Symbolic Bisimulation for the Applied Pi Calculus *

Stéphanie Delaune[1] Steve Kremer[1]
Mark D. Ryan[2]

[1]LSV, ENS Cachan & CNRS & INRIA, France
[2]School of Computer Science, University of Birmingham, UK

Abstract

We propose a symbolic semantics for the finite applied pi calculus. The applied pi calculus is a variant of the pi calculus with extensions for modelling cryptographic protocols. By treating inputs symbolically, our semantics avoids potentially infinite branching of execution trees due to inputs from the environment. Correctness is maintained by associating with each process a set of constraints on terms. We define a symbolic labelled bisimulation relation, which is shown to be sound but not complete with respect to standard bisimulation. We explore the lack of completeness and demonstrate that the symbolic bisimulation relation is sufficient for many practical examples. This work is an important step towards automation of observational equivalence for the finite applied pi calculus, e.g. for verification of anonymity or strong secrecy properties.

*This work has been partly supported by the EPSRC projects EP/E029833, *Verifying Properties in Electronic Voting Protocols* and EP/E040829/1, *Verifying Anonymity and Privacy Properties of Security Protocols*, the ARA SESUR project AVOTÉ and the ARTIST2 NoE. Preliminary versions of this paper appeared in [13] and [14].

1 Introduction **3**

2 The Applied Pi Calculus **4**
2.1 Syntax and Informal Semantics 5
2.2 Semantics 6
2.3 Equivalences 9

Part I: Intermediate Calculus **10**

3 Syntax and Semantics **11**
3.1 Syntax 11
3.2 Semantics 13

4 Soundness and Completeness **15**

5 Intermediate Bisimulation **17**

Part II: Symbolic Calculus **22**

6 Constraint systems **22**

7 Syntax and Semantics **24**
7.1 Syntax 24
7.2 Symbolic semantics 25

8 Soundness and Completeness **27**

9 Symbolic Equivalences **29**

Part III: Soundness of Symbolic Bisimulation **34**

10 Discussion **34**

11 Related and Future Work **39**

Appendix **42**

B Proofs of Part I – Intermediate Calculus **42**

C Proofs of Part II – Symbolic Calculus **50**

1 Introduction

The *applied pi calculus* [2] is a derivative of the pi calculus [21] that is specialised for modelling cryptographic protocols. Participants in a protocol are modelled as processes, and the communication between them is modelled by means of channels, names and message passing. The main difference with the pi calculus is that the applied pi calculus allows one to manipulate *complex data*, instead of just names. These data are generated by a term algebra and equality is treated modulo an *equational theory*. For instance the equation $\mathsf{dec}(\mathsf{enc}(x, y), y) = x$ models the fact that encryption and decryption with the same key cancel out in the style of the Dolev-Yao model [16]. Such complex data requires the use of a special kind of processes called *active substitutions*. As an example consider the following process and reduction step:

$$\nu a, k.\mathsf{out}(c, \mathsf{enc}(a, k)).P \xrightarrow{\nu x.out(c,x)} \nu a, k.(P \mid \{^{\mathsf{enc}(a,k)}/_{x}\}).$$

The process outputs a secret name a which has been encrypted with the secret key k on a public channel c. The active substitution $\{^{\mathsf{enc}(a,k)}/_{x}\}$ gives the environment the ability to access the term $\mathsf{enc}(a, k)$ via the fresh variable x without revealing a or k. The applied pi calculus also generalizes the *spi calculus* [3] which only allows a fixed set of built-in primitives (symmetric and public-key encryption), while the applied pi calculus allows one to define a variety of primitives by means of an equational theory.

One of the difficulties in automating the proof of properties of systems is the infinite number of possible behaviours of the attacker, even in the case that the process itself is finite. When the process requests an input from the environment, the attacker can give any term which can be constructed from freely available data and the terms it has learned so far in the protocol, and therefore the execution tree of the process is potentially infinite-branching. To address this problem, researchers have proposed *symbolic abstractions* of processes, in which terms input from the environment are represented as symbolic variables, together with some constraints. These constraints describe the knowledge of the attacker (and therefore, the range of possible values of the symbolic variable) at the time the input was performed.

Reachability properties can be verified by deciding satisfiability of constraint systems resulting from symbolic executions of process algebras (e.g. [20, 4]). Similarly, *off-line guessing attacks* coded as *static equivalence* between process states [5] can be decided using such symbolic executions, but this requires one to check the equivalence of constraint systems, rather than satisfiability. Decision procedures for both satisfiability [11] and equivalence [5] of constraint systems exist for significant families of equational theories. *Observational equivalence properties*, which can be characterized as a bisimulation, express the inability of the attacker to distinguish between two processes no matter how it interacts with them. These properties are useful for modelling anonymity and privacy properties (e.g. [12]), as well as strong secrecy. In the spi calculus [3] properties were actually expressed as a testing relation and bisimulation was used as a proof

technique. Symbolic methods have also been used for bisimulation in process algebras [18, 9]. In particular, Borgström *et al.* [10] define a sound symbolic bisimulation for the spi calculus.

In this paper we propose a symbolic semantics for the applied pi calculus together with a sound symbolic bisimulation. To show that a symbolic bisimulation implies the concrete one, we generally need to prove that the symbolic semantics is both sound and complete. The semantics of the applied pi calculus is not well suited for defining such a symbolic semantics. In particular, we argue at the beginning of Part I that defining a symbolic structural equivalence which is both sound and complete seems impossible. The absence of sound and complete symbolic structural equivalence significantly complicates the proof of our main result. We therefore split it into two parts. We define a more restricted semantics which will provide an *intermediate* representation of applied pi calculus processes (Part I). These intermediate processes are a selected (but sufficient) subset of the original processes. One may think of them as being processes in some kind of normal form. We equip these intermediate processes with a labelled bisimulation that coincides with the original one. Then we present a symbolic semantics which is both sound and complete with respect to the intermediate one and give a sound symbolic bisimulation (Part II).

To keep track of the constraints on symbolic variables we associate a constraint system to each symbolic process. Keeping these constraint systems separate from the process allows us to have a clean division between the bisimulation and the constraint solving part. In particular we can directly build on existing work [5] and obtain a decision procedure for our symbolic bisimulation for a significant family of equational theories whenever the constraint system does not contain disequalities. This corresponds to the fragment of the applied pi calculus without else branches in the conditional. For this fragment, one may also notice that our symbolic semantics can be used to verify reachability properties using the constraint solving techniques from [11]. Another side-effect of the separation between the processes and the constraint system is that we forbid α-conversion on symbolic processes as we lose the scope of names in the constraint system, but allow explicit renaming when necessary (using *naming environments*). We believe that the simplicity of our intermediate calculus (especially the structural equivalence) and the absence of α-conversion is appealing in view of an implementation.

Finally, one may note that as in [10, 8], our technique for deciding bisimulation is incomplete. However, we argue that our technique works for many interesting cases. This is the purpose of Part III. Most of the proofs are in Appendix. Those that are omitted can be found in [15].

2 The Applied Pi Calculus

The applied pi calculus [2] is a language for describing concurrent processes and their interactions. It is based on the pi calculus [21], but is intended to be less pure and therefore more convenient to use. In this paper we only consider the

finite applied pi calculus which does not have replication.

2.1 Syntax and Informal Semantics

To describe processes in the applied pi calculus, one starts with a set of *names* (which are used to name communication channels or other constants), a set of *variables*, and a *signature* Σ which consists of the function symbols which will be used to define terms. In the case of security protocols, typical *function symbols* will include enc for encryption, which takes a plaintext and a key and returns the corresponding ciphertext, and dec for decryption, taking a ciphertext and a key and returning the plaintext (if the decryption key matches the encryption). *Terms* are defined as names, variables, and function symbols applied to other terms. We write $vars(T)$ for the set of variables occurring in T. When $vars(T) = \emptyset$ we say that the term T is *ground*.

We rely on a sort system for terms. It includes a universal base type and a channel type. We denote by $\mathcal{N}$ the set of names and among those names we distinguish the set $\mathcal{N}_{ch}$ of channel names. Similarly, we denote by $\mathcal{X}$ the set of variables. Among those variables, we distinghuish two disjoint sets: $\mathcal{X}_b$ the set of variables of base type and $\mathcal{X}_{ch}$ the set of variables of channel type. Of course function symbol application must respect sorts and arities. Function symbols cannot be applied to variables or names of channel sort, and cannot return terms of that sort, so in fact the only terms of channel sort are variables and names of that sort.

We define the equations which hold on terms constructed from the signature as an *equational theory* E. We denote $=_{\mathsf{E}}$ the equivalence relation induced by E.

Example 2.1 *A typical example of an equational theory is defined by the equation* $\mathsf{dec}(\mathsf{enc}(x,k),k) = x$. *Let* $T_1 = \mathsf{dec}(\mathsf{enc}(\mathsf{enc}(n,k_1),k_2),k_2)$ *and* $T_2 = \mathsf{enc}(n,k_1)$. *We have that* $T_1 =_{\mathsf{E}} T_2$ *(while obviously the syntactic equality* $T_1 = T_2$ *does not hold).*

In the applied pi calculus, one has *plain processes*, denoted by P, Q, R and *extended processes*, denoted by A, B, C. Plain processes are built up in a similar way to processes in the pi calculus, except that messages can contain terms (rather than just names). In the grammar described below, M and N are terms, n is a name, x a variable and u is a metavariable, standing either for a name or a variable. Extended processes add *active substitutions*, and restriction on names and variables.

$P, Q, R :=$	plain processes	$A, B, C :=$	extended processes
	0		P
	$P \mid Q$		$A \mid B$
	$\nu n.P$		$\nu n.A$
	if $M = N$ then P else Q		$\nu x.A$
	$\mathsf{in}(u,x).P$		$\{^M/_x\}$
	$\mathsf{out}(u,N).P$		

The substitution $\{^M/_x\}$ replaces the variable x with the term M (x and M have the same sort which is required to be a base sort). Active substitutions generalise "let". The process $\nu x.(\{^M/_x\} \mid P)$ corresponds exactly to the process "let $x = M$ in P". As usual, names and variables have scopes, which are delimited by restrictions and by inputs. We write $fv(A)$, $bv(A)$, $fn(A)$ and $bn(A)$ for the sets of *free* and *bound variables* and *free* and *bound names* of A, respectively. In an extended process, there is at most one substitution for each variable, and there is exactly one when the variable is restricted. We say that an extended process is *closed* if all its variables are either bound or defined by an active substitution. We also allow the usual abuse of notations: we omit trailing 0 processes and "else 0" branches in conditionals and write $\nu u_1, u_2, \ldots, u_n$ instead of $\nu u_1.\nu u_2.\ \ldots\ .\nu u_n$.

Active substitutions are useful because they allow us to map an extended process A to its *frame* $\phi(A)$ by replacing every plain process in A with 0. A frame is an extended process built up from 0 and active substitutions by parallel composition and restriction. The frame $\phi(A)$ can be viewed as an approximation of A that accounts for the static knowledge A exposes to its environment, but not A's dynamic behaviour. The *domain* of a frame φ denoted by $\mathrm{dom}(\varphi)$, is the set of variables for which φ defines a substitution (those variables x for which φ contains a substitution $\{^M/_x\}$ not under a restriction on x). We also define the domain of an extended process A, written $\mathrm{dom}(A)$ to be the domain of its frame, i.e., $\mathrm{dom}(A) = \mathrm{dom}(\phi(A))$.

An *evaluation context* $C[\_]$ is an extended process with a hole instead of an extended process.

2.2 Semantics

The operational semantics of processes in the applied pi calculus is defined by structural rules defining two relations: *structural equivalence* and *internal reduction*.

Structural equivalence, noted $\equiv$, is the smallest equivalence relation on extended processes that is closed under α-conversion on names and variables, application of evaluation contexts, and such that:

Par-0	$A \mid 0$	$\equiv$	A	
Par-A	$A \mid (B \mid C)$	$\equiv$	$(A \mid B) \mid C$	
Par-C	$A \mid B$	$\equiv$	$B \mid A$	
New-0	$\nu n.0$	$\equiv$	0	
New-C	$\nu u.\nu v.A$	$\equiv$	$\nu v.\nu u.A$	
New-Par	$A \mid \nu u.B$	$\equiv$	$\nu u.(A \mid B)$	if $u \notin fn(A) \cup fv(A)$
Alias	$\nu x.\{^M/_x\}$	$\equiv$	0	
Subst	$\{^M/_x\} \mid A$	$\equiv$	$\{^M/_x\} \mid A\{^M/_x\}$	
Rewrite	$\{^M/_x\}$	$\equiv$	$\{^N/_x\}$	if $M =_{\mathsf{E}} N$

We also define $=_\alpha$ for equality closed under α-renaming.

Example 2.2 *Let* $P = \nu s, k.(\mathsf{out}(c_1, \mathsf{enc}(s,k)) \mid \mathsf{in}(c_1, y).\mathsf{out}(c_2, \mathsf{dec}(y,k)))$. *The first component publishes the message* $\mathsf{enc}(s,k)$ *by sending it on* c_1. *The second one receives a message on* c_1, *uses the secret key* k *to decrypt it, and forwards the resulting plaintext on* c_2. *The process* P *is structurally equivalent to the following extended process* A*:*

$$A = \nu s, k, x_1.(\mathsf{out}(c_1, x_1) \mid \mathsf{in}(c_1, y).\mathsf{out}(c_2, \mathsf{dec}(y,k)) \mid \{^{\mathsf{enc}(s,k)}/_{x_1}\})$$

We have $\phi(A) = \nu s, k, x_1.\{^{enc(s,k)}/_{x_1}\} \equiv 0$ *(since* x_1 *is under a restriction).*

As already noted in [2], any closed frame is structurally equivalent to a sequence of active substitutions under some restricted names $\nu\tilde{n}.\{^{M_1}/_{x_1}\} \mid \ldots \mid \{^{M_k}/_{x_k}\}$. Therefore, we sometimes refer to such a frame as $\nu\tilde{n}.\sigma$ where σ is the substitution of terms for variables obtained by taking the union of the active substitutions.

Internal reduction, noted $\rightarrow$, is the smallest relation on extended processes closed under structural equivalence and application of evaluation contexts such that

Comm	$\mathsf{out}(a, M).P \mid \mathsf{in}(a, x).Q$	$\rightarrow$	$P \mid Q\{^{M}/_{x}\}$	
Then	if $M = N$ then P else Q	$\rightarrow$	P	where $M =_{\mathsf{E}} N$
Else	if $M = N$ then P else Q	$\rightarrow$	Q	
	for any ground terms M and N such that $M \neq_{\mathsf{E}} N$			

Note that the presentation of the communication rule (Comm) slightly differs from the one given in [2], but our presentation is easily shown to be equivalent to theirs. The above presentation is closer to our symbolic semantics and therefore more convenient for the purpose of this paper. Comparisons (Then and Else) are kept unchanged and directly depend on the underlying equational theory; using Else sometimes requires that active substitutions in the context be applied first, to yield ground terms M and N. Terms M and N are required to be ground in the rule Else because disequality is not stable under substitution of terms for variables, unlike equality which explains the absence of this condition in the Then rule.

The operational semantics is extended by a *labelled* operational semantics enabling us to reason about processes that interact with their environment. Labelled operational semantics defines the relation $\xrightarrow{\alpha}$ where α is either $\mathsf{in}(a, M)$ (a is a channel name and M is a term that can contain names and variables), or $\nu x.\mathsf{out}(a, x)$ (x is variable of base type), or $\mathsf{out}(a, c)$ or $\nu c.\mathsf{out}(a, c)$ (c is a channel name). We adopt the following rules in addition to the internal reduction rules:

IN $$\mathsf{in}(a,x).P \xrightarrow{in(a,M)} P\{^M/_x\}$$

OUT-CH $$\mathsf{out}(a,c).P \xrightarrow{out(a,c)} P$$

OPEN-CH $$\frac{A \xrightarrow{out(a,c)} A' \qquad c \neq a}{\nu c.A \xrightarrow{\nu c.out(a,c)} A'}$$

OUT-T $$\mathsf{out}(a,M).P \xrightarrow{\nu x.out(a,x)} P \mid \{^M/_x\} \quad x \notin fv(P) \cup fv(M)$$

SCOPE $$\frac{A \xrightarrow{\alpha} A' \qquad u \text{ does not occur in } \alpha}{\nu u.A \xrightarrow{\alpha} \nu u.A'}$$

PAR $$\frac{A \xrightarrow{\alpha} A' \quad bv(\alpha) \cap fv(B) = bn(\alpha) \cap fn(B) = \emptyset}{A \mid B \xrightarrow{\alpha} A' \mid B}$$

STRUCT $$\frac{A \equiv B \quad B \xrightarrow{\alpha} B' \quad A' \equiv B'}{A \xrightarrow{\alpha} A'}$$

Example 2.3 *Consider the process P defined in Example 2.2. We have*

$$\begin{array}{rl} P & \nu s,k.(\mathsf{in}(c_1,y).\mathsf{out}(c_2,\mathsf{dec}(y,k)) \mid \{\mathsf{enc}(s,k)/x_1\}) \\ \xrightarrow{in(c_1,x_1)} & \nu s,k.(\mathsf{out}(c_2,\mathsf{dec}(x_1,k)) \mid \{\mathsf{enc}(s,k)/x_1\}) \\ \xrightarrow{\nu x_2.out(c_1,x_2)} & \nu s,k.(\{\mathsf{enc}(s,k)/x_1\} \mid \{\mathsf{dec}(x_1,k)/x_2\}) \end{array}$$

Let B be the extended process obtained after this sequence of reduction steps. We have that $\phi(B) \equiv \nu s,k.\{^{\mathsf{enc}(s,k)}/_{x_1}, ^s/_{x_2}\}$.

Our rules differ slightly from those described in [2]. Our rules OUT-CH and OPEN-CH can be used only when c is a channel name, whereas in [2] there are identical rules OUT-ATOM and OPEN-ATOM which can be used to output a channel name or a variable of base type. To handle variables of base types, we have the rule OUT-T instead. OUT-T can easily be derived from the rules in [2], and, conversely, any application of OUT-ATOM or OPEN-ATOM involving a variable of base type can be replaced by an application of OUT-T, though the label $out(c,x)$ will be replaced by a label $\nu y.out(c,y)$ where y is a fresh variable not appearing in the process. Any transition in our semantics is also a transition in [2], but they also allow output of free variables directly. For example, notice that in [2], the process $\mathsf{out}(c,M).P \mid \{^M/_x\}$ can transition by label $\nu y.out(c,y)$ to $P \mid \{^M/_x\} \mid \{^M/_y\}$, or by label $out(c,x)$ to $P \mid \{^M/_x\}$ (since $\mathsf{out}(c,M).P \mid \{^M/_x\} \equiv \mathsf{out}(c,x).P \mid \{^M/_x\}$). Our semantics only allows the former transition. In [15], we prove that labelled bisimulation in our system coincides with labelled bisimulation in [2].

2.3 Equivalences

We can now define what it means for two frames to be statically equivalent [2].

Definition 2.4 (static equivalence ($\sim$)) *Two closed frames φ_1 and φ_2 are* statically equivalent*, written $\varphi_1 \sim \varphi_2$, if and only if for some names $\tilde{n}_1$, $\tilde{n}_2$ and substitutions σ_1, σ_2, such that $\varphi_1 \equiv \nu\tilde{n}_1\sigma_1$, $\varphi_2 \equiv \nu\tilde{n}_2\sigma_2$, and $\mathrm{dom}(\sigma_i) \cap vars(\mathrm{img}(\sigma_i)) = \emptyset$ for $i = 1, 2$, we have*

(i) $\mathrm{dom}(\varphi_1) = \mathrm{dom}(\varphi_2)$,

(ii) for all terms M, N with variables included in $\mathrm{dom}(\varphi_i)$ and using no names occurring in $\tilde{n}_1$ or $\tilde{n}_2$, $M\sigma_1 =_{\mathsf{E}} N\sigma_1$ is equivalent to $M\sigma_2 =_{\mathsf{E}} N\sigma_2$.

Extended processes A and B are *statically equivalent*, noted by $A \sim B$, if we have that $\phi(A) \sim \phi(B)$.

Example 2.5 *Let $\varphi_0 = \nu k.\sigma_0$ and $\varphi_1 = \nu k.\sigma_1$ where $\sigma_0 = \{\mathsf{enc}(s_0, k)/x_1, k/x_2\}$, $\sigma_1 = \{\mathsf{enc}(s_1, k)/x_1, k/x_2\}$ and s_0, s_1 and k are names. Let E be the theory defined by the axiom $\mathsf{dec}(\mathsf{enc}(x, k), k) = x$. We have $\mathsf{dec}(x_1, x_2)\sigma_0 =_{\mathsf{E}} s_0$ but not $\mathsf{dec}(x_1, x_2)\sigma_1 =_{\mathsf{E}} s_0$. Therefore we have $\varphi_0 \not\sim \varphi_1$. However, note that we have $\nu k.\{\mathsf{enc}(s_0, k)/x_1\} \sim \nu k.\{\mathsf{enc}(s_1, k)/x_1\}$.*

Definition 2.6 (labelled bisimilarity ($\approx$)) *Labelled bisimilarity is the largest symmetric relation $\mathcal{R}$ on closed extended processes, such that $A\ \mathcal{R}\ B$ implies*

1. *$A \sim B$,*
2. *if $A \to A'$, then $B \to^* B'$ and $A'\ \mathcal{R}\ B'$ for some B',*
3. *if $A \xrightarrow{\alpha} A'$ and $fv(\alpha) \subseteq \mathrm{dom}(A)$ and $bn(\alpha) \cap fn(B) = \emptyset$, then $B \to^* \xrightarrow{\alpha} \to^* B'$ and $A'\ \mathcal{R}\ B'$ for some B'.*

The definition of labelled bisimilarity is like the usual definition of bisimilarity, except that at each step one additionally requires that the processes are statically equivalent. In [2], it is shown that labelled bisimilarity coincides with *observational equivalence*, which is a relation capturing the fact that two given processes cannot be distinguished by any context. In general, it is easier to work with labelled bisimilarity rather than observational equivalence because of the quantification over all contexts. As mentioned in the introduction, contextually defined equivalences are useful to formalize many security properties, in particular anonymity properties, such as those studied in [12].

— PART I: Intermediate Calculus —

The idea of a symbolic semantics is to have a notion of process in which terms that have been input from the environment are represented as variables. This allows us to reason in a way that abstracts away from the particular term that was input. Given such a process P_s, we can create a concrete process (an "instance") $P_s\sigma$ by applying a substitution σ that maps the input variables to terms. We define the symbolic semantics by means of counterparts $\equiv_s$, $\rightarrow_s$, $\xrightarrow{\alpha}_s$ for the concrete relations $\equiv$, $\rightarrow$, $\xrightarrow{\alpha}$ of applied pi calculus, and aim to show soundness and completeness results relating the symbolic and concrete semantics. Structural equivalence occupies a crucial role in our calculi because the transition relations are closed under structural equivalence; therefore, we would ideally like a notion of symbolic structural equivalence which is sound and complete in the following (informal) sense:

Soundness: $P_s \equiv_s Q_s$ implies for any valid instantiation σ, $P_s\sigma \equiv Q_s\sigma$;
Completeness: $P_s\sigma \equiv Q$ implies there exists Q_s s.t. $P_s \equiv_s Q_s$ and $Q_s\sigma = Q$.

Unfortunately, completeness in this sense appears to be unachievable. To see this, consider the following example:

Example 2.7 *Consider the following process:*

$$P = \mathsf{in}(c, x).\mathsf{in}(c, y).\mathsf{out}(c, f(x)).\mathsf{out}(c, g(y)).$$

The process P can be reduced to $P' = \mathsf{out}(c, f(M_1)).\mathsf{out}(c, g(M_2))$ where M_1 and M_2 are two arbitrary terms provided by the environment. In the case that $f(M_1) =_{\mathsf{E}} g(M_2)$ we have $P' \equiv \nu z.(\mathsf{out}(c, z).\mathsf{out}(c, z) \mid \{^{f(M_1)}/_z\})$, but this structural equivalence does not hold whenever $f(M_1) \neq_{\mathsf{E}} g(M_2)$. The aim of symbolic semantics is to avoid instantiating the variables x and y; the process P would reduce symbolically to $P'_s = \mathsf{out}(c, f(x)).\mathsf{out}(c, g(y))$. In this case we need to keep auxiliary information that allows us to infer that x and y may take arbitrary values. The process P'_s represents the two cases in which x and y are equal or distinct. Hence, the question of whether the symbolic structural equivalence $P'_s \equiv_s \nu z.(\mathsf{out}(c, z).\mathsf{out}(c, z) \mid \{^{f(x)}/_z\})$ is valid cannot be decided, as it depends on the concrete values of x and y.

Therefore, the notion of symbolic structural equivalence that we will introduce is sound but not complete in the sense above (we will give a weaker completeness result). This seems to be an inherent problem and it propagates to internal and labelled reduction, since they are closed under structural equivalence. In Example 2.7, the control flow is not affected by whether $f(x) =_{\mathsf{E}} g(y)$. When control flow is affected by conditions on input variables, we maintain those conditions as a set of constraints. This allows us to give a soundness result for symbolic labelled bisimulation.

Unfortunately, the fact that we are unable to have a notion of symbolic structural equivalence which is both sound and complete in the sense mentioned above

significantly complicates the proof of our main result. We therefore split it into two parts. In this part, we define a more restricted semantics which will provide an *intermediate* representation of applied pi calculus processes (Section 3). These intermediate processes are a selected (but sufficient) subset of the original processes. One may think of them as being processes in some kind of normal form. We equip these intermediate processes with a labelled bisimulation that coincides with the original one (Section 5).

3 Syntax and Semantics

3.1 Syntax

One has *intermediate plain processes* (denoted by P, Q, R), *intermediate framed processes* (F, G, H), and *intermediate extended processes* (A, B, C).

$$
\begin{array}{lll}
P, Q, R := & \text{inter. plain process} \\
\quad 0 & \\
\quad P \mid Q & \\
\quad \text{if } M = N \text{ then } P \text{ else } Q & \\
\quad \mathsf{in}(u, x).P & \\
\quad \mathsf{out}(u, N).P &
\end{array}
\qquad
\begin{array}{lll}
F, G, H := & \text{inter. framed process} \\
\quad P & \\
\quad \{^{M}/_{x}\} & \\
\quad F \mid G & \\
A, B, C := & \text{inter. extended process} \\
\quad F & \\
\quad \nu n.A &
\end{array}
$$

Additionally, we require intermediate extended processes to be

- *name and variable distinct* (nv-distinct): $bn(A) \cap fn(A) = bv(A) \cap fv(A) = \emptyset$ and any name and variable is at most bound once; and
- *applied*, meaning that each variable in $\mathrm{dom}(A)$ occurs only once in A (the occurrence in the substitution is the only one).

Intuitively, an intermediate process is applied if all active substitutions have been applied. Intermediate extended processes are a kind of normal form for extended processes. Because they are applied and nv-distinct, we do not need restriction νx on variables x, and all νn for names n occur at the beginning of the process (which is possible as our language does not have replication).

Example 3.1 *The extended process* $\mathsf{out}(c, x) \mid \{^{M}/_{x}\}$ *is not applied, as* x *occurs twice. The corresponding intermediate process would be* $\mathsf{out}(c, M) \mid \{^{M}/_{x}\}$.

As expected, an *intermediate context* is an intermediate extended process with a hole and similarly an intermediate framed context is an intermediate framed process with a hole. An *intermediate (framed) evaluation context* is a (framed) context whose hole is not under a conditional, an input or an output. We say that such a context $C[\_]$ is a context w.r.t. an extended intermediate process A if and only if $C[A]$ is an extended intermediate process. For instance, the context $\nu n.(B \mid \_)$ would not be a context for any A such that $n \in fn(A) \cup bn(A)$ as $\nu n.(B \mid A)$ would violate nv-distinctness.

As we do not allow α-conversion we explicitly run intermediate extended processes in a *naming environment*.

Definition 3.2 (naming environment) *A* naming environment

$$\mathsf{N} : \mathcal{N} \cup \mathcal{X} \to \{\mathsf{n}, \mathsf{f}, \mathsf{b}\}$$

is a function which maps each name and variable to one of n, f, b *(standing for "new", "free" and "bound" respectively), such that there are infinitely many names and infinitely many variables that are mapped to each of* n, f *and* b. *(More precisely, the sets* $\mathsf{N}^{-1}(\mathsf{n}) \cap \mathcal{X}$, $\mathsf{N}^{-1}(\mathsf{n}) \cap \mathcal{N}$, $\mathsf{N}^{-1}(\mathsf{f}) \cap \mathcal{X}$, $\mathsf{N}^{-1}(\mathsf{f}) \cap \mathcal{N}$, $\mathsf{N}^{-1}(\mathsf{b}) \cap \mathcal{X}$, *and* $\mathsf{N}^{-1}(\mathsf{b}) \cap \mathcal{N}$ *are all infinite.)*

Intuitively, $\mathsf{N}(u) = \mathsf{f}$ if the name or variable u occurs *free* in A, and $\mathsf{N}(u) = \mathsf{b}$ if the name or variable u has been *bound* and will not be used again. $\mathsf{N}(u) = \mathsf{n}$ means that the name or variable is *new* and has not been used before, either as free or bound. This discipline helps us avoid name and variable conflicts. We use standard notation for function updating: if $\mathsf{N}(u) = t$ then the naming environment $\mathsf{N}' = \mathsf{N}[u \mapsto t']$ is defined to be the same as N except that $\mathsf{N}'(u) = t'$; and $\mathsf{N}[U \mapsto t']$ is defined as $\mathsf{N}[u_1 \mapsto t', \ldots, u_n \mapsto t']$ if $U = \{u_1, \ldots, u_n\}$. If U is a set of names and variables then $\mathsf{N}(U) = \{\mathsf{N}(u) \mid u \in U\}$ and we write $\mathsf{N}(U) = t$ if $\mathsf{N}(U) \subseteq \{t\}$.

Definition 3.3 (compatible) *We say that a naming environment* N *is* compatible with an nv-distinct process A *if* $\mathsf{N}(bn(A) \cup bv(A)) = \mathsf{b}$ *and* $\mathsf{N}(fn(A) \cup fv(A)) = \mathsf{f}$. *A naming environment* N *is* compatible with a label α *if* $\mathsf{N}(bn(\alpha) \cup bv(\alpha)) = \mathsf{n}$ *and* $\mathsf{N}(fn(\alpha) \cup fv(\alpha)) = \mathsf{f}$.

Requesting that $\mathsf{N}(bn(\alpha) \cup bv(\alpha)) = \mathsf{n}$ may seem strange with respect to the original semantics. However, the intermediate semantics, that we present in the following subsection, should clarify this point.

We define an *intermediate process* to be a pair $(A \,;\, \mathsf{N})$ where A is an intermediate extended process and N a naming environment, compatible with A. Moreover, $(A \,;\, \mathsf{N})$ is closed if A is closed. We denote by $\psi(A)$ the substitution obtained by taking the union of the active substitutions $\{^{M}/_{x}\}$ occurring in A. Note that $\phi(A)$ denotes the frame of the process including the name restrictions, while $\psi(A)$ only refers to the substitution.

Throughout the paper we always suppose that substitutions are cycle-free and use the following notational conventions for substitution. Given substitutions $\sigma_1 = \{^{M_1}/_{x_1}, \ldots, ^{M_p}/_{x_p}\}$ and $\sigma_2 = \{^{N_1}/_{y_1}, \ldots, ^{N_q}/_{y_q}\}$ we write $\sigma_1 \cup \sigma_2$ for $\{^{M_1}/_{x_1}, \ldots, ^{M_p}/_{x_p}, ^{N_1}/_{y_1}, \ldots, ^{N_q}/_{y_q}\}$ and $\sigma_1\sigma_2$ for $\{^{M_1\sigma_2}/_{x_1}, \ldots, ^{M_p\sigma_2}/_{x_p}\}$. We define $\mathrm{img}(\sigma)$ to be the image of σ, e.g., $\mathrm{img}(\sigma_1) = \{M_1, \ldots, M_p\}$. Moreover, we write $\sigma^\star$ to emphasize that we iterate the substitution until obtaining idempotence. This is needed when $\mathrm{dom}(\sigma) \cap \mathit{vars}(\mathrm{img}(\sigma)) \neq \emptyset$.

We now define the $\downarrow$ operator which transforms an nv-distinct extended processes into an intermediate extended process.

Definition 3.4 ($A\downarrow$) *Given an nv-distinct extended process A, the intermediate extended process $A\downarrow$ is defined inductively as follows:*

$$\begin{array}{ccc} 0\downarrow = 0 & \mathsf{in}(u,x).P\downarrow = \nu\tilde{n}.\mathsf{in}(u,x).P' & (\nu n.A)\downarrow = \nu n.(A\downarrow) \\ \{^M/_x\}\downarrow = \{^M/_x\} & \mathsf{out}(u,N).P\downarrow = \nu\tilde{n}.\mathsf{out}(u,N).P' & (\nu x.A)\downarrow = \tilde{A} \end{array}$$

$$\begin{array}{rcl} \mathit{if}\ M = N\ \mathit{then}\ P\ \mathit{else}\ Q\downarrow & = & \nu\tilde{n}.\nu\tilde{m}.\mathit{if}\ M = N \mathit{then}\ P'\ \mathit{else}\ Q' \\ (A \mid B)\downarrow & = & \nu\tilde{n}.\nu\tilde{m}.(A' \mid B')(\psi(A') \cup \psi(B'))^\star \end{array}$$

where $P\downarrow = \nu\tilde{n}.P'$, $Q\downarrow = \nu\tilde{m}.Q'$, $A\downarrow = \nu\tilde{n}.A'$, $B\downarrow = \nu\tilde{m}.B'$, and $\tilde{A}$ is $A\downarrow$ but with the unique occurrence of $\{^M/_x\}$ replaced by 0.

The transformation $\downarrow$ consists of:

1. applying active substitutions as much as possible and allows us to get rid of restrictions on variables. This operation preserves structural equivalence since the rules SUBST and ALIAS allows us to do this.

2. pushing the restrictions on names in front of the process. This operation does not preserve structural equivalence (see Example 3.5) but only labelled bisimilarity (Lemma 3.6).

This operation is extended as expected to context. If $C[\_]$ is an evaluation context, then $C[\_]\downarrow$ is the intermediate evaluation context obtained by applying the above rules, with the additional rule $\_\downarrow = \_$.

Example 3.5 *Let $A = \nu x.(in(c,y).\nu b_1.out(a,b_1).out(a,x) \mid \{^{f(b_2)}/_x\})$. We have that*

$$A\downarrow = \nu b_1.(in(c,y).out(a,b_1).out(a,f(b_2)) \mid 0$$

Lemma 3.6 *Let A be an nv-distinct extended process. We have that $A\downarrow \approx A$.*

3.2 Semantics

Structural equivalence $\equiv_i$ is the smallest equivalence relation on intermediate processes such that:

$$\begin{array}{lrcl} \textsc{Par}\text{-}0_i & (A\ ;\mathsf{N}) & \equiv_i & (A \mid 0\ ;\mathsf{N}) \\ \textsc{Par}\text{-}\mathrm{A}_i & (A \mid (B \mid C)\ ;\mathsf{N}) & \equiv_i & ((A \mid B) \mid C\ ;\mathsf{N}) \\ \textsc{Par}\text{-}\mathrm{C}_i & (A \mid B\ ;\mathsf{N}) & \equiv_i & (B \mid A\ ;\mathsf{N}) \\ \textsc{New}\text{-}\mathrm{C}_i & (\nu n.\nu m.A\ ;\mathsf{N}) & \equiv_i & (\nu m.\nu n.A\ ;\mathsf{N}) \end{array}$$

and that is closed by application of intermediate evaluation context, i.e.

$$\frac{(A\ ;\mathsf{N}) \equiv_i (B\ ;\mathsf{N})}{(C[A]\ ;\mathsf{N}[bn(C[0]) \mapsto \mathsf{b}]) \equiv_i (C[B]\ ;\mathsf{N}[bn(C[0]) \mapsto \mathsf{b}])}$$

Note that, in this intermediate semantics, we have removed several structural equivalence rules such as SUBST, REWRITE and we do not allow α-renaming. In particular, Example 2.7 is not problematic anymore.

Internal reduction $\rightarrow_i$ is the smallest relation on intermediate processes closed by structural equivalence ($\equiv_i$), by application of intermediate evaluation contexts and such that:

$$\begin{array}{lrcll}
\textsc{Comm}_i & (\mathsf{out}(a,M).P \mid \mathsf{in}(a,x).Q \;;\; \mathsf{N}) & \rightarrow_i & (P \mid Q\{^M/_x\} \;;\; \mathsf{N}) & \\
\textsc{Then}_i & (\text{if } M = N \text{ then } P \text{ else } Q \;;\; \mathsf{N}) & \rightarrow_i & (P \;;\; \mathsf{N}) & \text{where } M =_{\mathsf{E}} N \\
\textsc{Else}_i & (\text{if } M = N \text{ then } P \text{ else } Q \;;\; \mathsf{N}) & \rightarrow_i & (Q \;;\; \mathsf{N}) & \\
 & & & \text{for any ground terms } M \text{ and } N \text{ such that } M \neq_{\mathsf{E}} N &
\end{array}$$

Labelled transition $\xrightarrow{\alpha}_i$. We also extend our intermediate semantics with the following labelled transition relation.

$$\textsc{In}_i \qquad (\mathsf{in}(a,x).P \;;\; \mathsf{N}) \xrightarrow{in(a,M)}_i (P\{^M/_x\} \;;\; \mathsf{N}) \quad \text{where } \mathsf{N}(fn(M) \cup fv(M)) = \mathsf{f}$$

$$\textsc{Out-Ch}_i \qquad (\mathsf{out}(a,c).P \;;\; \mathsf{N}) \xrightarrow{out(a,c)}_i (P \;;\; \mathsf{N})$$

$$\textsc{Out-T}_i \qquad (\mathsf{out}(a,M).P \;;\; \mathsf{N}) \xrightarrow{\nu x.out(a,x)}_i (P \mid \{^M/_x\}, \mathsf{N}[x \mapsto \mathsf{f}]) \quad \text{where } x \in \mathcal{X}_b \text{ and } \mathsf{N}(x) = \mathsf{n}$$

$$\textsc{Open-Ch}_i \qquad \frac{(A \;;\; \mathsf{N}) \xrightarrow{out(a,c)}_i (A' \;;\; \mathsf{N}') \qquad c \neq a,\ d \in \mathcal{N}_{ch},\ \mathsf{N}(d) = \mathsf{n}}{(\nu c.A, \mathsf{N}[c \mapsto \mathsf{b}]) \xrightarrow{\nu d.out(a,d)}_i (A'\{^d/_c\}, \mathsf{N}'[c \mapsto \mathsf{b}, d \mapsto \mathsf{f}])}$$

$$\textsc{Scope}_i \qquad \frac{(A \;;\; \mathsf{N}) \xrightarrow{\alpha}_i (A', \mathsf{N}') \qquad n \text{ does not occur in } \alpha}{(\nu n.A \;;\; \mathsf{N}[n \mapsto \mathsf{b}]) \xrightarrow{\alpha}_i (\nu n.A', \mathsf{N}'[n \mapsto \mathsf{b}])}$$

$$\textsc{Par}_i \qquad \frac{(A \;;\; \mathsf{N}) \xrightarrow{\alpha\psi(B)}_i (A', \mathsf{N}')}{(A \mid B \;;\; \mathsf{N}) \xrightarrow{\alpha}_i (A' \mid B, \mathsf{N}')}$$

$$\textsc{Struct}_i \qquad \frac{(A \;;\; \mathsf{N}) \equiv_i (B \;;\; \mathsf{N}) \xrightarrow{\alpha}_i (B', \mathsf{N}') \equiv_i (A', \mathsf{N}')}{(A \;;\; \mathsf{N}) \xrightarrow{\alpha}_i (A', \mathsf{N}')}$$

One may note two particularities in this semantics. The $\textsc{Open-Ch}_i$ rule requires an "on-the-fly renaming" at the point that we reveal a bound name. This will be needed in the bisimulation because we require both the left- and right-hand processes to use the same label without allowing α-conversion. The second unusual detail is the $\alpha\psi(B)$ label in the $\textsc{Par}_i$ rule which is needed to keep processes applied. Note that $\psi(B)$ can only affect labels that are of the form $\mathsf{in}(c, M)$ since for the other ones, the variables in $\mathrm{dom}(\psi(B))$ do not occur in the label α.

Example 3.7 *Let* $A = \nu a.(\mathsf{in}(c,x).P(x) \mid \{^a/_y\})$ *and* $B = \nu a.(P(a) \mid \{^a/_y\})$. *We have that* $A \xrightarrow{in(c,y)} B$. *Let* N *be a naming environment compatible with* $A\downarrow$ *and* $in(c,y)$. *We have also that* $(A\downarrow\ ;\mathsf{N}) \xrightarrow{in(c,y)} (B\downarrow\ ;\mathsf{N})$. *The derivation witnessing this reduction uses the fact that the label in the* $\textsc{Par}_i$ *rule can be instantiated along the derivation.*

$$\dfrac{\dfrac{(\mathsf{in}(c,x).P(x)\ ;\mathsf{N}) \xrightarrow{in(c,a)} (P(a)\ ;\mathsf{N})}{(\mathsf{in}(c,x).P(x) \mid \{^a/_y\}\ ;\mathsf{N}) \xrightarrow{in(c,y)} (P(a) \mid \{^a/_y\}\ ;\mathsf{N})}}{(\nu a.(\mathsf{in}(c,x).P(x) \mid \{^a/_y\})\ ;\mathsf{N}[a \mapsto \mathsf{b}]) \xrightarrow{in(c,y)} (\nu a.(P(a) \mid \{^a/_y\})\ ;\mathsf{N}[a \mapsto \mathsf{b}])}$$

In particular we note that if the first label had been $\mathsf{in}(c,y)$ *the obtained process would have been* $(P(y)\ ;\mathsf{N})$ *and the* $\textsc{Par}_i$ *rule could not have been used because* $P(y) \mid \{^a/_y\}$ *would not have been an intermediate process (as it is not applied).*

4 Soundness and Completeness

We now introduce the relation $\approxeq$ on intermediate processes. Intuitively, $\approxeq$ captures the structural equivalences that are "missing" in $\equiv_i$ with respect to $\equiv$. We show completeness of the intermediate semantics up to $\approxeq$. Note that the rule $\textsc{New-C}_i$ is in the relation $\approxeq$ because of some tricky interactions between the transformation $\downarrow$ and the relation $\equiv$ (see Example 4.2 given below).

Definition 4.1 ($\approxeq$) *We define* $\approxeq$ *to be the smallest equivalence relation on intermediate processes closed under bijective renaming of bound names and variables and such that*

$$\begin{array}{lrcll}
\textsc{New-N}_i & (\nu\tilde{n}.\nu m.A\ ;\mathsf{N}) & \approxeq & (\nu\tilde{n}.A\ ;\mathsf{N}) & \text{if } m \notin fn(A) \\
\textsc{Rew-N}_i & (A\{^M/_x\}\ ;\mathsf{N}) & \approxeq & (A\{^N/_x\}\ ;\mathsf{N}) & \text{if } M =_{\mathsf{E}} N \\
\textsc{New-C}_i & (\nu n.\nu m.A\ ;\mathsf{N}) & \approxeq & (\nu m.\nu n.A\ ;\mathsf{N}) &
\end{array}$$

Example 4.2 *Consider the processes:* $A = \mathsf{out}(c,n_1).\nu n_2.\nu n_2'.\mathsf{out}(c,\langle n_2,n_2'\rangle)$ *and* $C = \nu n_2'.\nu n_2.\mathsf{out}(c,n_1).\mathsf{out}(c,\langle n_2,n_2'\rangle)$. *We have that*

$$A\downarrow = \nu n_2.\nu n_2'.(\mathsf{out}(c,n_1).\mathsf{out}(c,\langle n_2,n_2'\rangle)) \quad \text{and} \quad A\downarrow \equiv C.$$

Note that there is no B *such that* $A \equiv B$ *and* $B\downarrow = C$. *Nevertheless, because of the rule* $\textsc{New-C}_i$ *there exists* B' *(e.g.* $B' = A$*) such that* $A \equiv B'$ *and* $(B'\downarrow\ ;\mathsf{N}) \approxeq (C\ ;\mathsf{N})$ *for any compatible naming environment* N. *This property is needed in the proof of Proposition 5.5.*

Lemma 4.3 *Let* $(A\ ;\mathsf{N})$ *and* $(A'\ ;\mathsf{N})$ *be two intermediate processes such that* $(A\ ;\mathsf{N}) \approxeq (A'\ ;\mathsf{N})$. *Then we have that* $A \equiv A'$.

We now show that our intermediate semantics is *sound* with respect to the original semantics: any structural equivalence and (labelled) reduction that holds in the intermediate context also holds in the original semantics.

Proposition 4.4 (soundness) *Let* $(A_i \; ; \mathsf{N})$ *and* $(B_i \; ; \mathsf{N}')$ *be two intermediate processes such that* $(A_i \; ; \mathsf{N}) \bowtie_i (B_i \; ; \mathsf{N}')$ *with* $\bowtie \in \{\equiv, \rightarrow, \xrightarrow{\alpha}\}$. *Then we have that* $A_i \bowtie B_i$.

The proofs when $\bowtie \in \{\equiv, \rightarrow\}$ are straightforward and are sketched in Appendix A. The proof for $\xrightarrow{\alpha}_i$ is more involved and detailed in Appendix A.

We also introduce a useful *commutation lemma*. As we show completeness for one step of each of these relations up to $\approxeq$ the commutation lemmas will allow us to lift the result to sequences of steps.

Lemma 4.5 (commutation) *Let* $(A \; ; \mathsf{N})$, $(A' \; ; \mathsf{N})$ *and* $(B \; ; \mathsf{N}')$ *be three intermediate processes such that* $(A' \; ; \mathsf{N}) \approxeq (A \; ; \mathsf{N}) \bowtie (B \; ; \mathsf{N}')$ *with* $\bowtie \in \{\equiv_i, \rightarrow_i, \xrightarrow{\alpha}_i\}$. *Then there exists* $(B' \; ; \mathsf{N}')$ *such that* $(A' \; ; \mathsf{N}) \bowtie (B' \; ; \mathsf{N}') \approxeq (B \; ; \mathsf{N}')$.

Proof.(sketch) This result can be proved by considering proofs in "linear form", i.e., by applying the rules directly under the evaluation context, resulting into a sequence of processes rather than a proof tree. Similarly, the proof of $(A' \; ; \mathsf{N}) \approxeq (A \; ; \mathsf{N})$ can be written as a sequence of steps. Now, it is easy to show, by case analysis on each pair of rules, that each time there is an application of $\approxeq$ occurring immediately to the left of an application of $\equiv_i$, this pair of rule applications can be commuted. □

Next we show completeness of the intermediate semantics: any structural equivalence and (labelled) reduction that holds in the original semantics should also hold for corresponding intermediate processes in the new semantics. As discussed previously completeness seems difficult to achieve. We therefore show completeness up to $\approxeq$.

Proposition 4.6 (completeness) *Let* A *and* B *be two nv-distinct extended processes such that* $A \bowtie B$ *with* $\bowtie \in \{\equiv, \rightarrow, \xrightarrow{\alpha}\}$. *Let* N *be a naming environment compatible with* $A{\downarrow}$ *and also with* α *when* $\bowtie = \xrightarrow{\alpha}$. *Let* N' *be a naming environment compatible with* $B{\downarrow}$ *and such that:*

- $\mathsf{N}' = \mathsf{N}[x \mapsto \mathsf{f}]$ *when* $\bowtie = \xrightarrow{\alpha}$ *and* α *is of the form* $\nu x.out(a, x)$;
- $\mathsf{N}' = \mathsf{N}[d \mapsto \mathsf{f}]$ *when* $\bowtie = \xrightarrow{\alpha}$ *and* α *is of the form* $\nu d.out(a, d)$;
- $\mathsf{N}' = \mathsf{N}$ *otherwise.*

Then there exists an intermediate process $(D_i \; ; \mathsf{N}')$ *such that*

- $(A{\downarrow} \; ; \mathsf{N}) \bowtie_i (D_i \; ; \mathsf{N}')$, *and*
- $(D_i \; ; \mathsf{N}') \approxeq (B{\downarrow} \; ; \mathsf{N}')$.

Note that, when $\bowtie \in \{\equiv, \rightarrow\}$, we have that $\mathsf{N}' = \mathsf{N}$. The proofs are done in Appendix A.

From Propositions 4.4 and 4.6 and the commutation lemma stated above, we derive the following corollaries. (Corollary 4.8 also requires Lemmas A.5 and A.8 in the appendix.)

Corollary 4.7 (soundness of $\rightarrow_i^*$ and $\rightarrow_i^* \xrightarrow{\alpha}_i \rightarrow_i^*$) *Let $(A_i \,;\, \mathsf{N})$ and $(B_i \,;\, \mathsf{N}')$ be two intermediate processes such that $(A_i \,;\, \mathsf{N}) \rightarrow_i^* (B_i \,;\, \mathsf{N}')$ (resp. $(A_i \,;\, \mathsf{N}) \rightarrow_i^* \xrightarrow{\alpha}_i \rightarrow_i^* (B_i \,;\, \mathsf{N}')$). Then we have that $A_i \rightarrow^* B_i$ (resp. $A_i \rightarrow^* \xrightarrow{\alpha} \rightarrow^* B_i$).*

Corollary 4.8 (completeness of $\rightarrow_i^*$ and $\rightarrow_i^* \xrightarrow{\alpha}_i \rightarrow_i^*$) *Let A and B be two nv-distinct extended processes such that $A \rightarrow^* B$ (resp. $A \rightarrow^* \xrightarrow{\alpha} \rightarrow^* B$) and N be a naming environment compatible with $A\downarrow$ (resp., and α). Let N' be a naming environment compatible with $B\downarrow$ and such that:*

- $\mathsf{N}' = \mathsf{N}[x \mapsto \mathsf{f}]$ *in the case $\rightarrow_i^* \xrightarrow{\alpha}_i \rightarrow_i^*$ when α is of the form $\nu x.out(a, x)$;*
- $\mathsf{N}' = \mathsf{N}[d \mapsto \mathsf{f}]$ *in the case $\rightarrow_i^* \xrightarrow{\alpha}_i \rightarrow_i^*$ when α is of the form $\nu d.out(a, d)$;*
- $\mathsf{N}' = \mathsf{N}$ *otherwise.*

Then there exists an intermediate process $(D_i \,;\, \mathsf{N}')$ such that $(A\downarrow \,;\, \mathsf{N}) \rightarrow_i^* (D_i \,;\, \mathsf{N}')$ (resp. $(A\downarrow \,;\, \mathsf{N}) \rightarrow_i^* \xrightarrow{\alpha}_i \rightarrow_i^* (D_i \,;\, \mathsf{N}')$) and $(D_i \,;\, \mathsf{N}') \cong (B\downarrow \,;\, \mathsf{N}')$.

5 Intermediate Bisimulation

We now define the intermediate labelled bisimulation. The definition is similar to the original one, but is stated with respect to our intermediate semantics. Moreover, note that the side condition $bn(\alpha) \cap fn(B) = \emptyset$ has been removed since the fact that both processes are running in the same naming environment ensures this condition.

Definition 5.1 (Intermediate labelled bisimilarity ($\approx_i$)) Intermediate labelled bisimilarity *is the largest symmetric relation $\mathcal{R}$ on closed intermediate processes with same naming environment, such that $(A_i, \mathsf{N})\ \mathcal{R}\ (B_i \,;\, \mathsf{N})$ implies*

1. *$A_i \sim B_i$,*
2. *if $(A_i \,;\, \mathsf{N}) \rightarrow_i (A_i' \,;\, \mathsf{N})$, then $(B_i \,;\, \mathsf{N}) \rightarrow_i^* (B_i' \,;\, \mathsf{N})$ and $(A_i' \,;\, \mathsf{N})\ \mathcal{R}\ (B_i' \,;\, \mathsf{N})$ for some B_i',*
3. *if $(A_i \,;\, \mathsf{N}) \xrightarrow{\alpha}_i (A_i' \,;\, \mathsf{N}')$ and $fv(\alpha) \subseteq \mathrm{dom}(A_i)$, then $(B_i \,;\, \mathsf{N}) \rightarrow_i^* \xrightarrow{\alpha}_i \rightarrow_i^* (B_i' \,;\, \mathsf{N}')$ and $(A_i' \,;\, \mathsf{N}')\ \mathcal{R}\ (B_i' \,;\, \mathsf{N}')$ for some B_i'.*

The following theorem states that the intermediate and the original bisimulations coincide.

Theorem 5.2 *Let A and B be two nv-distinct extended processes and N be a naming environment compatible with $A\downarrow$ and $B\downarrow$. We have that*

$$A \approx B \text{ if and only if } (A\downarrow \,;\, \mathsf{N}) \approx_i (B\downarrow \,;\, \mathsf{N})$$

Both directions of this result are proved separately by the following propositions.

Proposition 5.3 *Let A and B be two nv-distinct extended processes. We have*

$$A \approx B \text{ implies } (A\downarrow ; \mathsf{N}) \approx_i (B\downarrow ; \mathsf{N}).$$

for any naming environment N compatible with $A\downarrow$ and $B\downarrow$.

Proof. To prove this result, first we define a new relation $\mathcal{R}$. Next we will show that $\mathcal{R}$ is an intermediate labelled bisimulation witnessing $(A\downarrow ; \mathsf{N}) \approx_i (B\downarrow ; \mathsf{N})$.

(i) Definition of $\mathcal{R}$.
We define $\mathcal{R}$ as follows: $(A_i ; \mathsf{N})\ \mathcal{R}\ (B_i ; \mathsf{N})$ if $A_i \approx B_i$ and N is a naming environment compatible with A_i and B_i.

(ii) $\mathcal{R}$ is an intermediate bisimulation relation witnessing $(A\downarrow ; \mathsf{N}) \approx_i (B\downarrow ; \mathsf{N})$.
First we have to show that $(A\downarrow ; \mathsf{N})\ \mathcal{R}\ (B\downarrow ; \mathsf{N})$. We have that $A \approx B$ and hence $A\downarrow \approx B\downarrow$ (by Lemma 3.6) and N is a naming environment compatible with $A\downarrow$ and $B\downarrow$. Hence, by definition of $\mathcal{R}$, we easily conclude.

Now, we have to show that $\mathcal{R}$ satisfies the three points of the definition of intermediate labelled bisimilarity. Let $(A_i ; \mathsf{N})$ and $(B_i ; \mathsf{N})$ be two closed intermediate processes such that $(A_i ; \mathsf{N})\ \mathcal{R}\ (B_i ; \mathsf{N})$. By definition of $\mathcal{R}$, we have that $A_i \approx B_i$.
We have to show that:

1. $A_i \sim B_i$. By hypothesis, we have that $A_i \approx B_i$ which implies $A_i \sim B_i$.

2. If $(A_i ; \mathsf{N}) \rightarrow_i (A'_i ; \mathsf{N})$ then there exists $(B'_i ; \mathsf{N})$ with $(B_i ; \mathsf{N}) \rightarrow_i^* (B'_i ; \mathsf{N})$ and $(A'_i ; \mathsf{N})\ \mathcal{R}\ (B'_i ; \mathsf{N})$.
   By Proposition 4.4, we have that $A_i \rightarrow A'_i$. Since $A_i \approx B_i$ and $A_i \rightarrow A'_i$ there exists an extended process B' such that $B_i \rightarrow^* B'$ and $A'_i \approx B'$. In the remainder, we assume w.l.o.g. that B' is nv-distinct and compatible with N. By Corollary 4.8, there exists an intermediate extended process $(D ; \mathsf{N})$ such that

   - $(B_i\downarrow ; \mathsf{N}) \rightarrow_i^* (D ; \mathsf{N})$, and
   - $(D ; \mathsf{N}) \cong (B'\downarrow ; \mathsf{N})$.

   As B_i is an intermediate process we have that $B_i\downarrow = B_i$. Let $B'_i = D$. It remains to show that $(A'_i ; \mathsf{N})\ \mathcal{R}\ (B'_i ; \mathsf{N})$. As $A'_i \approx B'$, $B' \approx B'\downarrow$ (by Lemma 3.6), and $B'_i \equiv B'\downarrow$ (by Lemma 4.3 and the fact that $(B'_i ; \mathsf{N}) = (D ; \mathsf{N}) \cong (B'\downarrow ; \mathsf{N})$) we have that $A'_i \approx B'_i$. By definition of $\mathcal{R}$ we deduce that $(A'_i ; \mathsf{N})\ \mathcal{R}\ (B'_i ; \mathsf{N})$.

3. If $(A_i ; \mathsf{N}) \xrightarrow{\alpha}_i (A'_i ; \mathsf{N}')$ with $fv(\alpha) \subseteq \mathrm{dom}(A_i)$ then there exists $(B'_i ; \mathsf{N}')$ such that $(B_i ; \mathsf{N}) \rightarrow_i^* \xrightarrow{\alpha}_i \rightarrow_i^* (B'_i ; \mathsf{N}')$ and $(A' ; \mathsf{N}')\ \mathcal{R}'\ (B'_i ; \mathsf{N}')$.
   This case is similar to the previous one.

□

To show the converse direction, we need an additional lemma.

Lemma 5.4 *Let* $(A\ ;\ \mathsf{N})$ *and* $(B\ ;\ \mathsf{N})$ *be two intermediate processes, and* N' *a naming environment compatible with* A *and* B*. Then:*

1. $(A\ ;\ \mathsf{N}) \approx_i (B\ ;\ \mathsf{N})$ *implies* $(A\ ;\ \mathsf{N}') \approx_i (B\ ;\ \mathsf{N}')$*; and*
2. $(A\ ;\ \mathsf{N}) \cong (B\ ;\ \mathsf{N})$ *implies* $(A\ ;\ \mathsf{N}') \cong (B\ ;\ \mathsf{N}')$.

To see that the first part of the lemma holds it is sufficient to note that $(A\ ;\ \mathsf{N}) \bowtie_i (A'\ ;\ \mathsf{N}')$ if and only if $(A\rho\ ;\ \mathsf{N}\rho) \bowtie_i (A'\rho\ ;\ \mathsf{N}'\rho)$ where ρ is a bijective renaming of names and variables, $\bowtie \in \{\equiv, \rightarrow, \xrightarrow{\alpha}\}$ and $\mathsf{N}\rho(u) = \mathsf{N}(\rho^{-1}(u)))$. The second part is immediate.

Proposition 5.5 *Let* $(A_i\ ;\ \mathsf{N})$ *and* $(B_i\ ;\ \mathsf{N})$ *be two intermediate processes. We have that*

$$(A_i\ ;\ \mathsf{N}) \approx_i (B_i\ ;\ \mathsf{N}) \text{ implies } A_i \approx B_i$$

Proof. To prove this result, we first define a new relation $\mathcal{R}$ and then we will show that $\mathcal{R}$ witnesses $\approx$.

(i) Definition of $\mathcal{R}$*.*
$A\ \mathcal{R}\ B$ if there exist two intermediate processes $(\hat{A}\ ;\ \mathsf{N})$ and $(\hat{B}\ ;\ \mathsf{N})$, and two nv-distinct extended processes A_α and B_α such that

- $(\hat{A}\ ;\ \mathsf{N}) \approx_i (\hat{B}\ ;\ \mathsf{N})$,
- $(\hat{A}\ ;\ \mathsf{N}) \cong (A_\alpha{\downarrow}\ ;\ \mathsf{N})$ and $(\hat{B}\ ;\ \mathsf{N}) \cong (B_\alpha{\downarrow}\ ;\ \mathsf{N})$,
- $A_\alpha =_\alpha A$ and $B_\alpha =_\alpha B$.

(ii) $\mathcal{R}$ *witnesses* $\approx$*.*
First we show that $A_i\ \mathcal{R}\ B_i$. Let $\hat{A} = A_\alpha = A_i$ and $\hat{B} = B_\alpha = B_i$. By definition of $\mathcal{R}$ we easily conclude as $(A_i\ ;\ \mathsf{N}) \approx_i (B_i\ ;\ \mathsf{N})$.

Now, we show that $\mathcal{R}$ satisfies the three points of the definition of $\approx$. Let A and B be two closed intermediate extended processes such that $A\ \mathcal{R}\ B$. By definition of $\mathcal{R}$, we know that there exist two intermediate processes $(\hat{A}\ ;\ \mathsf{N})$ and $(\hat{B}\ ;\ \mathsf{N})$ such that

- $(\hat{A}\ ;\ \mathsf{N}) \approx_i (\hat{B}\ ;\ \mathsf{N})$,
- $(\hat{A}\ ;\ \mathsf{N}) \cong (A_\alpha{\downarrow}\ ;\ \mathsf{N})$ and $(\hat{B}\ ;\ \mathsf{N}) \cong (B_\alpha{\downarrow}\ ;\ \mathsf{N})$, and
- $A_\alpha =_\alpha A$ and $B_\alpha =_\alpha B$.

We have to show that:

1. $A \sim B$. Since $(\hat{A}\ ;\ \mathsf{N}) \approx_i (\hat{B}\ ;\ \mathsf{N})$, we have that $\hat{A} \sim \hat{B}$ by definition of $\approx_i$. As $A_\alpha =_\alpha A$ and $B_\alpha =_\alpha B$, we have that $\hat{A} \sim A$ and $\hat{B} \sim B$. We conclude by transitivity of $\sim$.

2. If $A \to A'$ for some extended process A' then there exists B' such that $B \to^* B'$ and $A' \mathcal{R} B'$.
   If $A \to A'$ we also have that $A_\alpha \to A'_\alpha$ for some A'_α such that $A'_\alpha =_\alpha A'$ and A'_α compatible with N. By Proposition 4.6, there exists an extended process $(D \,;\, \mathsf{N})$ such that

   - $(A_\alpha{\downarrow} \,;\, \mathsf{N}) \to_i (D \,;\, \mathsf{N})$, and
   - $(D \,;\, \mathsf{N}) \cong (A'_\alpha{\downarrow} \,;\, \mathsf{N})$.

   We have that $(\hat{A} \,;\, \mathsf{N}) \cong (A_\alpha{\downarrow} \,;\, \mathsf{N}) \to_i (D \,;\, \mathsf{N}) \cong (A'_\alpha{\downarrow} \,;\, \mathsf{N})$. By Lemma 4.5, there exists (D', N) such that

   $$(\hat{A} \,;\, \mathsf{N}) \to_i (D' \,;\, \mathsf{N}) \cong (D \,;\, \mathsf{N}) \cong (A'_\alpha{\downarrow} \,;\, \mathsf{N}).$$

   Since $(\hat{A} \,;\, \mathsf{N}) \approx_i (\hat{B} \,;\, \mathsf{N})$ and $(\hat{A} \,;\, \mathsf{N}) \to_i (D' \,;\, \mathsf{N})$, we have that there exists $(B'_i \,;\, \mathsf{N})$ such that $(\hat{B} \,;\, \mathsf{N}) \to_i^* (B'_i \,;\, \mathsf{N})$ and $(D' \,;\, \mathsf{N}) \approx_i (B'_i \,;\, \mathsf{N})$. By Corollary 4.7, we have that $\hat{B} \to^* B'_i$. Thus, we deduce that $B \to^* B''_i$, for some B''_i such that $(B''_i{\downarrow} \,;\, \mathsf{N}) \cong (B'_i \,;\, \mathsf{N})$. Let $B' = B''_i$, $B'_\alpha = B'$, $\hat{B}' = B'_i$ and $\hat{A}' = D'$. We have that $B \to^* B'$ and by definition of $\mathcal{R}$, we have that $A' \mathcal{R} B'$. Indeed, we have that:

   - $(\hat{A}' \,;\, \mathsf{N}) \approx_i (\hat{B}' \,;\, \mathsf{N})$, i.e. $(D' \,;\, \mathsf{N}) \approx_i (B'_i \,;\, \mathsf{N})$,
   - $(\hat{A}' \,;\, \mathsf{N}) \cong (A'_\alpha{\downarrow} \,;\, \mathsf{N})$ and $(\hat{B}' \,;\, \mathsf{N}) \cong (B'_\alpha{\downarrow} \,;\, \mathsf{N})$, and
   - $A'_\alpha =_\alpha A'$ and $B'_\alpha =_\alpha B'$.

3. If $A \xrightarrow{\alpha} A'$ and $fv(\alpha) \subseteq \mathrm{dom}(A)$ and $bn(\alpha) \cap fn(B) = \emptyset$ for some extended process A' then $B \to^* \xrightarrow{\alpha} \to^* B'$ and $A' \mathcal{R} B'$ for some process B'.

   First, we can assume w.l.o.g. that $(fn(\alpha) \cup bn(\alpha)) \cap (bn(A_\alpha) \cup bn(B_\alpha)) = \emptyset$ and $(fn(\alpha) \cup bn(\alpha)) \cap (bn(\hat{A}) \cup bn(\hat{B})) = \emptyset$ (since A_α, B_α $\hat{A}$, $\hat{B}$ can be chosen to make this true). For the same reason, we can assume that $bv(\alpha) \cap (bv(A_\alpha) \cup bv(B_\alpha)) = \emptyset$ and $bv(\alpha) \cap (bv(\hat{A}) \cup bv(\hat{B})) = \emptyset$. Let $\hat{\mathsf{N}} = \mathsf{N}[fn(\alpha) \mapsto \mathsf{f}][bn(\alpha), bv(\alpha) \mapsto \mathsf{n}]$. Note that $\hat{\mathsf{N}}$ is now compatible with A_α, B_α $\hat{A}$, $\hat{B}$ and α. Let A'_α be an nv-distinct extended process such that $A_\alpha \xrightarrow{\alpha} A'_\alpha$ and A'_α compatible with $\hat{\mathsf{N}}'$ where $\hat{\mathsf{N}}'$ is defined as follows:

   $$\hat{\mathsf{N}}' = \begin{cases} \hat{\mathsf{N}}[x \mapsto \mathsf{f}] & \text{when } \alpha \text{ is of the form } \nu x.out(a, x); \\ \hat{\mathsf{N}}[d \mapsto \mathsf{f}] & \text{when } \alpha \text{ is of the form } \nu d.out(a, d); \\ \hat{\mathsf{N}} & \text{otherwise.} \end{cases}$$

   By Proposition 4.6, there exists an extended process $(D \,;\, \hat{\mathsf{N}}')$ such that

   - $(A_\alpha{\downarrow} \,;\, \hat{\mathsf{N}}) \xrightarrow{\alpha}_i (D \,;\, \hat{\mathsf{N}}')$, and
   - $(D \,;\, \hat{\mathsf{N}}') \cong (A'_\alpha{\downarrow} \,;\, \hat{\mathsf{N}}')$.

We have that $(\hat{A}\ ;\ \mathsf{N}) \cong (A_\alpha\downarrow\ ;\ \mathsf{N})$ and thus $(\hat{A}\ ;\ \hat{\mathsf{N}}) \cong (A_\alpha\downarrow\ ;\ \hat{\mathsf{N}})$ (Lemma 5.4). Moreover, we have that $(A_\alpha\downarrow\ ;\ \hat{\mathsf{N}}) \xrightarrow{\alpha}_i (D\ ;\ \hat{\mathsf{N}}') \cong (A'_\alpha\downarrow\ ;\ \hat{\mathsf{N}}')$. By Lemma 4.5, there exists $(D', \hat{\mathsf{N}}')$ such that

$$(\hat{A}\ ;\ \hat{\mathsf{N}}) \xrightarrow{\alpha}_i (D'\ ;\ \hat{\mathsf{N}}') \cong (D\ ;\ \hat{\mathsf{N}}') \cong (A'_\alpha\downarrow\ ;\ \hat{\mathsf{N}}').$$

Since $(\hat{A}\ ;\ \mathsf{N}) \approx_i (\hat{B}\ ;\ \mathsf{N})$, we have also that $(\hat{A}\ ;\ \hat{\mathsf{N}}) \approx_i (\hat{B}\ ;\ \hat{\mathsf{N}})$ (Lemma 5.4). Moreover, we have that $(\hat{A}\ ;\ \hat{\mathsf{N}}) \xrightarrow{\alpha}_i (D'\ ;\ \hat{\mathsf{N}}')$, thus there exists $(B'_i\ ;\ \hat{\mathsf{N}}')$ such that $(\hat{B}\ ;\ \hat{\mathsf{N}}) \to_i^* \xrightarrow{\alpha}_i \to_i^* (B'_i\ ;\ \hat{\mathsf{N}}')$ and $(D'\ ;\ \hat{\mathsf{N}}') \approx_i (B'_i\ ;\ \hat{\mathsf{N}}')$. By Corollary 4.7, we have that $\hat{B} \to^* \xrightarrow{\alpha} \to^* B'_i$. Thus, we deduce that $B \to^* \xrightarrow{\alpha} \to^* B''_i$ for some B''_i such that $(B''_i\downarrow\ ;\ \hat{\mathsf{N}}') \cong (B'_i\ ;\ \hat{\mathsf{N}}')$. Let $B' = B''_i$, $B'_\alpha = B'$, $\hat{B}' = B'_i$ and $\hat{A}' = D'$. We have that $B \to^* \xrightarrow{\alpha} \to^* B'$ and by definition of $\mathcal{R}$, we have that $A'\ \mathcal{R}\ B'$. Indeed, we have that:

- $(\hat{A}'\ ;\ \hat{\mathsf{N}}') \approx_i (\hat{B}'\ ;\ \hat{\mathsf{N}}')$, i.e. $(D'\ ;\ \hat{\mathsf{N}}') \approx_i (B'_i\ ;\ \hat{\mathsf{N}}')$,
- $(\hat{A}'\ ;\ \hat{\mathsf{N}}') \cong (A'_\alpha\downarrow\ ;\ \hat{\mathsf{N}}')$ and $(\hat{B}'\ ;\ \hat{\mathsf{N}}') \cong (B'_\alpha\downarrow\ ;\ \hat{\mathsf{N}}')$, and
- $A'_\alpha =_\alpha A'$ and $B'_\alpha =_\alpha B'$.

□

— PART II: Symbolic Calculus —

A *symbolic process* is an intermediate process together with a *constraint system*. The aim of a symbolic semantics is to avoid the infinite branching due to the inputs of the environment. This is achieved by keeping variables rather than the input terms. The constraint system gives a finite representation of the value that these variables are allowed to take.

6 Constraint systems

Definition 6.1 (constraint system) *A* constraint system $\mathcal{C}$ *is a set of constraints where every constraint is of one of the following forms:*

- *"*$\varphi \Vdash x$*", where* $\varphi = \nu\tilde{u}.\sigma$ *for some tuple of names and variables* $\tilde{u}$ *and some substitution* σ*, and* x *is a variable which does not appear under a restriction of any frame nor in the domain of any frame;*
- *"*$M = N$*", where* M *and* N *are terms;*
- *"*$M \neq N$*", where* M *and* N *are terms;*
- *"*$\mathsf{gd}(M)$*" where* M *is a term;*

The constraint $\varphi \Vdash x$ is useful for specifying the information φ held by the environment when it supplies an input x. As we will see in the following section, these variables will be taken from a special set of variables. The constraint $\mathsf{gd}(M)$ means that the term M is ground. We denote by $\mathcal{D}ed(\mathcal{C})$ the *deducibility constraints* of $\mathcal{C}$, i.e. $\{\varphi \Vdash x \mid \text{"}\varphi \Vdash x\text{"} \in \mathcal{C}\}$. When $\mathcal{D}ed(\mathcal{C}) = \{\varphi_1 \Vdash x_1, \ldots, \varphi_\ell \Vdash x_\ell\}$, we define $cv(\mathcal{C}) = \{x_1, \ldots, x_\ell\}$ to be the *constraint variables* of $\mathcal{C}$. Moreover, we write $names(\mathcal{C})$ (resp. $vars(\mathcal{C})$) for the names (resp. variables) of $\mathcal{C}$.

The constraint systems that we consider arise while executing symbolic processes. We therefore restrict ourselves to *well-formed* constraint systems, capturing the fact that the knowledge of the environment always increases along the execution: we allow it to use more names and variables (less restrictions) or give it access to more terms (larger substitution). The fact that the constraint system is not arbitrary is useful when solving the constraints such as in [20].

Definition 6.2 (ordering on frames $\preceq$) *Let* $\varphi_1 = \nu\tilde{u}_1.\sigma_1$ *and* $\varphi_2 = \nu\tilde{u}_2.\sigma_2$ *be two frames. The frame* φ_1 *is smaller than or equal to* φ_2*, denoted by* $\varphi_1 \preceq \varphi_2$*, if* $\tilde{u}_1 \supseteq \tilde{u}_2$*,* $\mathrm{dom}(\sigma_1) \subseteq \mathrm{dom}(\sigma_2)$ *and* $y\sigma_1 = y\sigma_2$ *for any* $y \in \mathrm{dom}(\sigma_1)$*.*

Definition 6.3 (well-formed constraint system) *A constraint system* $\mathcal{C}$ *is* well-formed *if* $\mathcal{D}ed(\mathcal{C})$ *can be ordered* $\varphi_1 \Vdash x_1, \ldots, \varphi_\ell \Vdash x_\ell$ *in such a way that:*

1. *(monotonicity) for every* i *such that* $1 \leq i < \ell$*, we have* $\varphi_i \preceq \varphi_{i+1}$*, and*
2. *(origination) for every* i *such that* $1 \leq i \leq \ell$*, we have*

for all $x \in vars(\mathrm{img}(\varphi_i)) \cap cv(\mathcal{C})$*, there exists* $j < i$ *such that* $x = x_j$*.*

Moreover, if "$M \neq N$" $\in \mathcal{C}$ *then* "$\mathsf{gd}(M)$" $\in \mathcal{C}$ *and* "$\mathsf{gd}(N)$" $\in \mathcal{C}$*.*

In the remainder, when we consider a well-formed constraint system $\mathcal{C}$ and we write, for $\mathcal{D}ed(\mathcal{C})$, the sequence $\varphi_1 \Vdash x_1, \ldots, \varphi_\ell \Vdash x_\ell$, this implicitly means that we consider the ordering given by the monotonicity condition.

We say that two well-formed constraint systems $\mathcal{C}$ and $\mathcal{C}'$ have *same basis* if $\mathcal{D}ed(\mathcal{C}) = \{\varphi_1 \Vdash x_1, \ldots, \varphi_\ell \Vdash x_\ell\}$ and $\mathcal{D}ed(\mathcal{C}') = \{\varphi'_1 \Vdash x'_1, \ldots, \varphi'_\ell \Vdash x'_\ell\}$ are such that $x_i = x'_i$ and $\mathrm{dom}(\varphi_i) = \mathrm{dom}(\varphi'_i)$ for $1 \leq i \leq \ell$.

We now define the solutions of a well-formed constraint system. Intuitively, each solution defines an intermediate process which corresponds to a concrete instance of the corresponding symbolic process.

Definition 6.4 (E-solution) *Let* $\mathcal{C}$ *be a well-formed constraint system such that* $\mathcal{D}ed(\mathcal{C}) = \{\varphi_1 \Vdash x_1, \ldots, \varphi_\ell \Vdash x_\ell\}$ *where each* $\varphi_i = \nu\tilde{u}_i.\sigma_i$ *for some* $\tilde{u}_i$ *and some substitution* σ_i*. An* E-solution *of* $\mathcal{C}$ *is a substitution* θ *whose domain is* $cv(\mathcal{C})$ *and such that*

- $vars(x_i\theta) \cap cv(\mathcal{C}) = \emptyset$ *and* $vars(x_i\theta) \cap (\mathrm{dom}(\varphi_\ell) \setminus \mathrm{dom}(\varphi_i)) = \emptyset$*;*
- $names(x_i\theta) \cap \tilde{u}_i = \emptyset$ *and* $vars(x_i\theta) \cap \tilde{u}_i = \emptyset$*;*
- *for every constraint* "$M = N$" $\in \mathcal{C}$*, we have* $M(\theta\sigma_\ell)^\star =_{\mathsf{E}} N(\theta\sigma_\ell)^\star$*;*
- *for every constraint* "$M \neq N$" $\in \mathcal{C}$*, we have* $M(\theta\sigma_\ell)^\star \neq_{\mathsf{E}} N(\theta\sigma_\ell)^\star$*;*
- *for every constraint* "$\mathsf{gd}(M)$" $\in \mathcal{C}$*, the term* $M(\theta\sigma_\ell)^\star$ *is ground.*

We denote by $Sol_{\mathsf{E}}(\mathcal{C})$ *the set of* E*-solutions of* $\mathcal{C}$*. An* E*-solution* θ *of* $\mathcal{C}$ *is* closed *if* $vars(x_i\theta) \subseteq \mathrm{dom}(\varphi_i)$ *for any* $i \in \{1, \ldots, \ell\}$*. We denote by* $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C})$ *the set of closed* E*-solutions of* $\mathcal{C}$*.*

The condition that $vars(x_i\theta) \cap cv(\mathcal{C}) = \emptyset$ states that the image of θ should not use any variables that are in the domain of θ. The second condition, $vars(x_i\theta) \cap (\mathrm{dom}(\varphi_\ell) \setminus \mathrm{dom}(\varphi_i)) = \emptyset$, ensures that the environment does not use information that will only be revealed "in the future"; it can use only the entries of the frame that have previously been added. The conditions $names(x_i\theta) \cap \tilde{u}_i = \emptyset$ and $vars(x_i\theta) \cap \tilde{u}_i = \emptyset$ dissallow the environment to use restricted names and variables which are supposed to be secret; thus, they ensure that the value $x_i\theta$ can be *deduced* from public data. (These conditions are related to the definition of deduction given in [1].) The meaning of the remaining conditions should be clear.

Example 6.5 *Let* $\mathcal{C} = \{\nu k.\nu s.\{^{\mathsf{enc}(s,k)}/_{y_1}, {}^{k}/_{y_2}\} \Vdash x', \mathsf{gd}(c), x' = s\}$*. Let* E *be the equational theory* $\mathsf{dec}(\mathsf{enc}(x, y), y) = x$ *and* $\theta = \{^{\mathsf{dec}(y_1,y_2)}/_{x'}\}$*. We have that* θ *is a closed* E*-solution of* $\mathcal{C}$*. Note that* $\theta' = \{^{\mathsf{dec}(y_1,k)}/_{x'}\}$ *is not an* E*-solution of* $\mathcal{C}$*. Indeed,* $names(x'\theta') \cap \{k, s\} = \{k\}$ *and thus is not empty.*

We also define what it means to apply an evaluation context on a constraint system. This is needed because we define the semantics in a compositional way.

Definition 6.6 ($C[\mathcal{C}]$) *Let $C = \nu\tilde{n}.(\_ \mid D)$ be an intermediate evaluation context and e be a constraint. We have that*

- $C[e] = e$ *when e is a constraint of the form $M = N$, $M \neq N$ or $\mathsf{gd}(M)$;*
- $C[\nu\tilde{v}.\sigma \Vdash x] = \nu\tilde{n}.\nu\tilde{v}.(\sigma \cup \psi(D)) \Vdash x$ *otherwise.*

Similarly, for a variable $y \in \mathcal{X}$

- $\nu y.e = e$ *when e is a constraint of the form $M = N$, $M \neq N$ or $\mathsf{gd}(M)$;*
- $\nu y.e = \nu y, \tilde{v}.\sigma \Vdash x$ *when $e = \nu\tilde{v}.\sigma \Vdash x$.*

Given a constraint system $\mathcal{C}$, we have that $C[\mathcal{C}] = \{C[e] \mid e \in \mathcal{C}\}$ and $\nu y.\mathcal{C} = \{\nu y.e \mid e \in \mathcal{C}\}$.

7 Syntax and Semantics

7.1 Syntax

We first need to extend naming environments used in the intermediate semantics to the symbolic setting. For this, we introduce an infinite set $\mathcal{Y}$ of variables, to be used as constraint variables, disjoint from the set $\mathcal{X}$. We also distinghuish two disjoint subsets: $\mathcal{Y}_b$ for variables of base type and $\mathcal{Y}_{ch}$ for those of channel type. The functions *vars* and *fv* are updated to also return variables from $\mathcal{Y}$.

Definition 7.1 (Symbolic naming environment) *A* symbolic naming environment $\mathsf{N_s} : \mathcal{N} \cup \mathcal{X} \cup \mathcal{Y} \to \{\mathsf{n}, \mathsf{f}, \mathsf{b}, \mathsf{c}\}$ *is a function which maps each name and variable in $\mathcal{N} \cup \mathcal{X}$ to one of* n, f *and* b *and each variable in $\mathcal{Y}$ to one of* n *or* c. *It extends naming environments with an infinite set $\mathcal{Y}$ of variables that can be mapped to* c *(which stands for "constraint") or* n. *As before, we require that there are infinitely many names and infinitely many variables that are mapped to each of* n, f *and* b.

We say that a symbolic naming environment N_s is compatible *with an intermediate extended process A and a constraint system $\mathcal{C}$ if*

– $\mathsf{N}_s(fn(A)) = \mathsf{f}$ – $\mathsf{N}_s(bn(A) \cup bv(A)) = \mathsf{b}$ – $\mathsf{N}_s(names(\mathcal{C})) \subseteq \{\mathsf{f}, \mathsf{b}\}$
– $\mathsf{N}_s(fv(A)) \subseteq \{\mathsf{f}, \mathsf{c}\}$ – $\mathsf{N}_s(y) = \mathsf{c}$ *iff* $y \in cv(\mathcal{C})$ – $\mathsf{N}_s(vars(\mathcal{C})) \subseteq \{\mathsf{f}, \mathsf{c}, \mathsf{b}\}$

Intuitively, $\mathsf{N}_s(y) = \mathsf{c}$ means that the variable y is a *constraint* variable (i.e., an input from the environment subject to constraints in $\mathcal{C}$).

We are now ready to precisely define a symbolic process.

Definition 7.2 (Symbolic process) *A symbolic process is a triple $(A \; ; \; \mathcal{C} \; ; \; \mathsf{N}_s)$ where A is an intermediate extended process, $\mathcal{C}$ is a constraint system and N_s is a symbolic naming environment. We say that $(A \; ; \; \mathcal{C} \; ; \; \mathsf{N_s})$ is well-formed if*

- N_s *is compatible with* A *and* $\mathcal{C}$*;*
- *If* $\mathcal{D}ed(\mathcal{C}) = \{\varphi_1 \Vdash x_1, \ldots, \varphi_\ell \Vdash x_\ell\} \neq \emptyset$ *then* $\phi(A) \succeq \varphi_\ell$ *and* $bv(\varphi_1) \subseteq \mathrm{dom}(A)$*;*
- *for all* M, N *such that* $M = N$, $M \neq N$ *or* $\mathsf{gd}(M)$ *is in* $\mathcal{C}$ *we have that* $vars(M, N) \cap dom(A) = \emptyset$.

$(A\,;\mathcal{C}\,;\mathsf{N_s})$ *is said to be closed if* $fv(A) \subseteq \mathrm{dom}(A) \cup cv(\mathcal{C})$.

Given a well-formed symbolic process $(A\,;\mathcal{C}\,;\mathsf{N_s})$ we define by $Sol_{\mathsf{E}}(\mathcal{C}\,;\mathsf{N}_s)$ the set of solutions of $\mathcal{C}$ which are compatible with N_s, i.e.

$$Sol_{\mathsf{E}}(\mathcal{C}, \mathsf{N}_s) = \{\theta \mid \theta \in Sol_{\mathsf{E}}(\mathcal{C}), \mathsf{N}_s(names(\mathrm{img}(\theta)) \cup vars(\mathrm{img}(\theta))) = \mathsf{f}\}.$$

We denote by $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}\ ;\ \mathsf{Ns})$ the subset of $Sol_{\mathsf{E}}(\mathcal{C}, \mathsf{N}_s)$ containing closed E-solutions of $\mathcal{C}$.

Each of these solutions θ defines a corresponding (closed) intermediate process which we call the θ-concretization.

Definition 7.3 (θ-concretization) *Let* $(A_s\ ;\mathcal{C}\ ;\mathsf{N}_s)$ *be a well-formed symbolic process. Let* $\theta \in Sol_{\mathsf{E}}(\mathcal{C}, \mathsf{N}_s)$*. We say that an intermediate process* $(A\ ;\mathsf{N})$ *is the* θ*-concretization of* $(A_s\ ;\mathcal{C}\ ;\mathsf{N}_s)$ *if* $A = A_s(\theta\sigma)^\star$ *where* σ *is the maximal frame of* $\mathcal{C}$ *and* $\mathsf{N} = \mathsf{N}_s|_{\mathcal{N}\cup\mathcal{X}}$.

Example 7.4 *Let* $A_s = \nu b.(\mathsf{out}(c, x) \mid \{^b/_y\})$, $\mathcal{C} = \{\nu a.\nu b.\{^b/_y\} \Vdash x,\ x \neq c,\ \mathsf{gd}(x)\}$ *and* N_s *be a naming environment compatible with* A_s *and* $\mathcal{C}$ *such that* $\mathsf{N}_s(d) = \mathsf{f}$*. Let* $\theta_1 = \{^d/_x\}$, $\theta_2 = \{^y/_x\}$ *and* $\mathsf{N} = \mathsf{N}_s|_{\mathcal{N}\cup\mathcal{X}}$*. We have that* $\theta_1, \theta_2 \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}, \mathsf{N}_s)$*. Hence* $(\nu b.(\mathsf{out}(c, d) \mid \{^b/_y\})\ ;\mathsf{N})$ *(resp.* $(\nu b.(\mathsf{out}(c, b) \mid \{^b/_y\})\ ;\mathsf{N})$*) is the* θ_1 *(resp.* θ_2*) concretization of* $(A_s\ ;\mathcal{C}\ ;\mathsf{N}_s)$*. However,* $\nu b.(\mathsf{out}(c, a) \mid \{^b/_y\})$ *is not a concretization of* $(A_s\ ;\mathcal{C}\ ;\mathsf{N}_s)$ *since no* $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}, \mathsf{N}_s)$ *can have* a *in its image.*

7.2 Symbolic semantics

Symbolic structural equivalence ($\equiv_s$) is the smallest equivalence relation on well-formed symbolic processes such that:

$$\begin{array}{lrcl}
\textsc{Par-0}_s & (A\,;\mathcal{C}\,;\mathsf{N_s}) & \equiv_s & (A \mid 0\,;\mathcal{C}\,;\mathsf{N_s}) \\
\textsc{Par-A}_s & (A \mid (B \mid D)\,;\mathcal{C}\,;\mathsf{N_s}) & \equiv_s & ((A \mid B) \mid D\,;\mathcal{C}\,;\mathsf{N_s}) \\
\textsc{Par-C}_s & (A \mid B\,;\mathcal{C}\,;\mathsf{N_s}) & \equiv_s & (B \mid A\,;\mathcal{C}\,;\mathsf{N_s}) \\
\textsc{New-C}_s & (\nu n.\nu m.A\,;\mathcal{C}\,;\mathsf{N_s}) & \equiv_s & (\nu m.\nu n.A\,;\mathcal{C}\,;\mathsf{N_s})
\end{array}$$

$$\frac{(A\,;\mathcal{C}_A\,;\mathsf{N_s}) \equiv_s (B\,;\mathcal{C}_B\,;\mathsf{N_s})}{(C[A]\,;C[\mathcal{C}_A]\,;\mathsf{N_s}[bn(C[0]) \mapsto \mathsf{b}]) \equiv_s (C[B]\,;C[\mathcal{C}_B]\,;\mathsf{N_s}[bn(C[0]) \mapsto \mathsf{b}])}$$

Symbolic internal reduction $\rightarrow_s$ is the smallest relation on well-formed symbolic processes such that:

$\textsc{Comm}_s$ $\quad (\mathsf{out}(u,M).P \mid \mathsf{in}(v,x).Q \;;\; \mathcal{C} \;;\; \mathsf{N_s}) \rightarrow_s (P \mid Q\{^M/_x\} \;;\; \mathcal{C} \cup \{u = v \;,\; \mathsf{gd}(u) \;,\; \mathsf{gd}(v)\} \;;\; \mathsf{N_s})$
where $u, v \in \mathcal{N}_{ch} \cup (cv(\mathcal{C}) \cap \mathcal{Y}_{ch})$.

$\textsc{Then}_s$ $\quad (\text{if } M = N \text{ then } P \text{ else } Q \;;\; \mathcal{C} \;;\; \mathsf{N_s}) \rightarrow_s (P \;;\; \mathcal{C} \cup \{M = N\} \;;\; \mathsf{N_s})$

$\textsc{Else}_s$ $\quad (\text{if } M = N \text{ then } P \text{ else } Q \;;\; \mathcal{C} \;;\; \mathsf{N_s}) \rightarrow_s (Q \;;\; \mathcal{C} \cup \{M \neq N \;;\; \mathsf{gd}(M) \;;\; \mathsf{gd}(N)\} \;;\; \mathsf{N_s})$

$$\frac{(A \;;\; \mathcal{C}_A \;;\; \mathsf{N_s}) \rightarrow_s (B \;;\; \mathcal{C}_B \;;\; \mathsf{N_s})}{(C[A] \;;\; C[\mathcal{C}_A] \;;\; \mathsf{N_s}[bn(C[0]) \mapsto \mathsf{b}]) \rightarrow_s (C[B] \;;\; C[\mathcal{C}_B] \;;\; \mathsf{N_s}[bn(C[0]) \mapsto \mathsf{b}])}$$

$$\frac{(A \;;\; \mathcal{C}_A \;;\; \mathsf{N_s}) \equiv_s (B \;;\; \mathcal{C}_B \;;\; \mathsf{N_s}) \rightarrow_s (B' \;;\; \mathcal{C}'_B \;;\; \mathsf{N_s}) \equiv_s (B' \;;\; \mathcal{C}'_B \;;\; \mathsf{N_s})}{(A \;;\; \mathcal{C}_A \;;\; \mathsf{N_s}) \rightarrow_s (A' \;;\; \mathcal{C}'_A \;;\; \mathsf{N_s})}$$

In addition to the rules for symbolic structural equivalence and internal reduction, we adopt the following rules:

$\textsc{In}_s$ $\quad (\mathsf{in}(u,x).P \;;\; \mathcal{C} \;;\; \mathsf{N_s}) \xrightarrow{in(u,y)}_s (P\{^y/_x\} \;;\; \mathcal{C} \cup \{0 \Vdash y, \mathsf{gd}(u)\} \;;\; \mathsf{N_s}[y \mapsto \mathsf{c}])$
where $u \in \mathcal{N}_{ch} \cup (\mathcal{Y}_{ch} \cap cv(\mathcal{C}))$, $y \in \mathcal{Y}$, $\mathsf{N_s}(y) = \mathsf{n}$.

$\textsc{Out-Ch}_s$ $\quad (\mathsf{out}(u,v).P \;;\; \mathcal{C} \;;\; \mathsf{N_s}) \xrightarrow{out(u,v)}_s (P \;;\; \mathcal{C} \cup \{\mathsf{gd}(u), \mathsf{gd}(v)\} \;;\; \mathsf{N_s})$
where $u, v \in \mathcal{N}_{ch} \cup (\mathcal{Y}_{ch} \cap cv(\mathcal{C}))$.

$\textsc{Out-T}_s$
$(\mathsf{out}(u,M).P \;;\; \mathcal{C} \;;\; \mathsf{N_s}) \xrightarrow{\nu x.out(u,x)}_s (P \mid \{^M/_x\} \;;\; \nu x.\mathcal{C} \cup \{\mathsf{gd}(u)\} \;;\; \mathsf{N_s}[x \mapsto \mathsf{f}])$
where $x \in \mathcal{X}_b$, $\mathsf{N_s}(x) = \mathsf{n}$.

$\textsc{Open-Ch}_s$
$$\frac{(A \;;\; \mathcal{C} \;;\; \mathsf{N_s}) \xrightarrow{out(u,c)}_s (A' \;;\; \mathcal{C}' \;;\; \mathsf{N'_s}) \qquad u \neq c,\ d \in \mathcal{N}_{ch},\ \mathsf{N_s}(d) = \mathsf{n}}{(\nu c.A \;;\; \nu c.\mathcal{C} \;;\; \mathsf{N_s}[c \mapsto \mathsf{b}]) \xrightarrow{\nu d.out(u,d)}_s (A'\{^d/_c\} \;;\; \nu d.(\mathcal{C}'\{^d/_c\}) \;;\; \mathsf{N'_s}[c \mapsto \mathsf{b},\, d \mapsto \mathsf{f}])}$$

$\textsc{Scope}_s$ $\quad \dfrac{(A \;;\; \mathcal{C} \;;\; \mathsf{N_s}) \xrightarrow{\alpha}_s (A' \;;\; \mathcal{C}' \;;\; \mathsf{N'_s}) \qquad n \text{ does not occur in } \alpha}{(\nu n.A \;;\; \nu n.\mathcal{C} \;;\; \mathsf{N_s}[n \mapsto \mathsf{b}]) \xrightarrow{\alpha}_s (\nu n.A' \;;\; \nu n.\mathcal{C}' \;;\; \mathsf{N_s}[n \mapsto \mathsf{b}])}$

$\textsc{Par}_s$ $\quad \dfrac{(A \;;\; \mathcal{C} \;;\; \mathsf{N_s}) \xrightarrow{\alpha}_s (A' \;;\; \mathcal{C}' \;;\; \mathsf{N'_s})}{(A \mid B \;;\; \mathcal{C} \mid \psi(B) \;;\; \mathsf{N_s}) \xrightarrow{\alpha}_s (A' \mid B \;;\; \mathcal{C}' \mid \psi(B) \;;\; \mathsf{N'_s})}$

$\textsc{Struct}_s$ $\quad \dfrac{(A \;;\; \mathcal{C}_A \;;\; \mathsf{N_s}) \equiv_s (B \;;\; \mathcal{C}_B \;;\; \mathsf{N_s}) \xrightarrow{\alpha}_s (B' \;;\; \mathcal{C}'_B \;;\; \mathsf{N'_s}) \equiv_s (B' \;;\; \mathcal{C}'_B \;;\; \mathsf{N'_s})}{(A \;;\; \mathcal{C}_A \;;\; \mathsf{N_s}) \xrightarrow{\alpha}_s (A' \;;\; \mathcal{C}'_A \;;\; \mathsf{N'_s})}$

For reasons similar to those cited for $\textsc{Open-Ch}_i$, the rules $\textsc{In}_s$ and $\textsc{Open-Ch}_s$ require on-the-fly renaming. When a transition is executed under a context (by the rules $\textsc{Scope}_s$ and $\textsc{Par}_s$) the constraint system must also be put in the context (according to Definition 6.6). In $\textsc{Par}_s$ we avoid the substitution $\psi(B)$

which appears on a label in PAR_i since here we always have that the variables in $\text{dom}(\psi(B))$ do not occur in the label α. In the rule OPEN-CH$_s$, the restriction $\nu d.(\mathcal{C}'\{^d/_c\})$ is needed to ensure that the name d is not used to instantiate the previous inputs: those that are done before the disclosure of this name.

Example 7.5 *To illustrate our symbolic semantics, consider the process* $(A\ ;\ \emptyset\ ;\ \mathsf{N}_s)$ *where* $A = \nu k.\nu s.(\mathsf{in}(c,x).\textit{if } x = s \textit{ then } \mathsf{out}(c,ok) \mid \{^{\mathsf{enc}(s,k)}/_{y_1}\} \mid \{^k/_{y_2}\})$ *and* N_s *is a symbolic environment compatible with* A*. Let* $x' \in \mathcal{Y}_b$ *be a variable such that* $\mathsf{N}_s(x') = \mathsf{n}$*.*

$$\begin{array}{rll}(A\ ;\ \emptyset\ ;\ \mathsf{N}_s) & \xrightarrow{in(c,x')}_s & (A'\ ;\ \{\nu k.\nu s.\{^{\mathsf{enc}(s,k)}/_{y_1}, {}^k/_{y_2}\} \Vdash x'\ ,\ \mathsf{gd}(c)\}\ ;\ \mathsf{N}_s[x' \mapsto \mathsf{c}]) \\ & \longrightarrow_s & (\nu k.\nu s.(\mathsf{out}(c,ok) \mid \{^{\mathsf{enc}(s,k)}/_{y_1}\} \mid \{^k/_{y_2}\})\ ;\ \mathcal{C}\ ;\ \mathsf{N}_s[x' \mapsto \mathsf{c}])\end{array}$$

where $A' = \nu k.\nu s.(\textit{if } x' = s \textit{ then } \mathsf{out}(c,ok) \mid \{^{\mathsf{enc}(s,k)}/_{y_1}\} \mid \{^k/_{y_2}\})$ *and* $\mathcal{C}$ *is the system*

$$\{\nu k.\nu s.\{^{\mathsf{enc}(s,k)}/_{y_1}, {}^k/_{y_2}\} \Vdash x'\ ,\ \mathsf{gd}(c)\ ,\ x' = s\}$$

Let $\theta = \{^{\mathsf{dec}(y_1,y_2)}/_{x'}\}$*. We have* $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}\ ;\ \mathsf{N}_s[x' \mapsto \mathsf{c}])$ *(see Example 6.5) and*

$$(A\ ;\ \mathsf{N}) \xrightarrow{in(c,x'\theta)}_i \rightarrow_i (A'\theta'\ ;\ \mathsf{N})$$

where $\mathsf{N} = \mathsf{N}_s|_{\mathcal{N}\cup\mathcal{X}}$*.*

8 Soundness and Completeness

We now show soundness of $\equiv_s$, $\rightarrow_s$ and $\xrightarrow{\alpha}_s$ with respect to their counterparts in the intermediate semantics: whenever one of these relations holds between two symbolic processes, the corresponding relation in the intermediate semantics holds for each θ-concretization. The proofs can be found in Appendix B.1.

Proposition 8.1 (soundness of $\equiv_s$ and $\rightarrow_s$) *Let* $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ *and* $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$ *be two well-formed symbolic processes such that* $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s) \bowtie_s (B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$ *with* $\bowtie\ \in \{\equiv, \rightarrow\}$*. Let* $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B\ ;\ \mathsf{N}_s)$*. We have that* $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A\ ;\ \mathsf{N}_s)$ *and* $(A\ ;\ \mathsf{N}) \bowtie_i (B\ ;\ \mathsf{N})$ *where* $(A\ ;\ \mathsf{N})$ *(resp.* $(B\ ;\ \mathsf{N})$*) is the* θ*-concretization of* $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ *(resp.* $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$*).*

Proposition 8.2 (soundness of $\xrightarrow{\alpha}_s$) *Let* $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ *and* $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}'_s)$ *be two well-formed symbolic processes such that* $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s) \xrightarrow{\alpha_s}_s (B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}'_s)$*. Let* $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}_B\ ;\ \mathsf{N}'_s)$ *and* $\theta_A = \theta_B|_{cv(\mathcal{C}_A)}$*. We have that* $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_A\ ;\ \mathsf{N}_s)$ *and* $(A\ ;\ \mathsf{N}) \xrightarrow{\alpha_s\theta_B}_i (B\ ;\ \mathsf{N}')$*, where* $(A\ ;\ \mathsf{N})$ *and* $(B\ ;\ \mathsf{N}')$ *are respectively the* θ_A*-concretization and the* θ_B*-concretization of* $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ *and* $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}'_s)$*.*

We also show completeness of the symbolic semantics with respect to the intermediate one: each time a θ-concretization of a symbolic process is structurally equivalent, respectively reduces, to another intermediate process, the

symbolic process too is structurally equivalent, respectively reduces, to a corresponding symbolic process. The proofs of these two following propositions can be found in Appendix B.2.

Proposition 8.3 (completeness of $\equiv_s$ and $\rightarrow_s$) *Let $(A_s \ ; \ \mathcal{C}_A \ ; \ \mathsf{N}_s)$ be a well-formed symbolic process and $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A \ ; \ \mathsf{N}_s)$. Let $(A \ ; \ \mathsf{N})$ be the θ-concretization of $(A_s \ ; \mathcal{C}_A \ ; \ \mathsf{N}_s)$ and (A', N) be an intermediate process such that $(A \ ; \ \mathsf{N}) \bowtie_i (A' \ ; \ \mathsf{N})$ with $\bowtie \in \{\equiv, \rightarrow\}$. Then there exists a well-formed symbolic process $(A'_s \ ; \mathcal{C}'_A \ ; \ \mathsf{N}_s)$ such that:*

1. $(A_s \ ; \mathcal{C}_A \ ; \ \mathsf{N}_s) \bowtie_s (A'_s \ ; \mathcal{C}'_A \ ; \ \mathsf{N}_s)$,
2. $\theta \in Sol_{\mathsf{E}}(\mathcal{C}'_A \ ; \ \mathsf{N}_s)$,
3. $(A' \ ; \ \mathsf{N})$ *is the* θ*-concretization of* $(A'_s \ ; \mathcal{C}'_A \ ; \ \mathsf{N}_s)$.

Proposition 8.4 (completeness of $\xrightarrow{\alpha}_s$) *Let $(A_s \ ; \mathcal{C}_A \ ; \ \mathsf{N}_s)$ be a well-formed symbolic process and $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_A \ ; \ \mathsf{N}_s)$. Let $(A \ ; \ \mathsf{N})$ be the θ_A-concretization of $(A_s \ ; \mathcal{C}_A \ ; \ \mathsf{N}_s)$ and $(A' \ ; \ \mathsf{N}')$ be an intermediate process such that $(A \ ; \ \mathsf{N}) \xrightarrow{\alpha}_i (A' \ ; \ \mathsf{N}')$. Then there exists a well-formed symbolic process $(A'_s \ ; \mathcal{C}'_A \ ; \ \mathsf{N}'_s)$ and a substitution θ'_A such that:*

1. $(A_s \ ; \mathcal{C}_A \ ; \ \mathsf{N}_s) \xrightarrow{\alpha_s}_s (A'_s \ ; \mathcal{C}'_A \ ; \ \mathsf{N}'_s)$,
2. $\theta'_A \in Sol_{\mathsf{E}}(\mathcal{C}'_A \ ; \ \mathsf{N}'_s)$ *and* $\theta'_A|_{cv(\mathcal{C}_A)} = \theta_A$,
3. $(A' \ ; \ \mathsf{N}')$ *is the* θ'_A*-concretization of* $(A'_s \ ; \mathcal{C}'_A \ ; \ \mathsf{N}'_s)$, *and*
4. $\alpha_s\theta'_A = \alpha$.

From the propositions stated above, we easily derive the following corollaries.

Corollary 8.5 (soundness of $\rightarrow_s^*$) *Let $(A_s \ ; \mathcal{C}_A \ ; \ \mathsf{N}_s)$ and $(B_s \ ; \mathcal{C}_B \ ; \ \mathsf{N}_s)$ be two well-formed symbolic processes such that $(A_s \ ; \mathcal{C}_A \ ; \ \mathsf{N}_s) \rightarrow_s^* (B_s \ ; \mathcal{C}_B \ ; \ \mathsf{N}_s)$. Let $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B \ ; \ \mathsf{N}_s)$. We have that $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A \ ; \ \mathsf{N}_s)$ and $(A \ ; \ \mathsf{N}) \rightarrow_i^* (B \ ; \ \mathsf{N})$ where $(A \ ; \ \mathsf{N})$ (resp. $(B \ ; \ \mathsf{N})$) is the θ-concretization of $(A_s \ ; \ \mathcal{C}_A \ ; \ \mathsf{N}_s)$ (resp. $(B_s \ ; \mathcal{C}_B \ ; \ \mathsf{N}_s)$).*

Corollary 8.6 (soundness of $\rightarrow_s^* \xrightarrow{\alpha}_s \rightarrow_s^*$) *Let $(A_s \ ; \mathcal{C}_A \ ; \ \mathsf{N}_s)$ and $(B_s \ ; \mathcal{C}_B \ ; \ \mathsf{N}'_s)$ be two well-formed symbolic processes such that $(A_s \ ; \ \mathcal{C}_A \ ; \ \mathsf{N}_s) \rightarrow_s^* \xrightarrow{\alpha_s}_s \rightarrow_s^* (B_s \ ; \ \mathcal{C}_B \ ; \ \mathsf{N}'_s)$. Let $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}_B \ ; \mathsf{N}'s)$ and $\theta_A = \theta_B|_{cv(\mathcal{C}_A)}$. We have that $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_A \ ; \ \mathsf{N}_s)$ and $(A \ ; \ \mathsf{N}) \rightarrow_i^* \xrightarrow{\alpha_s\theta_B}_i \rightarrow_i^* (B \ ; \ \mathsf{N}')$, where $(A \ ; \ \mathsf{N})$ and $(B \ ; \ \mathsf{N}')$ are respectively the θ_A-concretization and the θ_B-concretization of $(A_s \ ; \mathcal{C}_A \ ; \ \mathsf{N}_s)$ and $(B_s \ ; \mathcal{C}_B \ ; \ \mathsf{N}'_s)$.*

9 Symbolic Equivalences

In this section, we define our notion of symbolic static equivalence and our notion of symbolic bisimulation. We also show the soundness of these equivalences w.r.t. their intermediate counterparts. We also show in Section 10 that our symbolic bisimulation is *not* complete.

We define symbolic static equivalence using an encoding similar to the one in [5]. The tests used to distinguish two frames in the definition of static equivalence are encoded by means of two additional deduction constraints on fresh variables x, y and by the equation $x = y$.

Definition 9.1 (symbolic static equivalence) *We say that two closed well-formed symbolic processes* $(A_s \; ; \mathcal{C}_A \; ; \mathsf{N}_s)$ *and* $(B_s \; ; \mathcal{C}_B \; ; \mathsf{N}_s)$ *are symbolically statically equivalent, written* $(A_s \; ; \mathcal{C}_A \; ; \mathsf{N}_s) \sim_s (B_s \; ; \mathcal{C}_B \; ; \mathsf{N}_s)$ *if for some variables* $x, y \in \mathcal{Y}_b$*, the constraint systems* $\mathcal{C}'_A$*,* $\mathcal{C}'_B$ *have the same basis and* $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \; ; \mathsf{N}'_s) = Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_B \; ; \mathsf{N}'_s)$ *where*

- $\mathsf{N}_s(\{x, y\}) = \mathsf{n}$,
- $\mathsf{N}'_s = \mathsf{N}_s[x, y \mapsto \mathsf{c}]$,
- $\mathcal{C}'_A = \mathcal{C}_A \cup \{\phi(A_s) \Vdash x \; , \phi(A_s) \Vdash y \; , x = y\}$, *and*
- $\mathcal{C}'_B = \mathcal{C}_B \cup \{\phi(B_s) \Vdash x \; , \phi(B_s) \Vdash y \; , x = y\}$.

The following proposition states the correctness of the symbolic static equivalence with respect to the concrete one.

Proposition 9.2 (soundness of symbolic static equivalence) *Let* $(A_s \; ; \mathcal{C}_A \; ; \mathsf{N}_s)$ *and* $(B_s \; ; \mathcal{C}_B \; ; \mathsf{N}_s)$ *be two closed and well-formed symbolic processes such that* $(A_s \; ; \mathcal{C}_A \; ; \mathsf{N}_s) \sim_s (B_s \; ; \mathcal{C}_B \; ; \mathsf{N}_s)$*. Then we have that:*

1. $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A \; ; \mathsf{N}_s) = Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_B \; ; \mathsf{N}_s)$*, and*
2. *for all* $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A \; ; \mathsf{N}_s)$ *we have that* $\phi(A_s(\theta\sigma_A)^\star) \sim \phi(B_s(\theta\sigma_B)^\star)$*, where* σ_A *(resp.* σ_B*) is the substitution corresponding to the maximal frame of* $\mathcal{C}_A$ *(resp.* $\mathcal{C}_B$*).*

Proof.

1. We show one direction, i.e., $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A \; ; \mathsf{N}_s) \subseteq Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_B \; ; \mathsf{N}_s)$. The other one can be proved in a similar way. Let $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A \; ; \mathsf{N}_s)$. Since $(A_s \; ; \mathcal{C}_A \; ; \mathsf{N}_s) \sim_s (B_s \; ; \mathcal{C}_B \; ; \mathsf{N}_s)$, we know that $\mathcal{C}'_A$, $\mathcal{C}'_B$ have same basis and $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \; ; \mathsf{N}'_s) = Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_B \; ; \mathsf{N}'_s)$ where
   - $x, y \in \mathcal{Y}_b$ and $\mathsf{N}_s(\{x, y\}) = \mathsf{n}$,
   - $\mathsf{N}'_s = \mathsf{N}_s[x, y \mapsto \mathsf{c}]$,
   - $\mathcal{C}'_A = \mathcal{C}_A \cup \{\phi(A_s) \Vdash x \; , \phi(A_s) \Vdash y \; , x = y\}$, and

- $\mathcal{C}'_B = \mathcal{C}_B \cup \{\phi(B_s) \Vdash x\ ,\ \phi(B_s) \Vdash y\ ,\ x = y\}$.

Let $\rho = \{x \mapsto a, y \mapsto a\}$ for some name a such that $\mathsf{N}_s(a) = \mathsf{f}$ and that does not occur in $A_s, \mathcal{C}_A, B_s, \mathcal{C}_B$. It is easy to see that $\theta \cup \rho \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}'_s)$. Since $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}'_s) = Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_B\ ;\ \mathsf{N}'_s)$, we deduce that $\theta \cup \rho \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_B\ ;\ \mathsf{N}'_s)$. It remains to check that $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_B\ ;\ \mathsf{N}_s)$.

Let $\phi(B_s) = \nu\tilde{n}'_B.\sigma'_B$. By hypothesis, we know that the idempotent substitution σ obtained by iterating $(\theta \cup \rho)\sigma'_B$ satisfies the constraints in $\mathcal{C}'_B$. Let $\nu\tilde{n}_A.\sigma_A$ be the maximal frame in $\mathcal{C}_A$ and $\nu\tilde{n}_B.\sigma_B$ be the maximal frame in $\mathcal{C}_B$. Since $\mathcal{C}_A$ and $\mathcal{C}_B$ have same basis, we have that $\mathrm{dom}(\sigma_A) = \mathrm{dom}(\sigma_B)$. Moreover, since $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A\ ;\ \mathsf{N}_s)$, we know that $\mathit{vars}(\mathrm{img}(\theta)) \subseteq \mathrm{dom}(\sigma_A) = \mathrm{dom}(\sigma_B)$. Since $\sigma = ((\theta \cup \rho)\sigma'_B)^\star$, we deduce that $\sigma = ((\theta \cup \rho)\sigma_B)^\star$, and actually $\sigma = (\theta\sigma_B)^\star \cup \rho$. This means that $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B\ ;\ \mathsf{N}_s)$ since x and y do not appear in $\mathcal{C}_B$.

2. Let $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A\ ;\ \mathsf{N}_s)$. Let σ_A be the substitution corresponding to the maximal frame of $\mathcal{C}_A$. Let $A = A_s(\theta\sigma_A)^\star$. Note that A is a closed intermediate extended process and $\phi(A)$ is a closed frame. Thanks to the previous point, we also have that $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_B\ ;\ \mathsf{N}_s)$. Let σ_B be the substitution corresponding to the maximal frame of $\mathcal{C}_B$. Let $B = B_s(\theta\sigma_B)^\star$. We have that B is a closed intermediate extended process and $\phi(B)$ is a closed frame. Assume that $\phi(A) \not\sim \phi(B)$. Since we have $\mathrm{dom}(\phi(A)) = \mathrm{dom}(\phi(B))$, this means that there exist two terms M and N such that

   - $\mathit{fv}(M, N) \subseteq \mathrm{dom}(\phi(A)) = \mathrm{dom}(\phi(A_s))$,
   - $\mathit{fn}(M, N) \cap (\mathit{bn}(A) \cup \mathit{bn}(B)) = \emptyset$, and
   - $(M\sigma'_A)(\theta\sigma_A)^\star =_{\mathsf{E}} (N\sigma'_A)(\theta\sigma_A)^\star$ and $(M\sigma'_B)(\theta\sigma_B)^\star \neq_{\mathsf{E}} (N\sigma'_B)(\theta\sigma_B)^\star$ (or vice-versa), where $\phi(A_s) = \nu\tilde{n}'_A.\sigma'_A$ and $\phi(B_s) = \nu\tilde{n}'_B.\sigma'_B$.

   Hence, we have that $\mathsf{N}'_s(\mathit{fv}(M, N)) = \mathsf{f}$ and we can also assume w.l.o.g. that $\mathsf{N}'_s(\mathit{fn}(M, N)) = \mathsf{f}$. Now, let $\rho = \{x \mapsto M,\ y \mapsto N\}$. First note that $\theta \cup \rho$ is closed w.r.t. $\mathcal{C}'_A$ and $\mathcal{C}'_B$. It remains to show that $\theta \cup \rho \in Sol_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}'_s)$ whereas $\theta \cup \rho \notin Sol_{\mathsf{E}}(\mathcal{C}'_B\ ;\ \mathsf{N}'_s)$ obtaining in this way a contradiction.

   - $\theta \cup \rho \in Sol_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}'_s)$.
     We want to show that $((\theta \cup \rho)\sigma'_A)^\star$ satisfies the constraints in $\mathcal{C}'_A$. We have that $((\theta \cup \rho)\sigma'_A)^\star = (\theta\sigma_A)^\star \cup (\rho\sigma'_A)(\theta\sigma_A)^\star$. By hypothesis, we know that $(\theta\sigma_A)^\star$ satisfies the constraints in $\mathcal{C}_A$. Hence, we conclude for the constraint in $\mathcal{C}_A$. We have also that $x((\theta \cup \rho)\sigma'_A)^\star =_{\mathsf{E}} y((\theta \cup \rho)\sigma'_A)^\star$, since we know that $(M\sigma'_A)(\theta\sigma_A)^\star =_{\mathsf{E}} (N\sigma'_A)(\theta\sigma_A)^\star$.
   - $\theta \cup \rho \notin Sol_{\mathsf{E}}(\mathcal{C}'_B\ ;\ \mathsf{N}'_s)$.
     We show that $((\theta \cup \rho)\sigma'_B)^\star$ does not satisfy the constraint $x = y$. We have that $x((\theta \cup \rho)\sigma'_B)^\star = (M\sigma'_B)(\theta\sigma_B))^\star$ and $y((\theta \cup \rho)\sigma'_B)^\star = (N\sigma'_B)(\theta\sigma_B)^\star$. We know that $(M\sigma'_B)(\theta\sigma_B)^\star) \neq_{\mathsf{E}} (N\sigma'_B)(\theta\sigma_B)^\star$. This allows us to conclude.

□

Although we do not need completeness of symbolic static equivalence for our result, we may note that it follows from Baudet's result [5]. By completeness we mean that $A \sim B$ implies that $(A\ ;\ \emptyset\ ;\ \mathsf{N}_s) \sim_s (B\ ;\ \emptyset\ ;\ \mathsf{N}_s)$ for any compatible naming environment N_s.
We now define symbolic labelled bisimulation using our symbolic semantics.

Definition 9.3 (Symbolic labelled bisimilarity ($\approx_s$)) *Symbolic labelled bisimilarity is the largest symmetric relation $\mathcal{R}$ on closed well-formed symbolic processes with same naming environment, such that $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)\ \mathcal{R}\ (B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$ implies*

1. $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s) \sim_s (B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$
2. *if $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s) \rightarrow_s (A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}_s)$ with $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}_s) \neq \emptyset$, then there exists a symbolic process $(B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}_s)$ such that*
   - $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s) \rightarrow^*_s (B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}_s)$, *and*
   - $(A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}_s)\ \mathcal{R}\ (B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}_s)$;
3. *if $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s) \xrightarrow{\alpha_s}_s (A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}'_s)$ with $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}'_s) \neq \emptyset$, then there exists a symbolic process $(B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'_s)$ such that*
   - $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s) \rightarrow^*_s \xrightarrow{\alpha_s}_s \rightarrow^*_s (B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'_s)$, *and*
   - $(A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}'_s)\ \mathcal{R}\ (B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'_s)$.

The side condition $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}'_s) \neq \emptyset$ ensures that we only consider symbolic executions that correspond to at least one concrete execution. The following theorem states the soundness of the symbolic bisimulation with respect to the intermediate one.

Theorem 9.4 (soundness of symbolic labelled bisimilarity) *Let $(A\ ;\ \mathsf{N})$ and $(B\ ;\ \mathsf{N})$ be two intermediate processes. Let N_s be a symbolic naming environment such that $\mathsf{N}_s|_{\mathcal{N}\cup\mathcal{X}} = \mathsf{N}$ and $\mathsf{N}_s(y) = \mathsf{n}$ for all $y \in \mathcal{Y}$. We have that*

$$(A\ ;\ \emptyset\ ;\ \mathsf{N}_s) \approx_s (B\ ;\ \emptyset\ ;\ \mathsf{N}_s) \Rightarrow (A\ ;\ \mathsf{N}) \approx_i (B\ ;\ \mathsf{N})$$

Proof. To prove this result, first we define a new relation $\mathcal{R}'$ and then we will show that $\mathcal{R}'$ is an intermediate labelled bisimulation witnessing $(A\ ;\ \mathsf{N}) \approx_i (B\ ;\ \mathsf{N})$. Let $\mathcal{R}$ be the relation witnessing $(A\ ;\ \emptyset\ ;\ \mathsf{N}_s) \approx_s (B\ ;\ \emptyset\ ;\ \mathsf{N}_s)$.

(i) Definition of $\mathcal{R}'$.
$(A\ ;\ \mathsf{N})\ \mathcal{R}'\ (B\ ;\ \mathsf{N})$ if there exists two closed well-formed symbolic processes $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ and $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$ such that

- $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)\ \mathcal{R}\ (B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$ with $\mathsf{N} = \mathsf{N}_s|_{\mathcal{N}\cup\mathcal{X}}$, and
- there exists $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A\ ;\ \mathsf{N}_s)$ such that $A_s(\theta\sigma_A)^\star = A$ and $B_s(\theta\sigma_B)^\star = B$ where σ_A (resp. σ_B) is the maximal frame of $\mathcal{C}_A$ (resp. $\mathcal{C}_B$).

(ii) $\mathcal{R}'$ is an intermediate bisimulation relation witnessing $(A \,;\, \mathsf{N}) \approx_i (B \,;\, \mathsf{N})$. First we have to show that $(A \,;\, \mathsf{N})\ \mathcal{R}'\ (B \,;\, \mathsf{N})$. To do this, it is sufficient to see that the two well-formed symbolic processes $(A \,;\, \emptyset \,;\, \mathsf{N}_s)$ and $(B \,;\, \emptyset \,;\, \mathsf{N}_s)$ satisfy the required conditions.

Now, we have to show that $\mathcal{R}'$ satisfies the three points of the definition of intermediate labelled bisimilarity. Let $(A \,;\, \mathsf{N})$ and $(B \,;\, \mathsf{N})$ be two closed intermediate processes such that $(A \,;\, \mathsf{N})\ \mathcal{R}'\ (B \,;\, \mathsf{N})$. By definition of $\mathcal{R}'$, we know that there exists two closed well-formed symbolic processes $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ and $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$ such that

- $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)\ \mathcal{R}\ (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$, with $\mathsf{N} = \mathsf{N}_s|_{\mathcal{N} \cup \mathcal{X}}$, and
- there exists $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$ such that $A_s(\theta\sigma_A)^\star = A$ and $B_s(\theta\sigma_B)^\star = B$.

We have to show that:

1. $\phi(A) \sim \phi(B)$.
   Thanks to Proposition 9.2, we deduce that
   - $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s) = Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}_s)$, and
   - for all $\theta' \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$ we have $\phi(A_s(\theta'\sigma_A)^\star) \sim \phi(B_s(\theta'\sigma_B)^\star)$.

   Since $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$ we deduce that $\phi(A_s(\theta\sigma_A)^\star) \sim \phi(B_s(\theta\sigma_B)^\star)$, *i.e* $\phi(A) \sim \phi(B)$.

2. If $(A \,;\, \mathsf{N}) \rightarrow_i (A' \,;\, \mathsf{N})$, then there exists $(B' \,;\, \mathsf{N})$ such that $(B \,;\, \mathsf{N}) \rightarrow_i^* (B' \,;\, \mathsf{N})$ and $(A' \,;\, \mathsf{N})\ \mathcal{R}'\ (B' \,;\, \mathsf{N})$.
   By definition of $\mathcal{R}'$, we know that $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ is a closed well-formed symbolic process such that $A_s(\theta\sigma_A)^\star = A$ and $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$. Hence, thanks to Proposition 8.3, we know that there exists a well-formed symbolic process $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)$ such that
   - $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \rightarrow_s (A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)$,
   - $\theta \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}_s)$, and
   - $A'_s(\theta\sigma'_A)^\star = A'$.

   We have that $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)\ \mathcal{R}\ (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$ and $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \rightarrow_s (A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)$. Moreover $\theta \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}_s)$, and actually $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}_s)$, thus we know that $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}_s) \neq \emptyset$. Hence, there exists a closed well-formed symbolic process $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}_s)$ such that
   - $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s) \rightarrow_s^* (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}_s)$, and
   - $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)\ \mathcal{R}\ (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}_s)$.

   Since $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)\ \mathcal{R}\ (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}_s)$, we deduce that $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}_s) = Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_B \,;\, \mathsf{N}_s)$ by using Proposition 9.2. We have that $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}_s)$ and hence, we deduce that $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_B \,;\, \mathsf{N}_s)$. Now, by Corollary 8.5, we deduce that $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}_s)$ and $(B_s(\theta\sigma_B)^\star \,;\, \mathsf{N}) \rightarrow^* (B'_s(\theta\sigma'_B)^\star \,;\, \mathsf{N})$. Let

$B' = B'_s(\theta\sigma'_B)^\star$. As $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)$ and $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}_s)$ are two closed well-formed symbolic processes such that $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)\ \mathcal{R}\ (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}_s)$ and $A'_s(\theta\sigma'_A)^\star = A'$ and $B'_s(\theta\sigma'_B)^\star = B'$, we have that $(A' \,;\, \mathsf{N})\ \mathcal{R}'\ (B' \,;\, \mathsf{N})$.

3. If $(A \,;\, \mathsf{N}) \xrightarrow{\alpha}_i (A' \,;\, \mathsf{N}')$ with $fv(\alpha) \subseteq \mathrm{dom}(A)$ then $(B \,;\, \mathsf{N}) \rightarrow_i^* \xrightarrow{\alpha}_i \rightarrow_i^* (B' \,;\, \mathsf{N}')$ and $(A' \,;\, \mathsf{N}')\ \mathcal{R}'\ (B' \,;\, \mathsf{N}')$ for some B'.

   By definition of $\mathcal{R}'$, we know that $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ is a closed well-formed symbolic process such that $A_s(\theta\sigma_A)^\star = A$ and $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$. Hence, thanks to Proposition 8.4, we know that there exist a well-formed symbolic process $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$, a substitution θ' and a label α_s such that

   - $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \xrightarrow{\alpha_s}_s (A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$ and $\mathsf{N}' = \mathsf{N}'_s|_{\mathcal{N}\cup\mathcal{X}}$,
   - $\theta' \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s)$ and $\theta'|_{cv(\mathcal{C}_A)} = \theta$,
   - $A'_s(\theta'\sigma'_A)^\star = A'$ where σ'_A is the substitution corresponding to the maximal frame in $\mathcal{C}'_A$, and
   - $\alpha_s\theta' = \alpha$.

   Actually, we have that $\theta' \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s)$, i.e. θ' is a closed solution. This is clear when the label α is not an input. In the case of an input, α_s is of the form $in(c, y)$ and we conclude by relying on the fact that $vars(y\theta') = fv(\alpha) \subseteq \mathrm{dom}(A)$.

   We have that $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)\ \mathcal{R}\ (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$ and $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \xrightarrow{\alpha}_s (A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$. Since $\theta' \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s)$, we know that $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s) \neq \emptyset$. Hence, there exists a closed well-formed symbolic process $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$ such that

   - $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s) \rightarrow_s^* \xrightarrow{\alpha_s}_s \rightarrow_s^* (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$, and
   - $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)\ \mathcal{R}\ (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$.

   Since $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)\ \mathcal{R}\ (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$, thanks to Proposition 9.2, we have that $Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s) = Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_B \,;\, \mathsf{N}'_s)$. We have that $\theta' \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s)$ and hence, we deduce that $\theta' \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\mathcal{C}'_B \,;\, \mathsf{N}'_s)$. Now, by Corollary 8.6, we deduce that $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}_s)$ and $(B_s(\theta\sigma_B)^\star \,;\, \mathsf{N}) \rightarrow^* (B'_s(\theta'\sigma'_B)^\star \,;\, \mathsf{N})$.

   As $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$ and $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$ are two closed well-formed symbolic processes such that $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)\ \mathcal{R}\ (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$ and $A'_s(\theta'\sigma'_A)^\star = A'$ and $B'_s(\theta'\sigma'_B)^\star = B'$, we have that $(A' \,;\, \mathsf{N})\ \mathcal{R}'\ (B' \,;\, \mathsf{N})$. This allows us to conclude.

□

PART III
— Soundness of Symbolic Bisimulation —

We now put the results of the previous section together (Theorem 5.2 and Theorem 9.4) to prove our main result.

Theorem 9.5 (Soundness of symbolic bisimulation) *Let A and B be two closed, nv-distinct extended processes. For any symbolic naming environment N_s compatible with $A{\downarrow}$, $B{\downarrow}$ and the empty constraint system we have that*

$$(A{\downarrow} \,;\, \emptyset \,;\, \mathsf{N}_s) \approx_s (B{\downarrow} \,;\, \emptyset \,;\, \mathsf{N}_s) \text{ implies } A \approx B$$

Note that limiting the theorem to nv-distinct processes is not an onerous restriction. If we want to prove that $A \approx B$, we can construct by α-conversion two nv-distinct processes A', B' such that $A' \equiv A$ and $B' \equiv B$. Showing $A' \approx B'$ implies that $A \approx B$, since $\approx$ is closed under structural equivalence.

10 Discussion

Our techniques suffer from the same sources of incompleteness as the ones described for the spi calculus in [10]. In a symbolic bisimulation the instantiation of input variables is postponed until the point at which they are actually used, leading to a finer relation. We illustrate this point on an example, similar to one given in [10].

Example 10.1 *Consider the two following processes:*

$$\begin{array}{rcl} P_1 & = & \nu c_1.\mathsf{in}(c_2, x).(\mathsf{out}(c_1, b) \mid \mathsf{in}(c_1, y) \mid \textit{if } x = a \textit{ then } \mathsf{in}(c_1, z).\mathsf{out}(c_2, a)) \\ Q_1 & = & \nu c_1.\mathsf{in}(c_2, x).(\mathsf{out}(c_1, b) \mid \mathsf{in}(c_1, y) \mid \mathsf{in}(c_1, z).\textit{if } x = a \textit{ then } \mathsf{out}(c_2, a)) \end{array}$$

We have that $P_1 \approx Q_1$ whereas $(P_1 \,;\, \emptyset \,;\, \mathsf{N}_s) \not\approx_s (Q_1 \,;\, \emptyset \,;\, \mathsf{N}_s)$ for any compatible naming environment N_s. To see the latter inequivalence, observe that $(Q_1 \,;\, \emptyset \,;\, \mathsf{N}_s)$ can make the transition $\xrightarrow{\mathsf{in}(c_2, x')}_s$ and then an internal transition to $(\nu c_1.(\mathsf{in}(c_1, y) \mid \textit{if } x' = a \textit{ then } \mathsf{out}(c_2, a)) \,;\, \mathcal{C} \,;\, \mathsf{N}'_s)$ with $\mathcal{C} = \{0 \Vdash x' \,,\, \mathsf{gd}(c_2)\}$; this process is still undecided about whether $x' = a$ or not. $(P_1 \,;\, \emptyset \,;\, \mathsf{N}_s)$ can make the transition $\xrightarrow{\mathsf{in}(c_2, x')}_s$ but then cannot make the corresponding internal transition without committing either to the constraint $x' = a$ or to the constraint $x' \neq a$. Whichever one it does, Q_1 can do the opposite, showing the inequivalence.

The second example shows that the requirement that the constraint systems must have the same solutions gives rise to some incompleteness.

Example 10.2 *Consider the two following processes:*

$$\begin{array}{rcl} P_2 & = & \mathsf{in}(c,x).\mathsf{out}(c,a) \\ Q_2 & = & \mathsf{in}(c,x).\mathit{if}\ x = a\ \mathit{then}\ \mathsf{out}(c,a)\ \mathit{else}\ \mathsf{out}(c,a) \end{array}$$

We have that $P_2 \approx Q_2$ *whereas* $(P_2 \,;\, \emptyset \,;\, \mathsf{N}_s) \not\approx_s (Q_2 \,;\, \emptyset \,;\, \mathsf{N}_s)$ *for any compatible naming environment* N_s*. Indeed, we have that* $(P_2 \,;\, \emptyset \,;\, \mathsf{N}_s) \xrightarrow{\mathsf{in}(c,x')}_s (\mathsf{out}(c,a) \,;\, \mathcal{C} \,;\, \mathsf{N}'_s)$ *with* $\mathcal{C} = \{0 \Vdash x' \,,\, \mathsf{gd}(c)\}$*) whereas* $(Q_2 \,;\, \emptyset \,;\, \mathsf{N}_s) \xrightarrow{\mathsf{in}(c,x')}_s$ *if* $x = a$ *then* $\mathsf{out}(c,a)$ *else* $\mathsf{out}(c,a)$ *and then can move to either* $(\mathsf{out}(c,a) \,;\, \mathcal{C}_1 \,;\, \mathsf{N}'_s)$ *or* $(\mathsf{out}(c,a) \,;\, \mathcal{C}_2 \,;\, \mathsf{N}'_s)$ *where* $\mathcal{C}_1 = \{0 \Vdash x' \,,\, \mathsf{gd}(c) \,,\, x' = a\}$ *and* $\mathcal{C}_2 = \{0 \Vdash x' \,,\, \mathsf{gd}(c) \,,\, \mathsf{gd}(x') \,,\, x' \neq a\}$*. However, neither* $\mathcal{C}_1$ *nor* $\mathcal{C}_2$ *is equivalent to* $\mathcal{C}$*.*

Although our symbolic bisimulation is not complete, as shown above, we are able to prove labelled bisimulation on interesting examples for which the method implemented in the state-of-the-art ProVerif tool [7] fails. For instance, ProVerif is unable to establish labelled bisimilarity between $\mathsf{out}(c,a) \mid \mathsf{out}(c,b)$ and $\mathsf{out}(c,b) \mid \mathsf{out}(c,a)$ whereas of course we are able to deal with such examples. A more interesting example, for which our symbolic semantics plays an important role is as follows.

Example 10.3 *Consider the following two processes.*

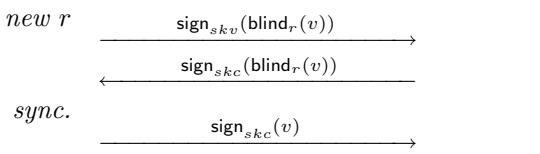
$$\begin{array}{rcl} P & = & \nu c_1.(\mathsf{in}(c_2,x).\mathsf{out}(c_1,x).\mathsf{out}(c_2,a) \mid \mathsf{in}(c_1,y).\mathsf{out}(c_2,y)) \\ Q & = & \nu c_1.(\mathsf{in}(c_2,x).\mathsf{out}(c_1,x).\mathsf{out}(c_2,x) \mid \mathsf{in}(c_1,y).\mathsf{out}(c_2,a)) \end{array}$$

These two processes are labelled bisimilar and our symbolic labelled bisimulation is complete enough to prove this. In particular, let $P' = \nu c_1.(\mathsf{out}(c_1,x').\mathsf{out}(c_2,a) \mid \mathsf{in}(c_1,y).\mathsf{out}(c_2,y))$ *and* $Q' = \nu c_1.(\mathsf{out}(c_1,x').\mathsf{out}(c_2,x') \mid \mathsf{in}(c_1,y).\mathsf{out}(c_2,a))$*. The relation* $\mathcal{R}$*, that witnesses the symbolic bisimulation, includes*

$$\begin{array}{rcl} (P \,;\, \emptyset \,;\, \mathsf{N}_s) & \mathcal{R} & (Q \,;\, \emptyset \,;\, \mathsf{N}_s) \\ (P' \,;\, \{\nu c_1.0 \Vdash x' \,,\, \mathsf{gd}(c_2)\} \,;\, \mathsf{N}'_s) & \mathcal{R} & (Q' \,;\, \{\nu c_1.0 \Vdash x' \,,\, \mathsf{gd}(c_2)\} \,;\, \mathsf{N}'_s) \\ (\nu c_1.(\mathsf{out}(c_2,a) \mid \mathsf{out}(c_2,x')) \,; \\ \{\nu c_1.0 \Vdash x' \,,\, \mathsf{gd}(c_2) \,,\, \mathsf{gd}(c_1)\} \,;\, \mathsf{N}'_s) & \mathcal{R} & (\nu c_1.(\mathsf{out}(c_2,x') \mid \mathsf{out}(c_2,a)) \,; \\ & & \{\nu c_1.0 \Vdash x' \,,\, \mathsf{gd}(c_2) \,,\, \mathsf{gd}(c_1)\} \,;\, \mathsf{N}'_s) \end{array}$$

The example above is inspired by the problems we encountered when we analysed a bisimulation representing the privacy property in an electronic voting protocol [19]. ProVerif is not able to prove this kind of equivalence; its algorithm is limited to cases that the two processes P, Q have the same structure, and differ only in the terms that are output. In this example, the processes differ in their structure, providing the motivation for our methods. Our symbolic bisimulation seems to be "sufficiently" complete to deal with examples of privacy and anonymity properties arising in protocol analysis. We demonstrate that more fully with the next example, which considers the privacy property of voting systems in more detail, and illustrates the equational reasoning aspects of the calculus.

Example 10.4 *We consider a simplified version of the voting protocol due to Fujioka, Okamoto and Ohta (see [17]) that is analysed in [19, 12]. In the simplification we consider here, the voter casts a vote in the first phase of the voting process by blinding his selected candidate v with a random value r, and signing the result with his private key. He sends this signature to the collector. Using this signature, the collector is able to check that the voter is entitled to vote, and the collector sends back his signature on the blinded choice of the voter. The voter now unblinds this value, obtaining the collector's signature on his vote. In the second phase, he anonymously sends this signature to the collector for counting.*



The blinding operation allows signatures to be performed blindly. Here, the collector signs the vote, but is not able to see its value. This helps to achieve the property of vote-privacy for the voter. To avoid traffic analysis attacks, the protocol is in two phases; the voter synchronises with other voters between the two phases (represented by "sync." in the figure). This synchronisation can easily be modelled using private channels, but we prefer to omit that detail to keep the example simple. Although it satisfies vote-privacy, this simple protocol would allow a voter to vote multiple times by repeatedly sending the last message. That problem is easily fixed, but we prefer to keep the protocol simple for the purpose of illustration.

We assume a signature containing the binary functions sign, getmess, blind, unblind, *and the unary function* pk *with the equations:*

$$\begin{aligned} \mathsf{getmess}_{\mathsf{pk}(x)}(\mathsf{sign}_x(y)) &= y \\ \mathsf{unblind}_x(\mathsf{sign}_y(\mathsf{blind}_x(z))) &= \mathsf{sign}_y(z) \end{aligned}$$

Note that key arguments are written as subscripts, to aid readability. A voter with signing key skv casting the vote v for the collector with public key pkc runs the process P described below. (Since all the communications take place over a public channel, we do not mention it for sake of readability.)

$$\begin{aligned} P(skv, v, pkc) = \nu r.\ (&\mathsf{out}(\mathsf{sign}_{skv}(\mathsf{blind}_r(v))).\mathsf{in}(x). \\ &\textit{if } \mathsf{getmess}_{pkc}(x) = \mathsf{blind}_r(v) \textit{ then } \mathsf{out}(\mathsf{unblind}_r(x))) \end{aligned}$$

The anonymity property we want to prove says that an observer (which may include the collector) cannot distinguish a situation in which the voter A votes v_a and the voter B votes v_b, from another one in which they vote the other way around. Roughly speaking, it is the following labelled bisimilarity:

$$\begin{aligned} &\nu ska, skb(\mathsf{out}(\mathsf{pk}(ska)).\mathsf{out}(\mathsf{pk}(skb)).(P(ska, v_a, \mathsf{pk}(skc)) \mid P(skb, v_b, \mathsf{pk}(skc)))) \\ &\qquad\qquad \approx \\ &\nu ska, skb(\mathsf{out}(\mathsf{pk}(ska)).\mathsf{out}(\mathsf{pk}(skb)).(P(ska, v_b, \mathsf{pk}(skc)) \mid P(skb, v_a, \mathsf{pk}(skc)))) \end{aligned} \tag{1}$$

with the proviso that the voters A and B have to synchronise at the "sync." point. Note that we treat the collector's private key skc as a public name; we prove privacy property even in presence of a corrupted collector who disclosed his private key.

Below, we illustrate some of the calculations to establish this equivalence (of course we prove $\approx_s$ to establish $\approx$). We will only consider deducibility and equality constraints and we do not give the naming environment associated to each symbolic process.

intermediate process	*some of the constraints*
$P(ska, v_a, \mathsf{pk}(skc))$	$\emptyset$
$\xrightarrow{\nu x_1.out(x_1)}_s$	
$\nu r.(\mathsf{in}(x).\mathit{if}\ \mathsf{getmess}_{\mathsf{pk}(skc)}(x) = \mathsf{blind}_r(v_a)$ $\mathit{then}\ \mathsf{out}(\mathsf{unblind}_r(x)) \mid \{^{M_1}/_{x_1}\})$	$\emptyset$
$\xrightarrow{in(y)}_s$ $\nu r.(\mathit{if}\ \mathsf{getmess}_{\mathsf{pk}(skc)}(y) = \mathsf{blind}_r(v_a)$ $\mathit{then}\ \mathsf{out}(\mathsf{unblind}_r(y)) \mid \{^{M_1}/_{x_1}\})$	$\nu r.\{^{M_1}/_{x_1}\} \Vdash y$
$\rightarrow_s$ $\nu r.(\mathsf{out}(\mathsf{unblind}_r(y)) \mid \{^{M_1}/_{x_1}\})$	$\nu r.\{^{M_1}/_{x_1}\} \Vdash y$ $\mathsf{getmess}_{\mathsf{pk}(skc)}(y) = \mathsf{blind}_r(v_a)$
$\xrightarrow{\nu x_2.out(x_2)}_s$ $\nu r.(\{^{M_1}/_{x_1}\} \mid \{^{M_2}/_{x_2}\})$	$\nu r.\nu x_2.\{^{M_1}/_{x_1}\} \Vdash y$ $\mathsf{getmess}_{\mathsf{pk}(skc)}(y) = \mathsf{blind}_r(v_a)$

where $M_1 = \mathsf{sign}_{ska}(\mathsf{blind}_r(v_a))$ and $M_2 = \mathsf{unblind}_r(y)$. Note that to derive the first step, we use the rule Out-T_s *and* Scope_s *to add the restriction $\nu r.$ in front of the process. To derive the other steps, for instance $\nu x_2.out(x_2)$ we also use the rule* Par_s *to put $\{^{M_1}/_{x_1}\}$ in parallel. About the naming environment, we have assumed among others that x_1, x_2 and y are marked as new (namely* n*) at the beginning. At the end, the variables x_1, x_2 are marked as* f *whereas y is marked as* c*.*

We can now consider some example evolutions for the two processes in the equivalence (1) we want to establish. One of the expected evolutions of the left hand side is as follows. The process $P(ska, v_a, \mathsf{pk}(skc))$ will first do an action directly followed by the corresponding action of the process $P(skb, v_b, \mathsf{pk}(skc))$. The $\downarrow$ operator allows us to put the restrictions in front of the process to have an intermediate process.

$(\nu ska, skb.(\mathsf{out}(\mathsf{pk}(ska)).\mathsf{out}(\mathsf{pk}(skb)).$
$(P(ska, v_a, \mathsf{pk}(skc)) \mid P(skb, v_b, \mathsf{pk}(skc))))\downarrow \ ; \ \emptyset \ ; \ \mathsf{N}_s)$

$\xrightarrow{\nu x_0.out(x_0)}_s \xrightarrow{\nu x'_0.out(x'_0)}_s$ (* outputs of the public keys *)

$\xrightarrow{\nu x_1.out(x_1)}_s \xrightarrow{\nu x'_1.out(x'_1)}_s$ (* outputs of the first message *)

$\xrightarrow{in(y)}_s \xrightarrow{in(y')}_s$ (* inputs of the second message *)

$\rightarrow_s \rightarrow_s$ (* conditional - then branch *)

$\xrightarrow{\nu x_2.out(x_2)}_s \xrightarrow{\nu x'_2.out(x'_2)}_s \quad (\varphi \ ; \mathcal{C} \ ; \mathsf{N}'_s)$

where $\varphi = \nu ska, skb, ra, rb.(\{^{\mathsf{pk}(ska)}/_{x_0}\} \mid \{^{\mathsf{pk}(skb)}/_{x'_0}\} \mid \{^{\mathsf{sign}_{ska}(\mathsf{blind}_{ra}(v_a))}/_{x_1}\} \mid \{^{\mathsf{sign}_{skb}(\mathsf{blind}_{rb}(v_b))}/_{x'_1}\} \mid \{^{\mathsf{unblind}_{ra}(y)}/_{x_2}\} \mid \{^{\mathsf{unblind}_{rb}(y')}/_{x'_2}\})$.
Among the constraints in $\mathcal{C}$, *we have the following deducibility and equality constraints.*

$$\nu x_2, x'_2.\varphi \Vdash y \qquad \mathsf{getmess}_{\mathsf{pk}(skc)}(y) = \mathsf{blind}_{ra}(v_a)$$
$$\nu x_2.x'_2.\varphi \Vdash y' \qquad \mathsf{getmess}_{\mathsf{pk}(skc)}(y') = \mathsf{blind}_{rb}(v_b)$$

The right hand side of equivalence (1) can evolve in a similar way, i.e. with the same labels, to the symbolic process $(\varphi' \ ; \mathcal{C}' \ ; \mathsf{N}'_s)$ *where*

- $\varphi' = \nu ska, skb, ra, rb.(\{^{\mathsf{pk}(ska)}/_{x_0}\} \mid \{^{\mathsf{pk}(skb)}/_{x'_0}\} \mid \{^{\mathsf{sign}_{ska}(\mathsf{blind}_{ra}(v_b))}/_{x_1}\} \mid \{^{\mathsf{sign}_{skb}(\mathsf{blind}_{rb}(v_a))}/_{x'_1}\} \mid \{^{\mathsf{unblind}_{rb}(y')}/_{x_2}\} \mid \{^{\mathsf{unblind}_{ra}(y)}/_{x'_2}\})$.
- *the system* $\mathcal{C}'$ *contains* $\nu x_2, x'_2.\varphi' \Vdash y, \quad \mathsf{getmess}_{\mathsf{pk}(skc)}(y) = \mathsf{blind}_{ra}(v_b)$
  $\nu x_2, x'_2.\varphi' \Vdash y', \ \mathsf{getmess}_{\mathsf{pk}(skc)}(y') = \mathsf{blind}_{rb}(v_a)$.

To fully show that the two processes are in symbolic bisimulation, we would have to consider other possible evolutions as well. We omit that here. To complete the picture for this path, we illustrate the calculations to show static equivalence for the final processes along the paths. Consider the extended constraint systems $\tilde{\mathcal{C}}$ *and* $\tilde{\mathcal{C}}'$ *as defined in Definition 9.1.*

- $\tilde{\mathcal{C}} = \mathcal{C} \cup \{\varphi \Vdash z_1 \ ; \varphi \Vdash z_2 \ ; z_1 = z_2\}$, *and*
- $\tilde{\mathcal{C}}' = \mathcal{C}' \cup \{\varphi' \Vdash z_1 \ ; \varphi' \Vdash z_2 \ ; z_1 = z_2\}$.

Let $\tilde{\mathsf{N}}_s = \mathsf{N}'_s[z_1, z_2 \mapsto \mathsf{f}]$ *where* z_1, z_2 *are constraint variables that are marked as* n *in* N'_s. *We have to establish that* $Sol^{\mathsf{cl}}_{\mathsf{E}}(\tilde{\mathcal{C}} \ ; \tilde{\mathsf{N}}_s) = Sol^{\mathsf{cl}}_{\mathsf{E}}(\tilde{\mathcal{C}}' \ ; \tilde{\mathsf{N}}_s)$. *We don't prove this fully, but just illustrate with an expected solution. A solution is a map* θ *from the constraint variables* $\{y, y', z_1, z_2\}$ *to terms not including the constraint variables or the restricted names* ska, skb, ra, rb, *and in the case of* y, y', *not including* x_2, x'_2. *Consider for instance the solution*

$$\theta := \left\{ \begin{array}{lcllcl} y & \mapsto & \mathsf{sign}_{skc}(\mathsf{getmess}_{x_0}(x_1)) & z_1 & \mapsto & x_2 \\ y' & \mapsto & \mathsf{sign}_{skc}(\mathsf{getmess}_{x'_0}(x'_1)) & z_2 & \mapsto & v_a \end{array} \right.$$

We have that $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\tilde{\mathcal{C}} \,;\, \tilde{\mathsf{N}}_s)$ *and also* $\theta \in Sol^{\mathsf{cl}}_{\mathsf{E}}(\tilde{\mathcal{C}}' \,;\, \tilde{\mathsf{N}}_s)$.

ProVerif can handle the equational theory of this example, but it is not able to prove the privacy property. The reason is that ProVerif is not able to construct the bisimulation that is required. In the first phase, the behaviour of voter A must be matched with the behaviour of voter A, and B's behaviour with B's behaviour, so that the signatures respect the static equivalence; while in the second phase, A's behaviour must be matched with B's behaviour, and B's behaviour with A's behaviour, so that the votes output respect the static equivalence. (However, our algorithms are not yet implemented!)

11 Related and Future Work

Pioneering work in symbolic bisimulations has been done by Henessy and Lin [18] for value-passing CCS. However, the result which is most closely related to ours is by Borgström *et al.* [10]: they define a symbolic bisimulation for the spi calculus with the same sources of incompleteness as we have. However, our treatment of general equational theories is non trivial as illustrated by the problems implied for structural equivalence.

For many important equational theories, static equivalence has been shown to be decidable in [1]. More interestingly, some work has also been done to automate observational equivalence. The ProVerif tool [7] automates observational equivalence checking for the applied pi calculus (with process replication), but since the problem is undecidable the technique it uses is necessarily incomplete. The tool aims at proving a finer equivalence relation and relies on easily matching up the execution paths of the two processes [8]. In his thesis, Baudet [6] presents a decision procedure for a similar equivalence, called diff-equivalence, in a simplified process calculus. Examples where this equivalence relation is too fine occur when proving the observational equivalence required to show vote-privacy [19, 12]. Although our symbolic bisimulation is not complete, we are able to conclude on examples where ProVerif fails (see Section 10).

The technique used in ProVerif will generally fail in the case where the two processes take different branches at some point. This is the case in Example 10.3: after a synchronisation (modelled by a communication on the private channel c_1) between the two parallel components of process P (resp. Q), the output action of the left component of P matches the output action of the right component of Q.

Concerning future work, the obvious next step is to study the equivalence of solutions for constraint systems under different equational theories. Promising results have already been shown in [5] for a significant class of equational theories but for constraint systems that do not have disequalities. These results readily apply for deciding our symbolic bisimulation on the fragment without else branches in conditionals. We plan to implement an automated tool for checking observational equivalence. In particular we aim at automating proofs

arising in case studies of electronic voting protocols which currently rely on hand proofs [12].

Another direction for future work is how to include process replication (the "!" operator), which is omitted entirely from this paper. Since we require to put the ν operator in outermost position in intermediate extended processes, one could first try to include replications that do not have ν in their scope. This corresponds to processes that may not terminate, but can only create finitely many names. Including replication having ν in its scope is certainly more challenging.

Acknowledgements. We would like to thank Martín Abadi, Cédric Fournet, Magnus Johansson and Bjorn Victor for interesting discussions. We also warmly thank the anonymous reviewers for their many detailed comments; they helped us significantly to improve the paper. Thanks also to Liu Jia, who asked us several questions that were instrumental in helping us prepare the final version.

References

[1] M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. *Theoretical Computer Science*, 387(1-2):2–32, 2006.

[2] M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *Proc. 28th Symposium on Principles of Programming Languages*, pages 104–115. ACM Press, 2001.

[3] M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: The spi calculus. In *Proc. 4th Conference on Computer and Communications Security*, pages 36–47. ACM, 1997.

[4] R. Amadio, D. Lugiez, and V. Vanackère. On the symbolic reduction of processes with cryptographic functions. *Theoretical Computer Science*, 290:695–740, 2002.

[5] M. Baudet. Deciding security of protocols against off-line guessing attacks. In *Proc. 12th Conference on Computer and Communications Security*, pages 16–25. ACM Press, 2005.

[6] M. Baudet. *Sécurité des protocoles cryptographiques : aspects logiques et calculatoires*. Thèse de doctorat, LSV, ENS Cachan, France, Jan. 2007.

[7] B. Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *Proc. 14th Computer Security Foundations Workshop*, pages 82–96. IEEE Comp. Soc. Press, 2001.

[8] B. Blanchet, M. Abadi, and C. Fournet. Automated Verification of Selected Equivalences for Security Protocols. In *Proc. 20th Symposium on Logic in Computer Science*, pages 331–340. IEEE Comp. Soc. Press, 2005.

[9] M. Boreale and R. D. Nicola. A symbolic semantics for the pi-calculus. *Information and Computation*, 126(1):34–52, 1996.

[10] J. Borgström, S. Briais, and U. Nestmann. Symbolic bisimulation in the spi calculus. In *Proc. 15th Int. Conference on Concurrency Theory*, volume 3170 of *LNCS*, pages 161–176. Springer, 2004.

[11] S. Delaune and F. Jacquemard. A decision procedure for the verification of security protocols with explicit destructors. In *Proc. 11th ACM Conference on Computer and Communications Security (CCS'04)*, pages 278–287. ACM Press, 2004.

[12] S. Delaune, S. Kremer, and M. D. Ryan. Coercion-resistance and receipt-freeness in electronic voting. In *Proc. 19th Computer Security Foundations Workshop*, pages 28–39. IEEE Comp. Soc. Press, 2006.

[13] S. Delaune, S. Kremer, and M. D. Ryan. Symbolic bisimulation for the applied pi calculus. In *Preliminary Proc. 5th International Workshop on Security Issues in Concurrency (SecCo'07)*, 2007.

[14] S. Delaune, S. Kremer, and M. D. Ryan. Symbolic bisimulation for the applied pi-calculus. In *Proc. 27th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'07)*, volume 4855 of *Lecture Notes in Computer Science*, pages 133–145. Springer, 2007.

[15] S. Delaune, S. Kremer, and M. D. Ryan. Symbolic bisimulation for the applied pi calculus. Research Report LSV-08-32, Laboratoire Spécification et Vérification, ENS Cachan, France, 2008. 73 pages.

[16] D. Dolev and A. C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, IT-29(12):198–208, 1983.

[17] A. Fujioka, T. Okamoto, and K. Ohta. A practical secret voting scheme for large scale elections. In *Advances in Cryptology – AUSCRYPT '92*, volume 718 of *Lecture Notes in Computer Science*, pages 244–251. Springer, 1992.

[18] M. Hennessy and H. Lin. Symbolic bisimulations. *Theoretical Computer Science*, 138(2):353–389, 1995.

[19] S. Kremer and M. D. Ryan. Analysis of an electronic voting protocol in the applied pi-calculus. In *Proc. 14th European Symposium on Programming*, volume 3444 of *LNCS*, pages 186–200. Springer, 2005.

[20] J. K. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *Proc. 8th Conference on Computer and Communications Security*, pages 166–175, 2001.

[21] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, part I. *Information and Computation*, 100(1):1–40, 1992.

A Proofs of Part I – Intermediate Calculus

A.1 Soundness Results

Proposition A.1 (soundness of $\equiv_i$) *Let $(A_i \;;\; \mathsf{N})$ and $(B_i \;;\; \mathsf{N})$ be two intermediate processes such that $(A_i \;;\; \mathsf{N}) \equiv_i (B_i \;;\; \mathsf{N})$. Then we have that $A_i \equiv B_i$.*

Proof. This proof is straightforward and can be done by induction on the proof tree witnessing $(A_i \;;\; \mathsf{N}) \equiv_i (B_i \;;\; \mathsf{N})$. The only point where we have to pay attention is closure by application of intermediate evaluation context. However, to deal with this case, it is sufficient to note that any intermediate evaluation context w.r.t. an intermediate extended process A is also an evaluation context w.r.t. A. □

Proposition A.2 (soundness of $\rightarrow_i$) *Let $(A_i \;;\; \mathsf{N})$ and $(B_i \;;\; \mathsf{N})$ be two intermediate processes such that $(A_i \;;\; \mathsf{N}) \rightarrow_i (B_i \;;\; \mathsf{N})$. Then we have that $A_i \rightarrow B_i$.*

Proof. This proof is straightforward and can be done by induction on the proof tree witnessing $(A_i \;;\; \mathsf{N}) \rightarrow_i (B_i \;;\; \mathsf{N})$. The base case, *i.e.* $\textsc{Comm}_i$, $\textsc{Then}_i$ and $\textsc{Else}_i$ are obvious since corresponding rules exist in the initial semantics. To deal with closure by application of evaluation context, it is sufficient to note that any intermediate evaluation context w.r.t. an intermediate extended process A is also an evaluation context w.r.t. A. Lastly, closure by structural equivalence can be easily done thanks to Proposition A.1. □

Proposition A.3 (soundness of $\xrightarrow{\alpha}_i$) *Let $(A_i \;;\; \mathsf{N})$ and $(B_i \;;\; \mathsf{N}')$ be two intermediate processes such that $(A_i \;;\; \mathsf{N}) \xrightarrow{\alpha}_i (B_i \;;\; \mathsf{N}')$. Then we have that $A_i \xrightarrow{\alpha} B_i$.*

Proof. To prove this result, we distinguish two cases depending on the fact whether α is an output or an input. In both cases, we perform the proof by induction on the prooftree witnessing the fact that $(A_i \;;\; \mathsf{N}) \xrightarrow{\alpha}_i (B_i \;;\; \mathsf{N}')$. However, if α is an input, we need to show an intermediate result in order to be able to deal with the inductive case $\textsc{Par}_i$. Note that in the case where α is a label of the form $\mathsf{out}(a, c)$, $\nu d.\mathsf{out}(a, d)$ or $\nu x.\mathsf{out}(a, x)$, we do not have such a problem since the rule $\textsc{Par}_i$ is equivalent to the rule described below. This is due to the fact that $fv(\alpha) = \emptyset$.

$$\textsc{Par-Out}_i \quad \frac{(A \;;\; \mathsf{N}) \xrightarrow{\alpha}_i (A', \mathsf{N}')}{(A \mid B \;;\; \mathsf{N}) \xrightarrow{\alpha}_i (A' \mid B, \mathsf{N}')}$$

First case: *α is of the form $\mathsf{out}(a, c)$, $\nu d.\mathsf{out}(a, d)$ or $\nu x.\mathsf{out}(a, x)$.* The two base cases, $\textsc{Out-Ch}_i$ and $\textsc{Out-T}_i$, are trivial. We only need to pay attention that the side condition of Out-T is satisfied. This is due to the fact that $\mathsf{N}(x) = \mathsf{n}$ whereas $\mathsf{N}(fv(P) \cup fv(M)) = \mathsf{f}$. Hence, we have that $x \notin fv(P) \cup fv(M)$. Now, we have to deal with the inductive cases.

Case Open-Ch$_i$. In such a case, we have that the prooftree witnessing the fact that $(A_i \ ; \mathsf{N}) \xrightarrow{\alpha}_i (B_i \ ; \mathsf{N}')$ ends with the following inference rule.

$$\frac{(A' \ ; \mathsf{N}'') \xrightarrow{out(a,c)}_i (B' \ ; \mathsf{N}''') \qquad c \neq a, \mathsf{N}''(d) = \mathsf{n}}{(\nu c.A', \mathsf{N}) \xrightarrow{\nu d.out(a,d)}_i (B'\{^d/_c\}, \mathsf{N}')}$$

By induction hypothesis, we know that $A' \xrightarrow{out(a,c)} B'$ and we deduce that $A'\{^d/_c\} \xrightarrow{out(a,d)} B'\{^d/_c\}$ since A' and B' are nv-distinct and d is fresh (it does not appear in A' nor in B'). By application of the rule Open-Ch, we obtain $\nu d.A'\{^d/_c\} \xrightarrow{\nu d.out(a,d)} B'\{^d/_c\}$. Since $\nu d.A'\{^d/_c\} \equiv \nu c.A'$, we deduce that $\nu c.A' \xrightarrow{\nu d.out(a,d)} B'\{^d/_c\}$, i.e. exactly what we want.

Case Scope$_i$. This case is completely straightforward.

Case Par-Out$_i$. The proof tree ends with the following inference rule

$$\textsc{Par-Out}_i \quad \frac{(A' \ ; \mathsf{N}) \xrightarrow{\alpha}_i (B', \mathsf{N}')}{(A' \mid D \ ; \mathsf{N}) \xrightarrow{\alpha}_i (B' \mid D, \mathsf{N}')}$$

In such a case, we need to pay attention that the side condition of the rule Par is satisfied. Since $\mathsf{N}(bn(\alpha) \cup bv(\alpha)) = \mathsf{n}$ and $\mathsf{N}(fn(D) \cup fv(D)) = \mathsf{f}$, we deduce that $bn(\alpha) \cap fn(D) = bv(\alpha) \cap fv(D) = \emptyset$.

Case Struct$_i$. We easily conclude for this case by using Proposition A.1.

Second case: *α is of the form* $\mathsf{in}(a, M)$. To deal with this case, we rely on the claim stated below.

> **Claim:** Let $(A_i \ ; \mathsf{N})$ and $(B_i \ ; \mathsf{N}')$ be two extended processes such that A_i and B_i are intermediate framed processes and $(A_i \ ; \mathsf{N}) \xrightarrow{in(a,M)}_i (B_i \ ; \mathsf{N}')$. Let D_i be an intermediate framed process such that $(A_i \mid D_i \ ; \mathsf{N})$ and $(B_i \mid D_i \ ; \mathsf{N}')$ are also intermediate processes. Then, for any term M' such that $M = M'\psi(D_i)$, we have that $A_i \mid D_i \xrightarrow{in(a,M')} B_i \mid D_i$.

Note that this result will allow us to conclude. Our claim allows us to deal with the case where A_i and B_i are intermediate framed processes. For this it is sufficient to apply the claim above with $D_i = 0$ and $M' = M$. Then it remains to notice that $A_i \mid 0 \equiv A_i$ and $B_i \mid 0 \equiv B_i$ in order to conclude. Now, let us consider the case where $(A_i \ ; \mathsf{N})$, $(B_i \ ; \mathsf{N}')$ are not intermediate framed processes. We show the result by induction on the prooftree witnessing $(A_i \ ; \mathsf{N}) \xrightarrow{in(a,M)}_i (B_i \ ; \mathsf{N}')$. In such a case, this prooftree ends either with an instance of Scope$_i$ or an instance of Struct$_i$. In both cases, we easily conclude by using the induction hypothesis and applying the corresponding rules, that is either Scope or Struct and using Proposition A.1.

It remains to establish the claim.

Proof of the claim. We show this result by induction on the prooftree witnessing $(A_i \ ; \mathsf{N}) \xrightarrow{in(a,M)}_i (B_i \ ; \mathsf{N}')$. First, we consider the base case, i.e. the rule IN_i. In such a case, we have that $A_i = \mathsf{in}(a,x).P$, $B_i = P\{^M/_x\}$ and $\mathsf{N}' = \mathsf{N}$. We have that

$$\dfrac{\dfrac{\mathsf{in}(a,x).P \xrightarrow{in(a,M')} P\{^{M'}/_x\}}{\mathsf{in}(a,x).P \mid D_i \xrightarrow{in(a,M')} P\{^{M'}/_x\} \mid D_i \ \equiv P\{^M/_x\} \mid D_i}}{A_i \mid D_i \xrightarrow{in(a,M')} B_i \mid D_i}$$

Now, we have to deal with the inductive cases, that is STRUCT_i and PAR_i since the other rules do not allow us to derive framed processes. For STRUCT_i the result can be easily obtained by applying the induction hypothesis and Proposition A.1. Hence, we focus on PAR_i. In such a case, we have that the proof tree witnessing $(A_i \ ; \mathsf{N}) \xrightarrow{in(a,M)}_i (B_i \ ; \mathsf{N}')$ ends with the following rule:

$$\dfrac{(A_i' \ ; \mathsf{N}) \xrightarrow{in(a,M\psi(D))}_i (B_i' \ ; \mathsf{N}')}{(A_i' \mid D \ ; \mathsf{N}) \xrightarrow{in(a,M)}_i (B_i' \mid D \ ; \mathsf{N}')}$$

Recall that D_i is an intermediate framed process such that $(A_i' \mid D \mid D_i \ ; \mathsf{N})$ and $(B_i' \mid D \mid D_i \ ; \mathsf{N}')$ are also intermediate framed processes. Let M' be a term such that $M = M'\psi(D_i)$. By induction hypothesis and since $M\psi(D) = M'\psi(D|D_i)$, we have that $A_i' \mid (D_i \mid D) \xrightarrow{in(a,M')} B_i' \mid (D_i \mid D)$ and we easily conclude by using the fact that $\xrightarrow{\alpha}$ is closed by structural equivalence. □

A.2 Completeness Results

Given a nv-distinct extended process A containing an active substitution $\{^M/_x\}$. The process $A_{\setminus x}$ is A but with the unique occurrence of $\{^M/_x\}$ replaced by 0. This notation is extended as expected to sequences of variables. Now, we introduce a lemma which allows us to describe the process $C[A]\downarrow$ from $C\downarrow$ and $A\downarrow$.

Lemma A.4 *Let C be an evaluation context which is nv-distinct. Let $\tilde{x}$ be the tuple of variables such that the hole is in the scope of an occurrence of "νx" in C. Then there exists some sequences of names $\tilde{n}_1$, $\tilde{n}_2$ and an intermediate framed evaluation context G such that*

- *$C\downarrow = \nu\tilde{n}_1.\nu\tilde{n}_2.G$, and*
- *for all extended process A such that $C[A]$ is nv-distinct, we have that*

$$C[A]\downarrow = \nu\tilde{n}_1.\nu\tilde{m}.\nu\tilde{n}_2.G[F_{\setminus\tilde{x}}](\psi(G) \cup \psi(F))^\star$$

where $A\downarrow = \nu\tilde{m}.F$ for some sequence of names $\tilde{m}$ and some intermediate framed process F.

Proof. We prove this result by induction on the structure of C. In the base case, i.e. $C = \_$, we can show that $\tilde{n}_1 = \emptyset$, $\tilde{n}_2 = \emptyset$ and $G = \_$ satisfy the requirements. Indeed, let A be an extended process such that $A\downarrow = \nu\tilde{m}.F$, we have $\nu\tilde{n}_1.\nu\tilde{m}.\nu\tilde{n}_2.G[F_{\setminus\tilde{x}}](\psi(G) \cup \psi(F))^\star = \nu\tilde{m}.F\psi(F)^\star = \nu\tilde{m}.F = A\downarrow = C[A]\downarrow$.

The inductive cases are $C = C' \mid B$, $C = B \mid C'$, $C = \nu n.C'$ and $C = \nu x.C'$. Let $\tilde{x}'$ be the tuple of variables x' such that the hole of C' is in the scope of an occurrence of $\nu x'$ in C'. By induction hypothesis, we know that there exists some sequences of names $\tilde{n}'_1$ and $\tilde{n}'_2$ and an intermediate framed evaluation context G' such that

- $C'\downarrow = \nu\tilde{n}'_1.\nu\tilde{n}'_2.G'$, and
- for all extended process A such that $C'[A]$ is nv-distinct, we have that

$$C'[A]\downarrow = \nu\tilde{n}'_1.\nu\tilde{m}.\nu\tilde{n}'_2.G'[F_{\setminus\tilde{x}'}](\psi(G') \cup \psi(F))^\star \text{ where } A\downarrow = \nu\tilde{m}.F.$$

Inductive case 1: $C = C' \mid B$. Let $B\downarrow = \nu\tilde{b}.B'$. In such a case, we have that

$$\begin{array}{rcl} C[A]\downarrow & = & (C'[A] \mid B)\downarrow \\ & = & \nu\tilde{n}'_1.\nu\tilde{m}.\nu\tilde{n}'_2.\nu\tilde{b}.(G'[F_{\setminus\tilde{x}'}](\psi(G') \cup \psi(F))^\star \mid B')(\psi(C'[A]) \cup \psi(B'))^\star \end{array}$$

Let $\tilde{n}_1 = \tilde{n}'_1, \tilde{n}_2 = \tilde{n}'_2, \tilde{b}$ and $G = G' \mid B'$. As $\tilde{x} = \tilde{x}'$ and $\psi(G') \cup \psi(F) = \psi(C'[A])$ we obtain the expected result.

Inductive case 2: $C = B \mid C'$. This case is similar to the previous one.

Inductive case 3: $C = \nu n.C'$. In such a case, we have that

$$\begin{array}{rcl} C[A]\downarrow & = & \nu n.(C'[A]\downarrow) \\ & = & \nu n.\nu\tilde{n}'_1.\nu\tilde{m}.\nu\tilde{n}'_2.G'[F_{\setminus\tilde{x}'}](\psi(G') \cup \psi(F))^\star \end{array}$$

Let $\tilde{n}_1 = n, \tilde{n}'_1$, $\tilde{n}_2 = \tilde{n}'_2$ and $G' = G$. We obtain the expected result.

Inductive case 4: $C = \nu x.C'$. In such a case, we have that

$$\begin{array}{rcl} C[A]\downarrow & = & (C'[A]\downarrow)_{\setminus x} \\ & = & (\nu\tilde{n}'_1.\nu\tilde{m}.\nu\tilde{n}'_2.G'[F_{\setminus\tilde{x}'}](\psi(G') \cup \psi(F))^\star)_{\setminus x} \end{array}$$

Let $\tilde{n}_1 = \tilde{n}'_1$, $\tilde{n}_2 = \tilde{n}'_2$ and $G' = G$. We obtain the expected result since we have that $\tilde{x} = \tilde{x}', x$. □

The following lemma relies on the notion of linear proof defined below. A proof in linear form of $A \equiv B$ is a sequence $A = A_1, \ldots, A_n = B$ such that for every $1 \leq j \leq n$, there exist an evaluation context C_j, two extended processes A'_j and A'_{j+1} such that:

- $A_j = C_j[A'_j]$, $A_{j+1} = C_j[A'_{j+1}]$, and
- either $A'_j \equiv A'_{j+1}$ (or $A'_{j+1} \equiv A'_j$) is an instance of PAR-0, PAR-A, PAR-C, NEW-0, NEW-C, NEW-PAR, ALIAS, SUBST, or REWRITE,

- or $A'_j =_\alpha A'_{j+1}$.

Similarly, a proof in linear form of $A \rightarrow^* B$ is a sequence $A = A_1, \ldots, A_n = B$ such that for every $1 \leq j \leq n$, there exist an evaluation context C_j, two extended processes A'_j and A'_{j+1} such that:

- $A_j = C_j[A'_j]$, $A_{j+1} = C_j[A'_{j+1}]$, and
- either $A'_j \equiv A'_{j+1}$ (or $A'_{j+1} \equiv A'_j$) is an instance of PAR-0, PAR-A, PAR-C, NEW-0, NEW-C, NEW-PAR, ALIAS, SUBST, or REWRITE,
- or $A'_j =_\alpha A'_{j+1}$,
- or $A'_j \rightarrow A'_{j+1}$ is an instance of COMM, THEN or ELSE.

Moreover, there must exist at least one j such that $A'_j \rightarrow A'_{j+1}$ is an instance of COMM, THEN or ELSE.

Lemma A.5 *Let A and B be two nv-distinct extended processes such that $A \bowtie B$ with $\bowtie \in \{\equiv, \rightarrow^*\}$ and N be a naming environment compatible with A and B. Then there exists a proof in linear form such that every process in the proof is nv-distinct and compatible with N.*

Proposition A.6 (completeness of $\equiv_i$) *Let A and B be two nv-distinct extended processes such that $A \equiv B$ and N be a naming environment compatible with $A\downarrow$ and $B\downarrow$. Then there exists an intermediate process $(D_i \ ; \mathsf{N})$ such that $(A\downarrow \ ; \mathsf{N}) \equiv_i (D_i \ ; \mathsf{N}) \approxeq (B\downarrow \ ; \mathsf{N})$.*

Proof. Let A and B be two nv-distinct extended processes such that $A \equiv B$. We consider the proof of structural equivalence in linear form. Thanks to Lemma A.5, we can assume that extended processes involved in this derivation are nv-distinct and compatible with N. We show the result by induction on the length ℓ of the derivation. We first show the result when $\ell = 1$ by considering each rule of structural equivalence in turn. Then, we show the inductive case, i.e. $\ell > 1$. We denote by C the evaluation context under which the structural equivalence rule is applied. We denote by $\tilde{n}_1$, $\tilde{n}_2$, (resp. $\tilde{x}$) and G the sequences of names (resp. variables) and the intermediate framed evaluation context which satisfy the condition stated in Lemma A.4.

Case PAR-0: $C[D] \equiv C[D \mid 0]$.
Let $D_i = B\downarrow$. Clearly, we have that $(D_i \ ; \mathsf{N}) \approxeq (B\downarrow \ ; \mathsf{N})$. Now, let $\tilde{m}$ be the sequence of names and F be the framed process such that $D\downarrow = \nu\tilde{m}.F$. We have that $(D \mid 0)\downarrow = \nu\tilde{m}.(F \mid 0)$ and thanks to Lemma A.4 we have

- $(A\downarrow \ ; \mathsf{N}) = (C[D]\downarrow \ ; \mathsf{N}) = (\nu\tilde{n}_1.\nu\tilde{m}.\nu\tilde{n}_2.G[F_{\setminus\tilde{x}}](\psi(G) \cup \psi(F))^\star \ ; \mathsf{N})$, and
- $(B\downarrow \ ; \mathsf{N}) = (C[D \mid 0]\downarrow \ ; \mathsf{N}) = (\nu\tilde{n}_1.\nu\tilde{m}.\nu\tilde{n}_2.G[(F \mid 0)_{\setminus\tilde{x}}](\psi(G) \cup \psi(F \mid 0))^\star \ ; \mathsf{N})$.

Hence, we have that $(A\downarrow\ ;\ \mathsf{N}) \equiv_i (B\downarrow\ ;\ \mathsf{N})$.

A similar reasoning allows us to conclude for Par-A, Par-C. For the rule New-C, if the commutation involves two names, we conclude as in the previous case since this rule has a counterpart in the intermediate semantics. Otherwise, we have that $A\downarrow = B\downarrow$ and we easily conclude.

The rule New-0, New-Par, Subst, Alias and Rewrite are also straightforward. Note also that if $A \equiv B$ is a renaming step, then $D_i = A\downarrow$ satisfies the requirement. This concludes the base cases.

Now, it remains to show the inductive case. Let A and A' be two extended processes such that $A \equiv A'$ by a derivation of length $\ell > 1$. Then there exists B such that $A \equiv B$ by a derivation of length 1 and $B \equiv A'$ by a derivation of length $\ell' < \ell$. Firstly, we know that there exists an intermediate extended process $(D_i\ ;\ \mathsf{N})$ such that $(A\downarrow\ ;\ \mathsf{N}) \equiv_i (D_i\ ;\ \mathsf{N})$ and $(D_i\ ;\ \mathsf{N}) \approx (B\downarrow\ ;\ \mathsf{N})$. By using our induction hypothesis, we also know that there exists an intermediate extended process $(D_i'\ ;\ \mathsf{N})$ such that $(B\downarrow\ ;\ \mathsf{N}) \equiv_i (D_i'\ ;\ \mathsf{N})$ and $(D_i'\ ;\ \mathsf{N}) \approx (A'\downarrow\ ;\ \mathsf{N})$. Hence by using Lemma 4.5, we deduce that there exists an intermediate extended process $(D_i''\ ;\ \mathsf{N})$ such that $(D_i\ ;\ \mathsf{N}) \equiv_i (D_i''\ ;\ \mathsf{N})$ and $(D_i''\ ;\ \mathsf{N}) \approx (D_i'\ ;\ \mathsf{N})$. Hence, the process $(D_i''\ ;\ \mathsf{N})$ satisfies the requirements. □

Proposition A.7 (completeness of $\rightarrow_i$) *Let A and B be two nv-distinct extended processes such that $A \rightarrow^* B$ (resp. $A \rightarrow B$) and N be a naming environment compatible with $A\downarrow$ and $B\downarrow$. Then there exists an extended process $(D_i\ ;\ \mathsf{N})$ such that:*

- $(A\downarrow\ ;\ \mathsf{N}) \rightarrow_i^* (D_i\ ;\ \mathsf{N})$ *(resp. $(A\downarrow\ ;\ \mathsf{N}) \rightarrow_i (D_i\ ;\ \mathsf{N})$) and,*
- $(D_i\ ;\ \mathsf{N}) \approx (B\downarrow\ ;\ \mathsf{N})$.

Proof. Let A and B be two nv-distinct extended processes such that $A \rightarrow^* B$. The case where $B = A$ (reflexivity) is trivial. Otherwise we consider the proof of $A \rightarrow^* B$ in linear form. Each step of this proof will be either a single reduction step or a sequence of steps of structural equivalence. Thanks to Lemma A.5, we can assume that extended processes involved in this derivation are nv-distinct and compatible with N. We show the result by induction on the length ℓ of the derivation. We first show the result when $\ell = 1$. In the case of structural equivalence, Proposition A.6 allows us to conclude. Hence, we only consider the three rules of internal reduction in turn. Then, we show the inductive case, i.e. $\ell > 1$. We denote by C the evaluation context under which the rule is applied. We denote by $\tilde{n}_1$, $\tilde{n}_2$, (resp. $\tilde{x}$) and G the sequences of names (resp. variables) and the intermediate framed evaluation context which satisfy the condition stated in Lemma A.4.

Case Comm: $\mathsf{out}(a, M).P \mid \mathsf{in}(a, x).Q \rightarrow P \mid Q\{^M/_x\}$. We have that $A = C[\mathsf{out}(a, M).P \mid \mathsf{in}(a, x).Q]$ and $B = C[P \mid Q\{^M/_x\}]$. Let $P\downarrow = \nu\tilde{n}_p.F_p$ and $Q\downarrow = \nu\tilde{n}_q.F_q$. By using Lemma A.4, we obtain

- $(A\downarrow\ ;\ \mathsf{N}) = (\nu\tilde{n}_1.\nu\tilde{n}_p.\nu\tilde{n}_q.\nu\tilde{n}_2.G[\mathsf{out}(a, M).F_p \mid \mathsf{in}(a, x).F_q]\psi(G)^\star\ ;\ \mathsf{N})$,

- $(B\downarrow\,;\mathsf{N}) = (\nu\tilde{n}_1.\nu\tilde{n}_p.\nu\tilde{n}_q.\nu\tilde{n}_2.G[F_p \mid F_q\{^M/_x\}]\psi(G)^\star\,;\mathsf{N})$.

Note that F_p and F_q are intermediate plain processes (as P, resp. Q, are prefixed by an input, resp. output) and hence $\psi(F_p)$ and $\psi(F_q\{^M/_x\})$ are empty. Let $D_i = B\downarrow$. It is easy to see that D_i satisfies the requirements.

We deal with the rules Then and Else in a similar way.

Now, it remains to show the inductive case. Let A and A' be two extended processes such that $A \to^* A'$ by a derivation of length $\ell > 1$. Then there exists B such that $A \equiv B$ (or $A \to B$) and $B \to^* A'$ by a derivation of length $\ell' < \ell$. In both case we conclude thanks to Lemma 4.5 and the induction hypothesis on $B \to^* A'$. □

Lemma A.8 *Let A and B be two nv-distinct extended processes such that $A \xrightarrow{\alpha} B$ and N be a naming environment compatible with A and α. Let N' be a naming environment compatible with B and such that:*

- $\mathsf{N}' = \mathsf{N}[x \mapsto \mathsf{f}]$ *when α is of the form $\nu x.out(a,x)$;*
- $\mathsf{N}' = \mathsf{N}[d \mapsto \mathsf{f}]$ *when α if of the form $\nu d.out(a,d)$;*
- $\mathsf{N}' = \mathsf{N}$ *otherwise.*

Then there exist two nv-distinct extended processes A' and B' such that:

- $A \equiv A' \xrightarrow{\alpha} B' \equiv B$ *(where $A' \xrightarrow{\alpha} B'$ does not rely on some structural equivalence steps); and*
- N *is compatible with A', and N' is compatible with B'.*

Proposition A.9 (completeness of $\xrightarrow{\alpha}_i$) *Let A and B be two nv-distinct extended processes such that $A \xrightarrow{\alpha} B$ and N be a naming environment compatible with $A\downarrow$ and α. Let N' be a naming environment compatible with $B\downarrow$ such that:*

- $\mathsf{N}' = \mathsf{N}[x \mapsto \mathsf{f}]$ *when α is of the form $\nu x.out(a,x)$;*
- $\mathsf{N}' = \mathsf{N}[d \mapsto \mathsf{f}]$ *when α if of the form $\nu d.out(a,d)$;*
- $\mathsf{N}' = \mathsf{N}$ *otherwise.*

Then there exists an intermediate process $(D_i\,;\mathsf{N}')$ such that

$$(A\downarrow\,;\mathsf{N}) \xrightarrow{\alpha}_i (D_i\,;\mathsf{N}') \approxeq (B\downarrow\,;\mathsf{N}').$$

Proof. Thanks to Lemma A.8, we can assume that A and B are two nv-distinct extended processes such that $A \xrightarrow{\alpha} B$ without involving any structural equivalence step. Otherwise, we will have that $A \equiv A' \xrightarrow{\alpha} B' \equiv B$ and we can easily conclude, thanks to Lemma 4.5, by applying the result on $A' \xrightarrow{\alpha} B'$ and by using Proposition A.6 on $A \equiv A'$ and $B \equiv B'$. We consider the different kind of labels in turn: $out(a,c)$, $\nu x.out(a,x)$, $in(a,M)$ and $\nu c.out(a,c)$.

We denote by C the evaluation context (constructed by successive applications of the rules $\textsc{Par}_i$ and $\textsc{Scope}_i$) under which the rule is applied. We denote by $\tilde{n}_1$, $\tilde{n}_2$, (resp. $\tilde{x}$) and G the sequences of names (resp. variables) and the intermediate framed evaluation context which satisfy the condition stated in Lemma A.4.

Case Out-Ch: $\mathsf{out}(a,c).P \xrightarrow{\mathsf{out}(a,c)} P$.
We have that $A = C[\mathsf{out}(a,c).P]$ and $B = C[P]$. Note that $a, c \notin bn(C[\mathsf{out}(a,c).P])$. Let $P{\downarrow} = \nu\tilde{n}_p.F_p$. By using Lemma A.4, we obtain

- $(A{\downarrow}\ ;\ \mathsf{N}) = (\nu\tilde{n}_1.\nu\tilde{n}_p.\nu\tilde{n}_2.G[\mathsf{out}(a,c).F_p]\psi(G)^\star\ ;\ \mathsf{N})$,
- $(B{\downarrow}\ ;\ \mathsf{N}') = (\nu\tilde{n}_1.\nu\tilde{n}_p.\nu\tilde{n}_2.G[F_p]\psi(G)^\star\ ;\ \mathsf{N}')$.

Let $D_i = B{\downarrow}$. Obviously, $(D_i\ ;\ \mathsf{N}') \approxeq (B{\downarrow}\ ;\ \mathsf{N}')$. Moreover, we see that $(\mathsf{out}(a,c).F_p)\psi(G)^\star \xrightarrow{out(a,c)}_i F_p\psi(G)^\star$. By successive applications of rules $\textsc{Par}_i$ and $\textsc{Scope}_i$ we obtain that $(A{\downarrow}\ ;\ \mathsf{N}) \xrightarrow{out(a,c)}_i (D_i\ ;\ \mathsf{N}')$.

Case Out-T: $\mathsf{out}(a,M).P \xrightarrow{\nu x.out(a,x)} P \mid \{^M/_x\}$ $\qquad x \notin fv(P) \cup fv(M)$.
We have that $A = C[\mathsf{out}(a,M).P]$ and $B = C[P \mid \{^M/_x\}]$. Let $P{\downarrow} = \nu\tilde{n}_p.F_p$. By using Lemma A.4, we obtain

- $(A{\downarrow}\ ;\ \mathsf{N}) = (\nu\tilde{n}_1.\nu\tilde{n}_p.\nu\tilde{n}_2.G[\mathsf{out}(a,M).F_p]\psi(G)^\star\ ;\ \mathsf{N})$
- $(B{\downarrow}\ ;\ \mathsf{N}') = (\nu\tilde{n}_1.\nu\tilde{n}_p.\nu\tilde{n}_2.G[F_p \mid \{^M/_x\}]\psi(G)^\star\ ;\ \mathsf{N}')$.

Note that applying the substitution $\psi(G)$ is sufficient as $\psi(F_p)$ is empty and x is a fresh variable. Let $D_i = B{\downarrow}$. Obviously, we have that $(D_i\ ;\ \mathsf{N}') \approxeq (B{\downarrow}\ ;\ \mathsf{N}')$. Similarly to the previous case we show that $(A{\downarrow}\ ;\ \mathsf{N}) \xrightarrow{\nu x.out(a,x)}_i (D_i\ ;\ \mathsf{N}')$.

Case In: $\mathsf{in}(a,x).P \xrightarrow{in(a,M)} P\{^M/_x\}$.
We have that $A = C[\mathsf{in}(a,x).P]$ and $B = C[P\{^M/_x\}]$. Let $P{\downarrow} = \nu\tilde{n}_p.F_p$. By using Lemma A.4, we obtain

- $(A{\downarrow}\ ;\ \mathsf{N}) = (\nu\tilde{n}_1.\nu\tilde{n}_p.\nu\tilde{n}_2.G[\mathsf{in}(a,x).F_p]\psi(G)^\star\ ;\ \mathsf{N})$
- $(B{\downarrow}\ ;\ \mathsf{N}') = (\nu\tilde{n}_1.\nu\tilde{n}_p.\nu\tilde{n}_2.G[F_p\{^M/_x\}]\psi(G)^\star\ ;\ \mathsf{N}')$.

Let $D_i = B{\downarrow}$. Obviously, we have that $(D_i\ ;\ \mathsf{N}') \approxeq (B{\downarrow}\ ;\ \mathsf{N}')$. Moreover, we see that (we omit the naming environment for the moment)

$$(\mathsf{in}(a,x).F_p)\psi(G)^\star \xrightarrow{in(a,M\psi(G)^\star)}_i (F_p\{^M/_x\})\psi(G)^\star.$$

By application of the rule $\textsc{Par}_i$ we obtain

$$(\mathsf{in}(a,x).F_p)\psi(G)^\star \mid G\psi(G)^\star \xrightarrow{in(a,M)}_i (F_p\{^M/_x\})\psi(G)^\star \mid G\psi(G)^\star, \text{i.e.}$$

$$G[\mathsf{in}(a,x).F_p]\psi(G)^\star \xrightarrow{in(a,M)}_i G[F_p\{^M/_x\}]\psi(G)^\star.$$

Note that $G\psi(G)^\star$ is an intermediate framed process and $\psi(G\psi(G)^\star) = \psi(G)^\star$. By successive applications of SCOPE_i, we obtain that $(A\downarrow\ ;\ \mathsf{N}) \xrightarrow{in(a,M)} (B\downarrow\ ;\ \mathsf{N})$. Note that since N is compatible with $A\downarrow$ and $\alpha = in(a, M)$, we have that $\tilde{n}_1$, $\tilde{n}_2$ and $\tilde{n}_p$ are marked as bound, i.e. b, whereas names that occur in M are marked as f, and thus $\tilde{n}_1$, $\tilde{n}_2$ and $\tilde{n}$ do not occur in α.

Case OPEN-CH. Here, we assume that $A = \nu d.C[\mathsf{out}(a, d).P]$ and $B = C[P]$. Otherwise this can be obtained using structural equivalence. These structural equivalence steps are handled as explained above. Let $P\downarrow = \nu\tilde{n}_p.F_p$. By using Lemma A.4, we obtain

- $(A\downarrow\ ;\ \mathsf{N}) = (\nu d.\nu\tilde{n}_1.\nu\tilde{n}_p.\nu\tilde{n}_2.G[\mathsf{out}(a, d).F_p]\psi(G)^\star\ ;\ \mathsf{N})$,
- $(B\downarrow\ ;\ \mathsf{N}') = (\nu\tilde{n}_1.\nu\tilde{n}_p.\nu\tilde{n}_2.G[F_p]\psi(G)^\star\ ;\ \mathsf{N}')$.

Let $D_i = B\downarrow$. Obviously, we have that $(D_i\ ;\ \mathsf{N}') \approx (B\downarrow\ ;\ \mathsf{N}')$. As above we show that $(A\downarrow\ ;\ \mathsf{N}) \xrightarrow{\nu d.out(a,d)}_i (D_i\ ;\ \mathsf{N}')$. This allows us to conclude. □

B Proofs of Part II – Symbolic Calculus

We first show a useful lemma which allows us to transfer solutions of symbolic processes when we apply evaluation contexts to these processes.

Lemma B.1 *Let $(A\ ;\ \mathcal{C}\ ;\ \mathsf{N}_s)$ be a symbolic process and $C = \nu\tilde{u}.(\_ \mid D)$ an intermediate evaluation context such that $(C[A]\ ;\ C[\mathcal{C}]\ ;\ \mathsf{N}_s[bn(C[0]) \mapsto \mathsf{b}])$ is a symbolic process. We have that*

$$\theta \in Sol_{\mathsf{E}}(C[\mathcal{C}], \mathsf{N}_s[bn(C[0]) \mapsto \mathsf{b}])$$
$$\textit{iff}$$
$$(\theta\psi(C[0]))^\star \in Sol_{\mathsf{E}}(\mathcal{C}, \mathsf{N}_s) \textit{ and } bn(C[0]) \cap names(\mathrm{img}(\theta)) = \emptyset$$

Proof. Let $\mathsf{N}'_s = \mathsf{N}_s[bn(C[0]) \mapsto \mathsf{b}]$.

($\Rightarrow$) We need to consider two cases.

Case $C[\_] = \nu\tilde{n}.\_$.

We have that $\theta \in Sol_{\mathsf{E}}(\nu\tilde{n}.\mathcal{C}, \mathsf{N}'_s)$. Let $\mathcal{D}ed(\mathcal{C}) = \{\phi_i \Vdash x_i \mid 1 \le i \le \ell\}$ with $\phi_i = \nu\tilde{u}_i.\sigma_i$. We have that $(\theta\psi(\nu n.0))^\star = \theta^\star = \theta$ (as $vars(x_i\theta) \cap cv(\mathcal{C}) = \emptyset$ and $\mathrm{dom}(\theta) = cv(\mathcal{C})$). It is easy to check that θ is an E-solution of $\mathcal{C}$. As $\mathsf{N}'_s(names(\mathrm{img}(\theta))) = \mathsf{f}$ and $\mathsf{N}'_s(vars(\mathrm{img}(\theta))) = \mathsf{f}$ we also have that $\mathsf{N}_s(names(\mathrm{img}(\theta))) = \mathsf{f}$ and $\mathsf{N}_s(vars(\mathrm{img}(\theta))) = \mathsf{f}$. Hence, $\theta \in Sol_{\mathsf{E}}(\mathcal{C}, \mathsf{N}_s)$ and $\tilde{n} \cap names(\mathrm{img}(\theta)) = \emptyset$ (as $\mathsf{N}'_s(\tilde{n}) = \mathsf{b}$).

Case $C[\_] = \_ \mid D$.

We have that $\theta \in Sol_{\mathsf{E}}(\mathcal{C} \mid D, \mathsf{N}'_s)$. Let $\mathcal{D}ed(\mathcal{C}) = \{\phi_i \Vdash x_i \mid 1 \le i \le \ell\}$ with $\phi_i = \nu\tilde{u}_i.\sigma_i$ and let $\theta' = \theta\psi(D)$. We have to show that $\theta'^\star \in Sol_{\mathsf{E}}(\mathcal{C}, \mathsf{N}_s)$. For this, we need to show that:

- $vars(x_i\theta'^\star) \cap cv(\mathcal{C}) = \emptyset$. Actually we have that $\mathrm{dom}(\theta') = \mathrm{dom}(\theta) = cv(\mathcal{C}) = cv(\mathcal{C} \mid D) = \{x_1, \ldots, x_\ell\}$. Hence the result.

- $vars(x_i\theta'^\star) \cap (\mathrm{dom}(\phi_\ell) \setminus \mathrm{dom}(\phi_i)) = \emptyset$. By hypothesis we have that $vars(x_i\theta) \cap (\mathrm{dom}(\phi_\ell \cup \psi(D)) \setminus \mathrm{dom}(\phi_i \cup \psi(D))) = \emptyset$. We have that $\mathrm{dom}(\phi_\ell \cup \psi(D)) \setminus \mathrm{dom}(\phi_i \cup \psi(D)) = \mathrm{dom}(\phi_\ell) \setminus \mathrm{dom}(\phi_i)$ and $vars(x_i\theta'^\star) \subseteq vars(x_i\theta) \cup vars(\mathrm{img}(\psi(D)))$.

  As $A \mid D$ is applied we have that $vars(\mathrm{img}(\psi(D))) \cap \mathrm{dom}(\phi_\ell) = \emptyset$. Hence $vars(x_i\theta'^\star) \cap (\mathrm{dom}(\phi_\ell) \setminus \mathrm{dom}(\phi_i)) = \emptyset$.

- $names(x_i\theta'^\star) \cap \tilde{u}_i = \emptyset$. By definition of an intermediate process we have that $bn(0 \mid D) = \emptyset$. Hence, $\tilde{u}_i$ is a sequence of variables and this condition trivially holds.

- $vars(x_i\theta'^\star) \cap \tilde{u}_i = \emptyset$. As the process $A \mid D$ is applied we have that $vars(\mathrm{img}(\psi(D))) \cap \mathrm{dom}(A) = \emptyset$. As for all $x \in \tilde{u}_i$ we have that $x \in \mathrm{dom}(A)$ we conclude that $vars(x_i\theta'^\star) \cap \tilde{u}_i = \emptyset$.

- For any constraint $\mathsf{gd}(M) \in \mathcal{C}$ we need to show that $M(\theta'^\star\sigma_\ell)^\star$ is ground. By hypothesis we have that $M(\theta\sigma_\ell)^\star$ is ground. Hence, as $\mathrm{dom}(\psi(D)) \cap \mathrm{dom}(\sigma_\ell) = \emptyset$ we have that $M(\theta\psi(D))^\star = M\theta$ which allows us to conclude.

- For any constraint $M = N \in \mathcal{C}$ we need to show that $M(\theta'^\star\sigma_\ell)^\star =_{\mathsf{E}} N(\theta'^\star\sigma_\ell)^\star$. By hypothesis $M(\theta\sigma_\ell)^\star =_{\mathsf{E}} N(\theta\sigma_\ell)^\star$. As E is closed under substitution of terms for variables we conclude.

- For any constraint $M \neq N \in \mathcal{C}$ we need to show that $M(\theta'^\star\sigma_\ell)^\star \neq_{\mathsf{E}} N(\theta'^\star\sigma_\ell)^\star$. By hypothesis $M(\theta\sigma_\ell)^\star \neq_{\mathsf{E}} N(\theta\sigma_\ell)^\star$. Moreover, we have that $\mathsf{gd}(M) \in \mathcal{C}$ and $\mathsf{gd}(N) \in \mathcal{C}$. Hence, we have that $M\theta'^\star = M\theta$ and $N\theta'^\star = N\theta$ which allows us to conclude.

- $\mathsf{N}_s(names(\mathrm{img}(\theta')) \cup vars(\mathrm{img}(\theta'))) = \mathsf{f}$. By hypothesis we have that $\mathsf{N}_s(names(\mathrm{img}(\theta)) \cup vars(\mathrm{img}(\theta))) = \mathsf{f}$. Moreover, as $bn(D) = \emptyset$ we have that $\mathsf{N}'_s(names(\mathrm{img}(\psi(D)))) = \mathsf{f}$ which implies that $\mathsf{N}_s(names(\mathrm{img}(\psi(D)))) = \mathsf{f}$. Hence we conclude that $\mathsf{N}_s(names(\mathrm{img}(\theta'^\star))) = \mathsf{f}$. We similarly conclude that $\mathsf{N}_s(vars(\mathrm{img}(\theta'^\star))) = \mathsf{f}$.

In order to conclude, it remains to show that $bn(D) \cap names(\mathrm{img}(\theta)) = \emptyset$. This trivially holds since $bn(D) = \emptyset$.

($\Leftarrow$) We again consider two cases

Case $C[\_] = \nu\tilde{n}.\_$.

By hypothesis, $(\theta\psi(C[0]))^\star = \theta \in Sol_{\mathsf{E}}(\mathcal{C}, \mathsf{N}_s)$. We need to show that $\theta \in Sol_{\mathsf{E}}(\nu\tilde{n}.\mathcal{C}, \mathsf{N}'_s)$. The only tricky case is to show that $\tilde{n} \cap names(\mathrm{img}(\theta)) = \emptyset$. However this is directly implied by the additional hypothesis, i.e. $bn(C[0]) \cap names(\mathrm{img}(\theta)) = \emptyset$.

Case $C[\_] = \_ \mid D$.

By hypothesis we have that $(\theta\psi(D))^\star \in Sol_{\mathsf{E}}(\mathcal{C}, \mathsf{N}_s)$. As $C[A]$ is an extended intermediate process, we have that $bn(C[0]) = \emptyset$. Hence, $\mathsf{N}_s = \mathsf{N}'_s$. Let $\mathcal{D}ed(\mathcal{C}) = \{\phi_i \Vdash x_i \mid 1 \leq i \leq \ell\}$ with $\phi_i = \nu\tilde{u}_i.\sigma_i$. Then $\mathcal{D}ed(\mathcal{C} \mid D) = \{\nu\tilde{u}_i.\sigma_i \cup \psi(D) \Vdash x_i \mid 1 \leq i \leq \ell\}$. We have to show that $\theta \in Sol_{\mathsf{E}}(C[\mathcal{C}], \mathsf{N}'_s)$. For this, we need to show that:

- $vars(x_i\theta) \cap cv(\mathcal{C} \mid D) = \emptyset$. By hypothesis $vars(x_i(\theta\psi(D))^\star) \cap cv(\mathcal{C}) = \emptyset$. As $\mathrm{dom}(\psi(D)) \cap cv(\mathcal{C}) = \emptyset$ and $cv(\mathcal{C} \mid D) = cv(\mathcal{C})$ we conclude.

- $vars(x_i\theta) \cap (\mathrm{dom}(\phi_\ell \cup \psi(D)) \setminus \mathrm{dom}(\phi_i \cup \psi(D))) = \emptyset$. By hypothesis we have that $vars(x_i(\theta\psi(D))^\star) \cap (\mathrm{dom}(\phi_\ell) \setminus \mathrm{dom}(\phi_i)) = \emptyset$. Moreover, $\mathrm{dom}(\phi_\ell) \setminus \mathrm{dom}(\phi_i) = \mathrm{dom}(\phi_\ell \cup \psi(D)) \setminus \mathrm{dom}(\phi_i \cup \psi(D))$. Hence it is sufficient to show that $(vars(x_i\theta) \setminus vars(x_i(\theta\psi(D))^\star)) \cap (\mathrm{dom}(\phi_\ell) \setminus \mathrm{dom}(\phi_i)) = \emptyset$. We have that $(vars(x_i\theta) \setminus vars(x_i(\theta\psi(D))^\star)) \subseteq \mathrm{dom}(\psi(D))$. As $\mathrm{dom}(\psi(D)) \cap \mathrm{dom}(\phi_\ell) = \emptyset$ and $\mathrm{dom}(\phi_\ell) \supseteq \mathrm{dom}(\phi_\ell) \setminus \mathrm{dom}(\phi_i)$ we conclude.

- $names(x_i\theta) \cap \tilde{u}_i = \emptyset$. By hypothesis we have that $names(x_i(\theta\psi(D))^\star) \cap \tilde{u}_i = \emptyset$. As $names(x_i\theta) \subseteq names(x_i(\theta\psi(D))^\star)$ we conclude.

- $vars(x_i\theta) \cap \tilde{u}_i = \emptyset$. By hypothesis we have that $vars(x_i(\theta\ \psi(D))^\star) \cap \tilde{u}_i = \emptyset$. We also have that $vars(x_i\theta) \setminus vars(x_i(\theta\psi(D))^\star) \subseteq vars(\mathrm{img}(\psi(D)))$. We have that $\tilde{u}_i \subseteq \mathrm{dom}(A \mid D)$ and, as $(A \mid D)$ is applied, $vars(\mathrm{img}(\psi(D))) \cap \mathrm{dom}(A \mid D) = \emptyset$. Hence $vars(\mathrm{img}(\psi(D))) \cap \tilde{u}_i = \emptyset$ and we conclude.

- For any constraint $\mathsf{gd}(M) \in C[\mathcal{C}]$ we need to show that $M(\theta(\sigma_\ell \cup \psi(D)))^\star$ is ground. Note that $\mathsf{gd}(M) \in \mathcal{C}$. By hypothesis we have that $M((\theta\psi(D))^\star\sigma_\ell)^\star$ is ground. As $C[A]$ is applied we have that $\mathrm{dom}(\sigma_\ell) \cap vars(\mathrm{img}(\psi(D))) = \mathrm{dom}(\psi(D)) \cap vars(\mathrm{img}(\sigma_\ell)) = \emptyset$. Hence, $M((\theta\psi(D))^\star\sigma_\ell)^\star = M(\theta(\sigma_\ell \cup \psi(D)))^\star$ and we conclude. The cases for $M = N$ and $M \neq N$ are similar.

Finally, as $bn(C[0]) = \emptyset$ we obtain that $\theta \in Sol_{\mathsf{E}}(\mathcal{C} \mid D, \mathsf{N}'_s)$. This allows us to conclude the proof. □

B.1 Soundness Results

Proposition B.2 (soundness of $\equiv_s$) *Let $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ and $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$ be two well-formed symbolic processes such that $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \equiv_s (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$. Then $\mathcal{C}_A = \mathcal{C}_B$ and for all $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$, we have that $(A \,;\, \mathsf{N}) \equiv_i (B \,;\, \mathsf{N})$ where $(A \,;\, \mathsf{N})$ (resp. $(B \,;\, \mathsf{N})$) is the θ-concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ (resp. $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$).*

Proof. We show this result by induction on the proof tree witnessing the fact that $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \equiv_s (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$. First we need to consider the following base cases:

Case Par-0$_s$: $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \equiv_s (A_s \mid 0 \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$.
Trivially, $\mathcal{C}_A = \mathcal{C}_B$. Let $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$ and $(A \,;\, \mathsf{N})$ (resp. $(B \,;\, \mathsf{N})$) be the θ-concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ (resp. $(A_s \mid 0 \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$). Let σ

be the substitution corresponding to the maximal frame in $\mathcal{C}_A$. We have that $(A\ ;\ \mathsf{N}) = (A_s(\theta\sigma)^\star\ ;\ \mathsf{N}) \equiv_i (A_s(\theta\sigma)^\star \mid 0\ ;\ \mathsf{N}) = ((A_s \mid 0)(\theta\sigma)^\star\ ;\ \mathsf{N}) = (B\ ;\ \mathsf{N})$.

We can deal with the rules PAR-A$)_s$, PAR-C$_s$, and NEW-C$_s$ in a similar way.

We now consider the inductive case, i.e. application of an evaluation context. The proof tree witnessing the fact that $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s) \equiv_s (B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$ ends with an application of the following inference rule.

$$\frac{(A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}'_s) \equiv_s (B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'_s)}{(C[A'_s]\ ;\ C[\mathcal{C}'_A]\ ;\ \mathsf{N}_s) \equiv_s (C[B'_s]\ ;\ C[\mathcal{C}'_B]\ ;\ \mathsf{N}_s)}$$

We have that $\mathsf{N}_s = \mathsf{N}'_s[bn(C[0]) \mapsto \mathsf{b}]$. As $\mathcal{C}'_A = \mathcal{C}'_B$ we directly have that $C[\mathcal{C}'_A] = C[\mathcal{C}'_B]$. Let $\theta \in Sol_{\mathsf{E}}(C[\mathcal{C}'_A]\ ;\ \mathsf{N}_s)$ and $(A\ ;\ \mathsf{N})$ (resp. $(B\ ;\ \mathsf{N})$) be the θ-concretization of $(C[A'_s]\ ;\ C[\mathcal{C}'_A]\ ;\ \mathsf{N}_s)$ (resp. $(C[B'_s]\ ;\ C[\mathcal{C}'_B]\ ;\ \mathsf{N}_s)$). Let σ be the substitution corresponding to the maximal frame in $\mathcal{C}_A = C[\mathcal{C}'_A]$. We have to show that $(A\ ;\ \mathsf{N}) \equiv_i (B\ ;\ \mathsf{N})$, i.e. $(C(\theta\sigma)^\star[A'_s(\theta\sigma)^\star]\ ;\ \mathsf{N}) \equiv_i (C(\theta\sigma)^\star[B'_s(\theta\sigma)^\star]\ ;\ \mathsf{N})$. Since $\equiv_i$ is closed under application of evaluation context, it is sufficient to show that $(A'_s(\theta\sigma)^\star\ ;\ \mathsf{N}') \equiv_i (B'_s(\theta\sigma)^\star\ ;\ \mathsf{N}')$ where $\mathsf{N}' = \mathsf{N}'_s|_{\mathcal{N}\cup\mathcal{X}}$.

Let $\theta' = (\theta\psi(C[0]))^\star$. By Lemma B.1 we have that $\theta' \in Sol_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}'_s)$ and $(A'\ ;\ \mathsf{N}') \equiv_i (B'\ ;\ \mathsf{N}')$ where $(A'\ ;\ \mathsf{N}')$ (resp. $(B'\ ;\ \mathsf{N}')$) is the θ'-concretization of $(A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}'_s)$ (resp. $(B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'_s)$). We have that $A' = A'_s(\theta'\sigma')^\star$ and $B' = B'_s(\theta'\sigma')^\star$ where σ' is the maximal frame of $\mathcal{C}'_A$. This allows us to conclude since $(\theta'\sigma')^\star = ((\theta\psi(C[0]))^\star\sigma')^\star = (\theta(\psi(C[0])\sigma'))^\star = (\theta\sigma)^\star$. □

Proposition B.3 (soundness of $\rightarrow_s$) *Let $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ and $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$ be two well-formed symbolic processes such that $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s) \rightarrow_s (B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$. Let $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B\ ;\ \mathsf{N}_s)$. We have that $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A\ ;\ \mathsf{N}_s)$ and $(A\ ;\ \mathsf{N}) \rightarrow_i (B\ ;\ \mathsf{N})$ where $(A\ ;\ \mathsf{N})$ (resp. $(B\ ;\ \mathsf{N})$) is the θ-concretization of $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ (resp. $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$).*

Proof. The proof is done by induction on the proof witnessing $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s) \rightarrow_s (B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}_s)$. We first consider the three base cases.

Case COMM$_s$. We have that $A_s = \mathsf{out}(u, M).P_s \mid \mathsf{in}(v, x).Q_s$, $B_s = P_s \mid Q_s\{^M/_x\}$ and $\mathcal{C}_B = \mathcal{C}_A \cup \{u = v, \mathsf{gd}(u), \mathsf{gd}(v)\}$. Let $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A \cup \{u = v\ ,\ \mathsf{gd}(u)\ ,\ \mathsf{gd}(v)\}\ ;\ \mathsf{N}_s)$. We also have that $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A\ ;\ \mathsf{N}_s)$. Let σ_A (resp. σ_B) be the substitution corresponding to the maximal frame of $\mathcal{C}_A$ (resp. $\mathcal{C}_B$). Trivially, we have that $\sigma_A = \sigma_B$. Let $\rho = (\theta\sigma_A)^\star$.

$$\begin{array}{rcll}
(A\ ;\ \mathsf{N}) & = & (A_s\rho\ ;\ \mathsf{N}) & \\
& = & (\mathsf{out}(u\rho, M\rho).P_s\rho \mid \mathsf{in}(v\rho, x).Q_s\rho\ ;\ \mathsf{N}) & \text{as } x \notin \mathrm{dom}(\rho) \\
& = & (\mathsf{out}(u\rho, M\rho).P_s\rho \mid \mathsf{in}(u\rho, x).Q_s\rho\ ;\ \mathsf{N}) & \text{as } \theta \in Sol_{\mathsf{E}}(\mathcal{C}_B\ ;\ \mathsf{N}_s) \\
& & & u, v \text{ are of channel type} \\
& \rightarrow_i & (P_s\rho \mid Q_s\rho\{^{M\rho}/_x\}\ ;\ \mathsf{N}) & \text{as } u\rho \text{ is a channel name} \\
& & & \text{as } \mathsf{gd}(u) \in \mathcal{C}_B \\
& = & ((P_s \mid Q_s\{^M/_x\})\rho\ ;\ \mathsf{N}) & \\
& = & (B_s\rho\ ;\ \mathsf{N}) & \\
& = & (B\ ;\ \mathsf{N}) & \text{as } \sigma_A = \sigma_B
\end{array}$$

Again, the rules $\textsc{Then}_s$ and $\textsc{Else}_s$ are similar to the previous case. We now consider the two inductive cases.

The case of the structural equivalence rule is straightforward.
Case Application of an evaluation context

The proof witnessing the fact that $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \rightarrow_s (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$ ends with an application of the following inference rule.

$$\frac{(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s) \rightarrow_s (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)}{(C[A'_s] \,;\, C[\mathcal{C}'_A] \,;\, \mathsf{N}_s) \rightarrow_s (C[B'_s] \,;\, C[\mathcal{C}'_B] \,;\, \mathsf{N}_s)}$$

We have that $\mathsf{N}_s = \mathsf{N}'_s[bn(C[0]) \mapsto \mathsf{b}]$. Let $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}_s)$ and $(A \,;\, \mathsf{N})$ (resp. $(B \,;\, \mathsf{N})$) be the θ-concretization of $(C[A'_s] \,;\, C[\mathcal{C}'_A] \,;\, \mathsf{N}_s)$ (resp. $(C[B'_s] \,;\, C[\mathcal{C}'_B] \,;\, \mathsf{N}_s)$). Let σ'_A (resp. σ'_B) be the substitution corresponding to the maximal frame in $\mathcal{C}'_A$ and σ_A (resp. σ_B) be the substitution corresponding to the maximal frame in $\mathcal{C}_A = C[\mathcal{C}'_A]$ (resp. $\mathcal{C}_B = C[\mathcal{C}'_B]$). Note that since $\rightarrow_s$ does never add deduction constraints we have that $\sigma'_A = \sigma'_B$ and hence $\sigma_A = \sigma_B$. We have to show that $(A \,;\, \mathsf{N}) \rightarrow_i (B \,;\, \mathsf{N})$, i.e. $(C(\theta\sigma_A)^\star[A'_s(\theta\sigma_A)^\star] \,;\, \mathsf{N}) \rightarrow_i (C(\theta\sigma_B)^\star[B'_s(\theta\sigma_B)^\star] \,;\, \mathsf{N})$. Since $\rightarrow_i$ is closed under application of evaluation context, it is sufficient to show that $(A'_s(\theta\sigma_A)^\star \,;\, \mathsf{N}') \rightarrow_i (B'_s(\theta\sigma_B)^\star \,;\, \mathsf{N}')$ where $\mathsf{N}' = \mathsf{N}'_s|_{\mathcal{N}\cup\mathcal{X}}$.

Let $\theta' = (\theta\psi(C[0]))^\star$. By Lemma B.1 we have that $\theta' \in Sol_{\mathsf{E}}(\mathcal{C}'_B \,;\, \mathsf{N}'_s)$ and hence by induction hypothesis, we deduce that $\theta' \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s)$ and $(A' \,;\, \mathsf{N}) \rightarrow_i (B' \,;\, \mathsf{N})$ where $(A' \,;\, \mathsf{N}')$ (resp. $(B' \,;\, \mathsf{N}')$) is the θ'-concretization of $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$ (resp. $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$). Hence, by Lemma B.1 we deduce that $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s) = Sol_{\mathsf{E}}(C[\mathcal{C}'_A] \,;\, \mathsf{N}_s)$. We have that $A' = A'_s(\theta'\sigma'_A)^\star$ and $B' = B'_s(\theta'\sigma'_B)^\star$. This allows us to conclude since $(\theta'\sigma'_A)^\star = (\theta\sigma_A)^\star$ and $(\theta'\sigma'_B)^\star = (\theta\sigma_B)^\star$. □

Proposition 8.2 (soundness of $\xrightarrow{\alpha}_s$) *Let $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ and $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}'_s)$ be two well-formed symbolic processes such that $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \xrightarrow{\alpha_s}_s (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}'_s)$. Let $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}'_s)$ and $\theta_A = \theta_B|_{cv(\mathcal{C}_A)}$. We have that $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$ and $(A \,;\, \mathsf{N}) \xrightarrow{\alpha_s\theta_B}_i (B \,;\, \mathsf{N}')$, where $(A \,;\, \mathsf{N})$ and $(B \,;\, \mathsf{N}')$ are respectively the θ_A-concretization and the θ_B-concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ and $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}'_s)$.*

Proof. The proof is done by induction on the proof tree witnessing the following reduction step $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \xrightarrow{\alpha_s}_s (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}'_s)$. We first consider the three base cases.

Case $\textsc{In}_s$. We have that $A_s = \mathsf{in}(u, x).P_s$, $B_s = P_s\{^y/_x\}$, $\alpha_s = in(u, y)$ for some $y \in \mathcal{Y}$ such that $\mathsf{N}_s(y) = \mathsf{n}$ and $\mathcal{C}_B = \mathcal{C}_A \cup \{0 \Vdash y, \mathsf{gd}(u)\})$. Moreover, we have that $\mathsf{N}'_s = \mathsf{N}_s[y \mapsto \mathsf{c}]$. Note that we have that $\mathsf{N} = \mathsf{N}'$. Let $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}'_s)$ and $\theta_A = \theta_B|_{cv(\mathcal{C}_A)}$. As $\mathcal{C}_A \subset \mathcal{C}_B$ we have that $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$. Let σ_A (resp. σ_B) be the substitution corresponding to the maximal frame of $\mathcal{C}_A$ (resp. $\mathcal{C}_B$). Trivially, we have that $\mathrm{dom}(\sigma_A) = \mathrm{dom}(\sigma_B) = \emptyset$ since $\phi(P_s) = 0$ and $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ is well-formed.

$$
\begin{array}{lcll}
(A\ ;\ \mathsf{N}) & = & (A_s\theta_A\ ;\ \mathsf{N}) & \\
 & = & (\mathsf{in}(u\theta_A, x).P_s\theta_A\ ;\ \mathsf{N}) & \text{as } x \notin dom(\theta_A) \\
 & \xrightarrow{in(u,y)\theta_B}_i & (P_s\theta_A\{^{y\theta_B}/_x\}\ ;\ \mathsf{N}') & \text{as } u\theta_B \in \mathcal{N}_{ch},\ \mathsf{N}(fv(y\theta_B) \cup fn(y\theta_B)) = \mathsf{f} \\
 & = & (P_s\theta_A\{^{y}/_x\}\{^{y\theta_B}/_y\}\ ;\ \mathsf{N}') & \\
 & = & (B_s\theta_B\ ;\ \mathsf{N}') & \text{as } \theta_B = \theta_A \cup \{y \mapsto y\theta_B\} \\
 & = & (B\ ;\ \mathsf{N}') &
\end{array}
$$

Case OUT-CH$_s$. This case is similar to the previous one.
Case OUT-T$_s$. We have that $A_s = \mathsf{out}(u, M).P_s$, $B_s = P_s \mid \{^M/_x\}$, $\alpha_s = \nu x.out(u, x)$ where $x \in \mathcal{X}_b$ and $\mathsf{N}_s(x) = \mathsf{n}$. We have also that $\mathcal{C}_B = \nu x.\mathcal{C}_A \cup \{\mathsf{gd}(u)\}$ and $\mathsf{N}'_s = \mathsf{N}_s[x \mapsto \mathsf{f}]$. Let $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}_B\ ;\ \mathsf{N}'_s)$ and $\theta_A = \theta_B|_{cv(\mathcal{C}_A)}$, i.e. $\theta_B = \theta_A$. As $\nu x.\mathcal{C}_A \subset \mathcal{C}_B$ we have that $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_A\ ;\ \mathsf{N}_s)$. Let σ_A (resp. σ_B) be the substitution corresponding to the maximal frame of $\mathcal{C}_A$ (resp. $\mathcal{C}_B$). Trivially, we have that $\mathrm{dom}(\sigma_A) = \mathrm{dom}(\sigma_B) = \emptyset$ since $\phi(P_s) = 0$ and $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ is well-formed.

$$
\begin{array}{lcll}
(A\ ;\ \mathsf{N}) & = & (A_s\theta_A\ ;\ \mathsf{N}) & \\
 & = & (\mathsf{out}(u\theta_A, M\theta_A).P_s\theta_A\ ;\ \mathsf{N}) & \\
 & \xrightarrow{\nu x.out(u,x)\theta_B}_i & (P_s\theta_A \mid \{^{M\theta_A}/_x\}\ ;\ \mathsf{N}') & \text{as } x \notin \mathrm{dom}(\theta_B) \\
 & = & (P_s\theta_B \mid \{^{M\theta_B}/_x\}\ ;\ \mathsf{N}') & \text{as } \theta_A = \theta_B \\
 & = & (B_s\theta_B\ ;\ \mathsf{N}) & \\
 & = & (B\ ;\ \mathsf{N}) &
\end{array}
$$

Moreover, as $\theta_B \in Sol_{\mathsf{E}}(\nu x.\mathcal{C}_A)$ we have that $x \notin \mathrm{img}(\theta_B)$ and hence, x occurs only once in B.

We now consider the inductive cases.

Case OPEN-CH$_s$.

$$
\frac{(A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}''_s) \xrightarrow{out(u,c)}_s (B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'''_s) \quad u \neq c,\ \mathsf{N}''_s(d) = \mathsf{n},\ d \in \mathcal{N}_{ch}}{(\nu c.A'_s\ ;\ \nu c.\mathcal{C}'_A\ ;\ \mathsf{N}_s) \xrightarrow{\nu d.out(u,d)}_s (B'_s\{^d/_c\}\ ;\ \nu d.(\mathcal{C}'_B\{^d/_c\})\ ;\ \mathsf{N}'_s)}
$$

We have that $A_s = \nu c.A'_s$, $B_s = B'_s\{^d/_c\}$, $\alpha_s = \nu d.out(u, d)$, $\mathcal{C}_A = \nu c.\mathcal{C}'_A$ and $\mathcal{C}_B = \nu d.\mathcal{C}'_B\{^d/_c\}$. Moreover, we have that

- $\mathsf{N}_s = \mathsf{N}''_s[c \mapsto \mathsf{b}]$, and
- $\mathsf{N}'_s = \mathsf{N}'''_s[c \mapsto \mathsf{b},\ d \mapsto \mathsf{f}]$.

Let $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}_B\ ;\ \mathsf{N}'_s)$. We also have that $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}'_B\ ;\ \mathsf{N}'''_s)$ and $c, d \notin names(\mathrm{img}(\theta_B))$ and $u\theta_B \neq c$. Let $\theta_A = \theta_B|_{cv(\mathcal{C}_A)}$, i.e. $\theta_B = \theta_A$. By induction hypothesis we deduce that $(A'\ ;\ \mathsf{N}'') \xrightarrow{out(u,c)\theta_B}_i (B'\ ;\ \mathsf{N}''')$ where $(A'\ ;\ \mathsf{N}'')$ (resp. $(B'\ ;\ \mathsf{N}''')$) are the θ_B-concretization of $(A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}''_s)$ (resp. $(B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'''_s)$) and $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}''_s)$. As $c \notin names(\mathrm{img}(\theta_B))$, we also have that

$c \notin names(\text{img}(\theta_A))$ and hence $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$. Since $(A' \,;\, \mathsf{N}'') \xrightarrow{out(u,c)\theta_B}_i (B' \,;\, \mathsf{N}''')$, we deduce that $(\nu c.A' \,;\, \mathsf{N}) \xrightarrow{\nu d.out(u\theta_B,d)}_i (B'\{^d/_c\} \,;\, \mathsf{N}''')$. Note that $c \neq u\theta_B$ and $d \in \mathcal{N}_{ch}$ and $\mathsf{N}''(d) = \mathsf{n}$.

Case $\textsc{Scope}_s$.

$$\frac{(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}''_s) \xrightarrow{\alpha}_s (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'''_s) \quad n \text{ does not occur in } \alpha}{(\nu n.A'_s \,;\, \nu n.\mathcal{C}'_A \,;\, \mathsf{N}_s) \xrightarrow{\alpha}_s (\nu n.B'_s \,;\, \nu n.\mathcal{C}'_B \,;\, \mathsf{N}'_s)}$$

We have that $A_s = \nu n.A'_s$, $B_s = \nu n.B'_s$, $\mathcal{C}_A = \nu n.\mathcal{C}'_A$ and $\mathcal{C}_B = \nu n.\mathcal{C}'_B$. Moreover, we have that $\mathsf{N}_s = \mathsf{N}''_s[n \mapsto \mathsf{b}]$ and $\mathsf{N}'_s = \mathsf{N}'''_s[n \mapsto \mathsf{b}]$. Let $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}'_s)$. We have that $n \notin names(\text{img}(\theta_B))$. Let $\theta'_B = \theta_B$. By Lemma B.1 we have that $\theta'_B \in Sol_{\mathsf{E}}(\mathcal{C}'_B \,;\, \mathsf{N}'''_s)$. Let $\theta'_A = \theta'_B|_{cv(\mathcal{C}'_A)}$. By induction hypothesis, we have that $(A' \,;\, \mathsf{N}'') \xrightarrow{\alpha\theta'_B}_i (B' \,;\, \mathsf{N}''')$ where $(A' \,;\, \mathsf{N}'')$ and $(B' \,;\, \mathsf{N}''')$ are respectively the θ'_A and the θ'_B-concretization of $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}''_s)$ and $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'''_s)$. As $n \notin names(\text{img}(\theta_B))$, n does not occur in $\alpha\theta'_B$ and $\theta_A = \theta'_A \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$ by Lemma B.1. Since $(A' \,;\, \mathsf{N}'') \xrightarrow{\alpha\theta'_B}_i (B' \,;\, \mathsf{N}''')$, $\theta_B = \theta'_B$ and n does not occur in $\alpha\theta_B$, we deduce that $(\nu n.A' \,;\, \mathsf{N}) \xrightarrow{\alpha\theta_B}_i (\nu n.B' \,;\, \mathsf{N}')$

Case $\textsc{Par}_s$.

$$\frac{(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s) \xrightarrow{\alpha}_s (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)}{(A'_s \mid D_s \,;\, \mathcal{C}'_A \mid \psi(D_s) \,;\, \mathsf{N}_s) \xrightarrow{\alpha}_s (B'_s \mid D_s \,;\, \mathcal{C}'_B \mid \psi(D_s) \,;\, \mathsf{N}'_s)}$$

We have that $A_s = A'_s \mid D_s$, $B_s = B'_s \mid D_s$, $\mathcal{C}_A = \mathcal{C}'_A \mid \psi(D_s)$ and $\mathcal{C}_B = \mathcal{C}'_B \mid \psi(D_s)$. Let $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}'_s)$. Then, by Lemma B.1 we also have that $\theta'_B = (\theta_B\psi(D_s))^\star \in Sol_{\mathsf{E}}(\mathcal{C}'_B \,;\, \mathsf{N}'_s)$. Let $\theta'_A = \theta'_B|_{cv(\mathcal{C}'_A)}$. By induction hypothesis we have that $\theta'_A \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}_s)$ and $(A' \,;\, \mathsf{N}) \xrightarrow{\alpha\theta_B\psi(D_s)}_i (B' \,;\, \mathsf{N}')$ where $(A' \,;\, \mathsf{N})$ and $(B' \,;\, \mathsf{N}')$ are respectively the θ'_A and the θ'_B concretization of $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)$ and $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$.

Let $\theta_A = \theta_B|_{cv(\mathcal{C}_A)}$. We have $\theta'_A = \theta_B\psi(D_s)|_{cv(\mathcal{C}_A)}$ and $\theta'_A \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}_s)$. Hence by Lemma B.1 we have that $\theta_B|_{cv(\mathcal{C}_A)} \in Sol_{\mathsf{E}}(\mathcal{C}'_A \mid \psi(D_s) \,;\, \mathsf{N}_s)$, i.e. $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$. Let σ_A (resp. σ_B, σ'_A and σ'_B) be the substitution corresponding to the maximal frame of $\mathcal{C}_A$ (resp. $\mathcal{C}_B$, $\mathcal{C}'_A$ and $\mathcal{C}'_B$), We also have that $\sigma_A = \sigma'_A \cup \psi(D_s)$ and $\sigma_B = \sigma'_B \cup \psi(D_s)$. As $\mathsf{N}'_s(\text{dom}(\psi(D_s))) = \mathsf{f}$ and $\mathsf{N}'_s(fv(\alpha)) = \mathsf{c}$, we have that $\alpha\theta_B\psi(D_s) = \alpha\theta_B$. Hence, we have that $(A \,;\, \mathsf{N}) \xrightarrow{\alpha\theta_B}_i (B \,;\, \mathsf{N}')$ where $(A \,;\, \mathsf{N})$ and $(B \,;\, \mathsf{N}')$ are respectively the θ_A and the θ_B concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ and $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}'_s)$.

Case $\textsc{Struct}_s$. This case is straightforward by relying on Proposition B.2.

□

B.2 Completeness Results

Proposition B.4 (completeness of $\equiv_s$) *Let $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ be a well-formed symbolic process and $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A, \mathsf{N}_s)$. Let $(A \,;\, \mathsf{N})$ be the θ-concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ and B be a process such that $(A \,;\, \mathsf{N}) \equiv_i (B \,;\, \mathsf{N})$. Then there exists a well-formed symbolic process $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$ such that:*

1. $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \equiv_s (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$,
2. $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}_s)$, *and*
3. $(B \,;\, \mathsf{N})$ *is the θ-concretization of* $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$.

Proof. We show this result by induction on the proof tree witnessing the fact that $(A, \mathsf{N}) \equiv_i (B, \mathsf{N})$. First we need to consider the following base cases:

Case PAR-0_i: $(D, \mathsf{N}) \equiv_i (D \mid 0, \mathsf{N})$. In such a case, we have that $A = D$ and $B = D \mid 0$. Moreover, since $(A \,;\, \mathsf{N})$ is the θ-concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ we have that $A = A_s(\theta\sigma_A)^\star$. Hence, we know that $A_s = D_s$ for some process D_s such that $D_s(\theta\sigma_A)^\star = D$. Let $B_s = D_s \mid 0$ and $\mathcal{C}_B = \mathcal{C}_A$. The symbolic process $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$ is well-formed. Moreover, we have

1. $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \equiv_s (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$,
2. $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}_s)$,
3. $B_s(\theta\sigma_B)^\star = (D_s \mid 0)(\theta\sigma_A)^\star = D \mid 0 = B$, i.e., $(B \,;\, \mathsf{N})$ is the θ-concretization of $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$.

Symmetrically, we have to consider the case where $A = D \mid 0$ and $B = D$. We know that $A = A_s(\theta\sigma_A)^\star$ and we deduce that $A_s = D_s|0$ for some process D_s such that $D_s(\theta\sigma_A)^\star = D$. Let $B_s = D_s$ and $\mathcal{C}_B = \mathcal{C}_A$. We easily conclude.

We can deal with the rules PAR-A, PAR-C and NEW-C in a similar way.

Now, we show the inductive case, i.e. application of an evaluation context. In such a case, we have that the proof tree witnessing the fact that $(A \,;\, \mathsf{N}) \equiv_i (B \,;\, \mathsf{N})$ ends with an application of the following inference rule.

$$\frac{(A' \,;\, \mathsf{N}') \equiv_i (B' \,;\, \mathsf{N}')}{(C[A'] \,;\, \mathsf{N}) \equiv_i (C[B'] \,;\, \mathsf{N})}$$

Moreover, since $(A \,;\, \mathsf{N})$ is the θ-concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ we have that $A_s(\theta\sigma_A)^\star = C[A']$. Hence, we deduce that $A_s = C_s[A'_s]$ for some evaluation context C_s and some process A'_s such that $C_s(\theta\sigma_A)^\star = C$ and $A'_s(\theta\sigma_A)^\star = A'$. Since $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ is well-formed, we have also that $\mathcal{C}_A = C_s[\mathcal{C}'_A]$ for some constraint system $\mathcal{C}'_A$. Let

$$\mathsf{N}'_s(u) = \begin{cases} \mathsf{N}'(u) & \text{if } u \in \mathcal{N} \cup \mathcal{X} \\ \mathsf{N}_s(u) & \text{if } u \in \mathcal{Y}. \end{cases}$$

Let $\theta' = (\theta\psi(C_s))^\star$. By Lemma B.1 we have that $\theta' \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s)$. We can apply our induction hypothesis on $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$ and $(A' \,;\, \mathsf{N}') \equiv_i (B' \,;\, \mathsf{N}')$.

We deduce that there exists a well-formed symbolic process $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$ such that $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s) \equiv_s (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$, $\theta' \in Sol_{\mathsf{E}}(\mathcal{C}'_B \,;\, \mathsf{N}'_s)$ and $B'_s(\theta'\sigma'_B)^\star = B'$. Let $B_s = C_s[B'_s]$ and $\mathcal{C}_B = C_s[\mathcal{C}'_B]$. We have that

1. $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \equiv (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$,
2. $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N_s})$ (by Lemma B.1),
3. $B_s(\theta\sigma_B)^\star = C_s(\theta\sigma_B)^\star[B'_s(\theta\sigma_B)^\star] = C[B'_s(\theta'\sigma'_B)^\star] = C[B'] = B$, i.e., $(B \,;\, \mathsf{N})$ is the θ-concretization of $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$.

This concludes our proof. □

Proposition B.5 (completeness of $\rightarrow_s$) *Let $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ be a well-formed symbolic process and $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$. Let $(A \,;\, \mathsf{N})$ be the θ-concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ and (A', N) be an intermediate process such that $(A \,;\, \mathsf{N}) \rightarrow_i (A' \,;\, \mathsf{N})$. Then there exists a well-formed symbolic process $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)$ such that:*

1. $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \rightarrow_s (A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)$,
2. $\theta \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}_s)$,
3. $(A' \,;\, \mathsf{N})$ *is the θ-concretization of* $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)$.

Proof. We show this result by induction on the proof tree witnessing the fact that $(A \,;\, \mathsf{N}) \rightarrow_i (A' \,;\, \mathsf{N})$. First, we need to consider the three base cases, i.e. the rules Then, Else and Comm. We detail the case of the rule Else, the two other ones are very similar.
Case Else: $(\text{if } M = N \text{ then } P \text{ else } Q \,;\, \mathsf{N}) \rightarrow_i (Q \,;\, \mathsf{N})$ with M, N ground terms such that $M \neq_{\mathsf{E}} N$. In such a case, we have that

- $A = \text{if } M = N \text{ then } P \text{ else } Q$ for some ground terms M, N such that $M \neq_{\mathsf{E}} N$ and some processes P and Q, and
- $A' = Q$.

Moreover, since $(A \,;\, \mathsf{N})$ is the θ-concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ we have that $A = A_s(\theta\sigma_A)^\star$. Hence, we deduce that

- $A_s = \text{if } M_s = N_s \text{ then } P_s \text{ else } Q_s$ for some terms M_s, N_s, some processes P_s and Q_s such that
- $M_s(\theta\sigma_A)^\star = M$, $N_s(\theta\sigma_A)^\star = N$, $P_s(\theta\sigma_A)^\star = P$ and $Q_s(\theta\sigma_A)^\star = Q$.

Let $A'_s = Q_s$ and $\mathcal{C}'_A = \mathcal{C}_A \cup \{M_s \neq N_s \,,\, \mathsf{gd}(M_s) \,,\, \mathsf{gd}(N_s)\}$. The symbolic process $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)$ is well-formed. Moreover, we have

1. $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \rightarrow_s (Q_s \,;\, \mathcal{C}_A \cup \{M_s \neq N_s \,,\, \mathsf{gd}(M_s) \,,\, \mathsf{gd}(N_s)\} \,;\, \mathsf{N}_s)$
2. $\theta \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}_s)$. Indeed, by hypothesis, we know that $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$. We know also that $M_s(\theta\sigma_A)^\star$ and $N_s(\theta\sigma_A)^\star$ are ground terms which are not equal modulo E.

3. $A'_s(\theta\sigma_A)^\star = Q_s(\theta\sigma_A)^\star = Q = A'$, i.e., $(A' \,;\, \mathsf{N})$ is the θ-concretization of $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}_s)$.

Now, we show the inductive cases. In case of the structural equivalence inductive rule, we easily conclude by induction and thanks to Proposition B.4.

Case Application of an evaluation context. In such a case, we have that the tree witnessing the fact that $(A \,;\, \mathsf{N}) \rightarrow_i (B \,;\, \mathsf{N})$ ends with an application of the following inference rule.

$$\frac{(A' \,;\, \mathsf{N}') \rightarrow_i (B' \,;\, \mathsf{N}')}{(C[A'] \,;\, \mathsf{N}) \rightarrow_i (C[B'] \,;\, \mathsf{N})}$$

Moreover, since $(A \,;\, \mathsf{N})$ is the θ-concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ we have that $A_s(\theta\sigma_A)^\star = C[A']$. Hence, we deduce that $A_s = C_s[A'_s]$ for some evaluation context C_s and some process A'_s such that $C_s(\theta\sigma_A)^\star = C$ and $A'_s(\theta\sigma_A)^\star = A'$. Since $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ is well-formed, we have also that $\mathcal{C}_A = C_s[\mathcal{C}'_A]$ for some constraint system $\mathcal{C}'_A$. Let

$$\mathsf{N}'_s(u) = \begin{cases} \mathsf{N}'(u) & \text{if } u \in \mathcal{N} \cup \mathcal{X} \\ \mathsf{N}_s(u) & \text{if } u \in \mathcal{Y} \end{cases}$$

Let $\theta' = (\theta\phi(C_s))^\star$. By Lemma B.1 we have that $\theta' \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s)$.

We can apply our induction hypothesis on $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$ and $(A' \,;\, \mathsf{N}') \rightarrow_i (B' \,;\, \mathsf{N}')$. We deduce that there exists a well-formed symbolic process $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$ such that $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s) \rightarrow_s (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$, $\theta' \in Sol_{\mathsf{E}}(\mathcal{C}'_B \,;\, \mathsf{N}'_s)$ and $(B' \,;\, \mathsf{N}')$ is the θ-concretization of $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$. Let $B_s = C_s[B'_s]$ and $\mathcal{C}_B = C_s[\mathcal{C}'_B]$. We have that

1. $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \rightarrow_s (B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$,
2. $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}_s)$ (by Lemma B.1),
3. $B_s(\theta\sigma_B)^\star = C_s(\theta\sigma_B)^\star[B'_s(\theta\sigma_B)^\star] = C[B'_s(\theta'\sigma'_B)^\star] = C[B'] = B$ i.e., $(B \,;\, \mathsf{N})$ is the θ-concretization of $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$.

This concludes our proof. □

Proposition 8.4 (completeness of $\xrightarrow{\alpha}_s$) *Let $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ be a well-formed symbolic process and $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_A \,;\, \mathsf{N}_s)$. Let $(A \,;\, \mathsf{N})$ be the θ_A-concretization of $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ and $(A' \,;\, \mathsf{N}')$ be an intermediate process such that $(A \,;\, \mathsf{N}) \xrightarrow{\alpha}_i (A' \,;\, \mathsf{N}')$. Then there exists a well-formed symbolic process $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$ and a substitution θ'_A such that:*

1. $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \xrightarrow{\alpha_s}_s (A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$,
2. $\theta'_A \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s)$ *and* $\theta'_A|_{cv(\mathcal{C}_A)} = \theta_A$,
3. $(A' \,;\, \mathsf{N}')$ *is the* θ'_A*-concretization of* $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$, *and*
4. $\alpha_s\theta'_A = \alpha$.

Proof. We show this result by induction on the tree witnessing the fact that $(A\ ;\ \mathsf{N}) \xrightarrow{\alpha}_i (A'\ ;\ \mathsf{N}')$. First, we need to consider the following base cases:

Case IN$_i$: $(\mathsf{in}(a,x).P\ ;\ \mathsf{N}) \xrightarrow{in(a,M)}_i (P\{^M/_x\}\ ;\ \mathsf{N})$. In such a case, we have that

- $A = \mathsf{in}(a,x).P$ for some channel name a, some variable x and some P,
- $A' = P\{^M/_x\}$,
- $\alpha = \mathsf{in}(a,M)$ for some term M, and
- $\mathsf{N}(fn(M) \cup fv(M)) = \mathsf{f}$.

Since $(A\ ;\ \mathsf{N})$ is the θ-concretization of $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ we have that $A_s(\theta\sigma_A)^\star = A$. Hence, we know that

- $A_s = \mathsf{in}(u,x).P_s$ for some metavariable u and some process P_s such that
- $u(\theta\sigma_A)^\star = a$ and $P_s(\theta\sigma_A)^\star = P$.

We have that u is either a channel name or a constraint variable of channel type since $u(\theta\sigma_A)^\star = a$ and a is a channel name.

Let $y \in \mathcal{Y}$ having the same type than M and such that $\mathsf{N}_s(y) = \mathsf{n}$. Let $A'_s = P_s\{^y/_x\}$, $\mathcal{C}'_A = \mathcal{C}_A \cup \{0 \Vdash y\ ,\ \mathsf{gd}(u)\}$, $\alpha_s = \mathsf{in}(u,y)$, $\theta' = \theta \cup \{y \mapsto M\}$ and $\mathsf{N}'_s = \mathsf{N}_s[y \mapsto \mathsf{c}]$. The symbolic process $(A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}'_s)$ is well-formed, and we have:

1. $$\begin{aligned}(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s) &= (\mathsf{in}(u,x).P_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)\\ &\xrightarrow{\alpha_s}_s (P_s\{^y/_x\}\ ;\ \mathcal{C}_A \cup \{0 \Vdash y\ ,\ \mathsf{gd}(u)\}\ ;\ \mathsf{N}'_s)\\ &= (A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}'s)\end{aligned}$$
2. We know that $\theta \in Sol_{\mathsf{E}}(\mathcal{C}_A\ ;\ \mathsf{N}_s)$. It remains to check that $\theta' \in Sol_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}'_s)$, i.e. θ' satisfies the constraints $\Vdash y$ and $\mathsf{gd}(u)$. This is clearly true due to the fact that $\mathsf{N}_s(fn(M) \cup fv(M)) = \mathsf{f}$ and $u(\theta\sigma_A)^\star = a$. Lastly, by definition of θ', we have that $\theta'|_{cv(\mathcal{C}_A)} = \theta$.
3. We have $A'_s(\theta'\sigma'_A)^\star = P_s\{^y/_x\}(\theta\sigma_A)^\star[y \mapsto M] = P\{^M/_x\} = A'$ since $\mathrm{dom}(\sigma_A) = \mathrm{dom}(\sigma'_A) = \emptyset$, i.e., $(A'\ ;\ \mathsf{N}')$ is the θ'-concretization of $(A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}'_s)$, and
4. $\alpha_s\theta' = \mathsf{in}(u,y)\theta' = \mathsf{in}(u\theta', y\theta') = \mathsf{in}(a,M) = \alpha$.

We can deal with the rules OUT-CH$_i$ and OUT-T$_i$ in a rather similar way.

We now consider the inductive cases.

Case OPEN-CH$_i$: In such a case, we have that the tree witnessing the fact that $(A\ ;\ \mathsf{N}) \xrightarrow{\alpha}_i (A'\ ;\ \mathsf{N}')$ ends with an application of the following inference rule

$$\frac{(B\ ;\ \mathsf{N}'') \xrightarrow{out(a,c)}_i (B'\ ;\ \mathsf{N}''') \quad c \neq a,\ \mathsf{N}''(d) = \mathsf{n} \text{ and } d \in \mathcal{N}_{ch}}{(\nu c.B\ ;\ \mathsf{N}) \xrightarrow{\nu d.out(a,d)}_i (B'\{^d/_c\}\ ;\ \mathsf{N}')}$$

Since $(\nu c.B \ ; \ \mathsf{N})$ is the θ_A-concretization of $(A_s \ ; \ \mathcal{C}_A \ ; \ \mathsf{N}_s)$ we have that $A_s(\theta_A\sigma_A)^\star = \nu c.B$. Hence, we know that $A_s = \nu c.B_s$ for some process B_s such that $B_s(\theta_A\sigma_A)^\star = B$. Since $(\nu c.B_s \ ; \mathcal{C}_A \ ; \mathsf{N}_s)$ is well-formed, we know that $\mathcal{C}_A = \nu c.\mathcal{C}_B$ for some well-formed constraint system $\mathcal{C}_B$. Let N''_s be the symbolic naming environment such that

$$\mathsf{N}''_s(u) = \begin{cases} \mathsf{N}''(u) & \text{if } u \in \mathcal{N} \cup \mathcal{X} \\ \mathsf{N}_s(u) & \text{if } u \in \mathcal{Y}. \end{cases}$$

Firstly, we have that $(B_s \ ; \mathcal{C}_B \ ; \mathsf{N}''_s)$ is well-formed. We have also that $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_B \ ; \mathsf{N}''_s)$. We can apply our induction hypothesis on $(B_s \ ; \ \mathcal{C}_B \ ; \ \mathsf{N}''_s)$, θ_A, $(B \ ; \mathsf{N}'') \xrightarrow{out(a,c)}_i (B' \ ; \mathsf{N}''')$. We deduce that there exist a well-formed symbolic process $(B'_s \ ; \mathcal{C}'_B \ ; \mathsf{N}'''_s)$, a substitution θ'_B and a label α^B_s such that

1. $(B_s \ ; \mathcal{C}_B \ ; \mathsf{N}''_s) \xrightarrow{\alpha^B_s}_s (B'_s \ ; \mathcal{C}'_B \ ; \mathsf{N}'''_s)$ and $\mathsf{N}''' = \mathsf{N}'''_s[\mathsf{N}'''^{-1}_s(\mathsf{c}) \mapsto \mathsf{b}]$,
2. $\theta'_B \in Sol_{\mathsf{E}}(\mathcal{C}'_B \ ; \mathsf{N}'''_s)$ and $\theta'_B|_{cv(\mathcal{C}_B)} = \theta_A$ and $\theta'_B|_{cv(\mathcal{C}_B)} = \theta'_B$ since constraint variables increase only after an input action.
3. $B'_s(\theta_A\sigma_A)^\star = B'$, i.e., $(B' \ ; \mathsf{N}''')$ is the θ'_B-concretization of $(B'_s \ ; \mathcal{C}'_B \ ; \mathsf{N}'''_s)$ and
4. $\alpha^B_s\theta_A = out(a,c)$. Note also that since $c \notin names(\mathrm{img}(\theta_A))$, we have that $\alpha^B_s = out(u,c)$ for some metavariable u such that $u\theta_A = a$.

Let $A'_s = B'_s\{^d/_c\}$, $\mathcal{C}'_A = \nu d.(\mathcal{C}'_B\{^d/_c\})$, $\mathsf{N}'_s = \mathsf{N}'''_s[c \mapsto \mathsf{b}, d \mapsto \mathsf{f}]$. Let $\theta'_A = \theta'_B = \theta_A$ and $\alpha_s = \nu d.out(u,d)$. We have that

1. $(A_s \ ; \mathcal{C}_A \ ; \mathsf{N}_s) \xrightarrow{\alpha_s}_s (A'_s \ ; \mathcal{C}'_A \ ; \mathsf{N}'_s)$. Indeed, we have that

$$\frac{(B_s \ ; \mathcal{C}_B \ ; \mathsf{N}''_s) \xrightarrow{out(u,c)}_s (B'_s \ ; \mathcal{C}'_B \ ; \mathsf{N}'''_s)}{(\nu c.B_s \ ; \nu c.\mathcal{C}_B \ ; \mathsf{N}_s) \xrightarrow{\nu d.out(u,d)}_s (B'_s\{^d/_c\} \ ; \nu d.(\mathcal{C}'_B\{^d/_c\}) \ ; \mathsf{N}'_s)}$$

2. $\theta'_A \in Sol_{\mathsf{E}}(\mathcal{C}'_A \ ; \mathsf{N}'_s)$ since $\theta'_B \in Sol_{\mathsf{E}}(\mathcal{C}'_B \ ; \mathsf{N}'''_s)$ and $c, d \notin names(\mathrm{img}(\theta'_B))$. We have also that $\theta'_A|_{cv(\mathcal{C}_A)} = \theta'_B|_{cv(\mathcal{C}_B)} = \theta_A$,
3. We have that $A'_s(\theta'_A\sigma'_A)^\star = (B'_s\{^d/_c\})(\theta_A\sigma'_A)^\star = (B'_s\{^d/_c\})(\theta_A(\sigma_A\{^d/_c\}))^\star = B'_s(\theta_A\sigma_A)^\star\{^d/_c\} = B'\{^d/_c\} = A'$, i.e., $(A' \ ; \mathsf{N}')$ is the θ'_A-concretization of $(A' \ ; \mathcal{C}'_A \ ; \mathsf{N}'_s)$,
4. $\alpha_s\theta'_A = (\nu d.out(u,d))\theta_A = \nu d.out(a,d) = \alpha$.

Case Scope$_i$: In such a case, we have that the proof tree witnessing the fact that $(A \ ; \mathsf{N}) \xrightarrow{\alpha} (A' \ ; \mathsf{N}')$ ends with an application of the following inference rule

$$\frac{(B \ ; \mathsf{N}'') \ \xrightarrow{\alpha}_i \ (B' \ ; \mathsf{N}''')}{(\nu n.B \ ; \mathsf{N}) \ \xrightarrow{\alpha}_i \ (\nu n.B' \ ; \mathsf{N}')} \qquad \text{with } n \text{ does not occur in } \alpha$$

Hence, we know that there exist a name n, a label α such that n does not occur in α and two intermediate extended processes $(B\ ;\ \mathsf{N}'')$ and $(B'\ ;\ \mathsf{N}''')$ such that $A = \nu n.B$, $A' = \nu n.B'$ and $(B\ ;\ \mathsf{N}'') \xrightarrow{\alpha} (B'\ ;\ \mathsf{N}''')$. Since $(\nu n.B\ ;\ \mathsf{N})$ is the θ_A-concretization of $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ we have that $A_s(\theta_A\sigma_A)^\star = \nu n.B$. Hence, we know that $A_s = \nu n.B_s$ for some process B_s such that $B_s(\theta_A\sigma_A)^\star = B$. Since $(\nu n.B_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ is well-formed, we know that $\mathcal{C}_A = \nu n.\mathcal{C}_B$ for some well-formed constraint system $\mathcal{C}_B$. Let N''_s be the symbolic naming environment such that

$$\mathsf{N}''_s(u) = \begin{cases} \mathsf{N}''(u) & \text{if } u \in \mathcal{N} \cup \mathcal{X} \\ \mathsf{N}_s(u) & \text{if } u \in \mathcal{Y}. \end{cases}$$

Firstly, we have $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}''_s)$ is well-formed. By Lemma B.1 we have $\theta_A \in Sol_{\mathsf{E}}(\mathcal{C}_B\ ;\ \mathsf{N}''_s)$. We apply our induction hypothesis on $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}''_s)$, θ_A, $(B\ ;\ \mathsf{N}'') \xrightarrow{\alpha} (B'\ ;\ \mathsf{N}''')$. We deduce that there exist a well-formed symbolic process $(B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'''_s)$, a substitution θ'_B and a label α^B_s such that:

1. $(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}''_s) \xrightarrow{\alpha^B_s}_s (B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'''_s)$ and $\mathsf{N}''' = \mathsf{N}'''_s|_{\mathcal{N}\cup\mathcal{X}}$.
2. $\theta'_B \in Sol_{\mathsf{E}}(\mathcal{C}'_B\ ;\ \mathsf{N}'''_s)$ and $\theta'_B|_{cv(\mathcal{C}_B)} = \theta_A$,
3. $B'_s(\theta'_B\sigma'_B)^\star = B'$, i.e., $(B'\ ;\ \mathsf{N}''')$ is the θ'_B-concretization of $(B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'''_s)$,
4. $\alpha^B_s\theta'_B = \alpha$.

Let $A'_s = \nu n.B'_s$, $\mathcal{C}'_A = \nu n.\mathcal{C}'_B$, $\mathsf{N}'_s = \mathsf{N}'''_s[n \mapsto \mathsf{b}]$. Let $\theta'_A = \theta'_B$ and $\alpha_s = \alpha^B_s$. Note that the symbolic process $(A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}'_s)$ is well-formed. Moreover, we have

1. $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s) \xrightarrow{\alpha_s} (A'_s\ ;\ \mathcal{C}'_A\ ;\ \mathsf{N}'_s)$. Indeed, we have that

$$\frac{(B_s\ ;\ \mathcal{C}_B\ ;\ \mathsf{N}''_s) \xrightarrow{\alpha_s}_s (B'_s\ ;\ \mathcal{C}'_B\ ;\ \mathsf{N}'''_s)}{(\nu n.B_s\ ;\ \nu n.\mathcal{C}_B\ ;\ \mathsf{N}_s) \xrightarrow{\alpha_s}_s (\nu n.B'_s\ ;\ \nu n.\mathcal{C}'_B\ ;\ \mathsf{N}'_s)}$$

2. $\theta'_A \in Sol_{\mathsf{E}}(\mathcal{C}'_A\ ;\ \mathsf{N}'_s)$ by Lemma B.1 since $\theta'_B \in Sol_{\mathsf{E}}(\mathcal{C}'_B\ ;\ \mathsf{N}'''_s)$ and $n \notin names(\mathrm{img}(\theta'_B))$. We have also that $\theta'_A|_{cv(\mathcal{C}_A)} = \theta'_B|_{cv(\mathcal{C}_B)} = \theta_A$,
3. We have that $A'_s(\theta'_A\sigma'_A)^\star = (\nu n.B'_s)(\theta'_B\sigma'_B)^\star = \nu n.B' = A'$, i.e., $(A'\ ;\ \mathsf{N}')$ is the θ'_A-concretization of $(A'_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}'_s)$,
4. $\alpha_s\theta'_A = (\alpha^B_s)\theta'_B = \alpha$.

Case $\textsc{Par}_i$: In such a case, we have that the proof tree witnessing the fact that $(A\ ;\ \mathsf{N}) \xrightarrow{\alpha}_i (A'\ ;\ \mathsf{N}')$ ends with an application of the following inference rule.

$$\frac{(B\ ;\ \mathsf{N}) \xrightarrow{\alpha\psi(D)}_i (B'\ ;\ \mathsf{N}')}{(B \mid D\ ;\ \mathsf{N}) \xrightarrow{\alpha}_i (B' \mid D\ ;\ \mathsf{N}')}$$

Since $(A_s\ ;\ \mathsf{N})$ is the θ_A-concretization of $(A_s\ ;\ \mathcal{C}_A\ ;\ \mathsf{N}_s)$ we have that $A = B \mid D = A_s(\theta_A\sigma_A)^\star$. Hence, we know that

- $A_s = B_s \mid D_s$ for some processes B_s and D_s such that
- $B_s(\theta_A\sigma_A)^\star = B$ and $D_s(\theta_A\sigma_A)^\star = D$.

Since $(B_s \mid D_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s)$ is well-formed, we deduce that $\mathcal{C}_A = \mathcal{C}_B \mid \psi(D_s)$ for some well-formed constraint system $\mathcal{C}_B$. We have that $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s)$ is well-formed. Let $\theta_B = (\theta_A\psi(D_s))^\star$. By Lemma B.1 we have that $\theta_B \in Sol_{\mathsf{E}}(\mathcal{C}_B \,;\, \mathsf{N}_s)$. We can apply our induction hypothesis. We deduce that there exists a well-formed symbolic process $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$, a substitution θ'_B and a label α_s^B such that:

1. $(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s) \xrightarrow{\alpha_s^B}_s (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$ and
$$\mathsf{N}'_s(u) = \begin{cases} \mathsf{N}'(u) & \text{if } u \in \mathcal{N} \cup \mathcal{X} \\ \mathsf{N}_s(u) & \text{if } u \in \mathcal{Y}. \end{cases}$$
2. $\theta'_B \in Sol_{\mathsf{E}}(\mathcal{C}'_B \,;\, \mathsf{N}'_s)$ and $\theta'_B|_{cv(\mathcal{C}_B)} = \theta_B$,
3. $B'_s(\theta'_B\sigma'_B)^\star = B'$, i.e., $(B' \,;\, \mathsf{N}')$ is the θ'_B-concretization of $(B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)$,
4. $\alpha_s^B\theta'_B = \alpha\psi(D)$.

Let $A'_s = B'_s \mid D_s$, $\mathcal{C}'_A = \mathcal{C}'_B \mid \psi(D_s)$. To define θ'_A, we distinguish two cases.

1. Either α is of the form $in(c, M)$ and $\alpha_s^B = in(u, y)$ for some metavariable u and some variable y with $\mathsf{N}_s(y) = \mathsf{n}$ such that $u\theta_A = u\theta_B = c$. In such a case, let $\theta'_A = \theta_A \cup \{y \mapsto M\}$. Moreover, as $\theta_B = (\theta_A\psi(D_s))^\star$, $\theta'_B|_{cv(\mathcal{C}_B)} = \theta_B$, $\alpha_s^B\theta'_B = \alpha\psi(D)$ and $D_s(\theta_A\sigma_A)^\star = D$ we have that $\theta'_B = (\theta'_A\psi(D_s))^\star$.
2. Otherwise, $\theta'_A = \theta_A$. Moreover in this case we have that $\theta'_B = \theta_B = (\theta_A\psi(D_s))^\star = (\theta'_A\psi(D_s))^\star$.

Let $\alpha_s = \alpha_s^B$. Note that the symbolic process $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$ is well-formed. Moreover, we have

1. $(A_s \,;\, \mathcal{C}_A \,;\, \mathsf{N}_s) \xrightarrow{\alpha_s} (A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$. Indeed, we have that
$$\frac{(B_s \,;\, \mathcal{C}_B \,;\, \mathsf{N}_s) \xrightarrow{\alpha_s}_s (B'_s \,;\, \mathcal{C}'_B \,;\, \mathsf{N}'_s)}{(B_s \mid D_s \,;\, \mathcal{C}_B \mid \psi(D_s) \,;\, \mathsf{N}_s) \xrightarrow{\alpha_s}_s (B'_s \mid D_s \,;\, \mathcal{C}'_B \mid \psi(D_s) \,;\, \mathsf{N}'_s)}$$
2. We have to show that $\theta'_A \in Sol_{\mathsf{E}}(\mathcal{C}'_A \,;\, \mathsf{N}'_s)$. As $\theta'_B = (\theta'_A\psi(D_s))^\star \in Sol(\mathcal{C}'_B \,;\, \mathsf{N}'_s)$ we have by Lemma B.1 that $\theta'_A \in Sol_{\mathsf{E}}(\mathcal{C}'_B \mid \psi(D_s) \,;\, \mathsf{N}'_s)$. It is clear that we have also $\theta'_A|_{cv(\mathcal{C}_A)} = \theta_A$.
3. We have that $A'_s(\theta'_A\sigma'_A)^\star = (B'_s \mid D_s)(\theta'_A\sigma'_A)^\star = B'_s(\theta'_A\sigma'_A)^\star \mid D_s(\theta'_A\sigma'_A)^\star = B'_s(\theta'_B\sigma'_B)^\star \mid D_s(\theta_A\sigma_A)^\star = B' \mid D = A'$, i.e., $(B' \mid D \,;\, \mathsf{N}')$ is a θ'_A-concretization of $(A'_s \,;\, \mathcal{C}'_A \,;\, \mathsf{N}'_s)$,

4. In the case where $\alpha = in(c, M)$, we have that $\alpha_s \theta'_A = in(u, y)\theta'_A = in(c, M) = \alpha$. Otherwise, the equality holds since $\psi(D)$ and $\psi(D_s)$ do not affect variables which occurs in a label since those variables are of type channel.

Lastly, we can deal with the rule $\textsc{Struct}_i$ by relying on our Proposition B.4. This allows us to conclude. □