha)} A'$ with $\mathrm{bn}(\alpha) \subseteq \{\min\{\mathcal{N}-\mathrm{fn}(A)\}\}$, then $A' \in S$ and $A \xrightarrow{(M,\alpha)} A' \in T$.

Sat-nonred If $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle \in T$ and $\langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', D''\rangle \in T$ with $M \rhd N$ and $\alpha = \beta\sigma_M$, then $\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\rangle \in S$.

Sat-bunch Let $y \stackrel{\mathrm{def}}{=} \min\{\mathcal{N} - \mathrm{fn}(A)\}$; if $A \xrightarrow{(M,\alpha)} A' \in T$ with $\mathrm{bn}(\alpha) = \{y\}$, then $A'\{^v/_y\, ^y/_v\} \in S$, for all $v < y$ (where $<$ is the strict order assumed on $\mathcal{N}$).

Sat-nonred and **sat-bunch** are necessary, respectively, for the run-time computation of non-redundant transitions and for the targets of normalised transitions. Note that **sat-trans** requires a single instantiation of bound names of actions. In **sat-bunch**, v is *strictly* below $\min\{\mathcal{N} - \mathrm{fn}(A)\}$, since the latter name, which is surely inactive for A, has already been considered in **sat-trans**.

Step 3: Each cycle corresponds to an application of function Φ of Section 6.

Step 3.1: Following Definition 5.1, to compute the non-redundant transitions quickly, for each transition $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ we can keep a list of the processes $\langle P''\sigma_M, D^P_{M,\alpha} \cap \mathrm{fn}(P''\sigma_M)\rangle$ such that $\langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', D''\rangle$ with $M \rhd N$ and $\alpha = \beta\sigma_M$; then the given transition is non-redundant for $\mathcal{P}$ if and only if none of the processes in the list is in the equivalence class of $\langle P', D'\rangle$.

Step 3.2: Following Definition 6.1, the active names can be efficiently computed via a transitive-closure algorithm (the non-redundant transitions have already been computed in Step 3.1).

Step 3.4: The normalised transitions are generated applying inference rules `Norm1` and `Norm2` to each transition in *NonRed*. (The derivatives of normalised transitions are in $S_P \cup S_Q$, so no new process needs to be added.)

The algorithm terminates if the saturated state spaces, produced from the input processes in Step 1, are finite. This is the case for finite-control processes (up to garbage collection of restrictions, i.e., the law "if $x \notin \mathrm{fn}(P)$ then $(\boldsymbol{\nu} x)P = P$").

The algorithm can be used to produce the minimal realisation of a process P w.r.t. a distinction D — for this case it suffices to generate only the saturated state graph of $\langle P, D \cap \mathrm{fn}(P)\rangle$ in Step 1. If we take processes with normalised transitions only (normalised transitions for processes are defined as for constrained processes — just replace A with P and A' with P' in Definition 6.1), then the process returned by the algorithm (i.e., the one extracted from the final partition, where transitions are those computed in the last execution of Step 3.4) is minimal in the number of states, normalised transitions, and the number of free names among all processes in the relation $\sim_D$ with P.

Now we present two examples of application of the algorithm. In both of them we suppose that names are so ordered: $a < b < c < \cdots$.

Example 8.1 *Figure 1 shows a subset of the saturated state graph for processes A_1 and A_2 of Example 4.7. The target states of the dashed arrows are added to the graph by rule* **sat-bunch***; the target states of non-dashed arrows are added by rule* **sat-trans***. For instance, state 1 can perform action $\overline{c}(b)$ and become state 4; so there is a non-dashed arrow from state 1 to 4. However, if name a is inactive in state 1, then the bound name of the action should be a itself, and the target state should be state 6, obtained from 4 applying substitution $\{^a/_b\, ^b/_a\}$; so there is a dashed arrow from state 1 to 6.*

Note that the two non-dashed transitions that exit from 1 satisfy the conditions for applying rule **sat-nonred***; as a consequence, state 10 is added to the graph (this state is also added by rule* **sat-trans** *applied to state 9).*

Initially all states are in the same class of the partition. In this situation the transition from state 1 to state 2 is redundant and, therefore, name a is inactive in state 1 (there is no sequence of non-redundant transitions in which a is free). When the Paige and Tarjan refinement algorithm is run (step 3.5 of our algorithm), the target of the right transition from state 1 is state 6, rather than state 4. Similarly, a is inactive in state 3 and the target of its transition is state 12.

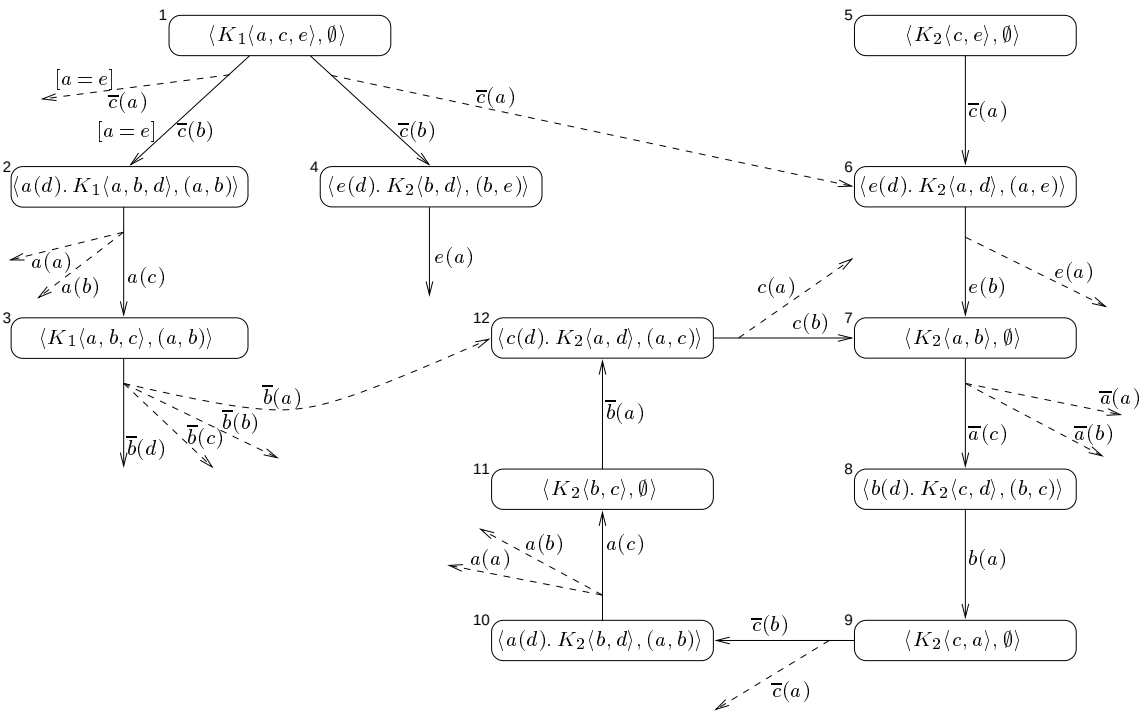

Figure 1: Saturated graph corresponding to the processes A_1 and A_2 of Example 8.1.

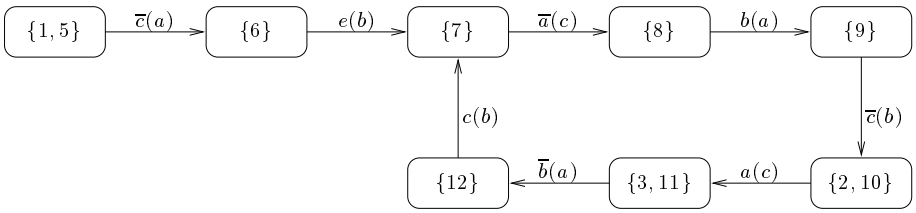

Figure 2: Minimal realisation for the processes A_1 and A_2 of Example 8.1.

The partition refinement algorithm refines the partition as follows:

$$\{1,5\}\quad\{2,10\}\quad\{3,11\}\quad\{4\}\quad\{6\}\quad\{7\}\quad\{8\}\quad\{9\}\quad\{12\}.$$

This is the stable partition: since states 2 and 10 are in the same class, transition from 1 to 2 remains redundant, and name a remains inactive in state 1. The algorithm correctly recognises A_1 and A_2 as open bisimilar, since states 1 and 5 end up in the same class.

The minimal realisation for A_1 and A_2 yielded by the algorithm is represented in Figure 2. Each state corresponds to a class of the partition; class $\{4\}$ does not appear, since it is not reachable from the initial state. The minimal realisation is identical to the graph generated by A_2 using normalised transitions. In this sense we can consider process A_2 as minimal.

Example 8.2 *Consider the processes $H\langle a,c\rangle$ and $K\langle c\rangle$, where constants H and K are so defined:*

$$H \stackrel{\text{def}}{=} (a,c)\big(c(b).\,(\tau.\,\overline{c}b.\,H\langle a,c\rangle + [a=b]\tau.\,\overline{c}c.\,K\langle a,c\rangle)\big) \qquad \textit{and} \qquad K \stackrel{\text{def}}{=} (c)\big(c(b).\,\tau.\,\overline{c}b.\,K\langle c\rangle\big).$$

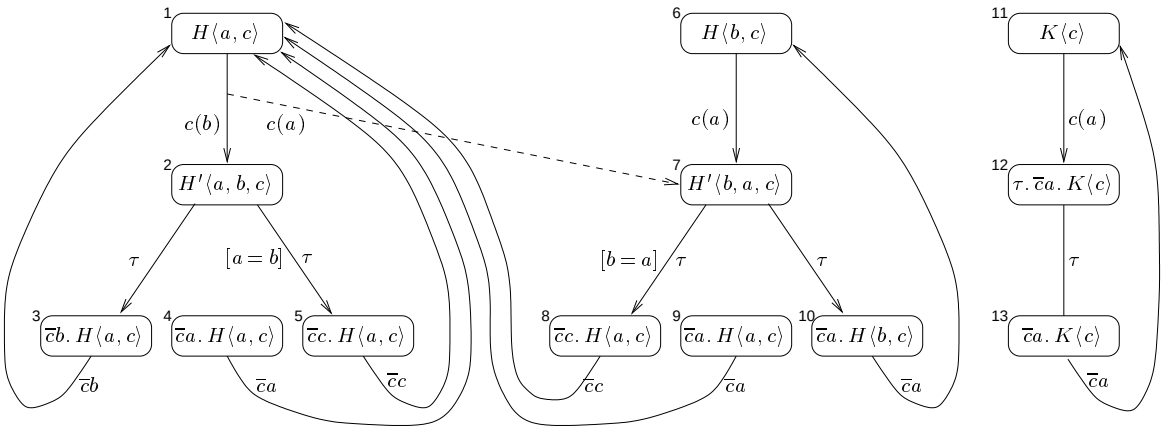

Figure 3: Saturated graph corresponding to the processes $H\langle a,c\rangle$ and $K\langle c\rangle$ of Example 8.2.

The saturated state graph corresponding to the processes[1] *is represented in Figure 3, where we have used the auxiliary constant*

$$H' \stackrel{\text{def}}{=} (a,b,c)\big(\tau.\,\overline{c}b.\,H\langle a,c\rangle + [a=b]\tau.\,\overline{c}c.\,H\langle a,c\rangle\big).$$

State 7 is added to the graph by rule **sat-bunch** *(this state is obtained from state 2 applying substitution* $\{{}^{a}\!/_{b}\,{}^{b}\!/_{a}\}$*). States 4 and 9 are added by rule* **sat-nonred**. *For instance, state 4 (namely* $\overline{c}a.\,H\langle a,c\rangle$*) is obtained from state 3 (namely* $\overline{c}b.\,H\langle a,c\rangle$*) applying substitution* $\{{}^{a}\!/_{b}\}$*; this substitution is generated by the condition in the transition from state 2 to state 5.*

In this example, two iterations of the loop (step 3) of our algorithm are needed for the partition to stabilise. In the first iteration, all the states are in the same class. The transition from 2 to 5 is redundant since covered by the transition from 2 to 3. Similarly, the transition from 7 to 8 is redundant. Name a is inactive in state 1, so it can be used as bound name in the normalised input transition from state 1.

At the end of the first iteration the partition consists of the following classes:

$$\{1,6,11\}\quad\{2\}\quad\{3\}\quad\{4,9,10,13\}\quad\{5,8\}\quad\{7,12\}.$$

At this point, state 1 and 11, corresponding to the processes $H\langle a,c\rangle$ *and* $K\langle c\rangle$*, are still in the same class.*

In the second iteration, since states 4 and 5 are not in the same class, the transition from 2 to 5 becomes non-redundant. As a consequence, name a becomes active in state 2 and, therefore, in state 1. In this situation, name b has to be used as bound name in the transition from 1, and the target of this transition becomes state 2. Also the transition from 7 to 8 becomes non-redundant, and name b becomes active in state 6; however, this does not affect the target of the transition from 6. At the end, the classes of the partition are as follows:

$$\{1\}\quad\{6\}\quad\{11\}\quad\{2\}\quad\{3\}\quad\{4,9\}\quad\{10\}\quad\{13\}\quad\{5,8\}\quad\{7\}\quad\{12\}.$$

The algorithm correctly recognises that $H\langle a,c\rangle$ *and* $K\langle c\rangle$ *are not open bisimilar, since states 1 and 11 end up in the different classes.*

9 Conclusions

We have presented various characterisations of open bisimulation. From the last one, we have extracted a partition refinement algorithm. It can be used on finite control processes for checking bisimilarity and for producing minimal realisations of processes.

[1] Since all the distinctions are empty, we have omitted them in the figure and in the discussion of this example.

Complexity. Paige-Tarjan algorithm has a worst-case running time $O(m \log n)$, on the number m of transitions and n of states; in our case, m and n are the number of transitions and of states obtained after initialisation (Step 1 in Table 2). The Paige-Tarjan routine may be applied several times (Step 3 in Table 2), but at most it runs n times, since at each iteration at least one split of the partition is made. Hence, we get a worst case running time $O(mn \log n)$.

However, the algorithm is exponential w.r.t. the syntactic length of the processes. The exponentiality is caused by the expansion of the parallel component of processes, as in CCS, and by value communication, as in data-dependent programs [6]. Surprisingly, the degree of exponentiation in the π-calculus is similar to that of CCS [4, 9]. Against this exponential bounds, the possibility of minimising process representations, offered by the algorithm, becomes important: As it happens in CCS, large-scale examples become tractable if process subcomponents are first minimised.

Active names and non-redundant transitions. The algorithm presented makes no attempts at uncovering active names and non-redundant transitions in initialisation. This may cause a large addition of states in the saturation procedure (operations **sat-nonred** and **sat-bunch** in Step 1) and a high number of iterations of the Paige-Tarjan routine.

However, a quick analysis of the processes could often produce reasonable estimates of active names and non-redundant transitions. For instance, all transitions with a true (i.e., empty) condition are non-redundant, and all free names in their labels are active. We believe that this direction can lead to significant improvements.

In non-trivial processes, like those used in the specification and the implementation of the handover protocol in the GSM Public Land Mobile Network [10], all free names are active, and all transitions are non-redundant. It would be useful to find syntactic characterisations of classes of processes with this property. An example is the fragment of language without parallel composition and matching. One could then envisage a two-speed algorithm, the first speed (faster) to be used when active names and non-redundant transitions can be quickly computed in initialisation. (Indeed, the speeds could even be three, the intermediate one to be used when active names are known but non-redundant transitions are not.) Again, the applicability of the first speed can be enhanced by first minimising the representation of process subcomponents whose active names and non-redundant transitions are hard to compute.

Extensions. The algorithm can be extended to the polyadic π-calculus (where tuples of names are communicated). It can also be adapted to early and late bisimulations, and to weak and branching bisimulations.

We briefly discuss the extension to early and late bisimulations. In these bisimulations, intuitively, free names of processes are used as constants rather than variables. Two processes $a(x).P$ and $a(x).Q$ are equivalent if P and Q are so under *all* possible instantiations of the placeholder x. Since names are viewed as constants, syntactically different free names of a process are considered distinct constants, and may never be identified. As a consequence, in (the standard definitions of) early and late bisimulation, transitions have no "condition" part, and distinctions are not used. Therefore the dependencies Dep.2 and Dep.3 discussed in the Introduction do not arise in the definitions of early and late bisimulations. The only dependency still present is (something like) Dep.1, that appears in input and bound output transitions. This dependency remains because, as in open bisimulation, the clause for these transitions impose a requirement on the processes being compared that depends on the union of the free names of the two processes. And, as for open bisimulation, bisimilar processes may have different sets of free names. Active names, rather than free names should be used; but computing the active names is as difficult as the bisimulation checking itself.

The algorithm for checking early or late bisimilarity between two given processes has the following structure[2]:

1 Generate the state graphs for the two processes; in the case of input transitions, the placeholder of the input must be instantiated with all the free names of the source processes plus the first name which does not appear in the source process.

2 Generate an initial partition in which all states are in the same class.

[2] The differences between the standard definitions of early and late bisimulations do not affect the structure of the algorithm: see [9].

3 Repeat the following steps until the partition becomes stable:

   3.1 Compute the active names of the processes w.r.t. the current partition.

   3.2 If necessary, refine the partition so that processes in the same class have the same set of active names.

   3.3 Apply the Paige and Tarjan refinement algorithm using, as instantiations for an input transition, all active names for the source process plus the first name which is not active.

4 Check if the initially given processes are in the same class.

This algorithm is similar to the one for open bisimulation presented in Table 2; however, here we do not have to deal with non-redundant transitions (all transitions are non-redundant) and the states of the graphs correspond to (unconstrained) processes.

The algorithm proposed in [9] for early and late bisimulations for a π-calculus without matching can be seen as a special case of the algorithm above. Indeed in these semantics, if matching is not allowed then active names are all identified during the first iteration of Step 3.1, so that Step 3 has to be executed just once.

The above sketched algorithm for early and late bisimulations is different from that proposed by Dam [4]. Our algorithm takes a process P as input and gives back a "canonical" minimal form for P as a result; this canonical minimal form has, in particular, a minimal number of free names (that is, no process with fewer free names can be bisimilar with P). By contrast, the minimisation that Dam's algorithm gives is *relative to a given set of names*; that is, Dam's algorithm takes as input, besides P, a set S of names that includes all free names of P, and returns a process whose set of free names will, in general, contain S.

In this paper, we have chosen to deal with the open semantics for several reasons: First, as explained above, the partitioning algorithm problem for open semantics is much more complex than for early or late semantics; it is easy to extend the techniques for open semantics to the other semantics, but not the converse. Second, since open bisimulation is stronger than late and early bisimulations, the minimisation of a process w.r.t. open bisimulation is also sound for the other bisimulations, but not the converse. Third, since the use of names in input transitions is much less heavy in open bisimulation,[3] checking open bisimulation will generally be faster than checking early or late bisimulations [2, 13, 15]. Fourth, by contrast with early and late bisimulations, open bisimulation is a congruence relation, which allows us to perform compositional minimisations of processes. Finally, open bisimulation has better algebraic properties; for instance a sound and complete axiomatisation for the finite processes exists; in the case of early and late bisimulations, axiomatisations exist only for the induced congruences, and they are more complex.

Implementations. The algorithm has been implemented by the authors. The implementation is still very rough, however it can handle some interesting examples, like the GSM protocol described in [10]. Experimentations with the implementation have suggested extensions and improvements, like those described above. Performance comparisons with the MWB, as well as the integration of the algorithm with the MWB and other tool sets like JACK [1] or AUTO/GRAPH [3], are planned.

Minimisation. The algorithm performs minimisation of processes on the normalised transition system. Roughly, normalised means that the bound name of transitions is forced to be the first inactive in the source process.

It would be interesting to see whether there are minimal forms and minimisation algorithms with more relaxed conditions on the choice of bound names. The challenge is that the minimal forms must be canonical, i.e., syntactically identical for bisimilar processes.

[3] Notice that, due to rule **sat-bunch** in the generation of the saturated state graph, our algorithm for open bisimulation may require more than one transition for any input of a process. However, in most of the cases this bunch of transitions is much smaller than a bunch obtained by allowing each possible free name of the process to be the value of any input transition, as it would be the case in the late and early semantics; this advantage increases if some estimation of the active names of processes is performed before generating the bunch of input transitions.

Acknowledgements. We are grateful to Ugo Montanari, for several helpful conversations especially in the initial stages of development of this work. We also benefited from comments by Amar Bouali and Robert de Simone, and from the suggestions of the two anonymous referees.

References

[1] A. Bouali, S. Gnesi and S. Larosa. The Integration Project for the JACK Environment. *Bullettin of the EATCS* 54, 1994.

[2] M. Boreale and R. De Nicola. A Symbolic Semantics for the π-calculus. In *Proc. CONCUR'94*, LNCS 836. Springer-Verlag, 1994.

[3] G. Boudol, V. Roy, R. de Simone and D. Vergamini. Process algebra and Systems of Communicating Processes. In *Proc. Automatic Verification Methods for Finite State Systems*, LNCS 407. Springer-Verlag, 1989.

[4] M. Dam. On the decidability of process equivalences for the π-calculus. SICS Research Report RR:94-20, 1994.

[5] J.-C. Fernandez and L. Mounier. "On-the-fly" verification of behavioural equivalences and preorders. In *Proc. CAV'91*, LNCS 575. Springer Verlag, 1994.

[6] B. Jonsson and J. Parrow. Deciding bisimulation equivalences for a class of non-finite-state programs. *Information and Computation*, 107:272–302, 1993.

[7] P. C. Kanellakis and S. A. Smolka. CCS expressions, finite state processes, and three problems of equivalence. *Information and Computation*, 86:43–68, 1990.

[8] R. Milner, J. Parrow and D. Walker. A calculus of mobile processes (parts I and II). *Information and Computation*, 100:1–77, 1992.

[9] U. Montanari and M. Pistore. Checking bisimilarity for finitary π-calculus. In *Proc. CONCUR'95*, LNCS 962. Springer Verlag, 1995.

[10] F. Orava and J. Parrow. An algebraic verification of a mobile network. *Formal Aspects of Computing*, 4:497–543, 1992.

[11] R. Paige and R. E. Tarjan. Three partition refinement algorithms. *SIAM Journal on Computing*, 16(6):973–989, 1987.

[12] D. Sangiorgi. On the bisimulation proof method. Tech. Rep. ECS-LFCS-94-299, LFCS, Department of Computer Science, University of Edinburgh, 1994. A summary in *Proc. MFCS'95*, LNCS 969. Springer-Verlag, 1995.

[13] D. Sangiorgi. A theory of bisimulation for π-calculus. *Acta Informatica*, 33:69–97, 1996. A summary in *Proc. CONCUR'93*, LNCS 715. Springer-Verlag, 1993.

[14] B. Victor and F. Moller. The Mobility Workbench — A tool for the π-calculus. In *Proc. CAV'94*, LNCS 818. Springer-Verlag, 1994.

[15] B. Victor. A Verification Tool for the Polyadic Pi-Calculus. Licentiate thesis, Department of Computer Systems, Uppsala University, Sweden, May 1994. Available as report DoCS 94/50.

A Basic lemmas

In this appendix we present some facts on conditions, distinctions, substitutions, and canonical transitions. The proofs are simple, so we only report a few cases.

Lemma A.1 *Let P be a process and σ be a substitution (resp. an injective substitution). There exists some substitution (resp. injective substitution) ρ such that $P\sigma = P\rho$ and the set*

$$\{x \in \mathcal{N} \ : \ \rho(x) \neq x\}$$

is finite.

The previous lemma is often used, implicitly, in the proofs that follow. In fact it is often required that, given a process P and a substitution σ, there exists some name x which is neutral for σ. There are substitutions for which no such name exists; however, by applying the lemma above, substitution σ can be replaced with an equivalent substitution for which such a name exists.

Lemma A.2 *If $P \xrightarrow{(M,\alpha)} P'$ then $\mathrm{fn}(P') \subseteq \mathrm{fn}(P)\sigma_M \cup \mathrm{bn}(\alpha)$.*

A.1 Lemmas on conditions and substitutions

Lemma A.3 *If $M \rhd N$, then $M\sigma \rhd N\sigma$ for all substitutions σ.*

Lemma A.4 *If $M' \lhd\rhd M\sigma$, then $\sigma_M \sigma \sigma_{M'} = \sigma\sigma_{M'}$.*

Proof: We have to show that $\sigma_{M'}(\sigma(\sigma_M(a))) = \sigma_{M'}(\sigma(a))$ for each name a. By definition of σ_M

$$M \rhd [\sigma_M(a) = a].$$

Hence, by applying substitution σ,

$$M\sigma \rhd [\sigma(\sigma_M(a)) = \sigma(a)].$$

From this and $M' \lhd\rhd M\sigma$ we obtain

$$M' \rhd [\sigma(\sigma_M(a)) = \sigma(a)]$$

which, in turn, by definition of $\sigma_{M'}$, implies

$$\sigma_{M'}(\sigma(\sigma_M(a))) = \sigma_{M'}(\sigma(a)).$$

□

Lemma A.5 *If $M \rhd N$, then $\sigma_M = \sigma_N \sigma_M$.*

Lemma A.6 *Suppose $M' \lhd\rhd M\sigma$, with σ an injective substitution, and let V be any set of names. Then $\sigma\sigma_{M'}$ is injective on $V\sigma_M$.*

A.2 Lemmas on distinctions

Lemmas A.7-A.9 relate distinctions and conditions.

Lemma A.7 *If M respects D and $D' \subseteq D$, then M respects D'.*

Lemma A.8 *If M respects D and $M \rhd N$, then N respects D.*

Lemma A.9 *If M respects D and $D' \cap \mathrm{n}(M) = D \cap \mathrm{n}(M)$, then M respects D'.*

Lemmas A.10-A.12 show distributivity properties of substitutions.

Lemma A.10 $(D \cup E)^M_{(P,Q),\alpha} = D^M_{P,\alpha} \cup E^M_{Q,\alpha}$.

Lemma A.11 *If* $\mathrm{n}(M) \subseteq \mathrm{fn}(P)$, *then* $D\sigma_M \cap \mathrm{fn}(P\sigma_M) = (D \cap \mathrm{fn}(P))\sigma_M$.

Lemma A.12 *If* σ *is injective on* $\mathrm{fn}(P)$, *then* $D\sigma \cap \mathrm{fn}(P\sigma) = (D \cap \mathrm{fn}(P))\sigma$.

Lemma A.13 *Suppose* $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P, D) = \emptyset$, $\mathrm{n}(M) \subseteq \mathrm{fn}(P)$ *and* $\mathrm{fn}(P') \subseteq \mathrm{fn}(P) \cup \mathrm{bn}(\alpha)$, *Then*

$$\big(D \cap \mathrm{fn}(P)\big)^M_{P,\alpha} \cap \mathrm{fn}(P') = D^M_{P,\alpha} \cap \mathrm{fn}(P').$$

The two lemmas below will be useful to push substitutions inside distinctions.

Lemma A.14 *Let* $D^M_{(P_1,\ldots,P_n),\alpha}$ *and* $(D\sigma)^{M'}_{(P_1\sigma,\ldots,P_n\sigma),\alpha'}$ *be two distinctions. If* $M' \lhd\rhd M\sigma$, $\mathrm{bn}(\alpha)$ *is neutral for* σ *and* $\alpha' = \alpha\sigma\sigma_{M'}$ *then*

$$D^M_{(P_1,\ldots,P_n),\alpha}\sigma\sigma_{M'} = (D\sigma)^{M'}_{(P_1\sigma,\ldots,P_n\sigma),\alpha'}.$$

Proof:

$$\begin{array}{rcl}
D^M_{(P_1,\ldots,P_n),\alpha}\sigma\sigma_{M'} & = & [\text{by definition of } D^M_{(P_1,\ldots,P_n),\alpha}] \\
\big(D \cup (\mathrm{en}(\alpha) \times \mathrm{fn}(P_1,\ldots,P_n))\big)\sigma_M\sigma\sigma_{M'} & = & [\text{since } \sigma_M\sigma\sigma_{M'} = \sigma\sigma_{M'} \text{ by Lemma A.4}] \\
\big(D \cup (\mathrm{en}(\alpha) \times \mathrm{fn}(P_1,\ldots,P_n))\big)\sigma\sigma_{M'} & = & [\text{by distributing } \sigma] \\
\big(D\sigma \cup (\mathrm{en}(\alpha)\sigma \times \mathrm{fn}(P_1\sigma,\ldots,P_n\sigma))\big)\sigma_{M'} & = & [\text{since } \mathrm{en}(\alpha)\sigma = \mathrm{en}(\alpha) = \mathrm{en}(\alpha')] \\
\big(D\sigma \cup (\mathrm{en}(\alpha') \times \mathrm{fn}(P_1\sigma,\ldots,P_n\sigma))\big)\sigma_{M'} & = & (D\sigma)^{M'}_{(P_1\sigma,\ldots,P_n\sigma),\alpha'}.
\end{array}$$

□

Lemma A.15 *Suppose* D *is a distinction,* M *respects* D, $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P_1,\ldots,P_n,M) = \emptyset$, *and* $M \rhd N$. *Then* M *respects* $D^N_{(P_1,\ldots,P_n),\alpha}$ *and*

$$D^N_{(P_1,\ldots,P_n),\alpha}\sigma_M = D^M_{(P_1,\ldots,P_n),\alpha}.$$

B Proofs for Section 3

In this appendix we give the proof of Theorem 3.5. As explained in the main text, the essence of the proof is represented by Lemma 3.6. We only report the proof of the third item of the lemma; the proof of the other items is based on similar techniques.

Lemma B.1 *If* $R \rightarrowtail \mathcal{S}$ *then* $\mathcal{C}(\mathcal{R}) \rightarrowtail \mathcal{S}ub(\mathcal{C}(\mathcal{S}))$.

Proof: Suppose $P\ (\mathcal{C}(\mathcal{R}))_D\ Q$. This means that there is some D' such that $P\ \mathcal{R}_{D\cup D'}\ Q$ and $D' \cap \mathrm{fn}(P,Q) = \emptyset$. Suppose that $P \xrightarrow{(M,\alpha)} P'$, that M respects D and that $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P,Q,D) = \emptyset$.
Since $\mathrm{n}(M) \subseteq \mathrm{fn}(P)$, and hence $\mathrm{n}(M) \cap \mathrm{n}(D') = \emptyset$, by Lemma A.9 we infer that M respects also $D \cup D'$. Hence there is some $Q \xrightarrow{(N,\beta)} Q'$ such that $M \rhd N$, $\alpha = \beta\sigma_M$ and

$$P'\ \mathcal{S}_{(D\cup D')^M_{(P,Q),\alpha}}\ Q'\sigma_M. \tag{9}$$

We now distinguish the cases $\mathrm{bn}(\alpha) \cap \mathrm{n}(D') = \emptyset$ and $\mathrm{bn}(\alpha) \cap \mathrm{n}(D') \neq \emptyset$.
Case 1. It holds that $\mathrm{bn}(\alpha) \cap \mathrm{n}(D') = \emptyset$. Then we can conclude

$$P'\ (\mathcal{C}(\mathcal{S}))_{D^M_{(P,Q),\alpha'}}\ Q'\sigma_M$$

from (9) and the definition of $\mathcal{C}$, since

$$(D \cup D')^M_{(P,Q),\alpha'} = D^M_{(P,Q),\alpha'} \cup D'\sigma_M$$

by definitions of $(D \cup D')^M_{(P,Q),\alpha'}$ and of $D^M_{(P,Q),\alpha'} \cup D'\sigma_M$, and

$$D'\sigma_M \cap \mathrm{fn}(P', Q'\sigma_M) = \emptyset$$

since:

$$\begin{array}{rcl} D'\sigma_M \cap \mathrm{fn}(P', Q'\sigma_M) & \subseteq & [\text{since } \mathrm{fn}(P', Q') \subseteq \big(\mathrm{fn}(P,Q) \cup \mathrm{bn}(\alpha)\big)\sigma_M \text{ by Lemma A.2}] \\ D'\sigma_M \cap (\mathrm{fn}(P,Q) \cup \mathrm{bn}(\alpha))\sigma_M & = & [\text{by Lemma A.11}] \\ \big(D' \cap (\mathrm{fn}(P,Q) \cup \mathrm{bn}(\alpha))\big)\sigma_M & = & [\text{since } \mathrm{n}(D') \cap \mathrm{fn}(P,Q) \text{ and } \mathrm{n}(D') \cap \mathrm{bn}(\alpha) = \emptyset \text{ by hypothesis}] \\ \emptyset. & & \end{array}$$

Case 2. It holds that $\mathrm{bn}(\alpha) \cap \mathrm{n}(D') \neq \emptyset$. We can take a variant $P \xrightarrow{(M,\alpha')} P''$ of $P \xrightarrow{(M,\alpha)} P'$, so that $\mathrm{bn}(\alpha') \cap \mathrm{n}(D') = \emptyset$. We can now use the result of Case 1, and infer that there is a transition $Q \xrightarrow{(N,\beta')} Q''$ such that $M \rhd N$, $\alpha' = \beta'\sigma_M$ and

$$P'' \; (\mathcal{C}(\mathcal{S}))_{D^M_{(P,Q),\alpha'}} \; Q''\sigma_M.$$

Choosing the variant $Q \xrightarrow{(N,\beta)} Q'$ of $Q \xrightarrow{(N,\beta')} Q''$ such that $\beta = \alpha\sigma_M$, and the substitution that maps $\mathrm{bn}(\alpha')$ into $\mathrm{bn}(\alpha)$, we obtain

$$P' \; (\mathcal{S}ub(\mathcal{C}(\mathcal{S})))_{D^M_{(P,Q),\alpha}} \; Q'\sigma_M.$$

□

Starting from Lemma 3.6, the proofs of Corollaries 3.7 and 3.8 are straightforward, as it is the proof of Theorem 3.5:

Proof of Theorem 3.5: Let $\mathcal{F}$ be any function on indexed relations defined from functions $\mathcal{S}ub$, $\mathcal{W}$, $\mathcal{C}$, $\mathcal{S}q$ and $\mathcal{S}ym$, and constructors $\cup$, $\circ$ and *. We have to show that $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ implies $\mathcal{R} \subseteq \sim$.
It is easy to observe, by induction on the definition of $\mathcal{F}$, that $\mathcal{F}(\mathcal{R}) \subseteq (\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R})$, so $\mathcal{R} \rightarrowtail \mathcal{F}(\mathcal{R})$ implies $\mathcal{R} \rightarrowtail (\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R})$. This implies $\mathcal{R} \subseteq \sim$ by Corollary 3.8. □

B.1 Proofs for Subsection 3.1

Lemma B.2 shows that if P and Q are open bisimilar, then a transition from P either can be matched by a transition from Q with the same condition, or it is covered by a transition by P with a logically weaker condition. This fact is used in the proof of Theorem 3.11.

Lemma B.2 *Suppose $P \sim_D Q$ and $P \xrightarrow{(M,\alpha)} P'$ with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P,Q,D) = \emptyset$. Then one of these cases is true:*

1. *There is some $P \xrightarrow{(N,\beta)} P''$ such that $M \not\lhd\rhd N$, $\alpha = \beta\sigma_M$ and $P' \sim_{D^M_{P,\alpha}} P''\sigma_M$.*
2. *There is some $Q \xrightarrow{(M,\alpha)} Q'$ such that $P' \sim_{D^M_{(P,Q),\alpha}} Q'$.*

Proof of Theorem 3.11: We have to prove that if $P \sim_D Q$ then $P \sim_E Q$ for $E \stackrel{\mathrm{def}}{=} (D \cap \mathrm{fn}(P)) \cup (D \cap \mathrm{fn}(Q))$. We show something more general: $P \sim_D Q$ implies $P \sim_E Q$ for $E \supseteq (D \cap \mathrm{fn}(P)) \cup (D \cap \mathrm{fn}(Q))$.
Let $\mathcal{R}$ be the indexed relation defined as follows:

$$\mathcal{R}_E \stackrel{\mathrm{def}}{=} \{(P,Q) \; : \; \exists D \text{ such that } P \sim_D Q \text{ and } E \supseteq \big(D \cap \mathrm{fn}(P)\big) \cup \big(D \cap \mathrm{fn}(Q)\big)\}.$$

We show that $\mathcal{R} \rightarrowtail (\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R})$; by Theorem 3.5, this implies $\mathcal{R} \subseteq \sim$.
Suppose $P \, \mathcal{R}_E \, Q$ and $P \sim_D Q$, for some D with $E \supseteq (D \cap \mathrm{fn}(P)) \cup (D \cap \mathrm{fn}(Q))$. Suppose also

$$P \xrightarrow{(M,\alpha)} P' \text{ with } \mathrm{bn}(\alpha) \cap \mathrm{fn}(P,Q,E) = \emptyset, \text{ and } M \text{ respects } E.$$

We show that there is matching transition from Q. It suffices to prove this assuming $\mathrm{bn}(\alpha) \cap \mathrm{n}(D) = \emptyset$; for, otherwise, we can take a variant which satisfies this condition and get a proof exploiting the closure of $(\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R})$ under substitutions.
We proceed by induction on M. Since $E \cap \mathrm{n}(M) = D \cap \mathrm{n}(M)$ and M respects E, by Lemma A.9, also M respects D. Therefore from $P \sim_D Q$ and Lemma B.2, one of these cases is true:

1. $Q \xrightarrow{(M,\alpha)} Q'$ with $P' \sim_{D^M_{(P,Q),\alpha}} Q'$.

2. $P \xrightarrow{(N,\beta)} P''$ with $M \not\lhd\rhd N$, $\alpha = \beta\sigma_M$, and
$$P' \sim_{D^M_{P,\alpha}} P''\sigma_M. \tag{10}$$

We consider each of these cases.
Case 1. We show that
$$P' \,\mathcal{R}_{E^M_{(P,Q),\alpha}}\, Q'.$$
Since $P' \sim_{D^M_{(P,Q),\alpha}} Q'$, by definition of $\mathcal{R}$ it is sufficient to prove
$$E^M_{(P,Q),\alpha} \supseteq \big(D^M_{(P,Q),\alpha} \cap \mathrm{fn}(P')\big) \cup \big(D^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q')\big). \tag{11}$$
We just show that $E^M_{(P,Q),\alpha} \supseteq D^M_{(P,Q),\alpha} \cap \mathrm{fn}(P')$, i.e., that
$$\big(E \cup (\mathrm{en}(\alpha) \cap \mathrm{fn}(P,Q))\big)\sigma_M \supseteq \big(D \cup (\mathrm{en}(\alpha) \cap \mathrm{fn}(P,Q))\big)\sigma_M \cap \mathrm{fn}(P').$$
This in turn is derived from
$$\begin{array}{rcl} E\sigma_M & \supseteq & D\sigma_M \cap \mathrm{fn}(P') \\ (\mathrm{en}(\alpha) \cap \mathrm{fn}(P,Q))\sigma_M & \supseteq & (\mathrm{en}(\alpha) \cap \mathrm{fn}(P,Q))\sigma_M \cap \mathrm{fn}(P'). \end{array}$$
The second inclusion is obvious. Now we consider the first one. Since $E \supseteq D \cap \mathrm{fn}(P)$, using Lemma A.11,
$$E\sigma_M \supseteq (D \cap \mathrm{fn}(P))\sigma_M = D\sigma_M \cap \mathrm{fn}(P)\sigma_M.$$
This is enough, since $\mathrm{fn}(P') \subseteq \mathrm{fn}(P)\sigma_M \cup \mathrm{bn}(\alpha)$ and $\mathrm{bn}(\alpha) \cap \mathrm{n}(D\sigma_M) = \emptyset$.
Case 2. By the inductive assumption, since $P \xrightarrow{(N,\beta)} P''$, there is some $Q \xrightarrow{(L,\gamma)} Q'$ such that $N \rhd L$, $\beta = \gamma\sigma_N$ and
$$P'' \,((\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R}))_{E^N_{(P,Q),\alpha}}\, Q'\sigma_N.$$
By Lemma A.15, M respects $E^N_{(P,Q),\alpha}$ and
$$P''\sigma_M \,((\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R}))_{E^M_{(P,Q),\alpha}}\, Q'\sigma_M. \tag{12}$$
Next, we show that (10) implies
$$P' \,\mathcal{R}_{E^M_{P,\alpha}}\, P''\sigma_M. \tag{13}$$
To obtain this we have to show that
$$E^M_{P,\alpha} \supseteq \big(D^M_{P,\alpha} \cap \mathrm{fn}(P')\big) \cup \big(D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\big)$$
and this is similar to the proof of (11).
By weakening (13) we obtain
$$P' \,(\mathcal{W}(\mathcal{R}))_{E^M_{(P,Q),\alpha}}\, P''\sigma_M.$$
From this and (12) we infer
$$P' \,((\mathcal{S}ub \cup \mathcal{W} \cup \mathcal{C} \cup \mathcal{S}q \cup \mathcal{S}ym)^\star(\mathcal{R}))_{E^M_{(P,Q),\alpha}}\, Q'\sigma_M.$$
This shows that $Q \xrightarrow{(L,\gamma)} Q'$ is a matching transition for $P \xrightarrow{(M,\alpha)} P'$, since it also holds that $\alpha = \gamma\sigma_M$ and $M \rhd L$. □

C Proofs for Section 4

Now we present some basic lemmas on constrained processes. Lemma C.1 gives us another formulation of the compatibility condition on constrained processes.

Lemma C.1 *Let $\langle P, D\rangle$ and $\langle Q, E\rangle$ be constrained processes. Then $\langle P, D\rangle \Downarrow \langle Q, E\rangle$ holds iff there exists some distinction F such that $D = F \cap \mathrm{fn}(P)$ and $E = F \cap \mathrm{fn}(Q)$.*

Lemma C.2 *Let A and B be two constrained processes such that $A \Downarrow B$ and let σ be an injective substitution. Then also $A\sigma \Downarrow B\sigma$.*

Lemma C.3 *Let $\langle P, D\rangle$ and $\langle Q, E\rangle$ be constrained processes and P' be a plain process. Suppose that $\mathrm{n}(M) \subseteq \mathrm{fn}(P)$, $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P, Q) = \emptyset$ and $\mathrm{fn}(P') \subseteq \mathrm{fn}(P)\sigma_M \cup \mathrm{bn}(\alpha)$. Then*

$$(D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(P') = D^M_{P,\alpha} \cap \mathrm{fn}(P').$$

Lemma C.4 shows the effect of injective substitutions on transitions of constrained processes.

Lemma C.4 *Let σ be an injective substitution. We have:*

1. *If $A \xrightarrow{(M,\alpha)} A'$ and $\mathrm{bn}(\alpha)$ is neutral for σ, then there is some $A\sigma \xrightarrow{(M',\alpha')} A''$ with $M' \lhd\rhd M\sigma$, $\alpha' = \alpha\sigma\sigma_{M'}$ and $A'' = A'\sigma\sigma_{M'}$.*
2. *The converse, i.e., if $A\sigma \xrightarrow{(M',\alpha')} A''$ and $\mathrm{bn}(\alpha')$ is neutral for σ, then there is some $A \xrightarrow{(M,\alpha)} A'$ with $M' \lhd\rhd M\sigma$, $\alpha' = \alpha\sigma\sigma_{M'}$ and $A'' = A'\sigma\sigma_{M'}$.*

C.1 Proofs for Subsection 4.1

Lemma C.5 *Assume $\langle P, D\rangle \Downarrow \langle Q, E\rangle$. Whenever $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ and $\langle Q, E\rangle \xrightarrow{(N,\beta)} \langle Q', E'\rangle$, with $M \rhd N$ and $\alpha = \beta\sigma_M$, then*

$$\langle P', D'\rangle \Downarrow \left\langle Q'\sigma_M, (D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M)\right\rangle.$$

Proof: By definition of compatible processes (Definition 4.1), we have to show that

$$\left(D^M_{P,\alpha} \cap \mathrm{fn}(P')\right) \cap \mathrm{fn}(Q'\sigma_M) = \left((D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M)\right) \cap \mathrm{fn}(P'). \tag{14}$$

By Lemma C.3 we obtain

$$(D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(P') = D^M_{P,\alpha} \cap \mathrm{fn}(P')$$

and hence (14) follows by intersecting both sides with $\mathrm{fn}(Q'\sigma_M)$. □

Lemma C.6 *Let $\mathcal{D}$ and $\mathcal{E}$ be relations such that $\mathcal{D} \rightarrowtail \mathcal{E}$. Then $(\mathcal{D} \cap \Downarrow) \rightarrowtail (\mathcal{E} \cap \Downarrow)$.*

Now we show that the two open bisimilarity relations we have defined — on plain and on constrained processes — coincide. This is done in the next two lemmas, which are the building blocks of the proof of Theorem 4.5.

Lemma C.7 *If $P \sim_D Q$ then $\langle P, D \cap \mathrm{fn}(P)\rangle \frown \langle Q, D \cap \mathrm{fn}(Q)\rangle$.*

Proof: Let $\mathcal{D}$ be the $\mathcal{CP}$-relations so defined:

$$\mathcal{D} \stackrel{\mathrm{def}}{=} \{(\langle P, D \cap \mathrm{fn}(P)\rangle, \langle Q, D \cap \mathrm{fn}(Q)\rangle) \ : \ P \sim_D Q\}.$$

We show that $\mathcal{D} \subseteq \Downarrow$ and $\mathcal{D} \rightarrowtail \mathcal{D}$. Then the thesis follows by Lemma 4.3.

Suppose $A \mathcal{D} B$. Then $A = \langle P, D \cap \mathrm{fn}(P)\rangle$ and $B = \langle Q, D \cap \mathrm{fn}(Q)\rangle$ for some P, Q and D such that $P \sim_D Q$. By strong strengthening (Theorem 3.11), $P \sim_E P$ with $E \stackrel{\mathrm{def}}{=} (D \cap \mathrm{fn}(P)) \cup (D \cap \mathrm{fn}(Q))$. Notice also that $D \cap \mathrm{fn}(P) = E \cap \mathrm{fn}(P)$ and, similarly, $D \cap \mathrm{fn}(Q) = E \cap \mathrm{fn}(Q)$. Hence $A \Downarrow B$ holds by Lemma C.1.
To conclude, we have to show that the transitions from A are matched by the transitions from B (the converse is similar).
Suppose $A \stackrel{(M,\alpha)}{\longrightarrow} A'$ with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(B) = \emptyset$. Then, by definition of transition on constrained processes, it holds that $P \stackrel{(M,\alpha)}{\longrightarrow} P'$, that M respects $D \cap \mathrm{fn}(P)$ and that $A' = \left\langle P', \left(D \cap \mathrm{fn}(P)\right)^M_{P,\alpha} \cap \mathrm{fn}(P')\right\rangle$.
We want to show that M respects E; this is proved by this sequence of implications:

$$\begin{array}{rcl} M \text{ respects } D \cap \mathrm{fn}(P) & \Rightarrow & [\text{since } D \cap \mathrm{fn}(P) = E \cap \mathrm{fn}(P)] \\ M \text{ respects } E \cap \mathrm{fn}(P) & \Rightarrow & [\text{by Lemma A.9}] \\ M \text{ respects } E. & & \end{array}$$

Since $\sim$ is an open bisimulation, $P \sim_E Q$ and M respects E, the transition $P \stackrel{(M,\alpha)}{\longrightarrow} P'$ must be matched by a transition from Q, say

$$Q \stackrel{(N,\beta)}{\longrightarrow} Q', \tag{15}$$

such that $M \rhd N$, $\alpha = \beta\sigma_M$ and $P' \sim_{E^M_{(P,Q),\alpha}} Q'\sigma_M$.
Given (15), by the definition of transition on constrained processes we can infer

$$B \stackrel{(N,\beta)}{\longrightarrow} \left\langle Q', \left(D \cap \mathrm{fn}(Q)\right)^N_{Q,\beta} \cap \mathrm{fn}(Q')\right\rangle$$

proviso N respects $D \cap \mathrm{fn}(Q)$; this fact is true, since

$$\begin{array}{rcl} M \text{ respects } E & \Rightarrow & [\text{by Lemma A.7}] \\ M \text{ respects } E \cap \mathrm{fn}(Q) & \Rightarrow & [\text{since } E \cap \mathrm{fn}(Q) = D \cap \mathrm{fn}(Q)] \\ M \text{ respects } D \cap \mathrm{fn}(Q) & \Rightarrow & [\text{by Lemma A.8}] \\ N \text{ respects } D \cap \mathrm{fn}(Q). & & \end{array}$$

It remains to show that

$$\left\langle P', \left(D \cap \mathrm{fn}(P)\right)^M_{P,\alpha} \cap \mathrm{fn}(P')\right\rangle \mathcal{D} \left\langle Q'\sigma_M, \left((D \cap \mathrm{fn}(P)) \cup (D \cap \mathrm{fn}(Q))\right)^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M)\right\rangle. \tag{16}$$

Since $D \cap \mathrm{fn}(P) = E \cap \mathrm{fn}(P)$ and $E = (D \cap \mathrm{fn}(P)) \cup (D \cap \mathrm{fn}(Q))$, (16) can be rewritten as

$$\left\langle P', \left(E \cap \mathrm{fn}(P)\right)^M_{P,\alpha} \cap \mathrm{fn}(P')\right\rangle \mathcal{D} \left\langle Q'\sigma_M, E^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M)\right\rangle. \tag{17}$$

By Lemma A.13, $\left(E \cap \mathrm{fn}(P)\right)^M_{P,\alpha} \cap \mathrm{fn}(P') = E^M_{P,\alpha} \cap \mathrm{fn}(P')$; so (17) can be rewritten as

$$\left\langle P', E^M_{(P,Q),\alpha} \cap \mathrm{fn}(P')\right\rangle \mathcal{D} \left\langle Q'\sigma_M, E^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M)\right\rangle,$$

which follows from $P' \sim_{E^M_{(P,Q),\alpha}} Q'\sigma_M$ by definition of $\mathcal{D}$. □

Lemma C.8 *If $\langle P, D\rangle \frown \langle Q, E\rangle$ then $P \sim_{D\cup E} Q$.*

Proof: Let $\mathcal{R}$ be the indexed relation on plain processes defined as follows:

$$\mathcal{R}_F \stackrel{\mathrm{def}}{=} \{(P, Q) \ : \ \langle P, D\rangle \frown \langle Q, E\rangle \text{ and } F = D \cup E\}.$$

We show that $\mathcal{R}$ is an open bisimulation up to weakening, i.e., $\mathcal{R} \rightarrowtail \mathcal{W}(\mathcal{R})$. By Theorem 3.5, this implies $\mathcal{R} \subseteq \sim$.

Suppose $P\ \mathcal{R}_F\ Q$. Then, by definition of $\mathcal{R}$, there are $\langle P, D\rangle$ and $\langle Q, E\rangle$ with $\langle P, D\rangle \frown \langle Q, E\rangle$ and $F = D \cup E$. By definition of $\frown$, it holds that $\langle P, D\rangle \Downarrow \langle Q, E\rangle$.

For each transition of P, we have to exhibit a matching transition of Q. Suppose $P \xrightarrow{(M,\alpha)} P'$, with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$ and M respects F. By Lemma A.7, M respects also D, so, by definition of transitions on constrained processes, $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D^M_{P,\alpha} \cap \mathrm{fn}(P')\rangle$.

Since $\langle P, D\rangle \frown \langle Q, E\rangle$, $\mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$, and $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D^M_{P,\alpha} \cap \mathrm{fn}(P')\rangle$, there is a transition $\langle Q, E\rangle \xrightarrow{(N,\beta)} \left\langle Q', E^M_{Q,\beta} \cap \mathrm{fn}(Q')\right\rangle$ such that $M \rhd N$, $\alpha = \beta\sigma_M$ and

$$\langle P', D^M_{P,\alpha} \cap \mathrm{fn}(P')\rangle \frown \left\langle Q'\sigma_M, (D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M)\right\rangle. \tag{18}$$

Then also $Q \xrightarrow{(N,\beta)} Q'$ must holds. It remains to show

$$P'\ \mathcal{W}(\mathcal{R})_{F^M_{(P,Q),\alpha}}\ Q'\sigma_M. \tag{19}$$

By definition of $\mathcal{R}$, from (18) we obtain

$$P'\ \mathcal{R}_{\left(D^M_{P,\alpha} \cap \mathrm{fn}(P')\right) \cup \left((D \cup E)^M_{P,Q,\alpha} \cap \mathrm{fn}(Q'\sigma_M)\right)}\ Q'\sigma_M.$$

Then (19) is obtained by weakening, since

$$\begin{array}{rcl} F^M_{(P,Q),\alpha} & = & [\text{by definition of } F] \\ (D \cup E)^M_{(P,Q),\alpha} & = & [\text{since } D^M_{P,\alpha} \subseteq (D \cup E)^M_{(P,Q),\alpha} \text{ by Lemma A.10}] \\ D^M_{P,\alpha} \cup (D \cup E)^M_{(P,Q),\alpha} & \supseteq & [\text{by set manipulations}] \\ \left(D^M_{P,\alpha} \cap \mathrm{fn}(P')\right) \cup \left((D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M)\right). & & \end{array}$$

□

C.2 Proofs for Subsection 4.2

Now we report the proof of Theorem 4.10. First, we need some auxiliary results.

Lemma C.9 *Suppose that:*

1. $\langle P, D\rangle \Downarrow \langle Q, E\rangle$,
2. M *respects* $D \cup E$, $\mathrm{n}(M) \subseteq \mathrm{fn}(P)$, $M \rhd N$ *and* $\mathrm{n}(N) \subseteq \mathrm{fn}(Q)$,
3. $\alpha = \beta\sigma_M$ *and* $\mathrm{bn}(\alpha) \cap \mathrm{fn}(P, Q) = \emptyset$,
4. $\mathrm{fn}(P') \subseteq \mathrm{fn}(P) \cup \mathrm{bn}(\alpha)$ *and* $\mathrm{fn}(Q') \subseteq \mathrm{fn}(Q) \cup \mathrm{bn}(\alpha)$,
5. $\left\langle P', D^N_{P,\beta} \cap \mathrm{fn}(P')\right\rangle\ \mathcal{D}\ \left\langle Q', E^N_{Q,\beta} \cap \mathrm{fn}(Q')\right\rangle$,

Then $\left\langle P'\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P'\sigma_M)\right\rangle\ \mathcal{S}ub_W(\mathcal{D})\ \left\langle Q'\sigma_M, (D \cup E)^M_{P,Q,\alpha} \cap \mathrm{fn}(Q'\sigma_M)\right\rangle$.

Proof: Conditions $M \rhd N$ and $\mathrm{n}(M) \subseteq \mathrm{fn}(P)$ imply $\mathrm{n}(N) \subseteq \mathrm{fn}(P)$. From Lemma C.3, we infer

$$(D \cup E)^N_{(P,Q),\beta} \cap \mathrm{fn}(P') = D^N_{P,\beta} \cap \mathrm{fn}(P')$$

and

$$(D \cup E)^N_{(P,Q),\beta} \cap \mathrm{fn}(Q') = E^N_{Q,\beta} \cap \mathrm{fn}(Q').$$

Hence, by hypothesis 5,

$$\Big\langle P', (D \cup E)^N_{(P,Q),\beta} \cap \mathrm{fn}(P') \Big\rangle \; \mathcal{D} \; \Big\langle Q', (D \cup E)^N_{(P,Q),\beta} \cap \mathrm{fn}(Q') \Big\rangle. \tag{20}$$

Since M respects $D \cup E$, we can apply Lemma A.15 and deduce that M respects also $(D \cup E)^N_{(P,Q),\beta}$ and that $(D \cup E)^N_{(P,Q),\beta}\sigma_M = (D \cup E)^M_{(P,Q),\alpha}$. By definition of $\mathcal{S}ub_W$, using σ_M as substitution, from (20) we obtain

$$\Big\langle P'\sigma_M, (D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(P'\sigma_M) \Big\rangle \; \mathcal{S}ub_W(\mathcal{D}) \; \Big\langle Q'\sigma_M, (D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M) \Big\rangle.$$

Finally, by Lemma C.3 $(D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(P'\sigma_M) = D^M_{P,\alpha} \cap \mathrm{fn}(P'\sigma_M)$, therefore

$$\big\langle P'\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P'\sigma_M) \big\rangle \; \mathcal{S}ub_W(\mathcal{D}) \; \big\langle Q'\sigma_M, (D^N_{P,\alpha} \cup E^M_{Q,\alpha}) \cap \mathrm{fn}(Q'\sigma_M) \big\rangle.$$

□

Corollary C.10 *Suppose that the hypothesis of Lemma C.9 hold and that, in addition,* $\mathrm{n}(M) \subseteq \mathrm{fn}(Q)$. *Then:*

$$\big\langle P'\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P'\sigma_M) \big\rangle \; \mathcal{S}ub_W(\mathcal{D}) \; \big\langle Q'\sigma_M, E^M_{Q,\alpha} \cap \mathrm{fn}(Q'\sigma_M) \big\rangle.$$

Lemma C.11 *Let $\mathcal{D}$ and $\mathcal{E}$ be two compatible $\mathcal{CP}$-relations such that $\mathcal{D} \rightarrowtail \mathcal{E}$. Suppose $\langle P, D\rangle \; \mathcal{D} \; \langle Q, E\rangle$ and $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$. Then there are transitions $\langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', D''\rangle$ and $\langle Q, E\rangle \xrightarrow{(N,\beta)} \langle Q'', E''\rangle$ such that $M \rhd N$, $\alpha = \beta\sigma_M$,*

$$\langle P'', D''\rangle \; \mathcal{E} \; \langle Q'', E''\rangle$$

and

$$\langle P', D'\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M) \big\rangle.$$

Proof: By induction on M. □

Lemma C.12 *Suppose that $\langle P, D\rangle \Downarrow \langle Q, E\rangle$ and that*

$$\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', Q'\rangle \qquad \langle Q, E\rangle \xrightarrow{(N,\beta)} \langle Q'', E''\rangle \qquad \langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', D''\rangle,$$

with $M \rhd N$, $\alpha = \beta\sigma_M$. If $\langle P'', D''\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \langle Q'', E''\rangle$ and

$$\langle P', D'\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M) \big\rangle \tag{21}$$

for some $\mathcal{CP}$-relation $\mathcal{D}$, then

$$\langle P', D'\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \big\langle Q''\sigma_M, (D^M_{P,\alpha} \cup E^M_{Q,\alpha}) \cap \mathrm{fn}(Q''\sigma_M) \big\rangle. \tag{22}$$

Proof: Applying Lemma C.9 onto $\langle P'', D''\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \langle Q'', E''\rangle$, we get

$$\big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M) \big\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \big\langle Q''\sigma_M, (D^M_{P,\alpha} \cap E^M_{Q,\alpha}) \cap \mathrm{fn}(Q''\sigma_M) \big\rangle.$$

From this and (21), we obtain (22) by weak transitivity (the compatibility between the two constrained processes is ensured by Lemma C.5). □

Definition C.13 *A transition $A \xrightarrow{(M,x(y))} A'$ (resp. $A \xrightarrow{(M,\overline{x}(y))} A'$) is a variant of the transition $A \xrightarrow{(M,x(w))} A''$ (resp. $A \xrightarrow{(M,\overline{x}(w))} A''$) if $A'' = A'\{{}^{w}/_{y}\,{}^{y}/_{w}\}$.*

In the previous definition, since $y, w \notin \mathrm{fn}(A)$, instead of $A'' = A'\{{}^{w}/_{y}\,{}^{y}/_{w}\}$ we could have equivalently required that $A'' = A'\{{}^{w}/_{y}\}$, or that $A' = A''\{{}^{y}/_{w}\}$.

Lemma C.14 *Suppose that $\mathcal{D}$ is a $\mathcal{CP}$-relation closed under injective substitutions. Suppose also that $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$, $\langle Q, E\rangle \xrightarrow{(N,\beta)} \langle Q', E'\rangle$ with $M \rhd N$, $\alpha = \beta\sigma_M$ and*

$$\langle P', D'\rangle \; \mathcal{D} \; \Big\langle Q'\sigma_M, (D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M)\Big\rangle.$$

If $\langle P, D\rangle \xrightarrow{(M,\alpha')} \langle P'', D''\rangle$ and $\langle Q, E\rangle \xrightarrow{(N,\beta')} \langle Q'', E''\rangle$ are variants of $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ and $\langle Q, E\rangle \xrightarrow{(N,\beta)} \langle Q', E'\rangle$, respectively, and $\alpha' = \beta'\sigma_M$, then also

$$\langle P'', D''\rangle \; \mathcal{D} \; \Big\langle Q''\sigma_M, (D \cup E)^M_{(P,Q),\alpha'} \cap \mathrm{fn}(Q''\sigma_M)\Big\rangle.$$

Now we are ready to prove Lemma 4.11, which will be the basis for the proof of Theorem 4.10. We just report the proof of the second item of the lemma, the one for function $\mathcal{S}q_W$.

Lemma C.15 *Suppose that $\mathcal{D}$ and $\mathcal{E}$ are two compatible $\mathcal{CP}$-relations. If $\mathcal{D} \rightarrowtail \mathcal{E}$ then it holds that $\mathcal{S}q_W(\mathcal{D}) \rightarrowtail (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})$.*

Proof: Suppose $\langle P, D\rangle \; \mathcal{S}q_W(\mathcal{D}) \; \langle Q, E\rangle$ and $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$. We show, by induction on M, that there are transitions $\langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', Q''\rangle$ and $\langle Q, E\rangle \xrightarrow{(N,\beta)} \langle Q'', E''\rangle$, with

$$M \rhd N \qquad \alpha = \beta\sigma_M \qquad \langle P'', D''\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \langle Q'', E''\rangle$$

and

$$\langle P', D'\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\big\rangle.$$

Then the thesis of the lemma follows by Lemma C.12. If $\langle P, D\rangle \; \mathcal{S}q_W(\mathcal{D}) \; \langle Q, E\rangle$ then $\langle P, D\rangle \Downarrow \langle Q, E\rangle$ and there is $\langle R, F\rangle$ such that $\langle P, D\rangle \; \mathcal{D} \; \langle R, F\rangle \; \mathcal{D} \; \langle Q, E\rangle$. Since $(\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star$ is closed under injective substitutions, by Lemma C.14 it suffices to find a matching transition from $\langle Q, E\rangle$ assuming that $\mathrm{bn}(\alpha)$ is fresh not only w.r.t. Q but also w.r.t. R.
Now we build a chain of transitions by repeatedly applying Lemma C.11 (whose uses will not be mentioned). First, $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ implies $\langle P, D\rangle \xrightarrow{(L,\gamma)} \langle P''', D'''\rangle$ and $\langle R, F\rangle \xrightarrow{(L,\gamma)} \langle R', F'\rangle$ with

$$\langle P''', D'''\rangle \; \mathcal{E} \; \langle R', F'\rangle \qquad M \rhd L \qquad \alpha = \gamma\sigma_M$$

and

$$\langle P', D'\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \big\langle P'''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P'''\sigma_M)\big\rangle.$$

This, in turn, implies $\langle R, F\rangle \xrightarrow{(N_0,\beta_0)} \langle R'_0, F'_0\rangle$ and $\langle Q, E\rangle \xrightarrow{(N_0,\beta_0)} \langle Q'_0, D'_0\rangle$ with

$$\langle R'_0, F'_0\rangle \; \mathcal{E} \; \langle Q'_0, E'_0\rangle \qquad L \rhd N_0 \qquad \gamma = \beta_0\sigma_L$$

and

$$\langle R', F'\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \big\langle R'_0\sigma_L, F^L_{R,\gamma} \cap \mathrm{fn}(R'_0\sigma_L)\big\rangle. \tag{23}$$

We distinguish two cases: $M \lhd\rhd N_0$ and $M \not\lhd\rhd N_0$.
Case 1. It holds that $M \lhd\rhd N_0$. Then $M \lhd\rhd L \lhd\rhd N_0$ and hence $M = L = N_0$. Then we can take $\langle P'', D''\rangle \stackrel{\mathrm{def}}{=} \langle P''', D'''\rangle$, $\langle Q'', E''\rangle \stackrel{\mathrm{def}}{=} \langle Q'_0, E'_0\rangle$, $N = L$ and $\beta = \alpha$. To conclude, it remains to show that

$$\langle P''', D'''\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \langle Q'_0, E'_0\rangle,$$

which is obtained by weak transitivity from $\langle P''', D'''\rangle \; \mathcal{E} \; \langle R', F'\rangle$, (23) and $\langle R'_0, F'_0\rangle \; \mathcal{E} \; \langle Q'_0, E'_0\rangle$ (all the involved states are compatible by Lemma C.5).
Case 2. It holds that $M \not\lhd\rhd N_0$. Then there are $\langle Q, E\rangle \xrightarrow{(L_1,\gamma_1)} \langle Q'_1, E'_1\rangle$ and $\langle R, F\rangle \xrightarrow{(L_1,\gamma_1)} \langle R'_1, F'_1\rangle$ with

$$\langle Q'_1, E'_1\rangle \; \mathcal{S}ym(\mathcal{E}) \; \langle R'_1, F'_1\rangle \qquad N_0 \rhd L_1 \qquad \beta_0 = \gamma_1\sigma_{N_0}$$

and

$$\langle Q'_0, D'_0\rangle\ (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})\ \Big\langle Q'_1\sigma_{N_0}, E^{N_0}_{Q,\beta_0} \cap \mathrm{fn}(Q'_1\sigma_{N_0})\Big\rangle.$$

Then, also $\langle R, F\rangle \xrightarrow{(M_1,\alpha_1)} \langle R'_2, F'_2\rangle$ and $\langle P, D\rangle \xrightarrow{(M_1,\alpha_1)} \langle P'_1, D'_1\rangle$ with

$$\langle R'_2, F'_2\rangle\ \mathcal{S}ym(\mathcal{E})\ \langle P'_1, D'_1\rangle \qquad L_1 \rhd M_1 \qquad \gamma_1 = \alpha_1\sigma_{L_1}$$

and

$$\langle R'_1, F'_1\rangle\ (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})\ \Big\langle R'_2\sigma_{L_1}, F^{L_1}_{R,\gamma_1} \cap \mathrm{fn}(R'_2\sigma_{L_1})\Big\rangle.$$

Since $M \not\lhd\rhd M_1$, we can apply the inductive hypothesis on $\langle P, D\rangle \xrightarrow{(M_1,\alpha_1)} \langle P'_1, D'_1\rangle$ and infer that there are transitions $\langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', D''\rangle$ and $\langle Q, E\rangle \xrightarrow{(N,\beta)} \langle Q'', E''\rangle$ such that

$$M_1 \rhd N \qquad \alpha_1 = \beta\sigma_{M_1} \qquad \langle P'', D''\rangle\ (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})\ \langle Q'', E''\rangle$$

and

$$\langle P'_1, D'_1\rangle\ (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})\ \Big\langle P''\sigma_{M_1}, D^{M_1}_{P,\alpha_1} \cap \mathrm{fn}(P''\sigma_{M_1})\Big\rangle.$$

It holds that $M \rhd N$ and $\alpha = \beta\sigma_M$. To conclude the proof, we have to show that

$$\langle P', D'\rangle\ (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})\ \big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\big\rangle.$$

This is easily obtained by combining the pairs of constrained states reported above, which are in relation $\mathcal{E}$, $\mathcal{S}ym(\mathcal{E})$, or $(\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})$. We obtain this by exploiting the closure of $(\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star$ by weak transitivity and by substitutions. This concludes the proof. □

Lemma C.16 *If $\mathcal{D} \subseteq \Downarrow$ then $(\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \subseteq \Downarrow$.*

Once Lemma 4.11 is proved, the proofs of Corollaries 4.12 and 4.13 are easily derived, as it is the proof of Theorem 4.10:

Proof of Theorem 4.10: Let $\mathcal{F}$ be any function on $\mathcal{CP}$-relations defined from functions $\mathcal{S}ub_W$, $\mathcal{S}q$ and $\mathcal{S}ym$ and constructors $\cup$, $\circ$ and *. We have to show that $\mathcal{D} \rightarrowtail \mathcal{F}(\mathcal{D})$ implies $\mathcal{D} \subseteq \sim$ for each compatible $\mathcal{CP}$-relation $\mathcal{D}$.
It is easy to observe, by induction on the definition of $\mathcal{F}$, that $\mathcal{F}(\mathcal{D}) \subseteq (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D})$, so $\mathcal{D} \rightarrowtail \mathcal{F}(\mathcal{D})$ implies $\mathcal{D} \rightarrowtail (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D})$. By Corollary 4.13 this implies that $\mathcal{D} \subseteq \frown$ whenever $\mathcal{D} \subseteq \Downarrow$. □

D Proofs for Section 5

Lemma D.1 *If $\mathcal{D} \subseteq \mathcal{E}$ then $\mathrm{nr}(\mathcal{D}) \supseteq \mathrm{nr}(\mathcal{E})$.*

Lemma D.2 *Let $\mathcal{D}$ be a $\mathcal{CP}$-relation and $\mathcal{E} \stackrel{\mathrm{def}}{=} \mathcal{D} \cap \Downarrow$. Then $\mathrm{nr}(\mathcal{D}) = \mathrm{nr}(\mathcal{E})$.*

Proof of Lemma 5.2: We have to prove that, for each $\mathcal{CP}$-relation $\mathcal{D}$ and each transition $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ there is a transition $\langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', D''\rangle \in \mathrm{nr}(\mathcal{D})$ such that $M \rhd N$, $\alpha = \beta\sigma_M$ and

$$\langle P', D'\rangle\ (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D})\ \big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\big\rangle.$$

We prove this by induction on M. If $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle \in \mathrm{nr}(\mathcal{D})$ then the lemma holds with $N \stackrel{\mathrm{def}}{=} M$, $\beta \stackrel{\mathrm{def}}{=} \alpha$ and $\langle P'', D''\rangle \stackrel{\mathrm{def}}{=} \langle P', D'\rangle$, since $(\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D})$ is reflexive. If $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle \notin \mathrm{nr}(\mathcal{D})$ then, by definition, there is some $\langle P, D\rangle \xrightarrow{(L,\gamma)} \langle P''', D'''\rangle$ such that

$$M \not\lhd\rhd L \qquad \alpha = \gamma\sigma_M \qquad \langle P', D'\rangle\ \mathcal{D}\ \big\langle P'''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P'''\sigma_M)\big\rangle. \tag{24}$$

By the inductive hypothesis applied to $\langle P, D\rangle \xrightarrow{(L,\gamma)} \langle P''', D'''\rangle$, there is some $\langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', D''\rangle \in \mathrm{nr}(\mathcal{D})$ with

$$L \rhd N \qquad \gamma = \beta\sigma_L \qquad \langle P''', D'''\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \big\langle P''\sigma_L, D^L_{P,\gamma} \cap \mathrm{fn}(P''\sigma_L)\big\rangle. \tag{25}$$

We are in the conditions to apply Corollary C.10, (taking $\langle Q, E\rangle \stackrel{\mathrm{def}}{=} \langle P, D\rangle$), thus obtaining

$$\big\langle P'''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P'''\sigma_M)\big\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\big\rangle.$$

From this and (24) we infer by weak transitivity

$$\langle P', D'\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\big\rangle,$$

where the processes are compatible by Lemma C.5. This concludes the proof since, from (24) and (25) we get $M \not\lhd\rhd N$ and $\alpha = \beta\sigma_M$. □

Lemmas D.3 and D.4 show that non-redundant transitions are closed w.r.t. injective substitutions and alpha conversion.

Lemma D.3 *Let $\mathcal{D}$ be a $\mathcal{CP}$-relation closed for injective substitutions and let σ be an injective substitution. We have:*

1. *If $A \xrightarrow{(M,\alpha)} A' \in \mathrm{nr}(\mathcal{D})$ and $\mathrm{bn}(\alpha)$ is neutral for σ, then there is some $A\sigma \xrightarrow{(M',\alpha')} A'' \in \mathrm{nr}(\mathcal{D})$ with $M' \lhd\rhd M\sigma$, $\alpha' = \alpha\sigma\sigma_{M'}$ and $A'' = A'\sigma\sigma_{M'}$;*
2. *The converse, i.e., if $A\sigma \xrightarrow{(M',\alpha')} A'' \in \mathrm{nr}(\mathcal{D})$ and $\mathrm{bn}(\alpha')$ is neutral for σ, then there is some $A \xrightarrow{(M,\alpha)} A' \in \mathrm{nr}(\mathcal{D})$ with $M' \lhd\rhd M\sigma$, $\alpha' = \alpha\sigma\sigma_{M'}$ and $A'' = A'\sigma\sigma_{M'}$.*

Lemma D.4 *Suppose that $\mathcal{D}$ is closed for injective substitutions and that $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ is a variant of $\langle P, D\rangle \xrightarrow{(M,\alpha')} \langle P'', D''\rangle$. Then $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle \in \mathrm{nr}(\mathcal{D})$ iff $\langle P, D\rangle \xrightarrow{(M,\alpha')} \langle P'', D''\rangle \in \mathrm{nr}(\mathcal{D})$.*

Proof of Lemma 5.3: Suppose $\langle P, D\rangle \frown \langle Q, E\rangle$ and $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle \in \mathrm{nr}(\frown)$ with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$. We have to show that there is some $\langle Q, E\rangle \xrightarrow{(M,\alpha)} \langle Q', E'\rangle \in \mathrm{nr}(\frown)$ such that $\langle P', D'\rangle \frown \langle Q', E'\rangle$. One proves this by induction on M. □

D.1 Proofs for Subsection 5.1

Lemma D.5 *If $\mathcal{D}$ is an $\frown_{\mathrm{nr}}$-bisimulation then also $\mathcal{D} \cap \Downarrow$ is.*

Proof of Theorem 5.8: We have to show that $\frown = \frown_{\mathrm{nr}}$. For the inclusion $\frown \subseteq \frown_{\mathrm{nr}}$ it is sufficient to observe that, by Lemma 5.3, $\frown$ is an $\frown_{\mathrm{nr}}$-bisimulation. Since $\frown \subseteq \Downarrow$, we obtain $\frown \subseteq \frown_{\mathrm{nr}}$ by definition of $\frown_{\mathrm{nr}}$. For the inclusion $\frown_{\mathrm{nr}} \subseteq \frown$, we show that if $\mathcal{D}$ is a $\frown_{\mathrm{nr}}$-bisimulation and if $\mathcal{D} \subseteq \Downarrow$ then $\mathcal{D} \rightarrowtail (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D})$. Then, by Theorem 4.10, we have $\mathcal{D} \subseteq \frown$. Since this is true for each compatible $\frown_{\mathrm{nr}}$-bisimulation, by Lemma 5.6 we obtain $\frown_{\mathrm{nr}} \subseteq \frown$.
Suppose $\langle P, D\rangle \; \mathcal{D} \; \langle Q, E\rangle$ and $\langle P, D\rangle \xrightarrow{(M,\alpha)} \langle P', D'\rangle$ with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(Q) = \emptyset$. We will find a matching transition from $\langle Q, E\rangle$. By Lemma 5.2, there is a transition $\langle P, D\rangle \xrightarrow{(N,\beta)} \langle P'', D''\rangle \in \mathrm{nr}(\mathcal{D})$ such that $M \rhd N$, $\alpha = \beta\sigma_M$ and

$$\langle P', D'\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\big\rangle. \tag{26}$$

By definition of $\frown_{\mathrm{nr}}$-bisimulation, some $\langle Q, E\rangle \xrightarrow{(N,\beta)} \langle Q', E'\rangle \in \mathrm{nr}(\mathcal{D})$ exists such that $\langle P'', D''\rangle \; \mathcal{D} \; \langle Q', E'\rangle$. Hence also $\langle P'', D''\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \langle Q', E'\rangle$. We can apply Lemma C.9, obtaining

$$\big\langle P''\sigma_M, D^M_{P,\alpha} \cap \mathrm{fn}(P''\sigma_M)\big\rangle \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \; \Big\langle Q'\sigma_M, (D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M)\Big\rangle.$$

Finally, from this and (26), by weak transitivity,

$$\left\langle P', D^M_{P,\alpha} \cap \mathrm{fn}(P') \right\rangle \, (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D}) \, \left\langle Q'\sigma_M, (D \cup E)^M_{(P,Q),\alpha} \cap \mathrm{fn}(Q'\sigma_M) \right\rangle$$

(these two states are compatible by Lemma C.5). We can therefore conclude that transition $\langle Q, E\rangle \stackrel{(N,\beta)}{\longrightarrow} \langle Q', E'\rangle$ matches transition $\langle P, D\rangle \stackrel{(M,\alpha)}{\longrightarrow} \langle P', D'\rangle$. □

The next lemma will be used in later proofs.

Lemma D.6 *Suppose that $\mathcal{D}$ is weak transitive and that $\mathcal{D} \supseteq \frown$. If $A \frown B$ and $A \stackrel{(M,\alpha)}{\longrightarrow} A' \in \mathrm{nr}(\mathcal{D})$ with $\mathrm{bn}(\alpha) \cap \mathrm{fn}(B) = \emptyset$ then there is B' such that $B \stackrel{(M,\alpha)}{\longrightarrow} B' \in \mathrm{nr}(\mathcal{D})$ and $A' \frown B'$.*

E Proofs for Section 6

We present some basic results on active names. Lemma E.1 assures us that the definition of active names makes sense; the other lemmas and corollaries of this section describe properties of active names that are exploited in later proofs.

Lemma E.1 *Functional ψ, introduced in Definition 6.1 to define $\mathrm{an}_\mathcal{D}$, is monotone on the CPO (complete partial order) of the functions on $\mathcal{CP}$-relations.*

Lemma E.2 *If $\mathcal{D} \supseteq \mathcal{E}$ then $\mathrm{an}_\mathcal{D} \subseteq \mathrm{an}_\mathcal{E}$.*

Proof: Let ψ be the functional introduced in Definition 6.1 to define $\mathrm{an}_\mathcal{D}$. We show that $\psi(\mathrm{an}_\mathcal{E}) \subseteq \mathrm{an}_\mathcal{E}$. This means that $\mathrm{an}_\mathcal{E}$ is a prefixed-point of functional ψ and hence that $\mathrm{an}_\mathcal{D} \subseteq \mathrm{an}_\mathcal{E}$, since $\mathrm{an}_\mathcal{D}$ is its least (pre)fixed-point. We have:

$$\begin{array}{rcll}
 & \psi(\mathrm{an}_\mathcal{E})(A) & = & [\text{by definition of } \psi] \\
\bigcup_{\{M,\alpha,A' \,:\, A \stackrel{(M,\alpha)}{\longrightarrow} A' \in \mathrm{nr}(\mathcal{D})\}} & \mathrm{fn}(M,\alpha) \cup (\mathrm{an}_\mathcal{E}(A') - \mathrm{bn}(\alpha)) & \subseteq & [\text{since } \mathrm{nr}(\mathcal{D}) \subseteq \mathrm{nr}(\mathcal{E}) \text{ by Lemma D.1}] \\
\bigcup_{\{M,\alpha,A' \,:\, A \stackrel{(M,\alpha)}{\longrightarrow} A' \in \mathrm{nr}(\mathcal{E})\}} & \mathrm{fn}(M,\alpha) \cup (\mathrm{an}_\mathcal{E}(A') - \mathrm{bn}(\alpha)) & = & [\text{by definition of } \mathrm{an}_\mathcal{E}] \\
 & \mathrm{an}_\mathcal{E}(A). & &
\end{array}$$

□

Lemma E.3 *Let $\mathcal{D}$ be any $\mathcal{CP}$-relation. Then $\mathrm{an}_\mathcal{D}(A) \subseteq \mathrm{fn}(A)$.*

Proof: Let ψ be the functional introduced in Definition 6.1 to define $\mathrm{an}_\mathcal{D}$. Proceeding as in the proof of Lemma E.2, it is possible to show that $\psi(\mathrm{fn}) \subseteq \mathrm{fn}$. So fn is a prefixed-point of functional ψ and hence $\mathrm{an}_\mathcal{D} \subseteq \mathrm{fn}$, since $\mathrm{an}_\mathcal{D}$ is its least (pre)fixed-point. □

Lemma E.4 *If $z \notin \mathrm{an}_\mathcal{D}(A)$ and $A \stackrel{(M,\alpha)}{\longrightarrow} A' \in \mathrm{nr}(\mathcal{D})$ with $z \notin \mathrm{bn}(\alpha)$, then also $z \notin \mathrm{an}_\mathcal{D}(A')$.*

Corollary E.5 *Let $\mathcal{D}$ be a $\mathcal{CP}$-relation and $\mathcal{E} \stackrel{\mathrm{def}}{=} \mathcal{D} \cap \Downarrow$. Then $\mathrm{an}_\mathcal{D} = \mathrm{an}_\mathcal{E}$.*

Lemma E.6 *Let $\mathcal{D}$ be a $\mathcal{CP}$-relation closed for injective substitution, and σ be an injective substitution. Then $\mathrm{an}_\mathcal{D}(A\sigma) = \mathrm{an}_\mathcal{D}(A)\sigma$.*

Proof: Let $\overline{\mathrm{an}} : \mathcal{CP} \to 2^{\mathcal{N}}$ be the function so defined

$$\overline{\mathrm{an}}(B) = \bigcap_{\{\rho \text{ injective substitution}\}} \mathrm{an}_{\mathcal{D}}(B\rho)\rho^{-1}.$$

We prove that $\overline{\mathrm{an}}$ is a prefixed-point of functional ψ introduced in Definition 6.1 to define $\mathrm{an}_{\mathcal{D}}$. From this, it follows that $\mathrm{an}_{\mathcal{D}}(B) \subseteq \overline{\mathrm{an}}(B)$ for all B, since $\mathrm{an}_{\mathcal{D}}$ is its least (pre)fixed-point. Therefore, $\mathrm{an}_{\mathcal{D}}(B) \subseteq \mathrm{an}_{\mathcal{D}}(B\rho)\rho^{-1}$ for each constrained process B and each injective substitution ρ. By taking $B \stackrel{\mathrm{def}}{=} A$ and $\rho \stackrel{\mathrm{def}}{=} \sigma$ we obtain $\mathrm{an}_{\mathcal{D}}(A)\sigma \subseteq \mathrm{an}_{\mathcal{D}}(A\sigma)$ and, for $B \stackrel{\mathrm{def}}{=} A\sigma$ and $\rho \stackrel{\mathrm{def}}{=} \sigma^{-1}$ we obtain $\mathrm{an}_{\mathcal{D}}(A\sigma) \subseteq \mathrm{an}_{\mathcal{D}}(A)\sigma$, which implies the thesis. To show that $\overline{\mathrm{an}}$ is a prefixed-point of the functional ψ, we need to prove

$$\psi(\overline{\mathrm{an}})(B) \subseteq \overline{\mathrm{an}}(B) = \bigcap_{\{\rho \text{ injective substitution}\}} \mathrm{an}_{\mathcal{D}}(B\rho)\rho^{-1}.$$

For this, we show that $\psi(\overline{\mathrm{an}})(B) \subseteq \mathrm{an}_{\mathcal{D}}(B\rho)\rho^{-1}$, for all injective substitutions ρ. Suppose $n \in \psi(\overline{\mathrm{an}})(B)$; then by definition of ψ there is some $B \xrightarrow{(M,\alpha)} B' \in \mathrm{nr}(\mathcal{D})$ with

$$n \in \mathrm{fn}(\alpha, M) \cup (\overline{\mathrm{an}}(B') - \mathrm{bn}(\alpha)). \tag{27}$$

We can suppose $\mathrm{bn}(\alpha)$ neutral for ρ, otherwise we can find a variant of $B \xrightarrow{(M,\alpha)} B'$ which satisfies this condition and such that (27) still holds. From the definition of $\overline{\mathrm{an}}$ and some simple set properties,

$$\mathrm{fn}(\alpha, M) \cup (\overline{\mathrm{an}}(B') - \mathrm{bn}(\alpha)) = \bigcap_{\{\varsigma \text{ injective substitution}\}} \mathrm{fn}(\alpha, M) \cup (\mathrm{an}_{\mathcal{D}}(B'\varsigma)\varsigma^{-1} - \mathrm{bn}(\alpha)).$$

From this and (27), we obtain that for each injective substitution ς,

$$n \in \mathrm{fn}(\alpha, M) \cup (\mathrm{an}_{\mathcal{D}}(B'\varsigma)\varsigma^{-1} - \mathrm{bn}(\alpha)).$$

In particular, this is true for $\varsigma \stackrel{\mathrm{def}}{=} \rho$, so we have

$$n \in \mathrm{fn}(\alpha, M) \cup (\mathrm{an}_{\mathcal{D}}(B'\rho)\rho^{-1} - \mathrm{bn}(\alpha)). \tag{28}$$

By Lemma D.3, $B \xrightarrow{(M,\alpha)} B' \in \mathrm{nr}(\mathcal{D})$ and $\mathrm{bn}(\alpha)$ neutral for ρ imply $B\rho \xrightarrow{(M',\alpha')} B'\rho \in \mathrm{nr}(\mathcal{D})$ with $M' \lhd\rhd M\rho$ and $\alpha' = \alpha\rho\sigma_{M'}$. Since $\mathrm{fn}(\alpha, M) = \mathrm{fn}(\alpha', M')\rho^{-1}$, from (28) we have

$$n \in \big(\mathrm{fn}(\alpha', M') \cup (\mathrm{an}_{\mathcal{D}}(B'\rho) - \mathrm{bn}(\alpha))\big)\rho^{-1} \subseteq \mathrm{an}_{\mathcal{D}}(B\rho)\rho^{-1},$$

that concludes the proof. □

Lemma E.7 *Suppose a $\mathcal{CP}$-relation $\mathcal{D}$ is weak transitive, closed for injective substitution and that $\frown \subseteq \mathcal{D}$. If $A \frown B$ then $\mathrm{an}_{\mathcal{D}}(A) = \mathrm{an}_{\mathcal{D}}(B)$.*

Proof: Let $\overline{\mathrm{an}} : \mathcal{CP} \to 2^{\mathcal{N}}$ be the function so defined

$$\overline{\mathrm{an}}(A) = \bigcap_{\{B \,:\, A \frown B\}} \mathrm{an}_{\mathcal{D}}(B).$$

Proceeding as in the proof of the previous lemma, we prove that $\overline{\mathrm{an}}$ is a prefixed-point of functional ψ (introduced in Definition 6.1 to define $\mathrm{an}_{\mathcal{D}}$). From this, it follows that $\mathrm{an}_{\mathcal{D}}(A) \subseteq \overline{\mathrm{an}}(A)$, since $\mathrm{an}_{\mathcal{D}}$ is its least (pre)fixed-point. Then we can conclude $\mathrm{an}_{\mathcal{D}}(A) \subseteq \mathrm{an}_{\mathcal{D}}(B)$ from these implications:

$$\begin{array}{rcl} \mathrm{an}_{\mathcal{D}}(A) \subseteq \overline{\mathrm{an}}(A) & \Rightarrow & [\text{by definition of } \overline{\mathrm{an}}] \\ \mathrm{an}_{\mathcal{D}}(A) \subseteq \bigcap_{\{C \,:\, A \frown C\}} \mathrm{an}_{\mathcal{D}}(C) & \Rightarrow & [\text{since } A \frown B \text{ implies } \bigcap_{\{C \,:\, A \frown C\}} \mathrm{an}_{\mathcal{D}}(C) \subseteq \mathrm{an}_{\mathcal{D}}(B)] \\ \mathrm{an}_{\mathcal{D}}(A) \subseteq \mathrm{an}_{\mathcal{D}}(B). & & \end{array}$$

Similarly, $\mathrm{an}_{\mathcal{D}}(B) \subseteq \mathrm{an}_{\mathcal{D}}(A)$, which concludes the proof. □

Proof of Lemma 6.2: We should prove that $A \frown B$ implies $\mathrm{an}_\frown(A) = \mathrm{an}_\frown(B)$. This is a consequence of Lemma E.7 (and of the closure of $\frown$ for substitutions and weak transitivity). □

Proof of Lemma 6.3: We have to prove that $x \in \mathrm{an}_\frown(A)$ iff $A \not\frown (\boldsymbol{\nu} x)A$.
For the *if* case, it is simple to show that $A \frown (\boldsymbol{\nu} x)A$ implies $x \notin \mathrm{an}_\frown(A)$. In fact, by Lemma 6.2, $A \frown (\boldsymbol{\nu} x)A$ implies $\mathrm{an}_\frown(A) = \mathrm{an}_\frown((\boldsymbol{\nu} x)A)$ and hence $x \notin \mathrm{an}_\frown((\boldsymbol{\nu} x)A)$ (which is true by Lemma E.3) implies $x \notin \mathrm{an}_\frown(A)$.
The *only if* case is more complex. Let $\mathcal{D}$ be so defined:

$$\mathcal{D} \stackrel{\mathrm{def}}{=} \{(A, (\boldsymbol{\nu} x)A) \ : \ x \notin \mathrm{an}_\frown(A)\}.$$

One shows that $\mathcal{D} \rightarrowtail \mathcal{S}q_W(\mathcal{S}ub_W(\mathcal{D}))$. □

E.1 Proofs for Subsection 6.1

Lemma E.8 *If $A \frown_{\mathrm{an}} B$ then $A \frown B$.*

Proof: Let $A_0 \frown_{\mathrm{an}} B_0$. Then $A_0 \Downarrow B_0$ and there is some $\frown_{\mathrm{an}}$-bisimulation $\mathcal{D}$ such that $A_0 \ \mathcal{D} \ B_0$.
Now we define function $\overline{\mathrm{an}} : \mathcal{CP} \to 2^{\mathcal{N}}$. The definition is similar to the definition of function $\mathrm{an}_{\mathcal{D}}$, but it is based on normalised transitions rather than on non-redundant ones. Intuitively, function $\overline{\mathrm{an}}$ returns the names which are active w.r.t. normalised transitions. Function $\overline{\mathrm{an}}$ is the least fixed-point of the monotone function $\psi : (\mathcal{CP} \to 2^{\mathcal{N}}) \to (\mathcal{CP} \to 2^{\mathcal{N}})$ defined as follows:

$$\psi(f)(A) = \bigcup_{\{M,\alpha,A' \ : \ A \stackrel{(M,\alpha)}{\longmapsto}_{\mathcal{D}} A'\}} \mathrm{fn}(M,\alpha) \cup (f(A') - \mathrm{bn}(\alpha)).$$

We omit the proof that function ψ is monotone, and hence that $\overline{\mathrm{an}}$ is well defined. We now define relation $\mathcal{E}$ as follows:

$$\mathcal{E} \stackrel{\mathrm{def}}{=} \{(A\sigma_A, B\sigma_B) \ : \ A\sigma_A \Downarrow B\sigma_B \text{ and } A \ \mathcal{D} \ B \text{ and } \sigma_A, \sigma_B \text{ are injective and coincide on } \overline{\mathrm{an}}(A) \cap \overline{\mathrm{an}}(B)\}.$$

We show that $\mathcal{E} \rightarrowtail (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})$. This is sufficient to conclude that $A_0 \frown B_0$, since $(A_0, B_0) \in \mathcal{D}$ (by hypothesis on $\mathcal{D}$), $\mathcal{D} \subseteq \mathcal{E}$ (by the definition of $\mathcal{E}$), and $\mathcal{E} \subseteq \frown$ (by Theorem 4.10).
Let $A \stackrel{\mathrm{def}}{=} \langle P, D\rangle$ and $B \stackrel{\mathrm{def}}{=} \langle Q, E\rangle$ be two constrained processes such that $A \ \mathcal{D} \ B$ and let σ_A and σ_B be two injective substitutions which coincide on $\overline{\mathrm{an}}(A) \cap \overline{\mathrm{an}}(B)$ and such that $A\sigma_A \ \mathcal{E} \ B\sigma_B$.
We should prove that, for each transition $A\sigma_A \stackrel{(M',\alpha')}{\longrightarrow} A'$, with $\mathrm{bn}(\alpha') \cap \mathrm{fn}(B\sigma_B) = 0$, there is a matching transition from B. We consider only bound output transitions, since input transitions are similar and free transitions are simpler.
Suppose $A\sigma_A \stackrel{(M',\overline{c}(w))}{\longrightarrow} A'$. We can suppose that w is fresh, i.e., $w \notin \mathrm{fn}(A,B)$ and neutral for σ_A and σ_B; if w is not fresh, we can work with variants. By Lemma C.4, $A\sigma_A \stackrel{(M',\overline{c}(w))}{\longrightarrow} A'$ implies $A \stackrel{(M,\overline{a}(w))}{\longrightarrow} A''$ with

$$M' \lhd\rhd M\sigma_A \qquad c = \sigma_{M'}(\sigma_A(a)) \qquad A' = A''\sigma_A\sigma_{M'}. \tag{29}$$

Let $y \stackrel{\mathrm{def}}{=} \min\{\mathcal{N} - \mathrm{fn}(P)\}$. Then, we can consider the variant of $A \stackrel{(M,\overline{a}(w))}{\longrightarrow} A''$ which uses y as bound name, namely $A \stackrel{(M,\overline{a}(y))}{\longrightarrow} A''\{y/w\, w/y\}$. Hence, by Lemma 5.2, there exists a transition $A \stackrel{(N,\overline{b}(y))}{\longrightarrow} A''' \in \mathrm{nr}(\mathcal{D} \cap \Downarrow)$ with $M \rhd N$, $\sigma_M(b) = a$ and, if $A''' \stackrel{\mathrm{def}}{=} \langle P''', D'''\rangle$,

$$A''\{y/w\, w/y\} \ (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{D} \cap \Downarrow) \ \left\langle P'''\sigma_M, D'''^{M}_{P,\overline{a}(y)} \cap \mathrm{fn}(P'''\sigma_M)\right\rangle. \tag{30}$$

By Lemma D.2, $\mathrm{nr}(\mathcal{D} \cap \Downarrow) = \mathrm{nr}(\mathcal{D})$, so $A \stackrel{(N,\overline{b}(y))}{\longrightarrow} A'''$ is also in $\mathrm{nr}(\mathcal{D})$. By definition of normalised transitions,

$$A \stackrel{(N,\overline{b}(v))}{\longmapsto}_{\mathcal{D}} A'''\{v/y\, y/v\},$$

where $v \stackrel{\text{def}}{=} \min\{\mathcal{N} - \text{an}_{\mathcal{D}}(A)\}$. Since $\mathcal{D}$ is an $\frown_{\text{an}}$-bisimulation and $A\ \mathcal{D}\ B$, there must be some transition

$$B \xrightarrow{(N,\overline{b}(v))}_{\mathcal{D}} B''$$

such that $A'''\{v/y\, y/v\}\ \mathcal{D}\ B''$. By definition of normalised transitions, we have

$$B \xrightarrow{(N,\overline{b}(z))} B''' \in \text{nr}(\mathcal{D}),$$

where $z \stackrel{\text{def}}{=} \min\{\mathcal{N} - \text{fn}(B)\}$, $v = \min\{\mathcal{N} - \text{an}_{\mathcal{D}}(B)\}$ and $B'' = B'''\{v/z\, z/v\}$. We can rewrite $A'''\{v/y\, y/v\}\ \mathcal{D}\ B''$ as follows:

$$A'''\{v/y\, y/v\}\ \mathcal{D}\ B'''\{v/z\, z/v\}. \tag{31}$$

Let us consider the variant of the transition of B which uses w as bound name, namely $B \xrightarrow{(N,\overline{b}(w))} B'''\{w/y\, y/w\}$. By Lemma C.4 there exists a transition

$$B\sigma_B \xrightarrow{(N',\overline{d}(w))} B'''\{w/y\, y/w\}\sigma_B\sigma_{N'}$$

with $N' \lhd\rhd N\sigma_B$ and $d = \sigma_{N'}(\sigma_B(b))$.
Let $B''' \stackrel{\text{def}}{=} \langle Q''', E'''\rangle$. We show that the above transition from $B\sigma_B$ matches that from $A\sigma_A$; for this, we need to prove that $M' \rhd N'$, $c = \sigma_{M'}(d)$ and

$$A'\ (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E})\ \left\langle Q'''\{w/y\, y/w\}\sigma_B\sigma_{M'}, (D\sigma_A \cup E\sigma_B)^{M'}_{(P\sigma_A, Q\sigma_B),\overline{c}(w)} \cap \text{fn}(Q'''\{w/y\, y/w\}\sigma_B\sigma_{M'})\right\rangle. \tag{32}$$

The proofs that $M' \rhd N'$ and $c = \sigma_{M'}(d)$ are easy; it remains to prove (32). We do this in the remainder of the proof. First, we prove that

$$A'''\{w/y\, y/w\}\sigma_A\ \mathcal{E}\ B'''\{w/z\, z/w\}\sigma_B. \tag{33}$$

By definition of $\mathcal{E}$, this holds if:

1 $A'''\{w/y\, y/w\}\sigma_A \Downarrow B'''\{w/z\, z/w\}\sigma_B$,

2 $A'''\{w/y\, y/w\}\sigma_A = (A'''\{v/y\, y/v\})\sigma'_A$ for $\sigma'_A \stackrel{\text{def}}{=} \{w/v\, y/w\, v/y\}\sigma_A$,

3 $B'''\{w/z\, z/w\}\sigma_A = (B'''\{v/z\, z/v\})\sigma'_B$ for $\sigma'_B \stackrel{\text{def}}{=} \{w/v\, z/w\, v/z\}\sigma_B$,

4 $A'''\{v/y\, y/v\}\ \mathcal{D}\ B'''\{v/z\, z/v\}$, and

5 σ'_A and σ'_B are injective and coincide on $\overline{\text{an}}(A'''\{v/y\, y/v\}) \cap \overline{\text{an}}(B'''\{v/z\, z/v\})$.

We show that each of these requirements is true.

Requirement 1. Above, we have derived transition $B\sigma_B \xrightarrow{(N',\overline{d}(w))} B'''\{w/z\, z/w\}\sigma_B\sigma_{N'}$ from transition $B \xrightarrow{(N,\overline{b}(z))} B'''$. Analogously, from $A \xrightarrow{(N,\overline{b}(y))} A'''$ it is possible to derive $A\sigma_A \xrightarrow{(N',\overline{d}(w))} A'''\{w/y\, y/w\}\sigma_A\sigma_{N'}$. Then, by $A\sigma_A \Downarrow B\sigma_B$ and Lemma C.5,

$$A'''\{w/y\, y/w\}\sigma_A\sigma_{N'} \Downarrow B'''\{w/z\, z/w\}\sigma_B\sigma_{N'}. \tag{34}$$

Since $B'''\{w/y\, y/w\} = B'''\{w/z\, z/w\}\sigma_N$ and since $N' \lhd\rhd N\sigma_B$, by Lemma A.6 substitution $\sigma_B\sigma_{N'}$ is injective on $B'''\{w/z\, z/w\}$. Since σ_B is an injective substitution by hypothesis, $\sigma_{N'}$ has to be injective on $B'''\{w/z\, z/w\}\sigma_B$.
Analogously, $\sigma_{N'}$ is injective on $A'''\{w/y\, y/w\}\sigma_A$. Hence, we can apply Lemma C.2 to (34), obtaining $A'''\{w/y\, y/w\}\sigma_A \Downarrow B'''\{w/z\, z/w\}\sigma_B$.

Requirement 2. We have:

$$\begin{array}{rcl} A'''\{v/y\, y/v\}\sigma'_A & = & [\text{by definition of } \sigma'_A] \\ A'''\{v/y\, y/v\}\{w/v\, y/w\, v/y\}\sigma_A & = & [\text{since } \{v/y\, y/v\}\{w/v\, y/w\, v/y\} = \{w/y\, y/w\}] \\ A'''\{w/y\, y/w\}\sigma_A. & & \end{array}$$

Requirement 3. Similar to the proof of Requirement 2.
Requirement 4. This is (31).
Requirement 5. Substitutions σ'_A and σ'_B are injective. Suppose now $n \in \overline{\mathrm{an}}(A'''\{v/y\, y/v\}) \cap \overline{\mathrm{an}}(B'''\{v/z\, z/v\})$. By definition of $\overline{\mathrm{an}}$,

$$\overline{\mathrm{an}}(A'''\{v/y\, y/v\}) \subseteq \overline{\mathrm{an}}(A) \cup \{v\} \qquad \text{and} \qquad \overline{\mathrm{an}}(B'''\{v/z\, z/v\}) \subseteq \overline{\mathrm{an}}(B) \cup \{v\}.$$

We now distinguish two cases: $n = v$ and $n \in \overline{\mathrm{an}}(A) \cap \overline{\mathrm{an}}(B)$.
Case $n = v$. Since w is neutral for σ_A and σ_B, we have $\sigma'_A(v) = \sigma_A(w) = w = \sigma_B(w) = \sigma'_B(v)$.
Case $n \in \overline{\mathrm{an}}(A) \cap \overline{\mathrm{an}}(B)$. It must be $n \notin \{w, y, z\}$; in fact $w, y \notin \mathrm{fn}(A)$ and hence $w, y \notin \overline{\mathrm{an}}(A)$ and, similarly, $w, z \notin \overline{\mathrm{an}}(B)$. Hence, by definition of σ'_A and of σ'_B, we have $\sigma'_A(n) = \sigma_A(n) = \sigma_B(n) = \sigma'_B(n)$.
Therefore σ'_A and σ'_B coincide on $(\overline{\mathrm{an}}(A) \cap \overline{\mathrm{an}}(B)) \cup \{v\} \supseteq \overline{\mathrm{an}}(A'''\{v/y\, y/v\}) \cap \overline{\mathrm{an}}(B'''\{v/z\, z/v\})$.
This concludes the proof of (33). From it, by applying substitution σ_N we can deduce

$$A'''\{w/y\, y/w\}\sigma_A\sigma_N \; \mathcal{S}ub_W(\mathcal{E}) \; B'''\{w/z\, z/w\}\sigma_B\sigma_N \tag{35}$$

(the two processes are compatible due to (34)).

Now we prove that

$$A' \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \Big\langle P'''\{w/y\, y/w\}\sigma_A\sigma_{M'}, (D\sigma_A)^{M'}_{P\sigma_A, \overline{c}(w)} \cap \mathrm{fn}(P'''\{w/y\, y/w\}\sigma_A\sigma_{M'}) \Big\rangle. \tag{36}$$

Since $\mathcal{D} \cap \Downarrow \subseteq \mathcal{E}$, from (30) we deduce

$$A''\{y/w\, w/y\} \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \Big\langle P'''\sigma_M, D^M_{P,\overline{a}(y)} \cap \mathrm{fn}(P'''\sigma_M) \Big\rangle.$$

Hence, by applying injective substitution $\{y/w\, w/y\}$,

$$A'' \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \Big\langle P'''\sigma_M\{y/w\, w/y\}, \big(D^M_{P,\overline{a}(y)} \cap \mathrm{fn}(P'''\sigma_M)\big)\{y/w\, w/y\} \Big\rangle.$$

By Lemma A.12, this can be rewritten as

$$A'' \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \Big\langle P'''\sigma_M\{y/w\, w/y\}, D^M_{P,\overline{a}(y)}\{y/w\, w/y\} \cap \mathrm{fn}(P'''\sigma_M\{y/w\, w/y\}) \Big\rangle.$$

Moreover, $P'''\sigma_M\{y/w\, w/y\} = P'''\{y/w\, w/y\}\sigma_M$ since $y, w \notin \mathrm{n}(M)$, and $D^M_{P,\overline{a}(y)}\{y/w\, w/y\} = D^M_{P,\overline{a}(w)}$ since $y, w \notin \mathrm{fn}(P) \cup \mathrm{n}(D, M) \cup \{c\}$. Therefore,

$$A'' \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \Big\langle P'''\{y/w\, w/y\}\sigma_M, D^M_{P,\overline{a}(w)} \cap \mathrm{fn}(P'''\{y/w\, w/y\}\sigma_M) \Big\rangle. \tag{37}$$

By definition of $D^M_{P,\overline{a}(w)}$ and Lemma A.6, substitution $\sigma_A\sigma_{M'}$ is injective on $D^M_{P,\overline{a}(w)}$. Then $\sigma_A\sigma_{M'}$ respects $D^M_{P,\overline{a}(w)}$, so we can apply this substitution to (37), obtaining

$$A''\sigma_A\sigma_{M'} \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \Big\langle P'''\{y/w\, w/y\}\sigma_M\sigma_A\sigma_{M'}, \big(D^M_{P,\overline{a}(w)} \cap \mathrm{fn}(P'''\{y/w\, w/y\}\sigma_M)\big)\sigma_A\sigma_{M'} \Big\rangle.$$

Since $\sigma_A\sigma'_M$ is injective on $\mathrm{fn}(P'''\{y/w\, w/y\}\sigma_M)$, we can apply Lemma A.12, obtaining

$$A''\sigma_A\sigma_{M'} \; (\mathcal{S}ub_W \cup \mathcal{S}q_W \cup \mathcal{S}ym)^\star(\mathcal{E}) \; \Big\langle P'''\{y/w\, w/y\}\sigma_M\sigma_A\sigma_{M'}, D^M_{P,\overline{a}(w)}\sigma_A\sigma_{M'} \cap \mathrm{fn}(P'''\{y/w\, w/y\}\sigma_M\sigma_A\sigma_{M'}) \Big\rangle.$$

But this is the same as (36), since

- $A' = A''\sigma_A\sigma_{M'}$ by (29),
- $\sigma_M\sigma_A\sigma_{M'} = \sigma_A\sigma_{M'}$ by Lemma A.4, and
- $D^M_{P,\overline{a}(w)}\sigma_A\sigma_{M'} = (D\sigma_A)^{M'}_{P\sigma_A,\overline{c}(w)}$ by Lemma A.14.

This concludes the proof of (36).

We can now finally close the proof of this lemma. By definition of $\mathcal{E}$ we have that $A\sigma_A \Downarrow B\sigma_B$. Moreover, transitions

$$A\sigma_A \xrightarrow{(M',\overline{c}(w))} A' \qquad A\sigma_A \xrightarrow{(N',\overline{d}(w))} A'''\{w/y\, y/w\}\sigma_A\sigma_{N'} \qquad B\sigma_B \xrightarrow{(N',\overline{d}(w))} B'''\{w/z\, z/w\}\sigma_B\sigma_{N'}$$

have been derived. We already know that $M' \rhd N'$ and that $c = \sigma_{M'}(d)$. Finally, we have proved (35) and (36). So, all the hypothesis of Lemma C.12 are satisfied, and (32) is the thesis of that Lemma. This concludes the proof. □

Lemma E.8 proves an implication of Theorem 6.7; we postpone the proof of the opposite implication to the next appendix.

F Proofs for Section 7

Lemma F.1 *Let $\mathcal{D}$ be an equivalence on constrained processes that is closed for injective substitutions. Then $A\ \Phi(\mathcal{D})\ B$ implies $\mathrm{an}_{\mathcal{D}}(A) = \mathrm{an}_{\mathcal{D}}(B)$.*

Lemma F.2 *If $\mathcal{D}_1$, $\mathcal{D}_2$ and $\mathcal{E}$ are $\mathcal{CP}$-relations such that $\mathcal{D}_1 \supseteq \mathcal{D}_2$ and $\mathcal{E}$ is closed for $\mathrm{an}_{\mathcal{D}_1}$-injective substitutions, then $\mathcal{E}$ is also closed for $\mathrm{an}_{\mathcal{D}_2}$-injective substitutions.*

Proof of Lemma 7.8: We have to show that, if $\mathcal{D}$ is an equivalence closed for $\mathrm{an}_{\mathcal{D}}$-injective substitutions, then also $\Phi(\mathcal{D})$ is an equivalence closed for $\mathrm{an}_{\Phi(\mathcal{D})}$-injective substitutions.
Starting from the hypothesis that $\mathcal{D}$ is an equivalence, the proof that $\Phi(\mathcal{D})$ is an equivalence is standard. So we just show that $\Phi(\mathcal{D})$ is closed for $\mathrm{an}_{\mathcal{D}}$-injective substitutions. This implies the thesis; in fact $\Phi(\mathcal{D})$ is closed also for $\mathrm{an}_{\Phi(\mathcal{D})}$-injective substitutions by Lemma F.2, since $\Phi(\mathcal{D}) \subseteq \mathcal{D}$.
To prove that $\Phi(\mathcal{D})$ is closed for $\mathrm{an}_{\mathcal{D}}$-injective substitutions, we consider $\mathcal{CP}$-relation $\mathcal{E}$ defined as follows

$$\mathcal{E} \stackrel{\mathrm{def}}{=} \{(A\sigma_A, B\sigma_B)\ :\ A\ \Phi(\mathcal{D})\ B \text{ and } \sigma_A, \sigma_B \text{ are injective and coincide on } \mathrm{an}_{\mathcal{D}}(A) \cap \mathrm{an}_{\mathcal{D}}(B)\}$$

and we show that $\mathcal{E}$ is a prefixed-point of function $\Psi_{\mathcal{D}}$. Then $\mathcal{E} = \Phi(\mathcal{D})$, since $\Phi(\mathcal{D}) \subseteq \mathcal{E}$ and $\Phi(\mathcal{D})$ is the greatest (pre)fixed-point of $\Psi_{\mathcal{D}}$.
Suppose $A_0\ \mathcal{E}\ B_0$. Then there are processes A and B and injective substitutions σ_A and σ_B which coincide on $\mathrm{an}_{\mathcal{D}}(A) \cap \mathrm{an}_{\mathcal{D}}(B)$, such that $A_0 = A\sigma_A$, $B_0 = B\sigma_0$ and $A\ \Phi(\mathcal{D})\ B$. We show that $A_0\ \Psi_{\mathcal{D}}(\mathcal{E})\ B_0$.
Suppose $A_0 \stackrel{(M_0,\alpha_0)}{\longmapsto}_{\mathcal{D}} A'_0$. We show that there is some $B_0 \stackrel{(M_0,\alpha_0)}{\longmapsto}_{\mathcal{D}} B'_0$ such that $A'_0\ \mathcal{E}\ B'_0$. We only consider the case of a bound output actions. Let

$$\begin{array}{lcllcl} v & \stackrel{\mathrm{def}}{=} & \min\{\mathcal{N} - \mathrm{an}_{\mathcal{D}}(A)\} & \qquad v_0 & \stackrel{\mathrm{def}}{=} & \min\{\mathcal{N} - \mathrm{an}_{\mathcal{D}}(A_0)\} \\ y & \stackrel{\mathrm{def}}{=} & \min\{\mathcal{N} - \mathrm{fn}(A)\} & \qquad y_0 & \stackrel{\mathrm{def}}{=} & \min\{\mathcal{N} - \mathrm{fn}(A_0)\} \\ z & \stackrel{\mathrm{def}}{=} & \min\{\mathcal{N} - \mathrm{fn}(B)\} & \qquad z_0 & \stackrel{\mathrm{def}}{=} & \min\{\mathcal{N} - \mathrm{fn}(B_0)\} \end{array}$$

and let w be a fresh name, i.e.,

$$w \notin \mathrm{fn}(A, B, A_0, B_0) \cup \{y, z, v, y_0, z_0, v_0\} \text{ and } w \text{ neutral for } \sigma_A \text{ and } \sigma_B.$$

Since $A_0 \stackrel{(M_0,\alpha_0)}{\longmapsto}_{\mathcal{D}} A'_0$, by definition of normalised transition it must be $\alpha_0 = \overline{x_0}(v_0)$ for some x_0; so we have

$$A_0 \stackrel{(M_0,\overline{x_0}(v_0))}{\longmapsto}_{\mathcal{D}} A'_0.$$

Again by definition of normalised transitions, we have $A_0 \xrightarrow{(M_0,\overline{x_0}(y_0))} A_0'\{y_0/v_0\ v_0/y_0\} \in \mathrm{nr}(\mathcal{D})$.
By Lemma D.4, $A_0 \xrightarrow{(M_0,\overline{x_0}(w))} A_0'' \in \mathrm{nr}(\mathcal{D})$ with

$$A_0'' = A_0'\{y_0/v_0\ v_0/y_0\}\{w/y_0\ y_0/w\} \qquad A_0' = A_0''\{w/y_0\ y_0/w\}\{y_0/v_0\ v_0/y_0\}. \tag{38}$$

By Lemma D.3, there is some $A \xrightarrow{(M,\overline{x}(w))} A'' \in \mathrm{nr}(\mathcal{D})$ such that

$$M_0 \lhd\rhd M\sigma_A \qquad x_0 = \sigma_{M_0}(\sigma_A(x)) \qquad A_0'' = A''\sigma_A\sigma_{M_0}. \tag{39}$$

By Lemma D.4, $A \xrightarrow{(M,\overline{x}(y))} A''\{w/y\ y/w\} \in \mathrm{nr}(\mathcal{D})$, and, by definition of normalised transitions,

$$A \xmapsto{(M,\overline{x}(v))}_{\mathcal{D}} A'$$

where

$$A' = A''\{w/y\ y/w\}\{y/v\ v/y\} \qquad A'' = A'\{y/v\ v/y\}\{w/y\ y/w\}. \tag{40}$$

Since $A\ \Phi(\mathcal{D})\ B$ and $\Phi(\mathcal{D})$ is a fixed-point of $\Psi_{\mathcal{D}}$, there is some

$$B \xmapsto{(M,\overline{x}(v))}_{\mathcal{D}} B'$$

such that $A'\ \Phi(\mathcal{D})\ B'$.
Now we work in the inverse direction, and deduce $B_0 \xmapsto{(M_0,\overline{x_0}(v_0))}_{\mathcal{D}} B_0'$ from $B \xmapsto{(M,\overline{x}(v))}_{\mathcal{D}} B'$.
By definition of normalised transitions and by Lemma D.4, we have

$$B \xrightarrow{(M,\overline{x}(w))} B'' \in \mathrm{nr}(\mathcal{D})$$

with

$$B' = B''\{w/z\ z/w\}\{z/v\ v/z\} \qquad B'' = B'\{z/v\ v/z\}\{w/z\ z/w\}. \tag{41}$$

By Lemma D.3, there is some $B_0 \xrightarrow{(M_0,\overline{x_0}(w))} B_0'' \in \mathrm{nr}(\mathcal{D})$ such that

$$B_0'' = B''\sigma_B\sigma_{M_0}. \tag{42}$$

Notice that, since $\mathrm{n}(M) \subseteq \mathrm{an}_{\mathcal{D}}(A) \cap \mathrm{an}_{\mathcal{D}}(B)$, then $M\sigma_A = M\sigma_B$, so $M_0 \lhd\rhd M\sigma_B$. Similarly, $x_0 = \sigma_{M_0}(\sigma_B(x))$.
Now we show that

$$v_0 = \min\{\mathcal{N} - \mathrm{an}_{\mathcal{D}}(B)\}. \tag{43}$$

We know that $\mathrm{an}_{\mathcal{D}}(A) = \mathrm{an}_{\mathcal{D}}(B)$ by Lemma F.1; since σ_A and σ_B coincide on $\mathrm{an}_{\mathcal{D}}(A) \cap \mathrm{an}_{\mathcal{D}}(B)$, it also holds that $\mathrm{an}_{\mathcal{D}}(A)\sigma_A = \mathrm{an}_{\mathcal{D}}(A)\sigma_B$. By Lemma E.6 this implies $\mathrm{an}_{\mathcal{D}}(A_0) = \mathrm{an}_{\mathcal{D}}(B_0)$. Then (43) is true since $v_0 = \min\{\mathcal{N} - \mathrm{an}_{\mathcal{D}}(A_0)\}$.
So, by Lemma D.4 and definition of normalised transitions,

$$B_0 \xmapsto{(M_0,\overline{x_0}(v_0))}_{\mathcal{D}} B_0'$$

with

$$B_0'' = B_0'\{z_0/v_0\ v_0/z_0\}\{w/z_0\ z_0/w\} \qquad B_0' = B_0''\{w/z_0\ z_0/w\}\{z_0/v_0\ v_0/z_0\}. \tag{44}$$

It remains to show that $A_0'\ \mathcal{E}\ B_0'$.
Since $A'\ \Phi(\mathcal{D})\ B'$, it is sufficient to prove that there exist two injective substitutions σ_A' and σ_B' which coincide on $\mathrm{an}_{\mathcal{D}}(A') \cap \mathrm{an}_{\mathcal{D}}(B')$ and such that $A_0' = A'\sigma_A'$ and $B_0' = B'\sigma_B'$.

Now we will exhibit two substitutions which satisfy all the required properties, except that they are not injective; however they are injective on $\mathrm{fn}(A', B')$, which is sufficient: in fact, it is possible to make the two substitutions injective by changing their behaviour on $\mathcal{N} - \mathrm{fn}(A', B')$, and this preserver the other properties. The two substitutions are

$$\sigma'_A = \{y/v\ v/y\}\{w/y\ y/w\}\sigma_A\sigma_{M_0}\{w/y_0\ y_0/w\}\{y_0/v_0\ v_0/y_0\}$$

and

$$\sigma'_B = \{z/v\ v/z\}\{w/z\ z/w\}\sigma_B\sigma_{M_0}\{w/z_0\ z_0/w\}\{z_0/v_0\ v_0/z_0\}.$$

Equality $A'_0 = A'\sigma'_A$ holds by (38), (39) and (40), whereas $B'_0 = B'\sigma'_B$ holds by by (44), (42) and (41). We omit the proof that σ'_A and σ'_B coincide on $\mathrm{an}_{\mathcal{D}}(A) \cap \mathrm{an}_{\mathcal{D}}(B)$ and that they are respectively injective on $\mathrm{fn}(A')$ and $\mathrm{fn}(B')$. This concludes the proof that $A'_0\ \mathcal{E}\ B'_0$.
We have shown that $A_0 \overset{(M_0,\alpha_0)}{\longmapsto}_{\mathcal{D}} A'_0$ implies $B_0 \overset{(M_0,\alpha_0)}{\longmapsto}_{\mathcal{D}} B'_0$ with $A'_0\ \mathcal{E}\ B'_0$. Similarly, $B_0 \overset{(M_0,\alpha_0)}{\longmapsto}_{\mathcal{D}} B'_0$ implies $A_0 \overset{(M_0,\alpha_0)}{\longmapsto}_{\mathcal{D}} A'_0$ with $A'_0\ \mathcal{E}\ B'_0$. We also have $A_0\ \mathcal{D}\ B_0$ since $A\ \mathcal{D}\ B$ and $\mathcal{D}$ is closed for $\mathrm{an}_{\mathcal{D}}$-injective substitutions. This implies $A_0\ \Phi_{\mathcal{D}}(\mathcal{E})\ B_0$ and concludes the proof. □

We omit the proof of Lemma 7.7 since it involves a reasoning which is similar to the previous proof.

Proof of Corollary 7.9: We have to show that $\frown \subseteq \simeq \cap \Downarrow$. This is done by proving, by transfinite induction, that $\frown \subseteq \Phi^i$: this proves that $\frown \subseteq \simeq$. Finally, $\frown \subseteq \Downarrow$ by definition of $\frown$. □

We can now prove Theorem 6.7, from Section 6.

Proof of Theorem 6.7: We have to show that $\frown$ and $\frown_{\mathrm{an}}$ coincide. Inclusion $\frown_{\mathrm{an}} \subseteq \frown$ is given by Lemma E.8. Inclusion $\frown \subseteq \frown_{\mathrm{an}}$ is given by Corollary 7.9 and Corollary 7.4. □

Finally, the proof of Theorem 7.5.

Proof of Theorem 7.5: We should prove that $\simeq \cap \Downarrow = \frown$. By Corollary 7.9, $\frown \subseteq \simeq \cap \Downarrow$. By Corollary 7.4, $\simeq$ is an $\frown_{\mathrm{an}}$-bisimulation and hence $\simeq \cap \Downarrow \subseteq \frown$ by Theorem 6.7. □