

A Trace Based Bisimulation for the Spi Calculus: An Extended Abstract

Alwen Tiu

Computer Sciences Laboratory
Australian National University

Abstract. A notion of open bisimulation is formulated for the spi-calculus, an extension of the π -calculus with cryptographic primitives. In this formulation, open bisimulation is indexed by pairs of symbolic traces, which represent the history of interactions between the environment with the pairs of processes being checked for bisimilarity. The use of symbolic traces allows for a symbolic treatment of bound input in bisimulation checking which avoids quantification over input values. Open bisimilarity is shown to be sound with respect to testing equivalence, and further, it is shown to be an equivalence relation on processes and a congruence on finite processes.

1 Introduction

The spi-calculus [2] is an extension of the π -calculus [9] with cryptographic primitives. This extension allows one to model cryptographic protocols and, via a notion of observational equivalence called *testing equivalence*, one can express security properties that a protocol satisfies. Testing equivalence is usually defined by quantifying the environment with which the processes interact: roughly, to show that two processes are testing equivalent, one shows that the two processes exhibit the same traces under arbitrary observers. As in the π -calculus, bisimulation techniques have been defined to check the observational equivalence of processes that avoids quantification over all possible observers. Unlike the π -calculus, in order to capture security notions such as secrecy, bisimulation in the spi-calculus needs to take into account the states of the environment (e.g., public networks) in its interaction with the processes being checked for equivalence. This gives rise to a more refined notion of equivalence of actions in the definition of bisimulation. In the π -calculus, to check whether two processes are bisimilar, one checks that an action by one process is matched by an equivalent action by the other process, and their continuations possess the same property. The differences between bisimulations for the π - and the spi-calculus lie in the interpretation of “equivalent actions”. Consider the processes $P = (\nu x)\bar{a}\langle\{b\}_x\rangle.0$ and $Q = (\nu x)\bar{a}\langle\{c\}_x\rangle.0$. P is a process that can output on channel a a message b , encrypted with a fresh key x , and terminates, while Q outputs a message c encrypted with x on the same channel. In the standard definitions of bisimulation for the π -calculus, e.g., late or early bisimulation [9], these two processes are not

bisimilar since they perform (syntactically) distinct actions. In the spi-calculus, when one is concerned only with whether an intruder (in its interaction with P and Q) can discover the message being encrypted, the two actions by P and Q are essentially equivalent; the intruder does not have access to the key x , hence cannot access the underlying messages.

Motivated by the above observation, different notions of bisimulation have been proposed, among others *framed bisimulation* [1], *environment-sensitive bisimulation* [4], *hedged bisimulation* [6], etc. (see [6] for a review on these bisimulations). All these notions of bisimulation share a similarity in that they are all indexed by some sort of structure representing the “knowledge” of the environment. This structure is called differently from one definition to another. We shall use the rather generic term *observer theory*, or *theory* for short, to refer to the knowledge structure used in this paper, which is just a finite set of pairs of messages. A theory represents the pairs of messages that are obtained through the interaction between the environment (observer) and the pairs of processes in the bisimulation set. The pairs of messages in the theory represent equivalent messages, from the point of view of the observer. This observer theory is then used as a theory in a deductive system for deducing messages (or actions) equivalence. Under this theory, equivalent messages need not be syntactically equivalent.

A main difficulty in bisimulation checking for spi-processes is in dealing with the input actions of the processes, where one needs to check that the processes are bisimilar for all equivalent pairs of input messages. One way of dealing with the infinite quantification is through a symbolic technique where one delays the instantiations of input values until they are needed. This technique has been applied to hedged bisimulation by Borgström et al.[5]. Their work on symbolic bisimulation for the spi-calculus is, however, mainly concerned with obtaining a sound approximation of hedged bisimulation, and less with studying meta-level properties of the symbolic bisimulation as an equivalence relation. *Open bisimulation* [10], on the other hand, makes use of the symbolic handling of input values, while at the same time maintains interesting meta-level properties, such as being a congruence relation on processes. Open bisimulation has so far been studied for the π -calculus and its extension to the spi-calculus has not been fully understood. There is a recent attempt at formulating an open-style bisimulation for the spi-calculus [8], which is shown to be sound with respect to hedged bisimulation. However, no congruence results have been obtained for this notion of open bisimulation. We propose a different formulation of open bisimulation, which is inspired by hedged bisimulation. A collection of *up-to techniques* are defined, and shown to be sound. These up-to techniques can be used to finitely check the bisimilarity of processes in some cases and, more importantly, they are used to show that open bisimilarity is a congruence on finite spi-processes. The latter allows for compositional reasoning about open bisimilarity.

A crucial part in theories of environment-sensitive bisimulation is that of the consistency of the observer theory. A consistent theory guarantees that the induced equality on messages satisfies the usual axioms of equality, most impor-

tantly, transitivity. A difficulty in formulating open bisimulation is in finding a good symbolic representation of observer theories. One needs to make sure that the symbolic observer theories in the bisimulation set can be properly instantiated to consistent theories. The symbolic representation of observer theories used in this paper is based on Boreale’s *symbolic traces* [3]. A symbolic trace is a compact representation of a set of traces of a process, where the input values are represented by parameters (which are essentially names). Associated with a symbolic trace is a notion of consistency, i.e., it should be possible to instantiate the symbolic trace to a set of concrete traces. The definition of open bisimulation in Section 4 is indexed by pairs of symbolic traces, called *bi-traces*. A symbolic trace is essentially a list, and the position of a particular name in the list constrains its possible instantiations. In this sense, its position in the list enforces an implicit scoping of the name. Bi-traces are essentially observer theories with added structures. The notion of consistency of bi-traces is therefore based on the notion of consistency for observer theories, with the added constraint on the possible instantiations of names in the bi-traces. The latter gives rise to the notion of respectful substitutions, much like the same notion that appears in the definition of open bisimulation for the π -calculus.

A good definition of open bisimulation for the spi-calculus should naturally address the issue of name distinction. As in the definition of open bisimulation for the π -calculus, the fresh names extruded by a bound output action of a process should be considered distinct from all other pre-existing names. We employ a syntactic device to encode this distinction implicitly. We extend the language of processes with a countably infinite set of *rigid names*. Rigid names are not subject to instantiations and therefore cannot be identified by substitutions. Note that it is possible to formulate open bisimulation without the use of rigid names, at a price of an added complexity. The role of rigid names will be clear when we discuss open bisimulation in Section 4.

Outline of the paper Section 2 reviews some notations and the operational semantics for the spi-calculus. Section 3 presents the notion of observer theories along with its properties. Section 4 defines the bi-trace structure and open bisimulation, and states its soundness with respect to testing equivalence. Section 5 defines several up-to techniques for open bisimulation. The main purpose of these techniques is to show that open bisimilarity is closed under parallel composition and respectful substitutions, from which the soundness of open bisimulation and its congruence results follow. Section 6 shows that open bisimilarity is an equivalence relation on processes and also a congruence relation on finite spi-processes without rigid names. Section 7 concludes the paper and outlines some directions for future work. The detailed proofs are omitted but they can be found in the extended version of the paper [11].

2 The Spi Calculus

In this section we review the syntax and the operational semantics for the spi-calculus, following its original presentation as in [2]. We consider a more re-

stricted language, i.e., the one with only the pairing and encryption operators. We assume that the reader has some familiarity with the spi-calculus, so the meaning of various constructs of the calculus will not be explained in detail.

We assume a denumerable set of names, denoted with \mathcal{N} . We use $m, n, x, y,$ and z to range over names. In order to simplify the presentation of open bisimulation, we introduce another infinite set of names which we call *rigid names*, denoted with \mathcal{RN} , which are assumed to be of a distinct syntactic category from names. Rigid names are a purely syntactic device to simplify presentation. It can be thought of as names which are created when restricted names in processes are extruded in their transitions. Rigid names embody a notion of *distinction*, as in open bisimulation for the π -calculus [10], in the sense that they cannot be instantiated, thus cannot be identified with other rigid names. Rigid names are ranged over by bold lower-case letters, e.g., as in **a**, **b**, **c**, etc.

Messages in the spi calculus are given by the following grammar:

$$M, N ::= x \mid \mathbf{a} \mid \langle M, N \rangle \mid \{M\}_N$$

where $\langle M, N \rangle$ denotes a pair consisting of messages M and N , and $\{M\}_N$ denotes the message M encrypted with the key N . The set of processes is defined by the grammar:

$$\begin{aligned} P, Q, R ::= & 0 \mid \bar{M}\langle N \rangle.P \mid M(x).P \mid P|Q \mid (\nu x)P \mid !P \\ & \mid [M = N]P \mid \text{let } \langle x, y \rangle = M \text{ in } P \mid \text{case } L \text{ of } \{x\}_N \text{ in } P \end{aligned}$$

The names x and y in the restriction, the ‘let’ and the ‘case’ constructs are binding occurrences. We assume the usual α -equivalence on process expressions. Given a syntactic expression E , e.g., a process, a set of process, pairs, etc., we write $\text{fn}(E)$ to denote the set of free names in E . Likewise, $\text{rn}(E)$ denote the set of rigid names in E . We call a process P *pure* if there are no occurrences of rigid names in P .

A *substitution* is a mapping from names to messages. Substitutions are ranged over by θ, σ and ρ . The domain of substitutions is defined as $\text{dom}(\theta) = \{x \mid \theta(x) \neq x\}$. We consider only substitutions with finite domains. The substitution with empty domain is denoted by ϵ . We often enumerate the mappings of a substitution on its finite domain, using the notation $[M_1/x_1, \dots, M_n/x_n]$. Substitutions are generalised straightforwardly to mappings between terms (processes, messages, etc.), with the usual proviso that the free names in the substitutions do not become bound as a result of the applications of the substitutions. Applications of substitutions to terms (processes or messages) are written in postfix notation, e.g., as in $M\theta$. Composition of two substitutions θ and σ , written $(\theta \circ \sigma)$, is defined as follows: $M(\theta \circ \sigma) = (M\theta)\sigma$. Given a substitution θ and a finite set of names V , we denote with $\theta_{\uparrow V}$ the substitution which coincides with θ on the set V , and is the identity map everywhere else.

We use the operational semantics of the spi calculus as it is given in [1], with one small modification: we allow communication channels to be arbitrary messages, instead of just names. We do this in order to get a simpler formulation

$$\begin{array}{c}
\frac{}{M(x).P \xrightarrow{M} (x)P} \quad \frac{}{\bar{M}\langle N \rangle.P \xrightarrow{\bar{M}} \langle N \rangle P} \quad \frac{P > Q \quad Q \xrightarrow{\alpha} A}{P \xrightarrow{\alpha} A} \quad \frac{P \xrightarrow{\alpha} A}{P \mid Q \xrightarrow{\alpha} A \mid Q} \\
\\
\frac{P \xrightarrow{M} F \quad Q \xrightarrow{\bar{M}} C}{P \mid Q \xrightarrow{\tau} F@C} \quad \frac{Q \xrightarrow{\bar{N}} C \quad P \xrightarrow{N} F}{P \mid Q \xrightarrow{\tau} C@F} \quad \frac{Q \xrightarrow{\alpha} A}{P \mid Q \xrightarrow{\alpha} P \mid A} \quad \frac{P \xrightarrow{\alpha} A \quad m \notin \text{fn}(\alpha)}{(\nu m)P \xrightarrow{\alpha} (\nu m)A}
\end{array}$$

Fig. 1. The operational semantics of the spi calculus.

of open bisimulation in Section 4, since we do not need to keep track of certain constraints related to channel names.

The one-step transition relations are not relating processes with processes, rather processes with *agents*. The latter is presented using the notion of *abstraction* and *concretion* of processes. Abstractions are expressions of the form $(x)P$ where P is a process and the construct (x) binds free occurrences of x in P , and concretions are expressions of the form $(\nu \vec{x})\langle M \rangle P$ where M is a message and P is a process. Agents are ranged over by A, B and C . As with processes, we call an agent A *pure* if $\text{rn}(A) = \emptyset$.

To simplify the presentation of the operational semantics, we define compositions between processes and agents as follows. In the definition below we assume that $x \notin \{\vec{y}, z\} \cup \text{fn}(R)$ and $\{\vec{y}\} \cap \text{fn}(R) = \emptyset$.

$$\begin{aligned}
(\nu x)(z)P &\triangleq (z)(\nu x)P, & R \mid (x)P &\triangleq (x)(R \mid P), \\
(\nu x)(\nu \vec{y})\langle M \rangle Q &\triangleq (\nu x, \vec{y})\langle M \rangle Q, & \text{if } x \in \text{fn}(M) \\
(\nu x)(\nu \vec{y})\langle M \rangle Q &\triangleq (\nu \vec{y})\langle M \rangle (\nu x)Q, & \text{if } x \notin \text{fn}(M) \\
R \mid (\nu \vec{y})\langle M \rangle Q &\triangleq (\nu \vec{y})\langle M \rangle (R \mid Q)
\end{aligned}$$

The dual composition $A \mid R$ is defined symmetrically.

Given an abstraction $F = (x)P$ and a concretion $(\nu \vec{y})\langle M \rangle Q$, where $\{\vec{y}\} \cap \text{fn}(P) = \emptyset$, the *interactions* of F and C are defined as follows.

$$F@C \triangleq (\nu \vec{y})(P[M/x] \mid Q) \quad C@F \triangleq (\nu \vec{y})(Q \mid P[M/x])$$

We define a reduction relation $>$ on processes as follows:

$$\begin{array}{ll}
!P > P \mid !P & \text{let } \langle x, y \rangle = \langle M, N \rangle \text{ in } P > P[M/x][N/y] \\
[M = M]P > P & \text{case } \{M\}_N \text{ of } \{x\}_N \text{ in } P > P[M/x]
\end{array}$$

The operational semantics of the spi calculus is given in Figure 1. The action α can be either the silent action τ , a term M , or a *co-term* \bar{M} , where M is a term. We note that as far as the operational semantics is concerned, there is no distinction between a name and a rigid name; both can be used as channel names and as messages.

Testing equivalence In order to define testing equivalence, we first define the notion of a *barb*. A barb is an input or an output channel on which a process can communicate. We assume that barbs contain no rigid names. We denote the reflexive-transitive closure of the silent transition $\xrightarrow{\tau}$ with $\xrightarrow{\tau}^*$.

Definition 1. *Two pure processes P and Q are said to be testing equivalent, written $P \sim Q$, when for every pure process R and every barb β , if*

$$P \mid R \xrightarrow{\tau}^* P' \xrightarrow{\beta} A$$

for some P' and A , then $Q \mid R \xrightarrow{\tau}^ Q' \xrightarrow{\beta} B$ for some Q' and B , and vice versa.*

Notice that testing equivalence is defined for pure processes only, therefore our definition of testing equivalence coincides with that in [2].

3 Observer theory

An *observer theory* is just a finite set of pairs of messages. These pairs of messages denote the pairs of indistinguishable messages from the observer point of view. We adopt the convention that *all* names are entities known to the observer. Rigid names, on the other hand, may or may not be known to the observer, depending on whether they are present in the observer theory.

Associated with an observer theory are certain proof systems representing the deductive capability of the observer. These proof systems allow for derivation of new knowledge from existing ones. Observer theories are ranged over by Γ and Δ . We often refer to observer theory simply as *theory*. Given a theory Γ , we write $\pi_1(\Gamma)$ to denote the set $\{M \mid \exists N.(M, N) \in \Gamma\}$, and likewise, $\pi_2(\Gamma)$ to denote the set $\{N \mid \exists M.(M, N) \in \Gamma\}$. The observer can encrypt and decrypt messages it has in order to either analyze or synthesize messages to deduce the equality of messages. This deductive capability is presented as a proof system in Figure 2. This proof system is a straightforward adaptation of the standard proof systems for message analysis and synthesis, usually presented in a natural-deduction style, e.g., as found in [3], to sequent calculus. We find sequent calculus a more natural setting to prove various properties of observer theories. The sequent $\Gamma \vdash M \leftrightarrow N$ means that the messages M and N are indistinguishable in the theory Γ . We shall often write $\Gamma \vdash M \leftrightarrow N$ to mean that the sequent $\Gamma \vdash M \leftrightarrow N$ is derivable using the rules in Figure 2. Notice that in the proof system in Figure 2, two names are indistinguishable if they are syntactically equal. This reflects the fact that names are entities known to the observer.

It is useful to consider the set of messages that can be constructed by an observer in its interaction with a particular process. This synthesis of messages follows the inference rules given in Figure 3. The symbol Σ denotes a finite set of messages. We overload the symbols \vdash and \vdash to denote, respectively, sequents and derivability relation of messages given a set of messages. The rules for message synthesis are just a projection of the rules for message equivalence.

$$\begin{array}{c}
\frac{x \in \mathcal{N}}{\Gamma \vdash x \leftrightarrow x} \text{ var} \quad \frac{}{\Gamma, (M, N) \vdash M \leftrightarrow N} \text{ id} \quad \frac{\Gamma \vdash M \leftrightarrow M' \quad \Gamma \vdash N \leftrightarrow N'}{\Gamma \vdash \langle M, N \rangle \leftrightarrow \langle M', N' \rangle} \text{ pr} \\
\\
\frac{\Gamma, (\langle M_1, N_1 \rangle, \langle M_2, N_2 \rangle), (M_1, M_2), (N_1, N_2) \vdash M \leftrightarrow N}{\Gamma, (\langle M_1, N_1 \rangle, \langle M_2, N_2 \rangle) \vdash M \leftrightarrow N} \text{ pl} \\
\\
\frac{\Gamma \vdash M \leftrightarrow M' \quad \Gamma \vdash N \leftrightarrow N'}{\Gamma \vdash \{M\}_N \leftrightarrow \{M'\}_{N'}} \text{ er} \\
\\
\frac{\Gamma' \vdash N_1 \leftrightarrow N_2 \quad \Gamma', (M_1, M_2), (N_1, N_2) \vdash M \leftrightarrow N}{\Gamma, (\{M_1\}_{N_1}, \{M_2\}_{N_2}) \vdash M \leftrightarrow N} \text{ el}
\end{array}$$

Fig. 2. Proof system for deriving message equivalence. In the rule *el*, Γ' is the set $\Gamma \cup \{(\{M_1\}_{N_1}, \{M_2\}_{N_2})\}$.

$$\begin{array}{c}
\frac{x \in \mathcal{N}}{\Sigma \vdash x} \text{ var} \quad \frac{}{\Sigma, M \vdash M} \text{ id} \quad \frac{\Sigma \vdash M \quad \Sigma \vdash N}{\Sigma \vdash \langle M, N \rangle} \text{ pr} \quad \frac{\Sigma \vdash M \quad \Sigma \vdash N}{\Sigma \vdash \{M\}_N} \text{ er} \\
\\
\frac{\Sigma, \langle M, N \rangle, M, N \vdash R}{\Sigma, \langle M, N \rangle \vdash R} \text{ pl} \quad \frac{\Sigma, \{M\}_N \vdash N \quad \Sigma, \{M\}_N, M, N \vdash R}{\Sigma, \{M\}_N \vdash R} \text{ el}
\end{array}$$

Fig. 3. Proof system for message synthesis

A nice feature of the sequent calculus formulation is that in any derivation of a judgment, every judgment in the derivation contains only subterms occurring in the judgment at the root of the derivation tree. This gives us immediately a bound on the depth of the derivation tree, hence the decidability of the proof systems.

Proposition 2. *Given any Γ , Σ , M and N , it is decidable whether the judgments $\Gamma \vdash M \leftrightarrow N$ and $\Sigma \vdash M$ hold.*

Consistency of observer theory. Recall that the motivation behind the notion of message equivalence \leftrightarrow is for it to replace syntactic equality in the definition of bisimulation. Since the relation \leftrightarrow is parameterised upon an observer theory, we shall investigate under what conditions an observer theory gives rise to a well-behaved relation \leftrightarrow . In the literature of bisimulation for the spi calculus, this notion is usually referred to as the *consistency* property of observer theories. We define an abstract notion of theory consistency, based on the entailment relation \vdash defined previously. We later show that this abstract notion of consistency is equivalent to a more concrete one which is finitely checkable.

Definition 3. *A theory Γ is consistent if for every M and N , if $\Gamma \vdash M \leftrightarrow N$ then the following hold:*

1. *M and N are of the same type of expressions, i.e., M is a pair (an encrypted message, a (rigid) name) if and only if N is.*

2. If $M = \{M_1\}_{M_2}$ and $N = \{N_1\}_{N_2}$ then $\pi_1(\Gamma) \vdash M_2$ implies $\Gamma \vdash M_2 \leftrightarrow N_2$ and $\pi_2(\Gamma) \vdash N_2$ implies $\Gamma \vdash M_2 \leftrightarrow N_2$.
3. For any R , $\Gamma \vdash M \leftrightarrow R$ implies $R = N$ and $\Gamma \vdash R \leftrightarrow N$ implies $R = M$.

The first condition in Definition 3 states that the equality relation \leftrightarrow respects types, i.e., it is not possible that an operation (pairing, encryption) on M succeeds while the same operation on N fails. The second condition states that both projections of the theory contain “equal” amount of knowledge, e.g., it is not possible that one message decrypts while the other fails to. The third condition states the unicity of \leftrightarrow .

Characterisation of consistent theories The notion of consistency as defined in Definition 3 is not obvious to check since it involves quantification over all equivalent pairs of messages. We show that a theory can be reduced to a certain normal form for which there exist finitely checkable properties that entail consistency of the original theory. For this purpose, we define a rewrite relation on theories.

Definition 4. *The rewrite relation \longrightarrow on observer theories is defined as follows:*

$$\begin{aligned} \Gamma, (\langle M, N \rangle, \langle M', N' \rangle) &\longrightarrow \Gamma, (M, M'), (N, N') \\ \Gamma, (\{M\}_N, \{M'\}_{N'}) &\longrightarrow \Gamma, (M, M'), (N, N') \\ &\text{if } \Gamma, (\{M\}_N, \{M'\}_{N'}) \vdash N \leftrightarrow N'. \end{aligned}$$

A theory Γ is irreducible if Γ cannot be rewritten to any other theory. Γ is an irreducible form of another theory Γ' if Γ is irreducible and $\Gamma' \longrightarrow^* \Gamma$.

The rewrite relation on theories defined above can be shown to be terminating and confluent, hence every theory Γ has a unique irreducible form, which we denote here with $\Gamma \Downarrow$. Moreover, the reduction can be shown to preserve consistency. Therefore to check the consistency of a theory, it is enough to check its irreducible form.

Proposition 5. *A theory Γ is consistent if and only if $\Gamma \Downarrow$ satisfies the following conditions: if $(M, N) \in \Gamma \Downarrow$ then*

- (a) M and N are of the same type of expressions,
- (b) if $M = \{M_1\}_{M_2}$ and $N = \{N_1\}_{N_2}$ then $\pi_1(\Gamma \Downarrow) \not\vdash M_2$ and $\pi_2(\Gamma \Downarrow) \not\vdash N_2$.
- (c) for any $(U, V) \in \Gamma \Downarrow$, $U = M$ if and only if $V = N$.

Closure under substitutions The entailment relation \vdash is in general not closed under arbitrary substitutions, the reason being the inclusion of the *var*-rule. Using this rule, we can prove, for instance, $\emptyset \vdash x \leftrightarrow x$. Now if we substitute \mathbf{a} for x , where \mathbf{a} is some rigid name, we do not have $\emptyset \vdash \mathbf{a} \leftrightarrow \mathbf{a}$, since the *var*-rule does not apply to rigid names.

We shall often work with substitution pairs in the following sections. Application of a substitution pair $\vec{\theta} = (\theta_1, \theta_2)$ to a pair of terms (M, N) is defined to be $(M\theta_1, N\theta_2)$. This extends straightforwardly to application of substitution pairs to sets or lists of pairs. The following lemma gives a class of substitutions under which the entailment relation is preserved.

Lemma 6. *Let $\Gamma \vdash M \leftrightarrow N$ and let $\vec{\theta} = (\theta_1, \theta_2)$ be a substitution pair such that for all $x \in \text{fn}(\Gamma, M, N)$ it holds that $\Gamma \vec{\theta} \vdash \theta_1(x) \leftrightarrow \theta_2(x)$. Then $\Gamma \vec{\theta} \vdash M\theta_1 \leftrightarrow N\theta_2$.*

4 Open bisimulation

Open bisimulation for the spi-calculus to be presented in this section is similar to other environment-sensitive bisimulations, in the sense that it is also indexed by some structure representing the knowledge of the environment. A candidate for representing this knowledge is the observer theory presented earlier. However, since the crucial feature of open bisimulation is the symbolic representation of input values, extra structures need to be added to observer theories to capture dependencies between various symbolic input values at different stages of bisimulation checking. The notion of *symbolic traces* as defined in [3] conveniently captures this sort of dependency. Open bisimulation is indexed by pairs of a variant of symbolic traces, called *bi-traces*. The important properties we need to establish regarding bi-traces are that they can be soundly interpreted as observer theories, and they behave well with respect to substitutions of input values.

In the following, we use the notation $[x_1, \dots, x_n]$ to denote a list whose elements are x_1, \dots, x_n . The empty list is denoted by $[\]$. Concatenation of a list l_1 with another list l_2 is denoted with $l_1.l_2$, if l_2 is appended to the end of l_1 . If l_2 is a singleton list, say $[x]$, then we write $l_1.x$ instead of $l_1.[x]$, likewise $x.l_1$ instead of $[x].l_1$.

Definition 7. *An I/O pair is a pair of messages marked with i (indicating input) or o (indicating output), i.e., it is of the form $(M, N)^i$ or $(M, N)^o$. A bi-trace is a list of I/O message pairs, ranged over by h . We denote with $\pi_1(h)$ the list obtained from h by taking the first components of the pairs in h . The list $\pi_2(h)$ is defined analogously. Bi-traces are subject to the following restriction: if $h = h_1.(M, N)^o.h_2$ then $\text{fn}(M, N) \subseteq \text{fn}(h_1)$. If h is $[(M_1, N_1)^{l_1}, \dots, (M_k, N_k)^{l_k}]$ then the inverse of h , written h^{-1} , is the list $[(N_1, M_1)^{l_1}, \dots, (N_k, M_k)^{l_k}]$. We write $\{h\}$ to denote the set $\{(M, N) \mid (M, N)^i \in h \text{ or } (M, N)^o \in h\}$.*

The underlying idea in the bi-trace representation is that *names are symbolic values*. This explains the requirement that the free names of an output pair in a bi-trace must appear before the output pair. In other words, input values (i.e., names) are created only at input pairs.

Given a bi-trace h , the underlying set $\{h\}$ is obviously an observer theory, hence bi-traces are essentially theories with added structures. As in symbolic traces [3], bi-traces consistency needs to take into account the fact that their instantiations correspond to concrete traces. Consistency conditions for bi-traces are more complicated since we need extra conditions ensuring the consistency of the underlying observer theory. We first define a notion of respectful substitutions for bi-traces, which is later used to define the notion of consistency for bi-traces. In the following we shall write $h \vdash M \leftrightarrow N$, instead of a more type-correct version $\{h\} \vdash M \leftrightarrow N$, when we consider an equivalent pair of messages

under the theory obtained from a bi-trace h . Application of a substitution pair (θ_1, θ_2) to a bi-trace is defined element-wise in a straightforward way.

Definition 8. A substitution pair $\vec{\theta} = (\theta_1, \theta_2)$ respects a bi-trace h if whenever $h = h_1.(M, N)^i.h_2$, then for every $x \in \text{fn}(M, N)$ it holds that $h_1\vec{\theta} \vdash x\theta_1 \leftrightarrow x\theta_2$.

Definition 9. We define the notion of consistent bi-traces inductively on the length of bi-traces as follows:

1. The empty bi-trace is consistent.
2. If h is a consistent bi-trace then $h.(M, N)^i$ is also a consistent bi-trace, provided that $h \vdash M \leftrightarrow N$.
3. If h is a consistent bi-trace, then $h' = h.(M, N)^o$ is a consistent bi-trace, provided that for every h -respectful substitution pair $\vec{\theta}$, if $h\vec{\theta}$ is a consistent bi-trace then $\{h'\vec{\theta}\}$ is a consistent theory.

The requirement that every input pair be deducible from its predecessors in the bi-trace captures the dependency of the names of the input pair on their preceding input/output pairs. At this point, it is instructive to examine the case where the elements of bi-traces are pairs of names or rigid names. Consider for example the bi-trace $(x, x)^i.(\mathbf{a}, \mathbf{a})^o.(y, y)^i.(\mathbf{b}, \mathbf{b})^o$. There is a respectful substitution that identifies x and y , or y with \mathbf{a} , but there are no respectful substitutions that identify x with \mathbf{a} , y with \mathbf{b} nor \mathbf{a} with \mathbf{b} . Thus this bi-trace captures a restricted notion of distinction [10]. Rigid names encode an implicit distinction: no two rigid names can be identified by substitutions, whereas the position of names encode their respective scopes.

Note that in item (3) in Definition 9, we quantify over all respectful substitutions. This is unfortunate from the viewpoint of bisimulation checking but it is unavoidable if we want the notion of consistency to be closed under respectful substitutions. Consider the bi-trace:

$$(\mathbf{a}, \mathbf{a})^o.(\mathbf{b}, \mathbf{b})^o.(x, x)^i.(\{x\}_{\mathbf{k}}, \{\mathbf{a}\}_{\mathbf{k}})^o.(\{\mathbf{b}\}_{\mathbf{k}}, \{x\}_{\mathbf{k}})^o.$$

If we drop the quantification on respectful substitutions, then this trace would be considered consistent. Under the respectful substitution pair $([\mathbf{b}/x], [\mathbf{b}/x])$, however, the above bi-trace becomes

$$(\mathbf{a}, \mathbf{a})^o.(\mathbf{b}, \mathbf{b})^o.(\mathbf{b}, \mathbf{b})^i.(\{\mathbf{b}\}_{\mathbf{k}}, \{\mathbf{a}\}_{\mathbf{k}})^o.(\{\mathbf{b}\}_{\mathbf{k}}, \{\mathbf{b}\}_{\mathbf{k}})^o$$

which gives rise to an inconsistent theory.

Definition 10. A traced process pair is a triple (h, P, Q) where h is a bi-trace, P and Q are processes such that $\text{fn}(P, Q) \subseteq \text{fn}(h)$. Let \mathcal{R} be a set of traced process pairs. We write $h \vdash P \mathcal{R} Q$ to denote the fact that $(h, P, Q) \in \mathcal{R}$. \mathcal{R} is consistent if for every $h \vdash P \mathcal{R} Q$, h is consistent. The inverse of \mathcal{R} , written \mathcal{R}^{-1} , is the set $\{(h^{-1}, Q, P) \mid (h, P, Q) \in \mathcal{R}\}$. \mathcal{R} is symmetric if $\mathcal{R} = \mathcal{R}^{-1}$.

Definition 11. A bi-trace h is called a universal bi-trace if h consists only of input-pairs of names, i.e., it is of the form $(x_1, x_1)^i \cdots (x_n, x_n)^i$, where each x_i is a name.

Definition 12. Open bisimulation. A set of traced process pairs \mathcal{R} is a strong open bisimulation if \mathcal{R} is consistent and symmetric, and if $h \vdash P \mathcal{R} Q$ then for all substitution pair $\vec{\theta} = (\theta_1, \theta_2)$ that respects h , the following hold:

1. If $P\theta_1 \xrightarrow{\tau} P'$ then there exists Q' such that $Q\theta_2 \xrightarrow{\tau} Q'$ and $h\vec{\theta} \vdash P' \mathcal{R} Q'$.
2. If $P\theta_1 \xrightarrow{M} (x)P'$, where $x \notin \text{fn}(h\vec{\theta})$, and $\pi_1(h\vec{\theta}) \vdash M$ then there exists Q' such that $Q\theta_2 \xrightarrow{N} (x)Q'$ and $h\vec{\theta}.(M, N)^i.(x, x)^i \vdash P' \mathcal{R} Q'$.
3. If $P\theta_1 \xrightarrow{\vec{M}} (\nu\vec{x})(M')P'$, and $\pi_1(h\vec{\theta}) \vdash M$ then there exist N, N' and Q' such that $Q\theta_2 \xrightarrow{\vec{N}} (\nu\vec{y})(N')Q'$, and

$$h\vec{\theta}.(M, N)^i.(M'[\vec{c}/\vec{x}], N'[\vec{d}/\vec{y}])^o \vdash P'[\vec{c}/\vec{x}] \mathcal{R} Q'[\vec{d}/\vec{y}],$$

where $\{\vec{c}, \vec{d}\} \cap \text{rn}(h\vec{\theta}, P\theta_1, Q\theta_2) = \emptyset$.

We denote with \approx_o the union of all open bisimulations. We say that P and Q are strong open h -bisimilar, written $P \sim_o^h Q$, if $(h, P, Q) \in \approx_o$. They are said to be strong open bisimilar, written $P \sim_o Q$, if $\text{rn}(P, Q) = \emptyset$ and $P \sim_o^h Q$ for a universal bi-trace h .

Notice that in the bound output case, the restricted names in the concretions are replaced by fresh rigid names. Notice also that strong open bisimilarity \sim_o is defined on pure processes, i.e., those processes without free occurrences of rigid names. We now show that open bisimilarity is sound with respect to testing equivalence. Its proof follows straightforwardly from the fact that open bisimilarity is closed under parallel composition (see Section 5 and Section 6).

Theorem 13. Soundness. If $P \sim_o Q$ then $P \sim Q$.

5 Up-to techniques

We define several up-to techniques for open bisimulation. The main purpose of these techniques is to prove congruence results for open bisimilarity, in particular, closure under parallel composition, and to prove soundness of open bisimilarity with respect to testing equivalence. Up-to techniques are also useful in checking bisimulation since in certain cases it allows one to finitely demonstrate bisimilarity of processes. The proof techniques used in this section derive mainly from the work of Boreale et al. [4]. We first need to introduce several notions, parallel to those in [4], and adapting their up-to techniques to open bisimulation.

Open bisimilarity for the spi-calculus is not closed under parallel composition with arbitrary processes, since these extra processes might introduce inconsistency into the observer theory or may reveal other knowledge that causes the composed processes to behave differently. Therefore, in defining closure under parallel composition, we need to make sure that the processes we are composing with do not reveal or add any extra information for the observer. This is done by restricting the composition to processes obtained by instantiating pure processes with the current knowledge of the observer. This is defined via a notion of equivalent substitutions.

Definition 14. Let h be a consistent bi-trace. Given two substitutions θ_1 and θ_2 , we say that θ_1 is h -equivalent to θ_2 , written $\theta_1 \leftrightarrow_h \theta_2$, if $\text{dom}(\theta_1) = \text{dom}(\theta_2)$ and for every $x \in \text{dom}(\theta_1)$, we have $h \vdash x\theta_1 \leftrightarrow x\theta_2$ and $\text{fn}(x\theta_1, x\theta_2) \subseteq \text{fn}(h)$. A substitution σ extends θ , written $\theta \preceq \sigma$, if $\sigma(x) = \theta(x)$ for every $x \in \text{dom}(\theta)$.

The next lemma is crucial to the soundness of up-to parallel composition. It shows that one-step transitions for pure processes are invariant under equivalent substitutions.

Lemma 15. Let h be a consistent bi-trace, let σ_1 and σ_2 be substitutions such that $\sigma_1 \leftrightarrow_h \sigma_2$, and let R be a process such that $\text{fn}(R) \subseteq \text{dom}(\sigma_1)$ and $\text{rn}(R) = \emptyset$. If $R\sigma_1 \xrightarrow{M} R'$ then there exist $\sigma'_1 \preceq \sigma_1$, $\sigma'_2 \preceq \sigma_2$, U and Q such that $\sigma'_1 \leftrightarrow_h \sigma'_2$, $\text{fn}(U, Q) \subseteq \text{dom}(\sigma'_1)$, $\text{rn}(U, Q) = \emptyset$, $M = U\sigma'_1$, $R' = Q\sigma'_1$ and $R\sigma_2 \xrightarrow{U\sigma'_2} Q\sigma'_2$.

We need a few relations on bi-traces to describe the up-to techniques.

Definition 16. The relations $<_i$, $<_o$ and $<_f$ on bi-traces are defined as follows. Given two bi-traces h and h' :

- weakening: $h <_w h'$ holds if $h = h_1.h_2$ and $h' = h_1.(M, N)^*.h_2$, where $* \in \{i, o\}$ and $\text{fn}(M, N) \subseteq \text{fn}(h_1)$,
- contraction: $h <_c h'$ holds if $h = h_1.(M, N)^*.h_2$ and $h' = h_1.h_2$, where $* \in \{i, o\}$, and $h_1 \vdash M \leftrightarrow N$, and
- flex-rigid: $h <_f h'$ holds if $h = h_1.(\mathbf{c}, \mathbf{c})^\circ.h_2[\mathbf{c}/x]$, $h' = h_1.(x, x)^i.h_2$, $x \notin \text{fn}(h_1)$ and $\mathbf{c} \notin \text{rn}(h_1.h_2)$.

The reflexive-transitive closures of $<_w$, $<_c$ and $<_f$ are denoted, respectively, by \sqsubseteq_w , \sqsubseteq_c and \sqsubseteq_f . If $h \sqsubseteq_f h'$ then $h = h'\theta$ for a unique substitution θ with $\text{dom}(\theta) \subseteq \text{fn}(h')$. We denote this substitution with $\theta_{h, h'}$.

Reading from right-to-left, the above relations read as follows: The relation $<_w$ removes an arbitrary pair from the bi-trace (hence possibly reducing the knowledge of the observer). The relation $<_c$ adds a redundant pair, i.e., one which is deducible from the current knowledge, hence adding no extra knowledge. The relation $<_f$ replaces a variable input pair with a fresh output pair of rigid names. It does not increase the knowledge of the observer, since the added pair is fresh value, but it does limit the possible respectful substitutions, since the fresh output pair cannot be substituted (they are rigid names). Thus, going from right to left in the relations, the knowledge of the observer does not increase.

In the following, the notation \equiv denotes the structural equivalence on processes as defined in [2].

Definition 17. Given a set of consistent traced process pairs \mathcal{R} , define \mathcal{R}_t , for $t \in \{\equiv, w, c, s, i, f, r, p\}$, as the least relations containing \mathcal{R} which satisfy the following rules:

1. up to structural equivalence:
$$\frac{P \equiv P', Q \equiv Q' \text{ and } h \vdash P' \mathcal{R} Q'}{h \vdash P \mathcal{R}_\equiv Q} \equiv$$

2. up to weakening:
$$\frac{h \vdash P \mathcal{R} Q, h' \sqsubseteq_w h \text{ and } h' \text{ is consistent}}{h' \vdash P \mathcal{R}_w Q} w$$
3. up to contraction:
$$\frac{h \vdash P \mathcal{R} Q, h' \sqsubseteq_c h \text{ and } h' \text{ is consistent}}{h' \vdash P \mathcal{R}_c Q} c$$
4. up to substitutions:
$$\frac{h \vdash P \mathcal{R} Q \text{ and } \vec{\theta} = (\theta_1, \theta_2) \text{ respects } h}{h\vec{\theta} \vdash P\theta_1 \mathcal{R}_s Q\theta_2} s$$
5. up to injective renaming of rigid names:
$$\frac{h \vdash P \mathcal{R} Q, \rho_1 \text{ and } \rho_2 \text{ are injective renaming on rigid names}}{h(\rho_1, \rho_2) \vdash P\rho_1 \mathcal{R}_i Q\rho_2} i$$
6. up to flex-rigid reversal of names:
$$\frac{h \vdash P \mathcal{R} Q, h' \sqsubseteq_f h}{h' \vdash P\theta_{h',h} \mathcal{R}_f Q\theta_{h',h}} f$$
7. up to restriction:
$$\frac{h \vdash P[\vec{c}/\vec{x}] \mathcal{R} Q[\vec{d}/\vec{y}], \quad \{\vec{c}\} \cap \text{rn}(\pi_1(h), P) = \emptyset, \quad \{\vec{d}\} \cap \text{rn}(\pi_2(h), Q) = \emptyset, \quad \{\vec{x}, \vec{y}\} \cap \text{fn}(h) = \emptyset}{h \vdash (\nu\vec{x})P \mathcal{R}_r (\nu\vec{y})Q} r$$
8. up to parallel composition:
$$\frac{h \vdash P \mathcal{R} Q, \quad h' \text{ is consistent, } h' \sqsubseteq_c h, \quad \sigma_1 \leftrightarrow_{h'} \sigma_2, \quad \text{fn}(R) \subseteq \text{dom}(\sigma_1), \quad \text{rn}(R) = \emptyset, \quad A \equiv (P \mid R\sigma_1) \text{ and } B \equiv (Q \mid R\sigma_2)}{h' \vdash A \mathcal{R}_p B} p$$

Strong open bisimulation up to structural equivalence is defined similarly to Definition 12, except that we replace the relation \mathcal{R} in items (1), (2) and (3) in Definition 12 with \mathcal{R}_\equiv . Strong open bisimulation up to weakening, contraction, substitutions, injective renaming, flex-rigid reversal, restrictions and parallel composition are defined analogously.

In those rules that concern weakening, contraction and flex-rigid reversal of names, the observer knowledge in the premise is always equal or greater than its knowledge in the conclusion. In other words, if the observer cannot distinguish two processes using its current knowledge, it cannot do so either in a reduced knowledge. In the rule for parallel composition, we allow only processes that can introduce no extra information to the observer. Notice that in the rule, for technical reason, we need to contract the bi-trace h to allow $R\sigma_i$ to contain new names not already in h .

Proposition 18. *Let \mathcal{R} be an open bisimulation up to structural equivalence (respectively, weakening, contraction, etc.). Then $\mathcal{R} \subseteq \mathcal{R}_\equiv \subseteq \approx_o$ (respectively, $\mathcal{R} \subseteq \mathcal{R}_t \subseteq \approx_o$, for $t \in \{w, c, s, i, f, r, p\}$).*

Example 19. This example demonstrates the use of the up-to techniques in proving bisimilarity. This example is adapted from a similar one in [5]. Let P and Q be the following processes:

$$P = \mathbf{a}(x).(\nu k)\bar{\mathbf{a}}\langle\{x\}_k\rangle.(\nu m)\bar{\mathbf{a}}\langle\{m\}_{\{\mathbf{a}\}_k}\rangle.\bar{m}\langle\mathbf{a}\rangle.0$$

$$Q = \mathbf{a}(x).(\nu k)\bar{\mathbf{a}}\langle\{x\}_k\rangle.(\nu m)\bar{\mathbf{a}}\langle\{m\}_{\{\mathbf{a}\}_k}\rangle.[x = \mathbf{a}]\bar{\mathbf{m}}\langle\mathbf{a}\rangle.0$$

Let \mathcal{R} be the least set such that:

$$\begin{aligned} &(\mathbf{a}, \mathbf{a})^\circ \vdash P \mathcal{R} Q, \quad (\mathbf{a}, \mathbf{a})^\circ.(x, x)^i \vdash P_1 \mathcal{R} Q_1, \\ &(\mathbf{a}, \mathbf{a})^\circ.(x, x)^i.\{\{x\}_k, \{x\}_k\}^\circ \vdash P_2 \mathcal{R} Q_2, \\ &(\mathbf{a}, \mathbf{a})^\circ.(x, x)^i.\{\{x\}_k, \{x\}_k\}^\circ.\{\{\mathbf{m}\}_{\{\mathbf{a}\}_k}, \{\mathbf{m}\}_{\{\mathbf{a}\}_k}\}^\circ \vdash P_3 \mathcal{R} Q_3, \\ &(\mathbf{a}, \mathbf{a})^\circ.(\mathbf{a}, \mathbf{a})^i.\{\{\mathbf{a}\}_k, \{\mathbf{a}\}_k\}^\circ.\{\{\mathbf{m}\}_{\{\mathbf{a}\}_k}, \{\mathbf{m}\}_{\{\mathbf{a}\}_k}\}^\circ.(\mathbf{m}, \mathbf{m})^i.(\mathbf{a}, \mathbf{a})^\circ \vdash 0 \mathcal{R} 0, \end{aligned}$$

where

$$\begin{aligned} P_1 &= (\nu k)\bar{\mathbf{a}}\langle\{x\}_k\rangle.(\nu m)\bar{\mathbf{a}}\langle\{m\}_{\{\mathbf{a}\}_k}\rangle.\bar{\mathbf{m}}\langle\mathbf{a}\rangle.0, \\ Q_1 &= (\nu k)\bar{\mathbf{a}}\langle\{x\}_k\rangle.(\nu m)\bar{\mathbf{a}}\langle\{m\}_{\{\mathbf{a}\}_k}\rangle.[x = \mathbf{a}]\bar{\mathbf{m}}\langle\mathbf{a}\rangle.0, \\ P_2 &= (\nu m)\bar{\mathbf{a}}\langle\{m\}_{\{\mathbf{a}\}_k}\rangle.\bar{\mathbf{m}}\langle\mathbf{a}\rangle.0, \quad Q_2 = (\nu m)\bar{\mathbf{a}}\langle\{m\}_{\{\mathbf{a}\}_k}\rangle.[x = \mathbf{a}]\bar{\mathbf{m}}\langle\mathbf{a}\rangle.0, \\ P_3 &= \bar{\mathbf{m}}\langle\mathbf{a}\rangle.0, \quad Q_3 = [x = \mathbf{a}]\bar{\mathbf{m}}\langle\mathbf{a}\rangle.0. \end{aligned}$$

Let \mathcal{R}' be the symmetric closure of \mathcal{R} . Then it is easy to see that \mathcal{R}' is an open bisimulation up-to contraction and substitutions. For instance, consider the traced process pair $h \vdash \bar{\mathbf{m}}\langle\mathbf{a}\rangle.0 \mathcal{R}' [x = \mathbf{a}]\bar{\mathbf{m}}\langle\mathbf{a}\rangle.0$

where $h = (\mathbf{a}, \mathbf{a})^\circ.(x, x)^i.\{\{x\}_k, \{x\}_k\}^\circ.\{\{\mathbf{m}\}_{\{\mathbf{a}\}_k}, \{\mathbf{m}\}_{\{\mathbf{a}\}_k}\}^\circ$. Let $\vec{\theta} = (\theta_1, \theta_2)$ be an h -respectful substitution. Since x is the only name in h , we have

$$h\vec{\theta} = (\mathbf{a}, \mathbf{a})^\circ.(s, t)^i.\{\{s\}_k, \{t\}_k\}^\circ.\{\{\mathbf{m}\}_{\{\mathbf{a}\}_k}, \{\mathbf{m}\}_{\{\mathbf{a}\}_k}\}^\circ,$$

where $s = x\theta_1$ and $t = x\theta_2$. We have to check that every detectable action from $\bar{\mathbf{m}}\langle\mathbf{a}\rangle.0$ can be matched by $[t = \mathbf{a}]\bar{\mathbf{m}}\langle\mathbf{a}\rangle.0$. If $t \neq \mathbf{a}$, then $s \neq \mathbf{a}$ (by the consistency of $h\vec{\theta}$), therefore, $\pi_1(h\vec{\theta}) \not\vdash \mathbf{m}$, i.e., the action \mathbf{m} is not detected by the environment, so this case is trivial. If $t = \mathbf{a}$, then $s = \mathbf{a}$ and $h\vec{\theta} \vdash \mathbf{m} \leftrightarrow \mathbf{m}$, so both $P_3\theta_1$ and $Q_3\theta_2$ can make a transition on channel \mathbf{m} . Their continuation is the traced process pair

$$(\mathbf{a}, \mathbf{a})^\circ.(\mathbf{a}, \mathbf{a})^i.\{\{\mathbf{a}\}_k, \{\mathbf{a}\}_k\}^\circ.\{\{\mathbf{m}\}_{\{\mathbf{a}\}_k}, \{\mathbf{m}\}_{\{\mathbf{a}\}_k}\}^\circ.(\mathbf{m}, \mathbf{m})^i.(\mathbf{a}, \mathbf{a})^\circ \vdash 0 \mathcal{R}' 0$$

which is in the set \mathcal{R}' , hence also in \mathcal{R}'_{cs} (up-to contraction and substitution on \mathcal{R}'). Therefore by Proposition 18, $(\mathbf{a}, \mathbf{a})^\circ \vdash P \approx_o Q$. \square

6 Congruence results for open bisimilarity

In this section we show that the relation \sim_o on pure processes is an equality relation (reflexive, symmetric, transitive) and is closed under arbitrary pure process contexts without replication. To show reflexivity, we define a bisimulation set indexed by *reflexive bi-traces*. Reflexive bi-traces are consistent bi-traces such that its first and second projections are the same list.

Lemma 20. *The following set is an open bisimulation:*

$$\{(h, P, P) \mid (h, P, P) \text{ is a traced process pair, } h \text{ is consistent and reflexive}\}.$$

To show transitivity, we first need to define composition of bi-traces.

Definition 21. Composition of bi-traces. *Two bi-traces can be composed if they have the same length and match element wise. More precisely, given two bi-traces*

$$h_1 = [(R_1, T_1)^{p_1}, \dots, (R_m, T_m)^{p_m}]$$

$$h_2 = [(U_1, V_1)^{q_1}, \dots, (U_n, V_n)^{q_n}]$$

we say h_1 is left-composable to h_2 (equivalently, h_2 is right-composable to h_1) if and only if $m = n$ and $T_k = U_k$ and $p_k = q_k$ for every $k \in \{1, \dots, n\}$. Their composition, written $h_1 \circ h_2$, is $[(R_1, V_1)^{p_1}, \dots, (R_m, V_m)^{p_m}]$

The important properties of composition are that it preserves consistency and that it behaves well with respect to respectful substitutions. The latter is made precise in the following lemma.

Lemma 22. Separating substitution. *Let h_1 and h_2 be consistent and composable bi-traces such that $h_1 \circ h_2$ is also consistent. Let (θ_1, θ_2) be a substitution pair that respects $h_1 \circ h_2$. Then there exists a substitution ρ such that (θ_1, ρ) respects h_1 and (ρ, θ_2) respects h_2 .*

Given two sets of traced process pairs \mathcal{R}_1 and \mathcal{R}_2 , their composition, written $\mathcal{R}_1 \circ \mathcal{R}_2$, is the set

$$\{(h_1 \circ h_2, P, R) \mid h_1 \vdash P \mathcal{R}_1 Q, h_2 \vdash Q \mathcal{R}_2 R \text{ and } h_1 \text{ is left-composable with } h_2\}.$$

Lemma 23. *If \mathcal{R}_1 and \mathcal{R}_2 are open bisimulations then $\mathcal{R}_1 \circ \mathcal{R}_2$ is also an open bisimulation.*

Theorem 24. *The relation \sim_o is an equivalence relation on pure processes.*

Proof. It follows straightforwardly from Lemma 20, Lemma 23 and Definition 12.

We now proceed to showing that it is also a congruence, for *finite* pure processes. This follows almost directly from Proposition 18.

Theorem 25. *The relation \sim_o is a congruence on finite pure processes.*

7 Conclusion and future work

We have shown a formulation of open bisimulation for the spi-calculus. In this formulation, bisimulation is indexed by pairs of symbolic traces that concisely encode the history of the interaction between the environment with the processes being checked for bisimilarity. We show that open bisimilarity is a congruence for finite pure processes and is sound with respect to testing equivalence. For the latter, we note that with some minor modifications, we can also show soundness of open bisimilarity with respect to barbed congruence. Our formulation is directly inspired by hedged bisimulation [6]. In fact, open bisimilarity can be shown to be sound with respect to hedged bisimulation. Comparison with

hedged bisimulation and other formulations of bisimulation for the spi-calculus is left for future work.

It would be interesting to see how the congruence results extend to the case with replications or recursions. This will probably require a more general definition of the rule for up-to parallel composition. The definition of open bisimulation and the consistency of bi-traces make use of quantification over respectful substitutions. We will investigate whether there is a finite characterisation of consistent bi-traces. One possibility is to use a symbolic transition system, i.e., a transition system parameterised upon certain logical constraints, the solution of which should correspond to respectful substitutions. Some preliminary study in this direction is done in [7] for a variant of open bisimulation based on hedged bisimulation. Since the bi-trace structure we use is a variant of symbolic traces, we will also investigate whether the techniques used for symbolic traces analysis [3] can be adapted to our setting.

Acknowledgment The author thanks the anonymous referees for their comments on earlier versions of the paper.

References

1. M. Abadi and A. D. Gordon. A bisimulation method for cryptographic protocols. *Nord. J. Comput.*, 5(4):267–303, 1998.
2. M. Abadi and A. D. Gordon. A calculus for cryptographic protocols: The spi calculus. *Information and Computation*, 148(1):1–70, 99.
3. M. Boreale. Symbolic trace analysis of cryptographic protocols. In *Proceedings of ICALP 2001*, volume 2076 of *LNCS*, pages 667 – 681. Springer-Verlag, 2001.
4. M. Boreale, R. D. Nicola, and R. Pugliese. Proof techniques for cryptographic processes. *SIAM Journal of Computing*, 31(3):947–986, 2002.
5. J. Borgström, S. Briaïs, and U. Nestmann. Symbolic bisimulation in the spi calculus. In P. Gardner and N. Yoshida, editors, *CONCUR*, volume 3170 of *Lecture Notes in Computer Science*, pages 161–176. Springer, 2004.
6. J. Borgström and U. Nestmann. On bisimulations for the spi calculus. *Mathematical Structures in Computer Science*, 15(3):487–552, 2005.
7. S. Briaïs. A symbolic characterisation of open bisimulation for the spi calculus. Technical Report LAMP-REPORT-2007-002, École Polytechnique Fédérale de Lausanne, 2007.
8. S. Briaïs and U. Nestmann. Open bisimulation, revisited. *Electr. Notes Theor. Comput. Sci.*, 154(3):109–123, 2006.
9. R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, Part II. *Information and Computation*, pages 41–77, 1992.
10. D. Sangiorgi. A theory of bisimulation for the π -calculus. *Acta Informatica*, 33(1):69–97, 1996.
11. A. Tiu. A trace based bisimulation for the spi calculus. Preprint, available on <http://rsise.anu.edu.au/~tiu/tbisim.pdf>, 2007.