Open bisimulation, revisited ⋆

Sébastien Briais ⋆⋆ Uwe Nestmann

School of Computer and Communication Sciences, EPFL, Switzerland

Abstract

In the context of the π-calculus, open bisimulation is prominent and popular due to its congruence properties and its easy implementability. Motivated by the attempt to generalise it to the spi-calculus, we offer a new, more refined definition and show in how far it coincides with the original one.

1 Introduction

Open bisimulation, as introduced by Sangiorgi [San96] is an attractive candidate notion of bisimulation for the π-calculus for a number of different reasons. First, it constitutes a reasonably full congruence, i.e., it is preserved by all operators including input prefix. Second, it allows for simple axiomatizations (for finite terms). Third, it is rather straightforward to build tools that check open bisimilarity (see the MWB [Vic94] or the ABC [Bri03]).

The current paper arose from our attempt to "smoothly" generalise the definition of open bisimulation from the π-calculus to the spi-calculus, an extension of the former by cryptographic primitives to be used in the description of security protocols. It turns out that this is not easily doable, for reasons that we try to explain in the remainder of this Introduction. Driven by the quest for a meaningful definition of open-style bisimulation for the spi-calculus, we came up with a proposal that we then observed can also be meaningfully projected down to the case of the π-calculus. The resulting notion and its comparison to the original definition is the main contribution of this paper.

The flurry of notions of bisimulation for the π-calculus [1], ranging from *ground* over *early* and *late* to *open*, results mainly from the different possible treatments of simulated symbolic input transitions, e.g., when

$$\text{simulating} \qquad P \xrightarrow{a(x)} P' \qquad \text{by} \qquad Q \xrightarrow{a(x)} Q'.$$

⋆ A long version is found at `http://lamp.epfl.ch/~sbriais/`.

⋆⋆Supported by the Swiss National Science Foundation, grant No. 21-65180.1

[1] Luckily, all of these notions collapse in certain sub-calculi, for example like the asynchronous π-calculus, that are still expressive enough for most practical purposes.

This is a preliminary version. The final version will be published in Electronic Notes in Theoretical Computer Science
URL: `www.elsevier.nl/locate/entcs`

The problem is that after the execution of a symbolic input on channel a, the input variable[2] x becomes free in the resulting continuation processes P' and Q'. Considering the possible instantiations of this input variable by received messages can be done either not at all (as in ground), or (as in early) before the simulating transition is chosen, or (as in late) right afterwards—or (as in open) considering all possible substitutions (not only affecting the just freed input variable) even before starting any bisimulation game. The latter case can also be seen as "very late" or "lazy" since all possible instantiations of the input variable will be checked the next time we try to continue with the bisimulation game with P' and Q'.

For clarity of the following explanations, in an application $P\{^{M}/_{x}\}$ of a substitution, where M replaces all (free) occurrences of x in P, let us use the terms *substitution subject* for x and *substitution object* for M.

What do we actually mean by *all possible instantiations*? By definition, only free names can ever be affected as substitution subjects. In a process, there are three kinds of free name. A free name may be free because:

(i) either it was already initially free,
(ii) or it has become free after having done an input (or been substituted),
(iii) or it has become free after having been created as a local name, and afterwards output to some observing process.

We argue[3] that names of the latter kind are constant, i.e., they should not be considered as substitution *subjects*, because they were created freshly and thus appropriately chosen. (We formally support this point of view in Lemma 3.6, and show that it gives rise to an equivalent *freshness-aware* notion of bisimulation.) In contrast, the first two kinds shall be considered. On the other hand, also not all substitution *objects* may be acceptable. More precisely: depending on the history of the ongoing bisimulation game, certain instantiations may sometimes be forbidden. There may be two different reasons for this.

The first reason concerns names names of kind (i) or (ii), say a, that were free in a process *before* another name, say b, got freshly created and extruded. Due to the freshness, any subsequent substitution for subject a must not mention b as substitution object, so not to retrospectively invalidate this freshness property. In open bisimulation, represented by an indexed family of binary relations, the indexing component is precisely a structure called *distinction* that keeps track of inequalities like $a \neq b$, as required above.

The second reason concerns only names of kind (ii) and resides on the intuition that substitution objects represent messages that may be sent from the observer to the observed process. In the π-calculus, there is no limitation beyond the above distinctions: the observer may send any name that it may have

[2] Note that we do not introduce different syntactic categories for (constant) names and variables. It is only for convenience of the explanation that we call receiving names in bound input position "input variables".
[3] And here we slightly differ from Sangiorgi's definition of open bisimulation.

received earlier, or it may simply invent names on its own. However, it is precisely here that severe difficulties arise when moving to the spi-calculus. The main reason there is the presence of complex messages $\mathrm{E}_{k_n}(\cdots \mathrm{E}_{k_1}(M)\cdots)$, which may dispose of some deeply nested structure that involves so-called encryption keys $k_1 \ldots k_n$. Substitution objects are then all messages that the observer (potentially a malicious attacker) could possibly have generated at the moment the message was input. This generation is not arbitrary; it is constrained by the *knowledge* that the observer has acquired up to the moment of interaction. For example, consider the spi-calculus process

$$P \stackrel{\text{def}}{=} (\nu k)\,(\nu m)\,\overline{a}\langle \mathrm{E}_k(m)\rangle.a(x).\overline{a}\langle k\rangle.[x{=}m]\overline{a}\langle a\rangle.\,\mathbf{0}$$

where (νk) denotes the generation of a fresh name, $\overline{a}\langle k\rangle$ the sending of name k over channel name a, $a(x)$ the reception of a message over channel name a with input variable x, $\mathrm{E}_k(m)$ the previously mentioned encryption of datum m with key k, and $[x{=}m]$ a test of equality of names. Intuitively, the output $\overline{a}\langle a\rangle$ is impossible, because it would require that x could have been substituted by m, which is itself impossible, because the private datum m was passed on to the observer only within message $\mathrm{E}_k(m)$ encrypted with the private key k; however, this key was unknown to the observer when it sent the message that got received by $a(x)$ — it was published only afterwards.

Here, a simple distinction $k \neq m$ is not sufficient to characterise disallowed substitutions because neither m, nor $\mathrm{E}_b(m)$, nor $\mathrm{E}_k(\mathrm{E}_b(m))$, etc., are permitted substitution objects. In contrast, the message $\mathrm{E}_k(m)$ that the observer learnt in the first exchange could have been sent back to the process.

The study of other notions of bisimulation for the spi-calculus (see an overview in [BN02]) resulted in careful analyses of observer (attacker) knowledge and various kinds of data structures for the representation of such knowledge. Typically, all messages that were emitted by an observed process in the course of a bisimulation game are stored. Likewise, in particular in the proposal of symbolic bisimulation of [BBN04], some timing or ordering information is stored that keeps track of which messages were known to the observer at the moment of the reception of a message by a process.

Together with the above-mentioned freshness-awareness, we choose to represent the observer knowledge for our new notion of open bisimulation by triples of the form $(O, V, \prec)$, where $\prec \subseteq O \times V$. O is the set of the emitted messages, while V is the set of the substitutable names. Note that the freshly created and subsequently extruded names are $C = \mathrm{n}(O) \setminus \mathrm{n}(V)$ and we add the condition that $O \cap V = \emptyset$. The relation $\prec$ indicates for each substitutable variable $x \in V$, which part of O was known when x was input. Thus, in bisimulation games, this kind of environment structure permits to treat substitutable names of the kinds (i) and (ii) in the same way.

While the above motivated way to characterise permissible substitutions was driven by an analysis of spi-calculus phenomena, it also makes sense to apply it to the much simpler π-calculus, which is the goal of this paper. In

$$P, Q \quad ::= \quad \mathbf{0} \;\Big|\; E(x).P \;\Big|\; \overline{E}\langle F\rangle.P \;\Big|\; \phi P \;\Big|\; P \,|\, Q \;\Big|\; P + Q \;\Big|\; !\, P \;\Big|\; (\nu x)\, P$$

Table 1
Syntax of processes $\mathcal{P}$

$$\begin{array}{rcll} M, N & ::= & a & (\text{messages } \mathcal{M}) \\ E, F & ::= & a & (\text{expressions } \mathcal{E}) \\ \phi, \psi & ::= & tt \;\big|\; \phi \wedge \psi \;\big|\; [E = F] & (\text{formulae } \mathcal{F}) \end{array}$$

Table 2
Syntax of messages, expressions and formulae for the π-calculus

$$\begin{array}{rcll} M, N & ::= & a \;\big|\; \mathrm{E}_N(M) & (\text{messages } \mathcal{M}) \\ E, F & ::= & a \;\big|\; \mathrm{E}_F(E) \;\big|\; \mathrm{D}_F(E) & (\text{expressions } \mathcal{E}) \\ \phi, \psi & ::= & tt \;\big|\; \phi \wedge \psi \;\big|\; [E = F] \;\big|\; [E : \mathcal{N}] & (\text{formulae } \mathcal{F}) \end{array}$$

Table 3
Syntax of messages, expressions and formulae for the spi-calculus

§2, we recall the original definition of open bisimulation in the π-calculus, for which we use a unified presentation of the π-calculus and the spi-calculus. In §3, we develop the details of our new proposal and prove its coincidence with the original notion. In §4, we comment on the advantages of our new notion.

2 Open bisimulation

2.1 Syntax of the π-calculus and the spi-calculus

A countably infinite set $a, b, c, \ldots, k, l, m, n, \ldots, x, y, z, \ldots$ of *names* $\mathcal{N}$ is presupposed. In the following, we write $\tilde{z}$ for a (possibly empty) finite sequence of names $z_1, z_2, \ldots, z_n$. If $\tilde{z}$ is such a sequence, then we write $\{\tilde{z}\}$ for the set of names appearing in the sequence $\tilde{z}$. In order to unify the presentation of the π-calculus and the spi-calculus, we have parametrised the syntax of *processes* Table 1 by *messages*, *expressions* and *formulae*. Table 2 read in conjunction with Table 1 gives the syntax of the π-calculus, whereas for the spi-calculus, Table 3 and Table 1 should be considered.

The set of names appearing in a message M is written $\mathrm{n}(M)$. In the case of the π-calculus, it is simply the singleton set containing M (since M is a name). Similarly, the set of the names appearing in an expression E is written $\mathrm{n}(E)$ and the set of the names appearing in a formula ϕ is written $\mathrm{n}(\phi)$. Finally, the set of free names $\mathrm{fn}(P)$ and bound names $\mathrm{bn}(P)$ of a process P are defined as usual taking into account that the name x is bound in P by the constructs $E(x).P$ and $(\nu x)\, P$. These notions are straightforwardly lifted to sets.

Definition of $[\![\cdot]\!] : \mathcal{E} \to \mathcal{M} \cup \{\bot\}$		
$[\![a]\!]$	$\stackrel{\text{def}}{=} a$	
$[\![\mathrm{E}_F(E)]\!]$	$\stackrel{\text{def}}{=} \mathrm{E}_N(M)$	if $[\![E]\!] = M \in \mathcal{M}$ and $[\![F]\!] = N \in \mathcal{M}$
$[\![\mathrm{D}_F(E)]\!]$	$\stackrel{\text{def}}{=} M$	if $[\![E]\!] = \mathrm{E}_N(M) \in \mathcal{M}$ and $[\![F]\!] = N \in \mathcal{M}$
$[\![E]\!]$	$\stackrel{\text{def}}{=} \bot$	in all other cases
Definition of $[\![\cdot]\!] : \mathcal{F} \to \{\mathbf{true}, \mathbf{false}\}$		
$[\![tt]\!]$	$\stackrel{\text{def}}{=} \mathbf{true}$	
$[\![\phi \wedge \psi]\!]$	$\stackrel{\text{def}}{=} [\![\phi]\!]$ and $[\![\psi]\!]$	
$[\![[\, E = F \,]]\!]$	$\stackrel{\text{def}}{=} \mathbf{true}$	if $[\![E]\!] = [\![F]\!] = M \in \mathcal{M}$
$[\![[\, E : \mathcal{N} \,]]\!]$	$\stackrel{\text{def}}{=} \mathbf{true}$	if $[\![E]\!] = a \in \mathcal{N}$
$[\![\phi]\!]$	$\stackrel{\text{def}}{=} \mathbf{false}$	in all other cases
Definition of $\mathfrak{c}(\cdot) : \mathcal{F} \to 2^{\mathcal{M} \cup \{\bot\}}$		
$\mathfrak{c}(tt)$	$\stackrel{\text{def}}{=} \emptyset$	
$\mathfrak{c}(\phi \wedge \psi)$	$\stackrel{\text{def}}{=} \mathfrak{c}(\phi) \cup \mathfrak{c}(\psi)$	
$\mathfrak{c}([\, E = F \,])$	$\stackrel{\text{def}}{=} \emptyset$	
$\mathfrak{c}([\, E : \mathcal{N} \,])$	$\stackrel{\text{def}}{=} \{[\![E]\!]\}$	

Table 4
Evaluation of expressions and formulae

2.2 Labelled (late) semantics

Table 4 defines the straightforward evaluation of expressions and formulae, as well as some name constraints of a given formula. Table 5 defines a labelled transition $P \xrightarrow{\mu}_S P'$ where μ is an action and S is a set of names. The set S collects the names that should be names in order for the transition to be enabled. In the π-calculus, where only names are considered, it can be simply ignored but it is useful for the case of spi-calculus. These names are those that are used as channels or that are assumed to be names by formulae.

Upon this transition system, the late semantics of the π-calculus and the spi-calculus is given by: $P \xrightarrow{\mu} P'$ if and only if there is S such that $P \xrightarrow{\mu}_S P'$.

The syntax of actions μ is given by:

$$\mu \quad ::= \quad \tau \;\Big|\; a(x) \;\Big|\; (\nu \tilde{z})\, \overline{a}\, M \qquad \text{(actions)}$$

The bound output actions $(\nu \tilde{z})\, \overline{a}\, M$ are such that $\{\tilde{z}\} \subseteq \mathrm{n}(M)$. In the case of the π-calculus, since messages M are reduced to names, we have two cases:

$$\textsc{Input}\ \frac{[\![E]\!] = a \in \mathcal{N}}{E(x).P \xrightarrow{a(x)}_{\{a\}} P} \qquad \textsc{Output}\ \frac{[\![E]\!] = a \in \mathcal{N} \qquad [\![F]\!] = M \in \mathcal{M}}{\overline{E}\langle F\rangle.P \xrightarrow{\overline{a}\,M}_{\{a\}} P}$$

$$\textsc{Close-l}\ \frac{P \xrightarrow{a(x)}_S P' \qquad Q \xrightarrow{(\nu\tilde{z})\,\overline{a}\,M}_{S'} Q'}{P \mid Q \xrightarrow{\tau}_{S\cup S'} (\nu\tilde{z})\,(P'\{^M/_x\} \mid Q')}\ \{\tilde{z}\} \cap \mathrm{fn}(P) = \emptyset$$

$$\textsc{Open}\ \frac{P \xrightarrow{(\nu\tilde{z})\,\overline{a}\,M}_S P'}{(\nu z')\,P \xrightarrow{(\nu z'\tilde{z})\,\overline{a}\,M}_{S\setminus\{z'\}} P'}\ z' \in \mathrm{n}(M) \setminus \{a, \tilde{z}\}$$

$$\textsc{Res}\ \frac{P \xrightarrow{\mu}_S P'}{(\nu z)\,P \xrightarrow{\mu}_{S\setminus\{z\}} (\nu z)\,P'}\ z \notin \mathrm{n}(\mu) \qquad \textsc{Guard}\ \frac{P \xrightarrow{\mu}_S P'}{\phi P \xrightarrow{\mu}_{S\cup\mathfrak{c}(\phi)} P'}\ [\![\phi]\!] = \mathbf{true}$$

$$\textsc{Par-l}\ \frac{P \xrightarrow{\mu}_S P'}{P \mid Q \xrightarrow{\mu}_S P' \mid Q}\ \mathrm{bn}(\mu) \cap \mathrm{fn}(Q) = \emptyset \qquad \textsc{Sum-l}\ \frac{P \xrightarrow{\mu}_S P'}{P + Q \xrightarrow{\mu}_S P'}$$

$$\textsc{Rep}\ \frac{P \mid !\,P \xrightarrow{\mu}_S P'}{!\,P \xrightarrow{\mu}_S P'} \qquad \textsc{Alpha}\ \frac{P =_\alpha P' \qquad P' \xrightarrow{\mu}_S P''}{P \xrightarrow{\mu}_S P''}$$

Table 5
The late semantics of the π-calculus

either $\tilde{z}$ is the empty sequence and $(\nu\tilde{z})\,\overline{a}\,M$ is simply written $\overline{a}\,M$ or $\tilde{z} = M$ and the bound output action is simply $(\nu z)\,\overline{a}\,z$ where $z = M$.

The set of names $\mathrm{n}(\mu)$ is defined by:

$$\mathrm{n}(\tau) := \emptyset, \qquad \mathrm{n}(a(x)) := \{a, x\}, \qquad \mathrm{n}((\nu\tilde{z})\,\overline{a}\,M) := \{a, \tilde{z}\} \cup \mathrm{n}(M).$$

The set of bound names $\mathrm{bn}(\mu)$ of μ is defined by:

$$\mathrm{bn}(\tau) := \emptyset, \qquad \mathrm{bn}(a(x)) := \{x\}, \qquad \mathrm{bn}((\nu\tilde{z})\,\overline{a}\,M) := \{\tilde{z}\}.$$

Moreover, if $\mu = a(x)$ or $\mu = (\nu\tilde{z})\,\overline{a}\,M$, we define $ch(\mu) \stackrel{\text{def}}{=} a$.

2.3 Open bisimulation in the π-calculus

As mentioned in the Introduction, open bisimulation was introduced by Sangiorgi [San96]. It relies on the notion of distinction to keep track of inequalities of names in order to constrain the set of substitutions to be considered in the respective bisimulation game.

Definition 2.1 (distinction) *A binary relation $D \subseteq \mathcal{N} \times \mathcal{N}$ on names is called* distinction *if it is finite, symmetric, and irreflexive.*

By $\mathrm{n}(D)$ we denote the set of names contained in D.

If A, B are two sets of names, we define the distinction $A \otimes B$ to be $\{(x,y) \in A \times B \cup B \times A \mid x \neq y\}$. $A^{\neq}$ abbreviates $A \otimes A$.

Definition 2.2 (substitution) *A substitution σ is a total function $\mathcal{N} \to \mathcal{M}$ such that its* support *$supp(\sigma) := \{x \mid x\sigma \neq x\}$ is a finite set.*

The co-support *of σ is $cosupp(\sigma) := \{x\sigma \mid x \in supp(\sigma)\}$.*

The set of names *of σ is $\mathrm{n}(\sigma) := supp(\sigma) \cup \mathrm{n}(cosupp(\sigma))$.*

As said previously, distinctions are to prevent substitutions to fuse two names that were assumed to be different at some point. Hence the definition of so-called respectful substitutions.

Definition 2.3 (respectfulness) *Let D be a distinction, σ a substitution.*

σ respects *D, written $\sigma \triangleright D$, if and only if $x\sigma \neq y\sigma$ for all $(x,y) \in D$.*

If σ respects D, then $D\sigma$ is defined as $\{(x\sigma, y\sigma) \mid (x,y) \in D\}$.

Note that since $\mathcal{M} = \mathcal{N}$ in the case of the π-calculus, $D\sigma$ is itself a distinction.

An open bisimulation is a distinction-indexed family of symmetric relations between processes that satisfies some condition.

Definition 2.4 (open bisimulation) *The family $(\mathcal{R}_D)_{D \in \mathcal{D}}$ (where $\mathcal{D}$ is a set of distinctions) of symmetric relations is an open bisimulation if for all $D \in \mathcal{D}$, for all substitutions σ such that $\sigma \triangleright D$, for all $(P,Q) \in \mathcal{R}_D$, whenever $P\sigma \xrightarrow{\mu} P'$ (with $\mathrm{bn}(\mu)$ fresh), there exists Q' such that $Q\sigma \xrightarrow{\mu} Q'$ and*

- *if $\mu = (\nu z)\,\overline{a}\,z$ for some a and z, $D' \in \mathcal{D}$ and $(P',Q') \in \mathcal{R}_{D'}$ where $D' = D\sigma \cup \{z\} \otimes (\mathrm{fn}((P+Q)\sigma) \cup \mathrm{n}(D\sigma))$*
- *otherwise, $D\sigma \in \mathcal{D}$ and $(P',Q') \in \mathcal{R}_{D\sigma}$.*

The induced equivalence is defined as usual, modulo the indexing component.

Definition 2.5 (open bisimilarity) *Let $P, Q \in \mathcal{P}$ and D a distinction. We say that P and Q are open D-bisimilar—written $P \approx_{\mathrm{O}}^{D} Q$—if there exists an open bisimulation $(\mathcal{R}_{\underline{D}})_{\underline{D} \in \mathcal{D}}$ such that $D \in \mathcal{D}$ and $(P,Q) \in \mathcal{R}_D$.*

Instead of families of binary relations between processes we may also use ternary relations, which is often done in the context of the spi-calculus. Thus, instead of $(P,Q) \in \mathcal{R}_D$, we then write $(D,P,Q) \in \mathcal{R}$, where D is usually called *environment*, and the ternary relation is called *environment-sensitive*. It is mainly for easier readability that we adopt the ternary style in the following, although a bit of care needs to be taken to lift the three equivalence properties to the ternary format. For example, a ternary environment-sensitive relation is called *symmetric* if and only if $(e,P,Q) \in \mathcal{R} \Leftrightarrow (e,Q,P) \in \mathcal{R}$.

3 Open bisimulation, reloaded

Before proceeding to our new proposal to define open-style bisimulation, we provide a slightly different, but equivalent variant of the previously given standard notion. This variant will make it easier to relate to our new proposal.

3.1 A freshness-aware variant of open bisimulation

In this section, we define the notion of F-open bisimulation. The simple idea is, as we mentioned already in the Introduction, to prevent names that were previously (in the course of a bisimulation game) created freshly from being considered as permissible substitution subjects.

The knowledgeable reader may be reminded of the notion of *quasi-open* bisimulation, proposed by Sangiorgi and Walker [SW01b], and later on revisited by Fu [Fu05]. There, the use of distinctions as environments was adapted to the use of a simple set of names that were once freshly created and therefore deemed to remain constant. The resulting quasi-open bisimulation was recognised as being strictly weaker than open bisimulation. Sangiorgi and Walker intuitively summarised this difference as: "*In open bisimilarity, when a name z is sent in a bound-output action, the distinction is enlarged to ensure that z is never identified with any name that is free in the processes that send it. In quasi-open bisimilarity, in contrast, at no point after the scope of z is extruded can a substitution be applied that identifies z with any other name.*" [SW01b].

Like quasi-open bisimulation, the following definition also explicitly keeps track of previously freshly created names. However, it does not use this information to prevent the fusion of such fresh names like quasi-open bisimulation does. It only use this information to implement the idea that fresh names can be considered as constant names once chosen, such that they should afterwards never be used as substitution subjects. In fact, Lemmas 3.6 and 3.7 show that this change still faithfully retains the equational power of open bisimulation.

Definition 3.1 (F-environment) *The pair (D, C) where D is a distinction and C is a finite subset of names is a* F-environment *if $C^{\neq} \subseteq D$. The set of all* F*-environments is written $\mathcal{F}$.*

The distinction D plays the same role as in open bisimulation, while the set C indicates which names can be considered as constant names. It is used to refine the notion of respectfulness, as follows.

Definition 3.2 (respectful substitution)
Let (D, C) be a F*-environment and σ a substitution. We say that σ respects (D, C) – written $\sigma \blacktriangleright (D, C)$ – if $\sigma \triangleright D$ and $supp(\sigma) \cap C = \emptyset$.*

Definition 3.3 (F-relation) *A* F-relation *$\mathcal{R}$ is a subset of $\mathcal{F} \times \mathcal{P} \times \mathcal{P}$.*

Definition 3.4 (F-open bisimulation) *A symmetric* F*-relation $\mathcal{R}$ is a* F-open bisimulation*, if for all $((D, C), P, Q) \in \mathcal{R}$ and for all substitutions σ such that $\sigma \blacktriangleright (D, C)$, whenever $P\sigma \xrightarrow{\mu} P'$ (with $\mathrm{bn}(\mu)$ fresh), there exists Q' such that $Q\sigma \xrightarrow{\mu} Q'$ and*

- *if $\mu = (\nu z)\,\overline{a}\,z$ for some a and z, $((D', C \cup \{z\}), P', Q') \in \mathcal{R}$ where $D' = D\sigma \cup \{z\} \otimes (\mathrm{fn}((P{+}Q)\sigma) \cup \mathrm{n}(D\sigma))$*
- *otherwise, $((D\sigma, C), P', Q') \in \mathcal{R}$*

The two only differences compared to open bisimulation is, first, that the

notion of respectfulness is slightly modified such that it takes into account the constant names of a F-environment and, second, that the extruded names are being accumulated in the pool of constant names of F-environments.

Definition 3.5 (F-open bisimilarity) *Let $P, Q \in \mathcal{P}$ and $(D, C) \in \mathcal{F}$.*

P and Q are F*-open (D, C)-bisimilar, written $P \approx_{\mathrm{F}}^{(D,C)} Q$, if there is a* F*-open bisimulation $\mathcal{R}$ such that $((D, C), P, Q) \in \mathcal{R}$.*

The two notions of bisimilarity are equivalent in the following sense.

Lemma 3.6 *Let $P, Q \in \mathcal{P}$ and $(D, C) \in \mathcal{F}$.*

If $P \approx_{\mathrm{F}}^{(D,C)} Q$, then $P \approx_{\mathrm{O}}^{D} Q$.

Proof. The key of the proof is that it is possible, if $\sigma \triangleright D$ and $C^{\neq} \subseteq D$, to find a substitution σ' and a bijective substitution θ such that $\sigma = \sigma'\theta$ and $\sigma' \blacktriangleright (D, C)$.

Lemma 3.7 *Let $P, Q \in \mathcal{P}$ and D a distinction.*

If $P \approx_{\mathrm{O}}^{D} Q$, then $\forall C : C^{\neq} \subseteq D \Rightarrow P \approx_{\mathrm{F}}^{(D,C)} Q$.

Proof. This result is obvious because $\sigma \blacktriangleright (D, C)$ implies $\sigma \triangleright D$.

3.2 A knowledge-aware variant of open bisimulation

As motivated in the Introduction, we propose a bisimulation that makes explicit the attacker who plays against the two players P and Q involved in the bisimulation game. The knowledge of the attacker is stored in K-environments of the form $(O, V, \prec)$. The set of names V represents all the substitutable free names (those that were initially free or become free after an input action). The set of messages O contains all the messages that were emitted by P and Q, except the names of V. Finally, the relation $\prec$ indicates for each substitutable name x the available knowledge acquired by the attacker at the moment the name x was input. This relation characterises the admissible messages received from the attacker.

Definition 3.8 (K-environment) *A* K*-environment is a triple $(O, V, \prec)$ such that $O \cup V$ is a finite subset of $\mathcal{N}$, $O \cap V = \emptyset$ and $\prec \subseteq O \times V$. The set of all* K*-environments is $\mathcal{K}$.*

If E is a K-environment, and $n \in \mathcal{N}$, it is possible to extend E with n in two ways. Either n is meant to be an emitted name and it is added to the constant part of E, or n is meant to be a received name and it is added to the variable part of E and put in relation with all already emitted names. If n is already contained in E, its addition to E has no effect.

Definition 3.9 (Extension of a K-environment) *Let $E = (O, V, \prec)$ be a* K*-environment and $n \in \mathcal{N}$. We define*

(i) $E \oplus_{\mathrm{O}} n \stackrel{\text{def}}{=} (O', V, \prec)$ *where* $O' \stackrel{\text{def}}{=} O \cup \{n\}$ *if* $n \notin V$ *and* $O' \stackrel{\text{def}}{=} O$ *otherwise.*

(ii) *if* $n \notin O \cup V$, $E \oplus_{\mathrm{V}} n \stackrel{\text{def}}{=} (O, V \cup \{n\}, \prec')$ *where* $\prec' \stackrel{\text{def}}{=} \prec \cup\, O \times \{n\}$.

Keeping in mind that a substitution represents the potential inputs the attacker could have generated, we define the set of respectful substitutions. A substitution σ respects a K-environment $E = (O, V, \prec)$ if it affects only substitutable names (those in V) and if for each $x \in V$, it takes only values that were generatable at the moment when x was input. This means that such a name x can use any name in V (this corresponds to fusing two substitutable names), or use any name in O that was known by the attacker when x was input (this is indicated by the relation $\prec$) or use any new fresh name not contained in E (this corresponds to the creation of free names by the attacker). In the π-calculus, since a substitution replaces a name by a name, this can be easily and concisely expressed by:

Definition 3.10 (respectful substitution)
A substitution σ respects a K*-environment $E = (O, V, \prec)$, written $\sigma \blacktriangleright\!\blacktriangleright E$, if:*

(i) $supp(\sigma) \subseteq V$
(ii) $\forall x \in V : x\sigma \in O \Rightarrow x\sigma \prec x$

Roughly speaking, in spi-calculus, $x\sigma$ is built using names from V, the messages from O that are permitted by $\prec$ and some freshly generated names. In π-calculus, this is simplified to $x\sigma \prec x$ because $x\sigma \in \mathcal{N}$.

Any K-environment $E = (O, V, \prec)$ may, under the impact of some a respectful substitution σ, be straightforwardly updated to E^σ. In general, the knowledge contained in O should be updated to $O\sigma$. However, in the π-calculus, substitution deals only with names, and since $O \cap V = \emptyset$ we have $O\sigma = O$. The set V of substitutable names should keep all the names that were not affected by σ, and in addition list all the new names that were created by the attacker, as visible in the substitution objects.[4] Particular care must be taken when computing the new relation $\prec'$ because of the possibility that σ fuses two names of V. Fusing two names x and y (by $x\sigma = y\sigma$) corresponds to a voluntary loss of power of the attacker: the only admissible values for the fused name are those that were admissible for *both* x *and* y.

Definition 3.11 (K-environment updating)
Let $E = (O, V, \prec)$ be a K*-environment and σ a substitution such that $\sigma \blacktriangleright\!\blacktriangleright E$. The updated environment is $E^\sigma \stackrel{\text{def}}{=} (O', V', \prec')$ of E by σ where*

$$V' \stackrel{\text{def}}{=} (V \setminus supp(\sigma)) \cup \{x\sigma \mid x \in supp(\sigma) \wedge x\sigma \notin O\}$$
$$\prec' \stackrel{\text{def}}{=} \{(n, x') \mid \forall x \in V : x' \in \mathrm{n}(x\sigma) \Rightarrow n \prec x\}$$

Definition 3.12 (K-relation) *A* K*-relation $\mathcal{R}$ is a subset of $\mathcal{K} \times \mathcal{P} \times \mathcal{P}$ such that $\forall((O, V, \prec), P, Q) \in \mathcal{R} : \mathrm{fn}(P{+}Q) \subseteq O \cup V$.*

The new variant of open bisimulation now simply keeps track of whether dynamically freed names are substitutable or not. If they are, then we explic-

[4] The fact that we put the names created by the environment in the substitutable part gives a "lazy" flavour to our definition, because it allows the attacker to uncover itself gradually.

itly state that previously created names may be used in future substitutions. Names that will be created later on—by the process—will not be permitted.

Definition 3.13 (K-open bisimulation) *A symmetric* K*-relation* $\mathcal{R}$ *is a* K-open bisimulation*, if for all* $(E, P, Q) \in \mathcal{R}$ *and for all substitutions* σ *such that* $\sigma \blacktriangleright\!\blacktriangleright E$*, whenever* $P\sigma \xrightarrow{\mu} P'$ *(with* $\mathrm{bn}(\mu)$ *fresh), there exists* Q' *such that* $Q\sigma \xrightarrow{\mu} Q'$ *and*

- *if* $\mu = \tau$*, then* $(E^\sigma, P', Q') \in \mathcal{R}$
- *if* $\mu = a(x)$ *then* $(E^\sigma \oplus_{\mathrm{V}} x, P', Q') \in \mathcal{R}$
- *if* $\mu = (\nu z)\,\overline{a}\,z$ *or* $\mu = \overline{a}\,z$ *then* $(E^\sigma \oplus_{\mathrm{O}} z, P', Q') \in \mathcal{R}$

We see in this definition that indeed O collects all the messages emitted by P and Q (but the addition $E^\sigma \oplus_{\mathrm{O}} z$ has only effect when $\mu = (\nu z)\,\overline{a}\,z$ because E contains all free names of P and Q) and V collects all substitutable names.

Definition 3.14 (K-open bisimilarity) *Let* $P, Q \in \mathcal{P}$ *and* $E \in \mathcal{K}$*.*

P and Q are K*-open E-bisimilar, written* $P \approx^E_{\mathrm{K}} Q$*, if there is a* K*-open bisimulation* $\mathcal{R}$ *such that* $(E, P, Q) \in \mathcal{R}$*.*

In the π-calculus, it is possible to represent any K-environment by some F-environment. The idea is that all names in O should be kept pairwise distinct (they were fresh names) and for all $(n, x) \in O \cup V$, if n cannot be used to generate x (i.e. $\neg n \prec x$), then n and x should be distinct ($n \neq x$).

Definition 3.15 (F-environment of a K-environment) *Let* $E = (O, V, \prec)$ *be a* K*-environment. We define* $\mathrm{f}(E) = (D, O)$ *where* $D = O^{\neq} \cup \bigcup_{n \in O \wedge x \in V \wedge \neg n \prec x} \{(n, x), (x, n)\}$*. Clearly,* $\mathrm{f}(E) \in \mathcal{F}$*.*

The K-open bisimilarity is sound with respect to F-open bisimilarity.

Lemma 3.16 *Let* $P, Q \in \mathcal{P}$ *and* $(O, V, \prec) \in \mathcal{K}$ *such that* $\mathrm{fn}(P{+}Q) \subseteq O \cup V$*. Then we have:*

$$P \approx^{(O,V,\prec)}_{\mathrm{K}} Q \Rightarrow P \approx^{\mathrm{f}((O,V,\prec))}_{\mathrm{F}} Q$$

Under the condition that the F-environment (D, C) is representable by a K-environment E, F-open (D, C)-bisimilarity is sound with respect to K-open E-bisimilarity.

Lemma 3.17 *Let* $P, Q \in \mathcal{P}$ *and* $(D, C) \in \mathcal{F}$*. Then we have*

$$P \approx^{(D,C)}_{\mathrm{F}} Q \Rightarrow \left(\forall V, \prec : \begin{array}{l} C \cap V = \emptyset \\ \wedge\ \mathrm{fn}(P{+}Q) \subseteq C \cup V \\ \wedge\ (D, C) = \mathrm{f}((C, V, \prec)) \end{array} \Rightarrow P \approx^{(C,V,\prec)}_{\mathrm{K}} Q \right)$$

The proof of this lemma also shows that F-environments that are not representable by any corresponding K-environment are negligible.

It is known that open D-bisimilarity is a D-congruence, i.e., it is preserved by all contexts in which the occurrence of the hole is not underneath an input prefix binding a name in D (cf. [SW01a]). We conjecture that, based on our new notion of K-open-bisimilarity and with respect to $(D, C) = \mathrm{f}((C, V, \prec))$,

we can define a bigger classes of contexts that preserve open bisimilarity. The idea is (1) to admit contexts with the same above condition w.r.t. names C as D-congruence imposes w.r.t. D, and furthermore (2) to admit contexts where the hole occurs underneath an input prefix that binds a name x of V, but only if, in addition, every name of $\{n \in C \mid \neg n \prec x\}$ appears underneath a respective restriction on the "path" from the hole-binding input prefix for x to the hole. We leave a formal treatment of this issue for future work, and just explain the conjecture by means of a simple example.

Example 3.18 Let $P = \overline{x} \,|\, y$ and $Q = \overline{x}.y + y.\overline{x}$.
It is known and easily verifiable that $P \approx_{\mathrm{O}}^{D} Q$ with $D = \{(x,y),(y,x)\}$.
Let $C = \{y\}$ and $V = \{x\}$, and note that $(D, C) = \mathrm{f}((C, V, \emptyset))$.
Observe that $P \approx_{\mathrm{K}}^{(C,V,\prec)} Q$.
Now, let us regard the context $X[\cdot] = a(x).(\nu y)\,[\cdot]$.
Then $X[P] \approx_{\mathrm{O}}^{\emptyset} X[Q]$, although $X[\cdot]$ is not considered by D-congruence.
However, $X[\cdot]$ follows our above informal rule of admissible contexts.
Finally, just note that also $X[P] \approx_{\mathrm{K}}^{(\emptyset,\{a\},\emptyset)} X[Q]$.

In summary, we can conclude from the previous results our new notion of open-style bisimilarity semantically coincides with the original style.

Theorem 3.19 $P \approx_{\mathrm{O}}^{\emptyset} Q \Leftrightarrow P \approx_{\mathrm{F}}^{(\emptyset,\emptyset)} Q \Leftrightarrow P \approx_{\mathrm{K}}^{(\emptyset,\mathrm{fn}(P+Q),\emptyset)} Q$

4 Conclusion and future work

The main contribution of this paper is the definition of a new notion of open-style bisimulation in the π-calculus guided by knowledge-sensitive notions of bisimulation that arose in the context of the spi-calculus. We have proved that the new notion corresponds to the original open bisimilarity in a precise and informative way that indicates improved congruence properties.

The new definition of open-style bisimulation can now indeed be smoothly extended in the spi-calculus (a first proposal is given in appendix but we can mention close work such as [Bri02] or [BBN04]). Our proposal in spi-calculus uses the same environment shape as our proposal in π-calculus. But it is necessary, as noticed by Abadi and Gordon in [AG98], to introduce also a notion of indistinguishability. Some type constraints should also be ensured: a free name used as a channel should never be substituted by anything else than a name. Hence, the environment we propose for spi-calculus are quadruple $(h, v, \prec, \gamma)$ where h stores all the emitted messages and moreover implements this notion of indistinguishability, v contains all the substitutable names, $\prec$ governs which messages can be used to generate inputs for names in v and γ stores which names should keep the type of names.

Next, we plan to study congruence properties of our K-open bisimilarity. We will do the same for our extension to the spi-calculus and also study its relation to symbolic bisimilarity as defined in [BBN04].

References

[AG98] M. Abadi and A. D. Gordon. A Bisimulation Method for Cryptographic Protocols. *Nordic Journal of Computing*, 5(4):267–303, Winter 1998. An extended abstract appeared in the *Proceedings of ESOP '98*, LNCS 1381, pages 12–26.

[AG99] M. Abadi and A. D. Gordon. A Calculus for Cryptographic Protocols: The Spi Calculus. *Information and Computation*, 148(1):1–70, 1999.

[BBN04] J. Borgström, S. Briais and U. Nestmann. Symbolic Bisimulation in the Spi Calculus. In P. Gardner and N. Yoshida, eds, *Proceedings of CONCUR 2004*, volume 3170 of *LNCS*, pages 161–176. Springer Verlag, Sept. 2004.

[BN02] J. Borgström and U. Nestmann. On Bisimulations for the Spi Calculus. In H. Kirchner and C. Ringeissen, eds, *Proc. AMAST'02*, volume 2422 of *Lecture Notes in Computer Science*, pages 287–303. Springer, 2002. Long version to appear in *Mathematical Structures in Computer Science*.

[Bri02] S. Briais. Towards open bisimulation in the spi calculus. Mémoire de D.E.A., Université Paris VII - Denis Diderot, 2002.

[Bri03] S. Briais. *ABC Bisimulation Checker*. EPFL, 2003. Available from `http://lamp.epfl.ch/~sbriais/abc/abc.html`.

[Bri04] S. Briais. Formal proofs about hedges using the Coq proof assistant, 2004. `http://lamp.epfl.ch/~sbriais/spi/hedges/hedge.html`.

[Fu05] Y. Fu. On Quasi-Open Bisimulation. *Theoretical Computer Science*, 338:96–126, 2005.

[San96] D. Sangiorgi. A Theory of Bisimulation for the π-calculus. *Acta Informatica*, 33:69–97, 1996. Earlier version published as Report ECS-LFCS-93-270, University of Edinburgh. An extended abstract appeared in the *Proceedings of CONCUR '93*, LNCS 715.

[SW01a] D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge University Press, 2001.

[SW01b] D. Sangiorgi and D. Walker. Some results on barbed equivalences in pi-calculus. In *Proc. CONCUR '01*, volume 2154 of *LNCS*. Springer Verlag, 2001.

[Vic94] B. Victor. *A Verification Tool for the Polyadic π-Calculus*. Licentiate thesis, Department of Computer Systems, Uppsala University, Sweden, May © 1994. Available as report DoCS 94/50.

A Proofs

Lemma A.1 *Let D be a distinction and σ a substitution such that $\sigma \triangleright D$. Let C be a finite set of names such that $C^{\neq} \subseteq D$. Then there exists σ' a substitution and θ a bijective substitution such that $\sigma' \blacktriangleright (D, C)$ and $\sigma = \sigma'\theta$ and $\mathrm{n}(\theta) \subseteq C \cup C\sigma$.*

Proof. We first prove that σ is injective on the finite set C.

Indeed, let $x, y \in C$ such that $x \neq y$. Since $C^{\neq} \subseteq D$, we have $(x, y) \in D$. Moreover, we have $\sigma \triangleright D$, so we have $x\sigma \neq y\sigma$. This proves that σ is injective on C.

According to Lemma 1.4.11 of [SW01a], we have the existence of a bijective substitution θ such that σ and θ agree on C. By construction, we have moreover that $\mathrm{n}(\theta) \subseteq C \cup C\sigma$.

Let $\sigma' = \sigma\theta^{-1}$. Then σ' is a substitution such that $\sigma = \sigma'\theta$.

It remains now to prove that $\sigma' \blacktriangleright (D, C)$.

We first show that $\sigma' \triangleright D$. Let $x, y \in D$. Since $\sigma \triangleright D$, we have that $x\sigma \neq y\sigma$. Now, since θ^{-1} is bijective, we have also that $x\sigma\theta^{-1} \neq y\sigma\theta^{-1}$, hence $x\sigma' \neq y\sigma'$ and $\sigma' \triangleright D$.

Now we show that $supp(\sigma') \cap C = \emptyset$. Let $x \in C$. Since σ and θ agree on C, we have $x\sigma = x\theta$. So $x\sigma' = x\sigma\theta^{-1} = x\theta\theta^{-1} = x$ and $x \notin supp(\sigma')$. Hence $supp(\sigma') \cap C = \emptyset$.

Finally, we have proved that $\sigma' \blacktriangleright (D, C)$.

Lemma A.2 (Lemma 3.6) *Let $P, Q \in \mathcal{P}$ and $(D, C) \in \mathcal{F}$.*

If $P \approx_{\mathrm{F}}^{(D,C)} Q$, then $P \approx_{\mathrm{O}}^{D} Q$.

Proof. Let $\mathcal{R}$ be a F-open bisimulation such that $((D, C), P, Q) \in \mathcal{R}$.

Let $\mathcal{D} = \{D \mid \exists C, P, Q : ((D, C), P, Q) \in \mathcal{R}\}$

For $D \in \mathcal{D}$ and θ a bijective substitution, let

$$\mathcal{R}'_{D\theta} = \{(P\theta, Q\theta) \mid \exists C : ((D, C), P, Q) \in \mathcal{R}\}$$

Let $\mathcal{D}' = \{D\theta \mid D \in \mathcal{D} \wedge \theta \text{ bijective substitution}\}$.

We have that $(\mathcal{R}'_D)_{D \in \mathcal{D}'}$ is an open bisimulation.

Indeed, let $D' \in \mathcal{D}'$, σ a substitution such that $\sigma \triangleright D'$ and $(P_0, Q_0) \in \mathcal{R}'_{D'}$. By definition, there is $D \in \mathcal{D}$ and θ a bijective substitution such that $D' = D\theta$. Moreover, there exists C such that $((D, C), P, Q) \in \mathcal{R}$ and $P_0 = P\theta$ and $Q_0 = Q\theta$.

Since $\sigma \triangleright D\theta$, we have $\theta\sigma \triangleright D$. We then use Lemma A.1 with $\theta\sigma$ and C. We have the existence of a substitution σ' and a bijective substitution θ' such that $\theta\sigma = \sigma'\theta'$, $\sigma' \blacktriangleright (D, C)$ and $\mathrm{n}(\theta') \subseteq C \cup C\theta$.

Assume now that $P_0\sigma \xrightarrow{\mu} P'_0$ (with $\mathrm{bn}(\mu)$ fresh), i.e. $P\theta\sigma \xrightarrow{\mu} P'_0$, i.e. $P\sigma'\theta' \xrightarrow{\mu} P'_0$. Since θ' is bijective, we have $P\sigma' \xrightarrow{\mu\theta'^{-1}} P'_0\theta'^{-1}$.

Since $((D, C), P, Q) \in \mathcal{R}$ and $\sigma' \blacktriangleright (D, C)$, by definition, there exists Q' such that $Q\sigma' \xrightarrow{\mu\theta'^{-1}} Q'$ and

- if $\mu\theta'^{-1} = (\nu z)\,\overline{a}\,z$ then $((D'', C \cup \{z\}), P'_0\theta'^{-1}, Q') \in \mathcal{R}$ where $D'' = D\sigma' \cup \{z\} \otimes (\mathrm{fn}((P+Q)\sigma') \cup \mathrm{n}(D\sigma'))$
- otherwise $((D\sigma', C), P'_0\theta'^{-1}, Q') \in \mathcal{R}$

Let $Q'_0 = Q'\theta'$, then we have $Q' = Q'_0\theta'^{-1}$ and $Q\sigma' \xrightarrow{\mu\theta'^{-1}} Q'_0\theta'^{-1}$.

Since θ'^{-1} is bijective, we then have $Q\sigma'\theta' \xrightarrow{\mu} Q'_0$, i.e. $Q\theta\sigma \xrightarrow{\mu} Q'_0$, i.e. $Q_0\sigma \xrightarrow{\mu} Q'_0$.

- if $\mu = (\nu z)\,\overline{a}\,z$, then $\mu\theta'^{-1} = (\nu z)\,\overline{a}\,z$ and we have by assumption $((D'', C \cup \{z\}), P'_0\theta'^{-1}, Q'_0\theta'^{-1}) \in \mathcal{R}$ where $D'' = D\sigma' \cup \{z\} \otimes (\mathrm{fn}((P+Q)\sigma') \cup \mathrm{n}(D\sigma'))$. So, by definition, we have $(P'_0, Q'_0) \in \mathcal{R}'_{D''\theta'}$. But $D''\theta' = D\sigma'\theta' \cup \{z\theta'\} \otimes (\mathrm{fn}((P+Q)\sigma')\theta' \cup \mathrm{n}(D\sigma'\theta'))$. So $D''\theta' = D\theta\sigma \cup \{z\theta\} \otimes (\mathrm{fn}((P+Q)\theta\sigma) \cup \mathrm{n}(D\theta\sigma))$, i.e. $D''\theta' = D'\sigma \cup \{z\} \otimes (\mathrm{fn}((P_0+Q_0)\sigma) \cup \mathrm{n}(D'\sigma))$ (because z is fresh and thus $z \notin \mathrm{n}(\theta')$).
- otherwise $((D\sigma', C), P'_0\theta'^{-1}, Q'_0\theta'^{-1}) \in \mathcal{R}$ so $(P'_0, Q'_0) \in \mathcal{R}'_{D\sigma'\theta'}$ and $D\sigma'\theta' = D\theta\sigma = D'\sigma$.

Hence, $(\mathcal{R}'_D)_{D \in \mathcal{D}'}$ is an open bisimulation.

Lemma A.3 *Let $E = (O, V, \prec)$ be a* K*-environment and σ a substitution. Then*

$$\sigma \blacktriangleright\!\blacktriangleright E \Leftrightarrow \mathit{supp}(\sigma) \subseteq V \wedge \sigma \blacktriangleright \mathrm{f}(E)$$

Proof. Let D such that $\mathrm{f}(e) = (D, O)$.

- First assume that $\sigma \blacktriangleright\!\blacktriangleright E$.
  By definition, we have $\mathit{supp}(\sigma) \subseteq V$ and $\forall x \in V : x\sigma \in O \Rightarrow x\sigma \prec x$.
  Since $\mathit{supp}(\sigma) \subseteq V$ and $O \cap V = \emptyset$, we have $\mathit{supp}(\sigma) \cap O = \emptyset$.
  Let $(x, y) \in D$. We have to show that $x\sigma \neq y\sigma$. There are four cases (according to the definition of D): either $x, y \in O$ with $x \neq y$, or $x \in O$, $y \in V$ and $\neq x \prec y$ or the two other symmetric cases.
  By case distinction, assume that $x, y \in O$ and $x \neq y$. Since $\mathit{supp}(\sigma) \cap O = \emptyset$, we have $x\sigma = x$, $y\sigma = y$, hence $x\sigma \neq y\sigma$.
  Now assume that $x \in O$, $y \in V$ and $\neg x \prec y$. Since $\mathit{supp}(\sigma) \cap O = \emptyset$, we have $x\sigma = x$. Assume by contradiction that $y\sigma = x\sigma = x$, then we have $y\sigma \in O$. Thus, we have $y\sigma \prec y$ which is equivalent to $x \prec y$ and thus leading to a contradiction. So $x\sigma \neq y\sigma$.
  The two other symmetric cases are treated in the same way.
  Hence $\sigma \blacktriangleright \mathrm{f}(E)$.
- Assume now that $\mathit{supp}(\sigma) \subseteq V \wedge \sigma \blacktriangleright \mathrm{f}(E)$.
  We have then that $\sigma \triangleright D$.
  By hypothesis, $\mathit{supp}(\sigma) \subseteq V$.
  Let $x \in V$ and assume that $x\sigma \in O$. We have to show that $x\sigma \prec x$. Assume by contradiction that $\neg x\sigma \prec x$. Then, by definition of D, we have that $(x\sigma, x) \in D$. Since σ respects D, we have $x\sigma\sigma \neq x\sigma$, but since $x\sigma \in O$

and $supp(\sigma) \cap O = \emptyset$, we have $x\sigma\sigma = x\sigma$, obtaining a contradiction.

Hence $\sigma \blacktriangleright\blacktriangleright E$.

Lemma A.4 *Let $E = (O, V, \prec)$ be a* K*-environment, D such that $\mathrm{f}(E) = (D, O)$ and σ a substitution such that $\sigma \blacktriangleright\blacktriangleright E$. Then $\mathrm{f}(E^\sigma) = (D\sigma, O)$.*

Proof. Let $(D', O) = \mathrm{f}(E^\sigma)$. We have to show that $D' = D\sigma$.

By definition, $D' = O^{\neq} \cup \bigcup_{n \in O \wedge x' \in V' \wedge \neg n \prec' x'} \{(n, x'), (x', n)\}$ where $V' = (V \setminus supp(\sigma)) \cup \{x\sigma \mid x \in supp(\sigma) \wedge x\sigma \notin O\}$ and $\prec'$ is defined by

$$n \prec' x' \Leftrightarrow \bigwedge_{x \in V \wedge x' \in \mathrm{n}(x\sigma)} n \prec x$$

Let $(x', y') \in D'$. If $(x', y') \in O \otimes O$ then $(x', y') \in D\sigma$ since $supp(\sigma) \cap O = \emptyset$. So, assume that $x' \in O$, $y' \in V'$ and $\neg x' \prec' y'$. By definition, we have that there exists in $y \in V$ such that $y' \in \mathrm{n}(y\sigma)$ and $\neg x' \prec y$. So, we have, by definition of D, $(x', y) \in D$ and since $x'\sigma = x'$ and $y\sigma = y'$, we have thus $(x', y') \in D\sigma$. So $D' \subseteq D\sigma$.

Let $(x', y') \in D\sigma$. By definition, there exists $(x, y) \in D$ such that $x' = x\sigma$ and $y' = y\sigma$. If $(x, y) \in O \otimes O$, then $x' = x$ and $y' = y$ and thus $(x', y') \in D'$. Now assume that $x \in O$, $y \in V$ and $\neg x \prec y$. Since $supp(\sigma) \cap O = \emptyset$, we have $x' = x$. If $y' \in O$ then $(x', y') \in O \otimes O$ and $(x', y') \in D'$. Assume that $y' \notin O$. Then, by definition of V', $y' \in V'$. We have, since $y' = y\sigma$, $y' \in \mathrm{n}(y\sigma)$ and since $\neg x' \prec y$, we have, by definition of $\prec'$, $\neg x' \prec' y'$ and thus $(x', y') \in D'$. So $D\sigma \subseteq D'$.

B Open bisimulation in the spi-calculus

In the following, we concentrate on how to extend K $-$*open* bisimilarity to the spi-calculus. This follows mainly ideas of [Bri02] and [BBN04] and the ideas already given in the main part of this article. Unfortunately, we did not have time to explain deeply our definitions. However, we have decided to put them in this appendix so that an interested reader can see how K $-$*open* bisimilarity extends to the spi-calculus. We first introduce the reader to spi-calculus and late hedged bisimulation. Then in Section B.3, we give the definition of our bisimulation in spi-calculus and then state a soundness theorem with respect to late hedged bisimilarity.

B.1 The spi-calculus

B.1.1 Syntax and semantics

The spi-calculus is a process calculus that was introduced by Abadi and Gordon [AG99] to model and study cryptographic protocols.

The syntax of the spi-calculus is given by Table 1 and Table 3. We have chosen to focus the study of this paper to a shared-key cryptosystem but the

language of messages can be easily extended to deal with public/private key, pairing and/or hashing (see [BBN04] or [Bri04] for more details).

Late semantics of the spi-calculus has been defined Section 2.2.

B.2 Late hedged bisimulation

We present in this section the late hedged bisimulation inspired by the definition of early hedged bisimulation that was defined in [BN02]. Abadi and Gordon first noticed that the classical notion of bisimulation as commonly used in the π-calculus was not really interesting for the spi-calculus and they proposed an environment-sensitive bisimulation: the *framed bisimulation*. *Hedged bisimulation* is another kind of environment-sensitive bisimulation where the environment (which can be understood as the knowledge of a potential attacker) is represented by a hedge. It has been shown in [BN02] that early hedged bisimilarity coincides with barbed equivalence.

B.2.1 Hedges

Definition B.1 *If $C \subseteq A \times B$ for some sets A and B, we define*

- $\pi_1(C) \stackrel{\text{def}}{=} \{a \in A \mid \exists b \in B : (a,b) \in C\}$,
- $\pi_2(C) \stackrel{\text{def}}{=} \{b \in B \mid \exists a \in A : (a,b) \in C\}$, *and*
- $C^{-1} = \{(b,a) \in A \times B \mid (a,b) \in C\}$.

We recall that the reader who is interested in a richer message language or in seeing formal definitions about hedges is invited to consult [Bri04] (in particular, the definition of analysis is given precisely and it is shown how to extend the definition of consistency).

Definition B.2 (hedge) *A* hedge *is a finite subset of $\mathcal{M} \times \mathcal{M}$. The set of all hedges is* $\mathbf{H}$.

If h is a hedge, we define the *synthesis* $\mathcal{S}(h)$ of h, the *analysis* $\mathcal{A}(h)$ of h and the *irreducibles* $\mathcal{I}(h)$ of h.

Definition B.3 (synthesis,analysis,irreducibles) *Let h be a hedge.*

The synthesis $\mathcal{S}(h)$ of h is the smallest subset of $\mathcal{M} \times \mathcal{M}$ containing h and satisfying:

$$(\text{SYN-ENC})\ \frac{(M,N) \in \mathcal{S}(h) \qquad (K,L) \in \mathcal{S}(h)}{(\mathrm{E}_K(M), \mathrm{E}_L(N)) \in \mathcal{S}(h)}$$

The analysis $\mathcal{A}(h)$ of h is the smallest hedge containing h and satisfying:

$$(\text{ANA-DEC})\ \frac{(\mathrm{E}_K(M), \mathrm{E}_L(N)) \in \mathcal{A}(h) \qquad (K,L) \in \mathcal{S}(\mathcal{A}(h))}{(M,N) \in \mathcal{A}(h)}$$

Finally, the irreducibles $\mathcal{I}(h)$ of h is defined by:

$$\mathcal{I}(h) \stackrel{\text{def}}{=} \mathcal{A}(h) \setminus \{(\mathrm{E}_K(M), \mathrm{E}_L(N)) \in \mathcal{A}(h) \mid (K, L) \in \mathcal{S}(\mathcal{A}(h))\}$$

Definition B.4 (left-consistency) *A hedge h is* left-consistent *if for all $(M, N) \in h$, we have*

(i) $M \in \mathcal{N} \Rightarrow N \in \mathcal{N}$
(ii) $\forall (M', N') \in h : M = M' \Rightarrow N = N'$
(iii) *if* $M = \mathrm{E}_K(M')$ *then* $K \notin \pi_1(\mathcal{S}(h))$

Definition B.5 (consistency) *A hedge h is* consistent *if h and h^{-1} are left-consistent.*

B.2.2 Late hedged bisimulation

Definition B.6 (hedged relation) *A* hedged-relation $\mathcal{R}$ *is a subset of* $\mathbf{H} \times \mathcal{P} \times \mathcal{P}$ *such that* $\forall (h, P, Q) \in \mathcal{R} : \mathrm{fn}(P) \subset \mathrm{n}(\pi_1(h)) \wedge \mathrm{fn}(Q) \subset \mathrm{n}(\pi_2(h))$.

A hedged relation $\mathcal{R}$ is called

- consistent *if* $\forall (h, P, Q) \in \mathcal{R} : h$ *is consistent;*
- symmetric *if* $\forall (h, P, Q) : (h, P, Q) \in \mathcal{R} \Leftrightarrow (h^{-1}, Q, P) \in \mathcal{R}$

Definition B.7 (late hedged bisimulation)
A symmetric consistent hedged-relation $\mathcal{R}$ is a late hedged bisimulation if for all $(h, P, Q) \in \mathcal{R}$, if $P \xrightarrow{\mu_1} P'$ with $\mathrm{bn}(\mu_1) \cap \mathrm{n}(\pi_1(h)) = \emptyset$ and $ch(\mu_1) \in \pi_1(h)$ (if $\mu_1 \neq \tau$), then there exists Q' and μ_2 such that $Q \xrightarrow{\mu_2} Q'$ with $\mathrm{bn}(\mu_2) \cap \mathrm{n}(\pi_2(h)) = \emptyset$ and

- *if $\mu_1 = \tau$ then $\mu_2 = \tau$ and $(h, P', Q') \in \mathcal{R}$*
- *if $\mu_1 = a_1(x_1)$ then $\mu_2 = a_2(x_2)$ where $(a_1, a_2) \in \mathcal{S}(h)$ and for all $B \subseteq \mathcal{N} \times \mathcal{N}$ consistent, $M_1, M_2 \in \mathcal{M}$ such that*
  - $\pi_1(B) \setminus \mathrm{n}(M_1) = \emptyset$
  - $\pi_1(B) \cap \mathrm{n}(\pi_1(h)) = \emptyset = \pi_2(B) \cap \mathrm{n}(\pi_2(h))$
  - $(M_1, M_2) \in \mathcal{S}(h \cup B)$

  we have $(h \cup B, P'\{^{M_1}/_{x_1}\}, Q'\{^{M_2}/_{x_2}\}) \in \mathcal{R}$
- *if $\mu_1 = (\nu \tilde{c})\,\overline{a_1}\,M_1$ then $\mu_2 = (\nu \tilde{d})\,\overline{a_2}\,M_2$ where $(a_1, a_2) \in \mathcal{S}(h)$ and $(\mathcal{I}(h \cup \{(M_1, M_2)\}), P', Q') \in \mathcal{R}$*

Definition B.8 (late hedged bisimilarity) *Let $P, Q \in \mathcal{P}$ and $h \in \mathbf{H}$ such that $\mathrm{fn}(P) \subseteq \mathrm{n}(\pi_1(h))$ and $\mathrm{fn}(Q) \subseteq \mathrm{n}(\pi_2(h))$. We say that P and Q are late h hedged bisimilar – written $P \approx^h_{\text{LH}} Q$ if there exists a late hedged bisimulation $\mathcal{R}$ such that $(h, P, Q) \in \mathcal{R}$.*

B.3 Open hedged bisimulation

Definition B.9 (S-environment)
The quadruple $(h, v, \prec, (\gamma_l, \gamma_r))$ is a S-environment if $h \subseteq \mathcal{M} \times \mathcal{M}$, $v \subseteq \mathcal{N} \times \mathcal{N}$ are two finite sets such that $h \cap v = \emptyset$, $\prec \subseteq h \times v$, $\gamma_l \subseteq \pi_1(v)$ and

$\gamma_r \subseteq \pi_2(v)$ *such that*

$$\forall (M,N) \in h, (x,y) \in v : (M,N) \prec (x,y) \Rightarrow x \notin \mathrm{n}(M) \wedge y \notin \mathrm{n}(N)$$

The set of all S-environments is $\mathcal{S}_{\mathrm{H}}$.
For $(x,y) \in v$, *we define* $h^{\prec}_{(x,y)} \stackrel{\text{def}}{=} \{(M,N) \mid (M,N) \prec (x,y)\}$.
We define $e^{-1} \stackrel{\text{def}}{=} (h^{-1}, v^{-1}, \prec^{-1}, (\gamma_r, \gamma_l))$
where $\prec^{-1} = \{((N,M),(y,x)) \mid (M,N) \prec (x,y)\}$.
We define $n_1(e) \stackrel{\text{def}}{=} \mathrm{n}(\pi_1(h \cup v))$ *and* $n_2(e) \stackrel{\text{def}}{=} \mathrm{n}(\pi_2(h \cup v))$.
We define $H(e) = \mathcal{I}(h \cup v)$ *and* $\mathcal{S}(e) = \mathcal{S}(H(e))$.

The intuition behind a S-environment $e = (h, v, \prec, (\gamma_l, \gamma_r))$ is as for K-environment. The hedge h represents the messages emitted by the two players, v represents the names input by these two players, the relation $\prec$ stores the time precedence between the emitted messages and the input names (thus a message containing x cannot have been emitted before the name x had been input) and (γ_l, γ_r) is an additional component that tells which input names should be really names and not arbitrary messages. For the π-calculus, this last component does not exist because messages are names.

Definition B.10 *Let* h *be a hedge and* (σ, ρ) *be a pair of substitutions. We define* $h(\sigma,\rho) \stackrel{\text{def}}{=} \{(M\sigma, N\rho) \mid (M,N) \in h\}$.

Definition B.11 (respectful substitutions) *Let* (σ,ρ) *be a pair of substitutions,* $e = (h, v, \prec, (\gamma_l, \gamma_r))$ *be a S-environment and* $B \subseteq \mathcal{N} \times \mathcal{N}$ *a consistent hedge. We say that* (σ,ρ) respects e with B *– written* $(\sigma,\rho) \rhd_B e$ *– if*

- $supp(\sigma) \subseteq \pi_1(v)$ *and* $supp(\rho) \subseteq \pi_2(v)$
- $\forall (x,y) \in v : x \in supp(\sigma) \Leftrightarrow y \in supp(\rho)$
- $\pi_1(B) \setminus \mathrm{n}(cosupp(\sigma)) = \emptyset$
- $\pi_1(B) \cap \mathrm{n}(\pi_1(h \cup (v \setminus v_{(\sigma,\rho)}))) = \emptyset = \pi_2(B) \cap \mathrm{n}(\pi_2(h \cup (v \setminus v_{(\sigma,\rho)})))$
- $\forall (x,y) \in v_{(\sigma,\rho)} : (x\sigma, y\rho) \in \mathcal{S}(\mathcal{I}(h^{\prec}_{(x,y)}(\sigma,\rho) \cup B \cup (v \setminus v_{(\sigma,\rho)})))$ *where* $v_{(\sigma,\rho)} = v \cap (supp(\sigma) \times supp(\rho))$
- $\forall x \in \gamma_l : x\sigma \in \mathcal{N}$
- $\forall y \in \gamma_r : y\rho \in \mathcal{N}$

Definition B.12 (S-environment updating) *Let* (σ,ρ) *be a pair of substitutions,* $e = (h, v, \prec, (\gamma_l, \gamma_r))$ *be a S-environment and* $B \subseteq \mathcal{N} \times \mathcal{N}$ *a consistent hedge such that* $(\sigma,\rho) \rhd_B e$. *The update* $e_B^{(\sigma,\rho)} = (h', v', \prec', (\gamma'_l, \gamma'_r))$ *of* e *by* (σ,ρ) *is defined as follows:*

- $h' = h(\sigma,\rho)$
- $v' = (v \setminus (supp(\sigma) \times supp(\rho))) \cup B$
- $\prec'$ *is defined by*

$$(M\sigma, N\rho) \prec' (x',y') \Leftrightarrow \bigwedge_{(x,y) \in v \,\wedge\, x' \in \mathrm{n}(x\sigma)} (M,N) \prec (x,y)$$

- $\gamma'_l = \gamma_l\sigma \cap \pi_1(v')$
- $\gamma'_r = \gamma_r\rho \cap \pi_2(v')$

Definition B.13 (consistency) *A S-environment $e = (h, v, \prec, (\gamma_l, \gamma_r))$ is* consistent *if for all (σ, ρ), B such that $(\sigma, \rho) \triangleright_B e$, we have:*

- *$\mathcal{I}(h' \cup v')$ is consistent*
- *$\forall (x, y) \in v' : x \in \gamma'_l \Leftrightarrow y \in \gamma'_r$*

where $(h', v', \prec', (\gamma'_l, \gamma'_r)) = e_B^{(\sigma,\rho)}$.

Definition B.14 (extension) *Let $e = (h, v, \prec, (\gamma_l, \gamma_r))$ be a S-environment.*

If $(M, N) \in \mathcal{M} \times \mathcal{M}$, we define $e \oplus_{\mathrm{O}} (M, N) \stackrel{\text{def}}{=} (h', v, \prec, (\gamma_l, \gamma_r))$ where $h' \stackrel{\text{def}}{=} \begin{cases} h \cup \{(M, N)\} & \text{if } (M, N) \notin v \\ h & \text{otherwise} \end{cases}$

Moreover, if $(x, y) \in \mathcal{N} \times \mathcal{N}$ such that $x \notin n_1(e)$ and $y \notin n_2(e)$, we define $e \oplus_{\mathrm{V}} (x, y) \stackrel{\text{def}}{=} (h, v', \prec', (\gamma_l, \gamma_r))$ where $v' \stackrel{\text{def}}{=} v \cup \{(x, y)\}$ and $\prec' \stackrel{\text{def}}{=} \prec \cup\, h \times \{(x, y)\}$.

Finally, if S_1 and S_2 are two finite sets of names, we define

$$e \oplus_{\mathrm{C}} (S_1, S_2) \stackrel{\text{def}}{=} (h, v, \prec, (\gamma_l \cup (S_1 \cap \pi_1(v)), \gamma_r \cup (S_2 \cap \pi_2(v)))).$$

Definition B.15 *An* open hedged-relation *$\mathcal{R}$ is a subset of $\mathcal{S}_{\mathrm{H}} \times \mathcal{P} \times \mathcal{P}$ such that $\forall (e, P, Q) \in \mathcal{R} : \mathrm{fn}(P) \subseteq n_1(e) \wedge \mathrm{fn}(Q) \subseteq n_2(e)$.*

It is called

- consistent *if $\forall (e, P, Q) \in \mathcal{R} : e$ is consistent*
- symmetric *if $\forall (e, P, Q) : (e, P, Q) \in \mathcal{R} \Leftrightarrow (e^{-1}, Q, P) \in \mathcal{R}$*

Definition B.16 (open hedged bisimulation)
A symmetric consistent open hedged-relation $\mathcal{R}$ is an open hedged bisimulation if for all $(e, P, Q) \in \mathcal{R}$, for all (σ, ρ) and B such that $(\sigma, \rho) \triangleright_B e$, if $P\sigma \xrightarrow{\mu_1}_{S_1} P'$ with $\mathrm{bn}(\mu_1) \cap n_1(e_B^{(\sigma,\rho)}) = \emptyset$ and $ch(\mu_1) \in \pi_1(\mathcal{S}(e_B^{(\sigma,\rho)}))$ (if $\mu_1 \neq \tau$), there exists Q', μ_2 and S_2 such that $Q\rho \xrightarrow{\mu_2}_{S_2} Q'$ with $\mathrm{bn}(\mu_2) \cap n_2(e_B^{(\sigma,\rho)}) = \emptyset$ and

- *if $\mu_1 = \tau$ then $\mu_2 = \tau$ and $(e_B^{(\sigma,\rho)} \oplus_{\mathrm{C}} (S_1, S_2), P', Q') \in \mathcal{R}$*
- *if $\mu_1 = a_1(x_1)$ then $\mu_2 = a_2(x_2)$ where $(a_1, a_2) \in \mathcal{S}(e_B^{(\sigma,\rho)})$ and $(e_B^{(\sigma,\rho)} \oplus_{\mathrm{V}} (x_1, x_2) \oplus_{\mathrm{C}} (S_1, S_2), P', Q') \in \mathcal{R}$*
- *if $\mu_1 = (\nu\tilde{c})\,\overline{a_1}\,M_1$ then $\mu_2 = (\nu\tilde{d})\,\overline{a_2}\,M_2$ where $(a_1, a_2) \in \mathcal{S}(e_B^{(\sigma,\rho)})$ and $(e_B^{(\sigma,\rho)} \oplus_{\mathrm{O}} (M_1, M_2) \oplus_{\mathrm{C}} (S_1, S_2), P', Q') \in \mathcal{R}$*

Definition B.17 (open hedged bisimilarity) *Let $P, Q \in \mathcal{P}$ and $e \in \mathcal{S}_{\mathrm{H}}$ such that $\mathrm{fn}(P) \subseteq n_1(e)$ and $\mathrm{fn}(Q) \subseteq n_2(e)$. We say that P and Q are open e hedged bisimilar – written $P \approx^e_{\mathrm{OH}} Q$ – if there exists an open hedged bisimulation $\mathcal{R}$ such that $(e, P, Q) \in \mathcal{R}$.*

Lemma B.18 *Let $P, Q \in \mathcal{P}$ and $e \in \mathcal{S}_{\mathrm{H}}$ such that $\mathrm{fn}(P) \subseteq n_1(e)$ and $\mathrm{fn}(Q) \subseteq$*

$n_2(e)$. *Then, we have*

$$P \approx^{e}_{\text{OH}} Q \Rightarrow (\forall(\sigma,\rho), B : (\sigma,\rho) \triangleright_B e \Rightarrow P \approx^{H(e_B^{(\sigma,\rho)})}_{\text{LH}} Q)$$