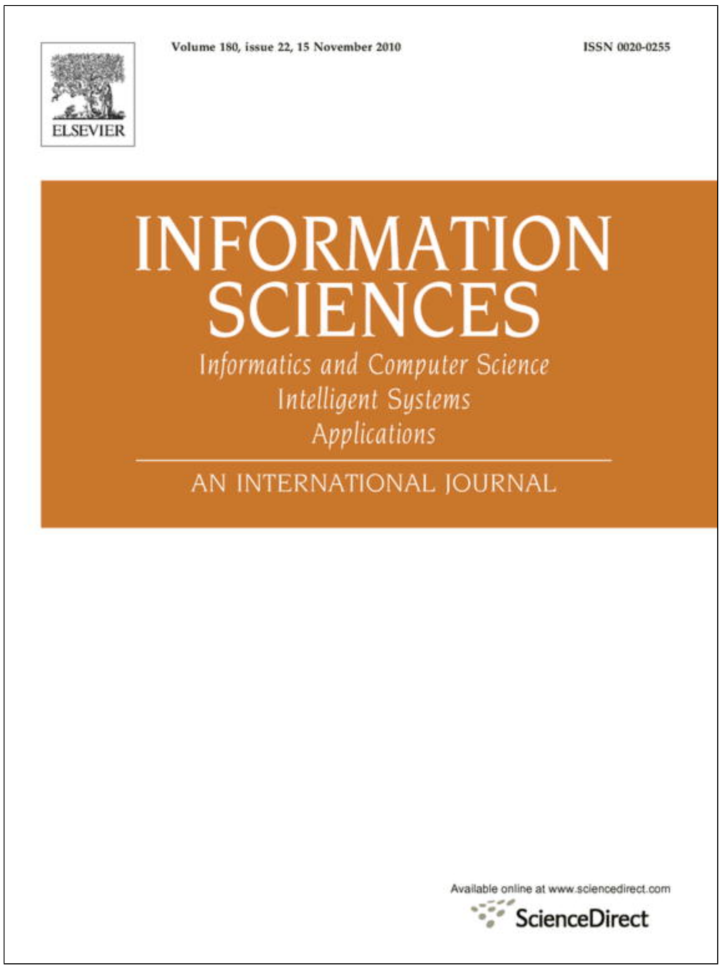

Information Sciences 180 (2010) 4436–4458

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A pure labeled transition semantics for the applied pi calculus ☆

Xiaojuan Cai

BASICS Lab, Department of Computer Science, Shanghai Jiao Tong University, Shanghai, China

ARTICLE INFO

Article history:
Received 11 November 2008
Received in revised form 10 June 2010
Accepted 16 July 2010

Keywords:
Labeled transition system
Bisimulation
Applied pi calculus
Zero-knowledge proof
Anonymity

ABSTRACT

The applied pi calculus proposed by Abadi and Fournet is successful in the analysis of security protocols. Its semantics mainly depends on several structural rules. Structural rules are convenient for specification, but inefficient for implementation. In this paper, we establish a new semantics for applied pi calculus based upon pure labeled transition system and propose a new formulation of labeled bisimulation. We prove that the new labeled bisimularity coincides with observational equivalence. A zero-knowledge protocol is given as an example to illustrate the effectiveness of this new semantics.

1. Introduction

Process calculi are a family of related approaches to modeling concurrent systems formally. They provide a tool for the high-level description of interactions and communications between agents or processes. They also provide algebraic laws to analyze process descriptions, and allow formal reasoning about equivalences between processes (e.g. using bisimulation). Leading examples of process calculi include CSP [25], CCS [31], and ACP [7,8]. More recent additions to the family include the pi calculus [32], the ambient calculus [11].

Using process calculi to analyze security protocols was first studied by [28] with CSP. Then Abadi and Gordon extended the pi calculus with cryptographic primitives and proposed the spi calculus [4]. The algebraic properties of the spi calculus were presented and proved in [5]. However, the spi calculus still has difficulties in describing some cryptographic operations, such as blind signature, hash function and so on. To meet expressiveness requirements, in [3] Abadi and Fournet introduced the applied pi calculus, which is a simple extension of the pi calculus with value passing, primitive functions and equations among terms. It has been used to model security protocols [1,26]. Verification of authentication using the applied pi calculus has been studied in [17]. Moreover, researchers begun to develop tools to check bisimulation in the applied pi calculus. One of the most important work was done by Blanchet et al. [10]. They implemented automation of checking bisimulation in the tool ProVerif [9].

We usually view the context as an active attacker when using process calculi to analyze protocols. In the spi calculus and the applied pi calculus, security properties such as secrecy, authenticity or anonymity are formally stated, and the verification of these properties is done by checking observational equivalence between two processes. Observational equivalence indicates that two given processes cannot be distinguished by any context. It is natural to use observational equivalence to capture some security properties. However, it is hard to check because one needs to consider all the contexts.

In the applied pi calculus, the operational semantics consists of some structural and internal reduction rules. Based on these rules, an auxiliary labeled transition system (LTS for short) [27] is defined to propose a labeled bisimulation, which

☆ This work is supported by the National Nature Science Foundation of China (60873034, 61011140074).
E-mail address: cxj@sjtu.edu.cn

is proved to coincide with observational equivalence. Therefore, instead of showing two processes are observationally equivalent, we prove that they are bisimilar.

To prove that two processes are bisimilar, the standard approach is to show that the pair of them is contained in some *bisimulation*. The structural rules will increase the size of this bisimulation relation dramatically which we have to check, because one needs to consider all the possible forms of a process (by applying structural rules repeatedly) and the corresponding derivatives. A pure labeled transition system without relying on structural rules will definitely free us from checking the ever increasing size of the bisimulation relation.

The contributions of this paper are threefold:

- We try to define the operational semantics purely on LTS and give a new formulation of labeled bisimulation.
- We study the algebraic theory of this calculus with new semantics, and the coincidence result in the domain of closed plain processes is established with detailed proofs.
- An example is given to illustrate the effectiveness of checking labeled bisimulation in this new semantics.

1.1. Related work

To our best knowledge, the study on the bisimulation of the applied pi calculus is limited. Among them there is symbolic bisimulation proposed by Delaune et al. [14]. A notion of intermediate process was defined to make sure all the restrictions are at the beginning of a process. The symbolic bisimulation is established based on the operational semantics and the labeled bisimulation in [3].

Pure LTS is beneficial for a lot of process calculi. For example, most studies [29,30,20] on ambient calculi are devoted to proposing an appropriate LTS to make bisimilarity equivalent to barbed congruence or observational equivalence which is based on structural rules.

The applied pi calculus has been successfully used to analyze some protocols [1,26]. Therefore, a more effective semantics is necessary. We think that pure LTS will make the application of the applied pi calculus more popular.

1.2. Outline of the paper

The rest of the paper is organized as follows: Section 2 briefly introduces the applied pi calculus. Section 3 defines the new operational semantics of the applied pi calculus and proposes a new labeled bisimulation. Section 4 shows that the labeled bisimulation coincides with the observational equivalence. Section 5 gives a practical example, and Section 6 concludes. The appendix includes some proofs.

2. The applied pi calculus

In this section, we introduce some basic concepts in the applied pi calculus. More details can be found in [3].

2.1. Some definitions

The set of *terms, plain processes* and *extended processes* are defined below:

$L, M, N ::=$		term
	$a, b, c, \ldots, k, \ldots, m, n$	name
	x, y, z	variable
	$f(M_1, \ldots, M_l)$	function application
$P, Q, R ::=$		plain processes
	$\mathbf{0}$	null process
	$P \vert Q$	parallel composition
	$!P$	replication
	$\nu n.P$	name restriction
	$\text{if } M = N \text{ then } P \text{ else } Q$	conditional
	$u(x).P$	message input
	$\bar{u}\langle N \rangle.P$	message output
$A,\ B,\ C:=$		extended processes
	P	plain process
	$A \vert B$	parallel composition
	$\nu n.A$	name restriction
	$\nu x.A$	variable restriction
	$\{M/x\}$	active substitution

$$\textsc{Subst}\quad \frac{}{\nu\,\tilde{n}.\sigma \vdash M}\ \text{if } \exists x \in dom(\sigma)\ s.t.\ x\sigma = M$$

$$\textsc{Nonce}\quad \frac{}{\nu\,\tilde{n}.\sigma \vdash s}\ \text{if } s \notin \tilde{n}$$

$$\textsc{Funct}\quad \frac{\phi \vdash M_1\ \cdots\ \phi \vdash M_k}{\phi \vdash f(M_1,\cdots,M_k)}\ f \in \Sigma$$

$$\textsc{Equiv}\quad \frac{\phi \vdash M \quad M =_E N}{\phi \vdash N}$$

Fig. 1. The deduction rules.

Here *terms* are extended with data names and function applications. We rely on a sort system for terms as in [3]. We use a, b and c as channel names, s, k as data names, and m, n as names of any sort. Data names are used to express data such as *keys, nonces, random numbers* and so on, and function applications are proposed to model all kinds of cryptographic operations such as *encryption* and *decryption*. A *ground* term is a term which has no free variables.

Equations are used to assert the relations of cryptographic primitives. For example, $\mathsf{dec}(\mathsf{enc}(x,y),y) = x$ models symmetric encryption where x represents a plaintext and y is a key. When describing security protocols, a signature Σ which consists of some function symbols must be given together with a set of equations E, and we call this set of equations *equational theory*. In the rest of this paper, we will use

$$\{\mathsf{fst}(\mathsf{pair}(x,y)) = x;\quad \mathsf{snd}(\mathsf{pair}(x,y)) = y;\quad \mathsf{dec}(\mathsf{enc}(x,y),y) = x\}$$

implicitly as the default equation theory.

We write $\Sigma \vdash M = N$ when the equation $M = N$ is in the equational theory associated with Σ, and we may write $M = N$ for simplicity when Σ is clear from context. The notation $\Sigma \nvdash M = N$ means we do not have $\Sigma \vdash M = N$.

The differences between plain processes and extended processes are active substitutions and variable restrictions. The notation $\{M/x\}$ is an active substitution which replaces the variable x with the term M. The active substitution $\{M/x\}$ typically appears when the term M has been sent to the environment. The variable restriction $\nu\,x$ restricts the scope of active substitutions.

The *frame* of an extended process consists of active substitutions and restrictions. Every extended process can be mapped to a frame. Frame can be viewed as static knowledge exposed by an extended process to its environment. We use $\phi, \psi, \ldots$ to range over frames and we usually write ϕ_A to denote the frame of process A.

The domain of a frame ϕ (resp. a process A), denoted by $dom(\phi)$ (resp. $dom(A)$) is the set of variables which appear in active substitutions of ϕ (resp. A) but not under a variable restriction. For example, If

$$A \stackrel{\text{def}}{=} \nu n, k.(P|\{\mathsf{enc}(n,k)/x\}|\{n/y\})$$

for some plain process P, then $\phi_A = \nu n, k.(\{\mathsf{enc}(n,k)/x\}|\{n/y\})$, $dom(\phi_A) = \{x, y\}$ and $dom(A) = dom(\phi_A)$. Given a set $\tilde{n} = \{n_1, n_2, \ldots, n_j\}$ and two tuples $\widetilde{M} = \langle M_1, M_2, \ldots, M_i\rangle$, $\tilde{x} = \langle x_1, x_2, \ldots, x_i\rangle$, we simply write $\nu\tilde{n}.\{\widetilde{M}/\tilde{x}\}$ instead of $\nu n_1, n_2, \ldots, n_j.(\{M_1/x_1\}|\{M_2/x_2\}|\cdots|\{M_i/x_i\})$.

We write $\phi \vdash M$ to mean M can be deduced from ϕ. This relation is called *deduction* [2], which is axiomatized by rules in Fig. 1.

We write $fn(A)$, and $bn(A)$ for free and bound names of A. We use $bv(A)$ to mean the bound variables of A. The free variables of A, denoted by $fv(A)$, are all the freely appearing variables in A excluding those variables in $dom(A)$. The names and variables of A are denoted as $n(A) \stackrel{\text{def}}{=} bn(A) \cup fn(A)$ and $v(A) \stackrel{\text{def}}{=} bv(A) \cup fv(A)$.

A process A is *closed* when $fv(A) = \emptyset$. An *evaluation context*, denoted by $C[\_]$ is a context whose hole is not under a replication, a condition, or a prefix. An evaluation context $C[\_]$ closes A when $C[A]$ is closed, and $C[\_]$ is called a *closing evaluation context.*

Definition 1. Given frame $\phi \stackrel{\text{def}}{=} \nu\tilde{n}.\sigma$ and two terms M, N such that $\tilde{n} \cap (fn(M) \cup fn(N)) = \emptyset$, we say that M and N are equal in ϕ, written $(M = N)\phi$, if and only if $M\sigma = N\sigma$.

Definition 2. We say that two closed frame ϕ and ψ are statically equivalent, denoted by $\phi \approx_s \psi$, if $dom(\phi) = dom(\psi)$ and, for all terms M and N, we have $(M = N)\phi$ if and only if $(M = N)\psi$.

Definition 3 (*Static equivalence* $(\approx_s)$). We say that two closed extended processes are statically equivalent, and write $A \approx_s B$, when their frames are statically equivalent.

2.2. Semantics

The original operational semantics of the applied pi calculus is presented in Appendix A. The structural rules denote an equivalence relation on extended processes that is closed under α-conversion on both names and variables, and under evaluation contexts. The internal reduction rules describe internal communication. And the labeled rules give the situation when the process is interacting with the environment.

We will write $\Rightarrow$ for $\rightarrow^*$ where $\rightarrow$ denotes one step of internal reduction, and $\overset{\lambda}{\Rightarrow}$ for $\Rightarrow\overset{\lambda}{\rightarrow}\Rightarrow$. We write $A{\downarrow_a}$, read as "A barbed on a", to mean that A can send or receive a message on channel a immediately. Note that if $A{\downarrow_a}$, then by using the structural rules of Appendix A, the process A can be rewritten to the form of $C[a(x).P_1]$ or $C'[\bar{a}\langle M\rangle.P_2]$ where $C[\_]$ and $C'[\_]$ are two evaluation contexts. We write $A{\Downarrow_a}$, read as "A weakly barbed on a", if there exists some A' such that $A \Rightarrow A'{\downarrow_a}$. We also write $A{\not\downarrow_a}$ (resp. $A{\not\Downarrow_a}$) if $A{\downarrow_a}$ (resp. $A{\Downarrow_a}$) does not hold.

Definition 4. A binary symmetric relation $\mathcal{R}$ on closed extended processes is a *labeled bisimulation* if for any $A\mathcal{R}B$:

(i) $A\approx_s B$.
(ii) if $A\xrightarrow{\alpha}A'$, and $fv(\alpha)\subseteq dom(A)$ and $bn(\alpha)\cap fn(B)=\emptyset$ then $\exists B'$, s.t. $B\overset{\alpha}{\Rightarrow}B'$ and $A'\mathcal{R}B'$.
(iii) if $A\rightarrow A'$, then $\exists B'$, s.t. $B\Rightarrow B'$ and $A'\mathcal{R}B'$.

The *bisimilarity*, denoted by $\approx_l$ is the largest labeled bisimulation.

Definition 5 (*Observational equivalence* ($\approx_o$)). Observational equivalence ($\approx_o$) is the largest symmetric relation $\mathcal{R}$ between closed extended processes with the same domain such that $A\ \mathcal{R}\ B$ implies:

(i) if $A{\downarrow_a}$, then $B{\Downarrow_a}$;
(ii) if $A\Rightarrow A'$, then $B\Rightarrow B'$ and $A'\ \mathcal{R}\ B'$ for some B';
(iii) $C[A]\ \mathcal{R}\ C[B]$ for all closing evaluation contexts $C[\_]$.

Theorem 1 [3]. $\approx_l = \approx_o$ *and* $\approx_o \subseteq \approx_s$.

3. Pure labeled semantics

In this section, we give a pure labeled transition system for applied pi. We also give a new definition of labeled bisimilarity based upon the new LTS.

3.1. New LTS

A pure LTS simplifies the manipulation and the analysis of process descriptions. We keep most of the syntax of the applied pi calculus in [3], but change the definition of *extended processes* by omitting the variable restriction operators:

$$A,B,C ::= P \mid A|B \mid \nu n.A \mid \{M/x\}$$

Readers will see in Fig. 2 that an active substitution appears when the output action is done. We need not use the structural rule $\mathbf{0}\equiv\nu x.\{M/x\}$ to produce the active substitutions, so we remove the operator νx in the syntax of extended processes.

The new semantics of the applied pi calculus is defined purely in terms of a LTS in Fig. 2 and no structural rules are preordained. We omit the symmetric rules for Int and Par. Here we make some comments:

1. $A\xrightarrow{\alpha}A'$. Here α can be an internal action τ, an input aM or an output $\bar{a}x$. We use the early semantics in order to model the environment as an active attacker and to simplify the definition of bisimulation. There are two reasons of using $\bar{a}x$ instead of $\bar{a}M$. First, after doing $\bar{a}x$, the process will be paralleled by an active substitution, and x denotes exactly the message sent in this action which is necessary when comparing two frames. Second, when distinguishing two processes we only care about the channels where this output happens. The output messages will be compared in the frames of the induced processes using *static equivalence*.
2. We have three rules to manage the restrictions, Res, Res-In and Res-Out. If the frame can deduce the message M then the message can be input. That is to say if the restricted name $m\in n(M)$, M still can be input because the environment (frame) knows it. For example, when a cipher text $\mathsf{enc}(m,k)$ was sent to the environment, then it can be received by anyone even though the plain text m and key k are under restrictions which means that they are unknown to anyone.
The condition $\phi_{\nu m.A}\vdash a$ considers the case $u=a$. This condition tells that if the channel name has been known by the environment, of course one can communicate on it. This is also the reason we have $\phi_{\nu a.A}\vdash a$ in the rule Res-Out.

$$\textsc{In}\ \frac{M \text{ is ground. } n(M)\cap bn(P)=\emptyset}{a(x).P \xrightarrow{a\,M} P\{M/x\}} \qquad \textsc{Out}\ \frac{x\notin v(N)\cup v(P)}{\bar a\,\langle N\rangle.P \xrightarrow{\bar a\,x} P \mid \{N/x\}}$$

$$\textsc{Par-L}\ \frac{A\xrightarrow{\alpha}A' \quad dom(A')\cap v(B)=\emptyset}{A\mid B\xrightarrow{\alpha}A'\mid B} \qquad \textsc{Res}\ \frac{A\xrightarrow{\tau}A'}{\nu\, b.A\xrightarrow{\tau}\nu\, b.A'} \qquad \textsc{Rep}\ \frac{P\xrightarrow{\alpha}P'}{!\,P\xrightarrow{\alpha}P'\mid !P}$$

$$\textsc{Res-In}\ \frac{A\xrightarrow{a\,M}A' \quad \phi_{\nu\, m.A}\vdash a \quad \phi_{\nu\, m.A}\vdash M}{\nu\, m.A\xrightarrow{a\,M}\nu\, m.A'} \qquad \textsc{Res-Out}\ \frac{A\xrightarrow{\bar a\,x}A' \quad \phi_{\nu\, m.A}\vdash a}{\nu\, m.A\xrightarrow{\bar a\,x}\nu\, m.A'}$$

$$\textsc{Con-T}\ \frac{\Sigma\vdash M=N \quad P\xrightarrow{\alpha}A}{\mathsf{if}\ M=N\ \mathsf{then}\ P\ \mathsf{else}\ Q\xrightarrow{\alpha}A} \quad \textsc{Con-F}\ \frac{M,N \text{ are ground. } \Sigma\nvdash M=N \quad Q\xrightarrow{\alpha}B}{\mathsf{if}\ M=N\ \mathsf{then}\ P\ \mathsf{else}\ Q\xrightarrow{\alpha}B}$$

$$\textsc{Int-L}\ \frac{A\xrightarrow{\bar a\,x}\nu\,\tilde n.(A'\mid\{M/x\}) \quad B\xrightarrow{a\,M}B' \quad \tilde n\subseteq n(M) \quad fn(M)\cap bn(A)=\emptyset}{A\mid B\xrightarrow{\tau}\nu\,\tilde n.(A'\mid B')}$$

Fig. 2. The LTS of the applied pi calculus.

3. Consider the rule Int-L. $A \overset{\bar a x}{\to} \nu\tilde n.(A'|\{M/x\})$ means that A has sent a message M to the environment, which is something like the "open" rule in the pi calculus. $B \overset{aM}{\to} B'$ states B receives a message M from the environment. When they are paralleled, an interaction happens but the active substitution $\{M/x\}$ disappears. This rule is similar to the "close" rule in the pi calculus. The appearance of $\{M/x\}$ indicates the openness of the message M. However, when this message is accepted by another process, the message M is closed meanwhile.
4. One may notice that the name restriction operator continues to exist no matter what the process does. We benefit from this fact in proving the theorems in Section 4.

Let's see some examples of derivations using the LTS in Fig. 2:

Example 3.1. Readers may find in the original semantics of applied pi (see Appendix A), the following two reductions are both available for $\bar a\langle m\rangle|\{m/x\}$.

$$\begin{array}{llll}
\bar a\langle m\rangle|\{m/x\} & \equiv & (\bar a\langle m\rangle|\mathbf{0})|\{m/x\} & \text{by } \textsc{Par-}\mathbf{0}\\
 & \equiv & (\bar a\langle m\rangle|\nu y.\{m/y\})|\{m/x\} & \text{by } \textsc{Alias}\\
 & \equiv & \nu y.(\bar a\langle m\rangle|\{m/y\})|\{m/x\} & \text{by } \textsc{New-Par}\\
 & = & \nu y.(\bar a\langle y\rangle\{m/y\}|\{m/y\})|\{m/x\} & \\
 & \equiv & \nu y.(\bar a\langle y\rangle|\{m/y\})|\{m/x\} & \text{by } \textsc{Subst}\\
 & \overset{\nu y.\bar a\langle y\rangle}{\to} & \{m/y\}|\{m/x\} & \text{by } \textsc{Open-Atom}\\
\bar a\langle m\rangle|\{m/x\} & \equiv & \bar a\langle x\rangle\{m/x\}|\{m/x\} & \\
 & \equiv & \bar a\langle x\rangle|\{m/x\} & \text{by } \textsc{Subst}\\
 & \overset{\bar a\langle x\rangle}{\to} & \{m/x\} & \text{by } \textsc{Out}
\end{array}$$

Our new LTS rules out the second case to avoid the unnecessary, though harmless, ambiguity.

$$\textsc{Out},\textsc{Par-L}\ \frac{y \textit{ is fresh}}{\bar a\langle m\rangle|\{m/x\}\overset{\bar a y}{\to}\{m/y\}|\{m/x\}}$$

Example 3.2. This example considers the restrictions of channel names after being sent out.

$$\textsc{Res}\ \frac{\textsc{Out}\ \dfrac{x \textit{ is fresh}}{\bar a\langle c\rangle.\bar c\langle m\rangle \overset{\bar a x}{\to} \bar c\langle m\rangle|\{c/x\}} \quad c\notin n(\bar a x)}{\nu c.(\bar a\langle c\rangle.\bar c\langle m\rangle)\overset{\bar a x}{\to}\nu c.(\bar c\langle m\rangle|\{c/x\})}$$

Note that νc still remains after the output of c on channel a. As in the pi calculus and the spi calculus, the process $\nu c.\bar c\langle m\rangle$ cannot do any actions. However, here because c, as a message, has been "open" to the environment with the active substitution $\{c/x\}$, we have the following reduction:

$$\textsc{Res}-\textsc{Out}\,\dfrac{\textsc{Par}-\textsc{L}\,\dfrac{\bar{c}\langle m\rangle \overset{\bar{c}\,y}{\rightarrow}\{m/y\}}{\bar{c}\langle m\rangle|\{c/x\} \overset{\bar{c}\,y}{\rightarrow}\{m/y\}|\{c/x\}} \qquad \phi_{\nu c.(\bar{c}\langle m\rangle|\{c/x\})} \vdash c}{\nu c.(\bar{c}\langle m\rangle|\{c/x\}) \rightarrow^{\bar{c}\,y} \nu c.(\{m/y\}|\{c/x\})}$$

Example 3.3. This example shows the interaction between processes.

$$\textsc{Int}-\textsc{L}\,\dfrac{\nu k,m.\bar{a}\langle \mathsf{enc}(m,k)\rangle.\bar{a}\,\langle k\rangle \overset{\bar{a}\,x}{\rightarrow} \nu k,m.(\{\mathsf{enc}(m,k)/x\}|\bar{a}\langle k\rangle) \qquad a(x).b(y) \overset{a\,\mathsf{enc}(m,k)}{\rightarrow} b(y)}{\nu k,m.\bar{a}\langle \mathsf{enc}(m,k)\rangle.\bar{a}\,\langle k\rangle|a(x).b(y) \overset{\tau}{\rightarrow} \nu k,m.(\bar{a}\langle k\rangle|b(y))}$$

Note that in the new semantics, the interaction $\overset{\tau}{\rightarrow}$ is different from internal reduction $\rightarrow$ in the original semantics. So in order not to confuse these two kinds of transitions, we write $\overset{\hat{\tau}}{\Rightarrow}$ for $\overset{\tau}{\rightarrow}{}^{*}$, and $\overset{\lambda}{\Rightarrow}$ for $\overset{\hat{\tau}}{\Rightarrow}\overset{\lambda}{\rightarrow}\overset{\hat{\tau}}{\Rightarrow}$. We write $A{\downarrow}_a$, read as "A barbed on a", to mean that A can send or receive a message on channel a, and $A{\Downarrow}_a$, read as "A weakly barbed on a", for the fact that there exists some A' such that $A\overset{\hat{\tau}}{\Rightarrow}A'{\downarrow}_a$. Similarly, $A{\downarrow}_a$ (resp. $A{\Downarrow}_a$) means $A{\downarrow}_a$ (resp. $A{\Downarrow}_a$) does not hold. We also write $A\Downarrow$ if $\forall a.A{\Downarrow}_a$.

It is worth mentioning that $A{\downharpoonright}_a$ implies $A{\downarrow}_a$ but the reverse is not true due to Con-T and Con-F rules in Fig. 2. For example, $P \overset{\text{def}}{=}$ $\mathsf{if}\ M=M\ \mathsf{then}\ a.\mathbf{0}$, we have $A{\downarrow}_a$ but $A|\!\swarrow_a$. However, the following lemma shows that weak barbs of plain processes are equivalent in the original and new semantics.

Lemma 2. *For any closed plain process P and any name a, $P{\downdownarrows}_a$ if and only if $P{\Downarrow}_a$.*

Proof. First of all, according to the original semantics in Appendix A, for any closed plain process P, if $P\rightarrow A'$, then there must exist some closed plain process P' s.t. $A'\equiv P'$.

Now we proceed the proof inductively on the structure of P.

1. The null process, prefix, restriction, and replication form are direct.
2. $P\overset{\text{def}}{=}$ $\mathsf{if}\ M=N\ \mathsf{then}\ P_1\ \mathsf{else}\ P_2$,
   - if for some a, $P{\downdownarrows}_a$, then $P_1{\downdownarrows}_a$ if $\Sigma\vdash M=N$, $P_2{\downdownarrows}_a$ if $\Sigma\nvdash M=N$. By induction we have $P_1{\Downarrow}_a$ if $\Sigma\vdash M=N$, $P_2{\Downarrow}_a$ otherwise. So according to Con-T and Con-F we have $P{\Downarrow}_a$.
   - if for some a, $P{\Downarrow}_a$, then $P_1{\Downarrow}_a$ if $\Sigma\vdash M=N$, $P_2{\Downarrow}_a$ if $\Sigma\nvdash M=N$. So according to Then and Else we have $P\rightarrow P_1{\downdownarrows}_a$ when
     $\Sigma\vdash M=N$. Otherwise $P\rightarrow P_2{\downdownarrows}_a$.
3. $P\overset{\text{def}}{=}P_1|P_2$,
   - if for some name a, $P{\downdownarrows}_a$,
     – if $P_1{\downdownarrows}_a$, then $P_1{\Downarrow}_a$, therefore, $P{\Downarrow}_a$; It is similar when $P_2{\downdownarrows}_a$.
     – if $P\rightarrow P'{\downdownarrows}_a$ and this transition is caused by Comm rule.
       Then there must exist some $b,P'_1,P'_2,M,\tilde{n}$ such that

$$\begin{aligned} P_1|P_2 &\equiv \nu\tilde{n}.\nu x.(\bar{b}\langle x\rangle.P'_1|b(x).P'_2|\{M/x\}) \\ &\rightarrow \nu\tilde{n}.\nu x.(P'_1|P'_2|\{M/x\}) \\ &\equiv \nu\tilde{n}.(P'_1|P'_2\{M/x\}){\downdownarrows}_a \end{aligned}$$

M is ground, $\tilde{n}\subseteq n(M)$, P'_1 and $P'_2\{M/x\}$ are closed plain processes. It is easy to see by rule Int-L or Int-R, we have

$$P_1|P_2 \overset{\tau}{\rightarrow} \nu\tilde{n}.(P'_1|P'_2\{M/x\})$$

So $P{\Downarrow}_a$.

   - if for some name a, $P{\Downarrow}_a$,
     – if $P_1{\Downarrow}_a$, then $P_1{\downdownarrows}_a$, therefore, $P{\downdownarrows}_a$; It is similar when $P_2{\Downarrow}_a$.
     – if $P\overset{\tau}{\rightarrow}P'{\Downarrow}_a$ and this transition is caused by Int-Ls rule, i.e.

$$\dfrac{P_1 \overset{\bar{b}\,x}{\rightarrow} \nu\tilde{n}.(P'_1|\{M/x\}) \quad P_2 \overset{b\,M}{\rightarrow} P'_2 \quad \tilde{n}\subseteq n(M) \quad fn(M)\cap bn(P_1)=\emptyset}{P_1|P_2 \overset{\tau}{\rightarrow} \nu\tilde{n}.(P'_1|P'_2)}$$

Considering all the rules in Fig. 2, we have the conclusion that there must exist P_3, P_4, P_2'' such that

$$\begin{array}{llll} P_1|P_2 & \Rightarrow & P_3|P_4 & \text{by Then or Else} \\ & \equiv & \nu\tilde{n}.\nu x.(\bar{b}\langle x\rangle.P_1'|b(x).P_2''|\{M/x\}) & \text{by structural rules} \\ & \rightarrow & \nu\tilde{n}.\nu x.(P_1'|P_2''|\{M/x\}) & \text{by Comm} \\ & \equiv & \nu\tilde{n}.(P_1'|P_2')\downdownarrows_a & \text{by Subst, Alias} \end{array}$$

So $P \Rightarrow \nu\tilde{n}.(P_1'|P_2')\downdownarrows_a$. It is similar when the transition $P \xrightarrow{\tau} P'$ is caused by Int-r rule. And we are done. □

In the rest of the paper we will use "d is a fresh name" or "for a fresh name d", to mean that name d does not occur in any syntactical objects under consideration. We also write $\bar{d}.P$ instead of $\nu m.(\bar{d}\langle m\rangle.P)$ when $m \notin fn(P)$; and $d.Q$ instead of $d(z).Q$ when $z \notin fv(Q)$.

By the Out rule,

$$\bar{d}.P \overset{\text{def}}{=} \nu m.(\bar{d}\langle m\rangle.P) \xrightarrow{\bar{d}x} \nu m.(P|\{m/x\})$$

Since $m \notin fn(P)$, the active substitution $\{m/x\}$ does not influence the frames of P and P's successors. By the In rule we have for any term M $d(z).Q \xrightarrow{dM} Q$ when $z \notin fn(Q)$. Therefore, we write $\bar{d}.P \xrightarrow{d} P$ and $d.Q \xrightarrow{d} Q$ for simplicity.

3.2. Labeled bisimulation

We define a new labeled bisimulation based on the pure LTS discussed above. First we recall a characterization of deduction proposed by Abadi and Cortier (Proposition 1 in [2]).

Proposition 3. *Let M be a closed term and $\nu\tilde{n}.\sigma$ be a frame. Then $\nu\tilde{n}.\sigma \vdash M$ if and only if there exists a term ζ such that $fn(\zeta) \cap \tilde{n} = \emptyset$ and $M = \zeta\sigma$.*

According to this proposition we define a relation to relate two terms.

Definition 6. Assume two frames $\phi_A \overset{\text{def}}{=} \nu\tilde{m}.\{\widetilde{M}/\tilde{x}\}$ and $\phi_B \overset{\text{def}}{=} \nu\tilde{n}.\{\widetilde{N}/\tilde{x}\}$ are statically equivalent, then given two terms M, N which can be deduced from ϕ_A and ϕ_B respectively, we say M and N are *correspondingly equivalent*, denoted $M \asymp_{\{\phi_A,\phi_B\}} N$, if whenever $M = \zeta\{\widetilde{M}/\tilde{x}\}$ for some term ζ and $n(\zeta) \cap (\tilde{n} \cup \tilde{m}) = \emptyset$, we have $N = \zeta\{\widetilde{N}/\tilde{x}\}$.

Definition 7. A binary symmetric relation $\mathcal{R}$ between closed extended processes is a labeled bisimulation if for any $A\mathcal{R}B$:

1. $A \approx_s B$.
2. if $A \xrightarrow{\tau} A'$, then $\exists B'$, s.t. $B \overset{\hat{\tau}}{\Rightarrow} B'$ and $A'\mathcal{R}B'$.
3. if $A \xrightarrow{\bar{a}x} A'$, then $\exists b, B'$, s.t. $B \overset{\bar{b}x}{\Rightarrow} B'$ with $a \asymp_{\{\phi_A,\phi_B\}} b$ and $A'\mathcal{R}B'$.
4. if $A \xrightarrow{aM_1} A'$,then $\exists b, B'$ and $\phi_B \vdash M_2$, s.t. $B \overset{bM_2}{\Rightarrow} B'$, $M_1 \asymp_{\{\phi_A,\phi_B\}} M_2$, $a \asymp_{\{\phi_A,\phi_B\}} b$ and $A'\mathcal{R}B'$,

We write $\approx_L$ to denote the largest labeled bisimulation in order to distinguish from the labeled bisimilarity $\approx_l$ in [3]. We give two simple examples to explain the definition of labeled bisimulation.

Example 3.4.

$$\begin{array}{ccccc} \nu c.\bar{a}\langle c\rangle.\bar{c}\langle m\rangle & \approx_L & \nu d.\bar{a}\langle d\rangle.\bar{d}\langle m\rangle & \\ \bar{a}x\downarrow & & \bar{a}x\downarrow & \\ \nu c.(\bar{c}\langle m\rangle|\{c/x\}) & & \nu d.(\bar{d}\langle m\rangle|\{d/x\}) & c \asymp_{\{\nu c.\{c/x\},\nu d.\{d/x\}\}} d \\ \bar{c}y\downarrow & & \bar{d}y\downarrow & \\ \nu c.(\{m/y\}|\{c/x\}) & \approx_L & \nu d.(\{m/y\}|\{d/x\}) & \end{array}$$

This example looks nonsense since $\nu c.\bar{a}\langle c\rangle.\bar{c}\langle m\rangle$ is α-convertible to $\nu d.\bar{a}\langle d\rangle.\bar{c}\langle m\rangle$. However, when implementing an algorithm to compute if two processes are bisimilar, α-convertibility is hard to check.

Example 3.5.

$$\begin{aligned}&\nu k, m.(\bar{a}\langle \mathsf{enc}(m,k)\rangle.c(x).\mathsf{if}\ x = \mathsf{enc}(m,k)\ \mathsf{then}\ \bar{b}\langle m'\rangle\ \mathsf{else}\ \mathbf{0})\\ &\qquad \approx_L \nu n.(\bar{a}\langle n\rangle.c(x).\mathsf{if}\ x = n\ \mathsf{then}\ \bar{b}\langle m'\rangle\ \mathsf{else}\ \mathbf{0})\end{aligned}$$

Assume

$$\begin{aligned}A &\overset{\mathrm{def}}{=} \nu k, m.(\bar{a}\langle \mathsf{enc}(m,k)\rangle.c(x).\mathsf{if}\ x = \mathsf{enc}(m,k)\ \mathsf{then}\ \bar{b}\langle m'\rangle\ \mathsf{else}\ \mathbf{0})\\ B &\overset{\mathrm{def}}{=} \nu n.(\bar{a}\langle n\rangle.c(x).\mathsf{if}\ x = n\ \mathsf{then}\ \bar{b}\langle m'\rangle\ \mathsf{else}\ \mathbf{0})\end{aligned}$$

Then we have the following reduction:

$$\begin{array}{ccc} A & \approx_L & B\\ \bar{a}x\downarrow & & \bar{a}x\downarrow\\ \nu k, m.(c(x).\mathsf{if}\cdots|\{\mathsf{enc}(m,k)/x\}) & & \nu n.(c(x).\mathsf{if}\cdots|\{n/x\})\\ c\,\mathsf{enc}(m,k)\downarrow & & c\,n\downarrow\\ \bar{b}z\downarrow & & \bar{b}z\downarrow\\ \nu k, m.(\{m'/z\}|\{\mathsf{enc}(m,k)/x\}) & \approx_L & \nu n.(\{m'/z\}|\{n/x\}) \end{array}$$

Note that $\mathsf{enc}(m,k) \asymp_{\{\nu k,m.\{\mathsf{enc}(m,k)/x\},\nu n.\{n/x\}\}} n$ and

$$\nu k, m.(\{m'/z\}|\{\mathsf{enc}(m,k)/x\}) \approx_s \nu n.(\{m'/z\}|\{n/x\}).$$

This example shows that the cipher text which cannot be decrypted is the same as a random number.

4. Algebraic theory

In this part, we first define the observational equivalence between *plain* processes, and then prove labeled bisimilarity coincides with observational equivalence in the domain of plain processes. In Theorem 5, we reveal labeled bisimilarity implies observational equivalence and in Theorem 6 we show the opposite direction. The coincidence result is given in Corollary 7.

We consider observational equivalence between *plain* processes rather than *extended* ones for two reasons. First, the initial processes with which we model security protocols are always plain ones. Second, without structural rules the static knowledge of processes cannot be captured by the environment using the rules in Fig. 2. For example, let

$$\begin{aligned}A &\overset{\mathrm{def}}{=} \nu m, n.(\{m/x\}|\{n/y\})\\ B &\overset{\mathrm{def}}{=} \nu m, n.(\{m/x\}|\{m/y\})\end{aligned}$$

If we try to construct contexts to distinguish A and B, all possible ways can be reduced to the following two cases: (i) $C_1[\_] \overset{\mathrm{def}}{=} \_|\mathsf{if}\ x = y\ \mathsf{then}\ \bar{d}$; (ii) $C_2[\_] \overset{\mathrm{def}}{=} \_|\bar{x}|y.\bar{d}$. Since we now have no structural rules as SUBST to substitute x, y with ground names, and we have $\Sigma \nvdash x = y$, so CON-T and CON-F rules can not be applied here. We can have the conclusion that $C_1[A]\downarrow$ and $C_1[B]\downarrow$. Now let's consider $C_2[\_]$. As we know, IN, OUT rules can only be applied when the channel name has been instantiated, so $C_2[A] \Downarrow$ and $C_2[B] \Downarrow$. That is to say, there is not contexts can differentiate A and B without structural rules. But $A \not\approx_L B$ since $A \not\approx_s B$.

In the rest of this paper we only consider plain processes and those extended processes that can be derived from some plain ones with rules in Fig. 2. These processes are enough for us to model security protocols and capture all the abilities of adversaries (or the environment). An coincidence result between observational equivalence and labeled bisimilarity restricted to plain processes is shown in next two subsections.

We now restrict the original observational equivalence to plain processes, and give the modified definition where *plain closing evaluation contexts* means closing evaluation contexts without active substitutions.

Definition 8. Observational equivalence ($\approx_{obs}$) is the largest symmetric relation $\mathcal{R}$ between closed plain processes such that $P\mathcal{R}\ Q$ implies:

1. if $P\downarrow_a$, then $Q\Downarrow_a$;
2. if $P \xrightarrow{\tau} P'$, then $Q \overset{\hat{\tau}}{\Rightarrow} Q'$ and $P'\ \mathcal{R}\ Q'$ for some Q';
3. $C[P]\ \mathcal{R}\ C[Q]$ for all plain closing evaluation contexts $C[\_]$.

Recall rules in Fig. 2. The interaction $\xrightarrow{\tau}$ will not change a plain process into an extended one. That is to say, τ-actions will never introduce active substitutions. And this is the reason why we write $P \xrightarrow{\tau} P'$ instead of $P \xrightarrow{\tau} A'$ in above definition.

Relation $\approx_{obs}$ is shown to be equal to $\approx_o$ in the domain of plain processes by the following lemma.

Lemma 4. *For any closed plain processes P, Q, $P \approx_o Q$ if and only if $P \approx_{obs} Q$.*

Proof. We prove by contradiction. This lemma is equivalent to

$$\text{For any closed plain processes } P, Q,\ P \not\approx_o Q \text{ if and only if } P \not\approx_{obs} Q. \tag{1}$$

According to Definitions 5 and 8, if $P \not\approx_o Q$ $(P \not\approx_{obs} Q)$, then there must exists some (plain) closing evaluation context $C[\_]$ and some name a such that $C[P]\Downarrow_a$ $(C[P]\Downarrow_a)$ but $C[Q]\not\Downarrow_a$ $(C[Q]\not\Downarrow_a)$ or $C[P]\not\Downarrow_a$ $(C[P]\not\Downarrow_a)$ but $C[Q]\Downarrow_a$ $(C[Q]\Downarrow_a)$. Since $\approx_o$ and $\approx_{obs}$ are symmetric relations, so the following statement implies (1).

For any closed plain processes P, Q, if there exists some closing evaluation context $C[\_]$ and name a such that $C[P]\Downarrow_a$ but $C[Q]\not\Downarrow_a$, then we can construct a plain closing evaluation context $C'[\_]$ such that $C'[P]\Downarrow_a$ but $C'[Q]\not\Downarrow_a$; and *vice versa.*

Using structural equivalence, any closing evaluation context $C[\_]$ can be rewritten to

$$\nu\tilde{n}.(\{\widetilde{M}/\tilde{x}\}|C_1[\_])$$

for some $\tilde{n}, \widetilde{M}, \tilde{x}$ and some plain closing evaluation context $C_1[\_]$.

In all the structural rules, only the following rule makes active substitutions influence the possible actions of processes.

$$\textsc{Subst}\ \{M/x\}|A \equiv \{M/x\}|A\{M/x\}$$

However, for any closed plain process P, $fv(C_1[P]) = \emptyset$ and the active substitutions $\{\widetilde{M}/\tilde{x}\}$ will not influence the transitions of $C_1[P]$. So if $C[P]\Downarrow_a$ then $\nu\tilde{n}.C_1[P]\Downarrow_a$; and if $C[P]\not\Downarrow_a$ then $\nu\tilde{n}.C_1[P]\not\Downarrow_a$. So it is enough to show:

For any closed plain process P, for any plain closing evaluation context $C[\_]$ and name a, $C[P]\Downarrow_a$ if and only if $C[P]\Downarrow_a$.

This is direct by Lemma 2. □

In the rest of this paper we will use $\approx_{obs}$ instead of $\approx_o$ when we say two plain processes are observationally equivalent because the definition of $\approx_{obs}$ is convenient for the proofs.

The fact that labeled bisimilarity $(\approx_L)$ implies observational equivalence is proved by showing labeled bisimilarity is closed under closing evaluation context.

Theorem 5. *Given any two processes A, B, if $A \approx_L B$, then for any closing evaluation context $C[\_]$ we have $C[A] \approx_L C[B]$.*

To prove this theorem, we need to show the following relation is a labeled bisimulation:

$$\mathcal{R} \stackrel{\text{def}}{=} \approx_L \cup \{(C[A], C[B])|A \approx_L B \text{ and } C[\_] \text{ is a closing evaluation context}\}$$

We proceed by considering the structure of context $C[\_]$. See the detailed proof in Appendix B.

Theorem 6. *For any plain processes P, Q, if $P \approx_{obs} Q$, then $P \approx_L Q$.*

Proof (*Main idea*). The proof of this theorem is more technical than Theorem 5. We extend the method of proving the coincidence between barbed congruence and weak bisimulation of π-calculus by Sangiorgi in his Ph.D thesis [33].

The main idea is to build up a set of context pairs, $\mathcal{C}$, powerful enough to make sure observational equivalence on these context pairs implies labeled bisimulation, i.e., we try to prove the following relation is a labeled bisimulation.

$$\mathcal{R}^* \stackrel{\text{def}}{=} \{(P, Q)|C_1[P] \approx_{obs} C_2[Q]\}$$

where P, Q are any closed plain processes and $(C_1, C_2) \in \mathcal{C}$. The key property of $\mathcal{C}$ is that: Suppose $P \xrightarrow{\alpha} P'$, then we can find $(C_1', C_2') \in \mathcal{C}$ such that

$$C_1[P] \stackrel{\tau}{\Rightarrow} C_1'[P']$$

And $C_2[Q]$ has to simulate the above transition with

$$C_2[Q] \stackrel{\hat{\tau}}{\Rightarrow} C_2'[Q']$$

where

$$Q \stackrel{\hat{\alpha}}{\Rightarrow} Q' \text{ and } C_1'[P'] \approx_{obs} C_2'[Q']$$

Then $(P', Q') \in \mathcal{R}^*$ and $\mathcal{R}^*$ is a labeled bisimulation.

However, the labeled transition action will introduce active substitutions which turn a plain process to an extended one. In other words, the transition $\xrightarrow{\alpha}$ might turn P to some extended process A' which will never appear in any pair of $\mathcal{R}^*$. So instead of proving

$$\mathcal{R}^* \stackrel{\text{def}}{=} \{(P, Q)|C_1[P] \approx_{obs} C_2[Q]\}$$

is a labeled bisimulation, we show that the following relation is a labeled bisimulation.

$$\mathcal{R} \overset{\text{def}}{=} \{(D_1[P], D_2[Q]) | C_1[P] \approx_{obs} C_2[Q]\}$$

where D_i is an active substitution context of the form $\nu\tilde{n}.(\_|\{\widetilde{M}/\tilde{x}\})$ and is determined by C_i.

Suppose $D_1[P] \xrightarrow{\alpha} D_1'[P']$, then we can find $(C_1', C_2') \in \mathcal{R}$ such that:

1. D_1' is an active substitution context determined by C_1'.
2. $C_1[P] \xrightarrow{\tau} C_1'[P']$ and $C_2[Q]$ has to simulate the above transition with

$$C_2[Q] \overset{\hat{\tau}}{\Rightarrow} C_2'[Q'] \approx_{obs} C_1'[P'].$$

Then we can find some D_2' determined by C_2' and

$$D_2[Q] \overset{\hat{\alpha}}{\Rightarrow} D_2'[Q']$$

So $(D_1'[P'], D_2'[Q']) \in \mathcal{R}$.

The construction of $\mathcal{C}$ depends on the active substitutions we need and some fresh names to make the simulation transition obedient. We give the detailed proof in Appendix C. □

The following corollary gives the coincidence result, and the proof is directly by Theorems 5 and 6.

Corollary 7. *For any plain processes P, Q, $P \approx_{obs} Q$ if and only if $P \approx_L Q$.*

5. A classic zero-knowledge protocol

In this section, we analyze a classic zero-knowledge protocol [21] with two purposes: the first is to reflect the effectiveness of the new semantics, and the second is to define some interesting properties in the applied pi calculus for zero-knowledge protocols.

Zero-knowledge protocols or proofs are hot topics in cryptography. The notion of zero-knowledge was proposed by Goldwasser et al. [23], and then turned out to be one of the most beautiful and attractive concepts in computer science. Informally speaking, a proof is zero-knowledge if the prover can convince the verifier that he/she has some secret but without revealing it. This incredibility made its applications ranging from signature schemes [12,15] to proving that many NP-complete problems are hard even to approximate [16].

Many efforts were made to analyze zero-knowledge in formal frameworks. One of the most important work was done by Backes et al. [6]. They established an equational theory to model the zero-knowledge property as a primitive in cryptography. However, in this section, we do not view zero-knowledge as a primitive. Instead, we try to prove a protocol is (interactive) zero-knowledge. To our best knowledge, this is the first attempt to define properties for zero-knowledge protocols in the applied pi calculus.

5.1. The protocol

It is shown by Goldreich et al. [22] that every language in NP has a zero-knowledge proof system with the assumption of the existence of one-way functions. The protocol considered here is a very classic one given in [21].

There are two participants, prover P and verifier V. The prover P knows the hamiltonian cycle c for graph G, and P tries to make the verifier V believe that he/she has the secret but without revealing it. The protocol depicts as follows:

- P chooses a random permutation σ, generates G's isomorphic graph G', and sends G, G' to V.
- V nondeterministically chooses 0 or 1 and sends the chosen one to P.
- If P gets 0 from V, P sends the permutation σ to V; if P gets 1, P applies σ to the hamiltonian cycle c, gets c' and sends it to V;
- If V sent 0 in step 2, V checks whether the received message is the permutation from G to G'; Otherwise, V checks whether the received message is the hamiltonian cycle of G'.

Note that in the original version V chooses 0 or 1 with probability 0.5 respectively in Step 2. Here we eliminate the probability and use nondeterminism instead. However, there is no nondeterministic choice in the syntax of applied pi. So we define the following "internal" nondeterministic process:

$$P + Q \overset{\text{def}}{=} \nu a.(\bar{a}|a.P|a.Q)$$

This process acts either as P or as Q because there is only one output on private channel a. Note that this process is different from the classical non-deterministic operator of the pi calculus [32] where the semantics of $P + Q$ is given by

$$\frac{P \xrightarrow{\alpha} P'}{P + Q \xrightarrow{\alpha} P'} \quad \frac{Q \xrightarrow{\alpha} Q'}{P + Q \xrightarrow{\alpha} Q'}$$

However, this kind of "internal" nondeterminism is enough for us to model the protocol.

Let true, false, 0, and 1 be special names. The first step is to set up the signature and equational theory.

$$\begin{array}{ll}
\textbf{fun} & \mathsf{g}/1 \\
\textbf{fun} & \mathsf{iso}/2 \\
\textbf{fun} & \mathsf{hc}/2 \\
\textbf{fun} & \mathsf{dei}/2 \\
\textbf{equation} & \mathsf{hc}(\mathsf{g}(x), x) = \mathsf{true} \\
\textbf{equation} & \mathsf{hc}(\mathsf{iso}(\mathsf{g}(x), y), \mathsf{iso}(x, y)) = \mathsf{true} \\
\textbf{equation} & \mathsf{dei}(\mathsf{iso}(x, y), y) = x
\end{array}$$

In the signature we define four functions: $\mathsf{g}(c)$ is a graph with its hamiltonian cycle c; $\mathsf{iso}(G,\sigma)$ denotes the isomorphic graph of G given the permutation σ; And $\mathsf{hc}(G,c)$ is a hamiltonian cycle checker. $\mathsf{hc}(G,c)$ will return true if c is a hamiltonian cycle of G; If we view the hamiltonian cycle c as a graph then $\mathsf{iso}(c,\sigma)$ actually is the hamiltonian cycle of graph $\mathsf{g}(c)$, and this is the meaning of the second equation. If G' is an isomorphic graph of G by applying the permutation σ, then function $\mathsf{dei}(G',\sigma)$ returns G.

The three equations illustrate the mathematical properties in the protocol. The first equation reveals the fact that for any cycle x and graph $\mathsf{g}(x)$ the hamiltonian cycle checker hc will always return true. The second equation shows that the hamiltonian cycle checker will return true if we apply the same permutation y to both graph $\mathsf{g}(x)$ and cycle x. The third one reflects the function $\mathsf{dei}/2$.

Assume P has the hamiltonian cycle c of the graph $\mathsf{g}(c)$, and a_1, a_2, a_3, a_4 are channels for communications, then we use process P and V to model the behaviors of the prover and the verifier.

$$\begin{aligned}
P &\overset{\text{def}}{=} \overline{a_1}\langle \mathsf{g}(c)\rangle.\nu\sigma.\overline{a_2}\langle \mathsf{iso}(\mathsf{g}(c),\sigma)\rangle.a_3(x). \\
&\quad (\mathsf{if}\ x = 0\ \mathsf{then}\ \overline{a_4}\langle\sigma\rangle\ \mathsf{else}\ \overline{a_4}\langle \mathsf{iso}(c,\sigma)\rangle) \\
V &\overset{\text{def}}{=} a_1(x).a_2(y).(\overline{a_3}\langle 0\rangle.a_4(z).\mathsf{if}\ \mathsf{dei}(y,z) = x\ \mathsf{then}\ \bar{b} \\
&\qquad + \overline{a_3}\langle 1\rangle.a_4(z).\mathsf{if}\ \mathsf{hc}(y,z) = \mathsf{true}\ \mathsf{then}\ \bar{b})
\end{aligned}$$

The action $\bar{b}$ reflects that V believes P has the secret.

As we know there may be many instances of such proofs running in parallel. In order to make sure there is no interference between any two instances, the channels $\tilde{a} = \{a_1, a_2, a_3, a_4\}$ are restricted in every instance. So the whole system is as follows:

$$ZK \overset{\text{def}}{=} \nu c.!\nu\tilde{a}.(P|V)$$

A zero-knowledge proof must satisfy three properties:

1. *Completeness*. An honest prover will always convince an honest verifier that he/she has the secret.
2. *Soundness*. No cheating prover can convince the honest verifier that he/she has the secret.
3. *Zero-knowledge*. No cheating verifier can learn anything about the secret.

The soundness does not hold in this non-deterministic protocol. Assume there is a cheating prover P^* who knows nothing about the hamiltonian cycle of some graph G. He just sends the permutation σ in the last step no matter what he received in the third step.

$$P^* \overset{\text{def}}{=} \overline{a_1}\langle G\rangle.\nu\sigma.\overline{a_2}\langle \mathsf{iso}(G,\sigma)\rangle.a_3(x).\overline{a_4}\langle\sigma\rangle$$

P^* can convince the verifier V by the following sequence:

$$\begin{aligned}
P^*|V \quad &\to \quad \nu\sigma.\overline{a_2}\langle \mathsf{iso}(G,\sigma)\rangle.a_3(x).\overline{a_4}\langle\sigma\rangle \\
&\qquad |a_2(y).(\overline{a_3}\langle 0\rangle.a_4(z).\mathsf{if}\ \mathsf{dei}(y,z) = G\ \mathsf{then}\ \bar{b} \\
&\qquad\qquad + \overline{a_3}\langle 1\rangle.a_4(z).\mathsf{if}\ \mathsf{hc}(\mathsf{iso}(G,\sigma),z) = \mathsf{true}\ \mathsf{then}\ \bar{b}) \\
&\to \quad a_3(x).\overline{a_4}\langle\sigma\rangle \\
&\qquad |(\overline{a_3}\langle 0\rangle.a_4(z).\mathsf{if}\ \mathsf{dei}(\mathsf{iso}(G,\sigma),z) = G\ \mathsf{then}\ \bar{b} \\
&\qquad\qquad + \overline{a_3}\langle 1\rangle.a_4(z).\mathsf{if}\ \mathsf{hc}(\mathsf{iso}(G,\sigma),z) = \mathsf{true}\ \mathsf{then}\ \bar{b}) \\
&\to \quad \overline{a_4}\langle\sigma\rangle|a_4(z).\mathsf{if}\ \mathsf{dei}(\mathsf{iso}(G,\sigma),z) = G\ \mathsf{then}\ \bar{b} \\
&\to \quad \mathsf{if}\ \mathsf{dei}(\mathsf{iso}(G,\sigma),\sigma) = G\ \mathsf{then}\ \bar{b}
\end{aligned}$$

However, in the probabilistic version, the soundness property holds with probability $1/2^n$ for n rounds.

In the applied pi calculus, frames denote the revealed information to the environment. So the zero-knowledge property holds if the frames of all the derivatives of process $\nu c.!P$ never reveal the secret c. In this paper, we will not focus on the proof of zero-knowledge properties because it can not demonstrate the effectiveness of the new semantics. We concentrate on the completeness of the protocol.

5.2. Completeness

Completeness means that the honest verifier will always be convinced by honest prover. So it is not difficult to write the specification for completeness:

$$V_{spec} \overset{\text{def}}{=} \bar{b},\ ZK_{spec} \overset{\text{def}}{=} !V_{spec}$$

In the specification, the verifier V_{spec} always outputs on channel b. So if $ZK \approx_{obs} ZK_{spec}$, then we are sure that V is convinced. The following theorem tells this protocol is complete and we will illustrate the effectiveness of the new LTS in the proof.

Theorem 8. *$ZK \approx_{obs} ZK_{spec}$.*

Proof. We have $\approx_L$ implies $\approx_{obs}$ by Theorem 5. So we prove $ZK \approx_L ZK_{spec}$.

We first introduce some abbreviation:

$$\begin{aligned}
P_1 &\overset{\text{def}}{=} \nu\sigma.\overline{a_2}\langle \mathsf{iso}(\mathsf{g}(c),\sigma)\rangle.a_3(x).\\
&\quad (\mathsf{if}\ x=0\ \mathsf{then}\ \overline{a_4}\langle\sigma\rangle\ \mathsf{else}\ \overline{a_4}\langle \mathsf{iso}(c,\sigma)\rangle)\\
V_1 &\overset{\text{def}}{=} a_2(y).(\overline{a_3}\langle 0\rangle.a_4(z).\mathsf{if}\ \mathsf{dei}(y,z)=\mathsf{g}(c)\ \mathsf{then}\ \bar{b}\\
&\qquad + \overline{a_3}\langle 1\rangle.a_4(z).\mathsf{if}\ \mathsf{hc}(y,z)=\mathsf{true}\ \mathsf{then}\ \bar{b})\\
P_2(\sigma) &\overset{\text{def}}{=} a_3(x).\\
&\quad (\mathsf{if}\ x=0\ \mathsf{then}\ \overline{a_4}\langle\sigma\rangle\ \mathsf{else}\ \overline{a_4}\langle \mathsf{iso}(c,\sigma)\rangle)\\
V_2(\sigma) &\overset{\text{def}}{=} \overline{a_3}\langle 0\rangle.a_4(z).\mathsf{if}\ \mathsf{dei}(\mathsf{iso}(\mathsf{g}(c),\sigma),z)=\mathsf{g}(c)\ \mathsf{then}\ \bar{b}\\
&\qquad + \overline{a_3}\langle 1\rangle.a_4(z).\mathsf{if}\ \mathsf{hc}(\mathsf{iso}(\mathsf{g}(c),\sigma),z)=\mathsf{true}\ \mathsf{then}\ \bar{b}\\
P_3(\sigma) &\overset{\text{def}}{=} \mathsf{if}\ 0=0\ \mathsf{then}\ \overline{a_4}\langle\sigma\rangle\ \mathsf{else}\ \overline{a_4}\langle \mathsf{iso}(c,\sigma)\rangle\\
P_3'(\sigma) &\overset{\text{def}}{=} \mathsf{if}\ 1=0\ \mathsf{then}\ \overline{a_4}\langle\sigma\rangle|\mathsf{if}\ 1=1\ \mathsf{then}\ \overline{a_4}\langle \mathsf{iso}(c,\sigma)\rangle\\
V_3(\sigma) &\overset{\text{def}}{=} a_4(z).\mathsf{if}\ \mathsf{dei}(\mathsf{iso}(\mathsf{g}(c),\sigma),z)=\mathsf{g}(c)\ \mathsf{then}\ \bar{b}\\
V_3'(\sigma) &\overset{\text{def}}{=} a_4(z).\mathsf{if}\ \mathsf{hc}(\mathsf{iso}(\mathsf{g}(c),\sigma),z)=\mathsf{true}\ \mathsf{then}\ \bar{b}\\
V_4(\sigma) &\overset{\text{def}}{=} \mathsf{if}\ \mathsf{dei}(\mathsf{iso}(\mathsf{g}(c),\sigma),\sigma)=\mathsf{g}(c)\ \mathsf{then}\ \bar{b}\\
V_4'(\sigma) &\overset{\text{def}}{=} \mathsf{if}\ \mathsf{hc}(\mathsf{iso}(\mathsf{g}(c),\sigma),\mathsf{iso}(c,\sigma))=\mathsf{true}\ \mathsf{then}\ \bar{b}
\end{aligned}$$

Let $ZK(i,j,k,k',r,r')$ be the following process

$$\begin{aligned}
&\nu c.(!\nu\tilde{a}.(P|V)|\underbrace{\nu\tilde{a}.(P_1|V_1)|\cdots|\nu\tilde{a}.(P_1|V_1)}_{i\ \text{times}}\\
&\quad |\underbrace{\nu\tilde{a}.\nu\sigma(P_2(\sigma)|V_2(\sigma))|\cdots|\nu\tilde{a}.\nu\sigma(P_2(\sigma)|V_2(\sigma))}_{j\ \text{times}}\\
&\quad |\underbrace{\nu\tilde{a}.\nu\sigma(P_3(\sigma)|V_3(\sigma))|\cdots|\nu\tilde{a}.\nu\sigma(P_3(\sigma)|V_3(\sigma))}_{k\ \text{times}}\\
&\quad |\underbrace{\nu\tilde{a}.\nu\sigma(P_3'(\sigma)|V_3'(\sigma))|\cdots|\nu\tilde{a}.\nu\sigma(P_3'(\sigma)|V_3'(\sigma))}_{k'\ \text{times}}\\
&\quad |\underbrace{\nu\tilde{a}.\nu\sigma(V_4(\sigma))|\cdots|\nu\tilde{a}.\nu\sigma(V_4(\sigma))}_{r\ \text{times}}\\
&\quad |\underbrace{\nu\tilde{a}.\nu\sigma(V_4'(\sigma))|\cdots|\nu\tilde{a}.\nu\sigma(V_4'(\sigma)))}_{r'\ \text{times}}
\end{aligned}$$

and let $\mathcal{R}$ be

$$\{(ZK(i,j,k,k',r,r'),ZK_{spec})|\forall i,j,k,k',r,r' \geqslant 0\}$$

we prove $\mathcal{R}\cup\mathcal{R}^{-1}$ be a labeled bisimulation.

From the LTS defined in Fig. 2, the process $ZK(i,j,k,k',r,r')$ can only have the following transitions:

$$\begin{array}{lll}
ZK(i,j,k,k',r,r') \xrightarrow{\tau} ZK(i+1,j,k,k',r,r') & \\
ZK(i,j,k,k',r,r') \xrightarrow{\tau} ZK(i-1,j+1,k,k',r,r') & \text{if } i>0\\
ZK(i,j,k,k',r,r') \xrightarrow{\tau} ZK(i,j-1,k+1,k',r,r') & \text{if } j>0\\
ZK(i,j,k,k',r,r') \xrightarrow{\tau} ZK(i,j-1,k,k'+1,r,r') & \text{if } j>0\\
ZK(i,j,k,k',r,r') \xrightarrow{\tau} ZK(i,j,k-1,k',r+1,r') & \text{if } k>0\\
ZK(i,j,k,k',r,r') \xrightarrow{\tau} ZK(i,j,k,k'-1,r,r'+1) & \text{if } k'>0\\
ZK(i,j,k,k',r,r') \xrightarrow{\bar{b}} ZK(i,j,k,k',r-1,r') & \text{if } r>0\\
ZK(i,j,k,k',r,r') \xrightarrow{\bar{b}} ZK(i,j,k,k',r,r'-1) & \text{if } r'>0
\end{array}$$

At the same time, the only possible transition of process ZK_{spec} is $ZK_{spec}\xrightarrow{\bar{b}}ZK_{spec}$. It is easy to check that $\mathcal{R}\cup\mathcal{R}^{-1}$ is a labeled bisimulation. So $ZK\approx_L ZK_{spec}$. $\square$

Now we consider the same proof by using the old LTS. We continue using the abbreviations and definitions in above proof. Recall the Comm rule in Appendix A:

$$\textsc{Comm}\quad \bar{a}\langle x\rangle.P|a(x).Q \to P|Q$$

So the interaction in old LTS happens only when the output message is a variable. The transitions of ZK must have the following form:

$$\begin{array}{lll}
ZK & \overset{\text{def}}{=} \nu c.!\nu\tilde{a}.(P|V) & \\
 & \equiv \nu c.(\nu\tilde{a}.(P|V)|!\nu\tilde{a}.(P|V)) & \textsc{Repl}\\
 & \equiv \nu c.(\nu\tilde{a}.(\overline{a_1}\langle \mathrm{g}(c)\rangle.P_1|\mathbf{0}|V)|\cdots) & \textsc{Par-o}\\
 & \equiv \nu c.(\nu\tilde{a}.(\overline{a_1}\langle \mathrm{g}(c)\rangle.P_1|\nu x.\{\mathrm{g}(c)/x\}|V)|\cdots) & \textsc{Alias}\\
 & \equiv \nu c.(\nu\tilde{a}.\nu x.(\overline{a_1}\langle \mathrm{g}(c)\rangle.P_1|\{\mathrm{g}(c)/x\}|V)|\cdots) & \textsc{New-Par}\\
 & \equiv \nu c.(\nu\tilde{a}.\nu x.(\overline{a_1}\langle x\rangle.P_1\{\mathrm{g}(c)/x\}|\{\mathrm{g}(c)/x\}|V)|\cdots) & \\
 & \equiv \nu c.(\nu\tilde{a}.\nu x.(\overline{a_1}\langle x\rangle.P_1|\{\mathrm{g}(c)/x\}|V)|\cdots) & \textsc{Subst}\\
 & \to ZK_1 \overset{\text{def}}{=} \nu c.(\nu\tilde{a}.\nu x.(P_1|\{\mathrm{g}(c)/x\}|V_1)|\cdots) & \textsc{Comm}\\
 & \equiv ZK_2 \overset{\text{def}}{=} \nu c.(\nu\tilde{a}.(P_1|\nu x.\{\mathrm{g}(c)/x\}|V_1)|\cdots) & \textsc{New-Par}\\
 & \equiv ZK_3 \overset{\text{def}}{=} \nu c.(\nu\tilde{a}.(P_1|\mathbf{0}|V_1)|\cdots) & \textsc{Alias}\\
 & \equiv ZK_4 \overset{\text{def}}{=} \nu c.(\nu\tilde{a}.(P_1|V_1)|\cdots) & \textsc{Par-o}
\end{array}$$

According to the rule

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

we have four possible transitions $ZK\to ZK_1,\ ZK\to ZK_2,\ ZK\to ZK_3$ and $ZK\to ZK_4$ for a single interaction. In the new LTS, for the same interaction we only need to add $(ZK(1,0,0,0,0,0),\ ZK_{spec})$ into the relation. However, in this case, we need to add four pairs into the relation. It gets more and more complex when more internal reductions are available. So in order to prove $ZK\approx_l ZK_{spec}$ we need to construct a far more complicated relation. And this is why the old semantics is ineffective.

6. Conclusion

In this paper, we establish a new pure labeled transition system and a new definition of bisimulation for the applied pi calculus. We study the algebraic theory of this calculus with new semantics, and prove the coincidence result. As we see in the example the new semantics given in this paper can decrease the size of the bisimulation relation when proving two processes are bisimilar.

As concurrency, interaction, and complexity become main characteristics of modern protocols [34], formal automatic analysis frameworks based on process calculi deserve more studies because process calculi were born to model concurrent, interactive systems. However, checking bisimulation is undecidable not only because of the replication in syntax but also due to the arbitrary equational theories. So we would like to investigate the decidability of bisimulation checking on the sub-calculus of applied pi as one possible future work. In the example we modeled the non-deterministic version of a classic probabilistic zero-knowledge proof. Another possible future work is to analyze the probabilistic version in probabilistic applied pi calculus proposed by Goubault-Larrecq et al. [24].

Acknowledgements

The author thanks Prof. Yuxi Fu and Dr. Yuxin Deng for their helpful advice on this paper. She is grateful for careful reading by Charles Liu, and for pleasant discussions with members of BASICS lab. In addition, the author wishes to thank anonymous referees for their perceptive comments on almost every page of this paper's earlier versions.

Appendix A. The original operational semantics of the applied pi calculus

Structural Rule

PAR-**0**	A	$\equiv$	$A\|\mathbf{0}$
PAR-A	$A\|(B\|C)$	$\equiv$	$(A\|B)\|C$
PAR-C	$A\|B$	$\equiv$	$B\|A$
REPL	$!P$	$\equiv$	$P\|!P$
NEW-**0**	$\nu n.\mathbf{0}$	$\equiv$	$\mathbf{0}$
NEW-C	$\nu u.\nu v.A$	$\equiv$	$\nu v.\nu u.A$
NEW-PAR	$A\|\nu u.B$	$\equiv$	$\nu u.(A\|B) \quad u \notin fv(A) \cup fn(A)$
ALIAS	$\nu x.\{M/x\}$	$\equiv$	$\mathbf{0}$
SUBST	$\{M/x\}\|A$	$\equiv$	$\{M/x\}\|A\{M/x\}$
REWRITE	$\{M/x\}$	$\equiv$	$\{N/x\} \quad \Sigma \vdash M = N$

Internal Reduction Rule

COMM	$\bar{a}\langle x\rangle.P\|a(x).Q$	$\rightarrow$	$P\|Q$
THEN	if $M = M$ then P else Q	$\rightarrow$	P
ELSE	if $M = N$ then P else Q	$\rightarrow$	Q for any ground terms M and N $\Sigma \nvdash M = N$

Labeled Rule

IN $\quad a(x).P \overset{a(M)}{\longrightarrow} P\{M/x\}$

OUT – ATOM $\quad \bar{a}\langle u\rangle.P \rightarrow^{\bar{a}\langle u\rangle} P$

OPEN – ATOM $\quad \dfrac{A \overset{\bar{a}\langle u\rangle}{\rightarrow} A' \quad u \neq a}{\nu u.A \overset{\nu u.\bar{a}\langle u\rangle}{\rightarrow} A'}$

SCOPE $\quad \dfrac{A \overset{\alpha}{\rightarrow} A' \quad u \notin n(\alpha) \cup v(\alpha)}{\nu u.A \overset{\alpha}{\rightarrow} \nu u.A'}$

PAR $\quad \dfrac{A \overset{\alpha}{\rightarrow} A' \quad bv(\alpha) \cap fv(B) = bn(\alpha) \cap fn(B) = \emptyset}{A|B \overset{\alpha}{\rightarrow} A'|B}$

STRUCT $\quad \dfrac{A \equiv B \quad B \overset{\alpha}{\rightarrow} B' \quad B' \equiv A'}{A \overset{\alpha}{\rightarrow} A'}$

Appendix B. The Proof of Theorem 5

In order to prove Theorem 5, we need some lemmas. The first one shows static equivalence is closed under evaluation context.

Lemma 9. *For any two processes A, B, if $A \approx_s B$, then for any evaluation context $C[\_]$ we have $C[A] \approx_s C[B]$.*

This lemma follows directly from Lemma 1 in [3]. The second one captures some properties of *corresponding equivalence*.

Lemma 10. *The corresponding equivalence has the following properties:*

1. *If $M \asymp_{\{\phi_A,\phi_B\}} N$, and $\nu n.\phi_A \vdash M$, then $\nu n.\phi_B \vdash N$ and $M \asymp_{\{\nu n.\phi_A, \nu n.\phi_B\}} N$;*

2. *If $M \asymp_{\{\phi_A,\phi_B\}} N$, then $M \asymp_{\{\phi_A|\phi_C,\phi_B|\phi_C\}} N$ for any frame ϕ_C;*
3. *If $M \asymp_{\{\phi_A,\phi_B\}} N$ where $\phi_A \overset{\text{def}}{=} \nu\tilde{m}.\{\widetilde{M}/\tilde{x}\}$ and $\phi_B \overset{\text{def}}{=} \nu\tilde{n}.\{\widetilde{N}/\tilde{x}\}$, then $\nu\tilde{m}.(\{M/x\}|\{\widetilde{M}/\tilde{x}\}) \approx_s \nu\tilde{n}.(\{N/x\}|\{\widetilde{N}/\tilde{x}\})$ for some fresh x.*

Proof. We denote the active substitutions of ϕ_A, ϕ_B, ϕ_C as $\sigma_A, \sigma_B, \sigma_C$.

1. If $M \asymp_{\{\phi_A,\phi_B\}} N$ and $\nu n.\phi_A \vdash M$, then there exists some term ζ such that $M = \zeta\sigma_A$ and $n \notin fn(\zeta)$. By the definition of $\asymp$, we have $N = \zeta\sigma_B$. By Lemma 9 one gets $\nu n.\phi_A \approx_s \nu n.\phi_B$. So $\nu n.\phi_B \vdash N$ and $M \asymp_{\{\nu n.\phi_A,\nu n.\phi_B\}} N$.
2. This case is very easy.
3. First, there exists a term ζ such that $M = \zeta\sigma_A$. Secondly, assume two sets:

$$T_1 \overset{\text{def}}{=} \{(M,N)|(M=N)\phi_A \text{ and } (fn(M)\cup fn(N))\cap\tilde{m}=\emptyset\}$$
$$T_2 \overset{\text{def}}{=} \{(M,N)|(M=N)\phi'_A \text{ and } (fn(M)\cup fn(N))\cap\tilde{m}=\emptyset\}$$

where ϕ'_A is the frame of $\nu\tilde{m}.(\{M/x\}|\{\widetilde{M}/\tilde{x}\})$.
Obliviously, $T_1 \subseteq T_2$, and $T_2 \subseteq T_1$ by replacing every x in terms of T_2 with ζ. Similar result holds for ϕ_B. So we get

$$\nu\tilde{m}.(\{M/x\}|\{\widetilde{M}/\tilde{x}\}) \approx_s \nu\tilde{n}.(\{N/x\}|\{\widetilde{N}/\tilde{x}\}). \qquad \square$$

The following lemma says if two frames are statically equivalent then they will also be statically equivalent after both removing an active substitution with same variable.

Lemma 11. *If $\nu\tilde{m}.(\{M/x\}|A) \approx_s \nu\tilde{n}.(\{N/x\}|B)$, then $\nu\tilde{m}.A \approx_s \nu\tilde{n}.B$.*

Proof. It is much easier to prove the converse-negative version of this lemma:

$$\nu\tilde{m}.A \not\approx_s \nu\tilde{n}.B \text{ implies } \nu\tilde{m}.(\{M/x\}|A) \not\approx_s \nu\tilde{n}.(\{N/x\}|B).$$

If we have a witness to show that $\nu\tilde{m}.A \not\approx_s \nu\tilde{n}.B$, then the same witness can show $\nu\tilde{m}.(\{M/x\}|A) \approx_s \nu\tilde{n}.(\{N/x\}|B)$. $\square$

Lemma 12. *The symmetric relation*

$$\mathcal{R} \overset{\text{def}}{=} \left\{ (\nu\tilde{m}.(A|D_1), \nu\tilde{n}.(B|D_2)) \;\middle|\; \begin{array}{c} \nu\tilde{m}.(\{\widetilde{M}/\tilde{x}\}|A) \approx_L \nu\tilde{n}.(\{\widetilde{N}/\tilde{x}\}|B), \\ D_1 = D\{\widetilde{M}/\tilde{x}\} \text{ and } D_2 = D\{\widetilde{N}/\tilde{x}\} \\ \text{for some } D, \widetilde{M}, \widetilde{N}, \tilde{x} \\ \text{and } fn(D)\cap(\tilde{m}\cup\tilde{n}) = \emptyset \end{array} \right\}$$

is a *labeled bisimulation.*

Proof. Given some $D, \widetilde{M}, \widetilde{N}, \tilde{x}, \tilde{m}$, and $\tilde{n}$ such that $fn(D)\cap(\tilde{m}\cup\tilde{n}) = \emptyset$, and

$$\nu\tilde{m}.(\{\widetilde{M}/\tilde{x}\}|A) \approx_L \nu\tilde{n}.(\{\widetilde{N}/\tilde{x}\}|B) \tag{2}$$

where $D_1 = D\{\widetilde{M}/\tilde{x}\}$ and $D_2 = D\{\widetilde{N}/\tilde{x}\}$. We use ϕ_1 to denote the frame of $\nu\tilde{m}.(\{\widetilde{M}/\tilde{x}\}|A)$ and ϕ_2 for $\nu\tilde{n}.(\{\widetilde{N}/\tilde{x}\}|B)$. ϕ'_1, ϕ'_2 are the frames of $\nu\tilde{m}.(A|D_1), \nu\tilde{n}.(B|D_2)$ respectively.

1. We denote the frames of A, B, D with ϕ_A, ϕ_B, ϕ_D. By Lemma 11, one can get $\nu\tilde{m}.A \approx_s \nu\tilde{n}.B$. So

$$\nu\tilde{m}.\phi_A \approx_s \nu\tilde{n}.\phi_B$$

By Lemma 9

$$(\nu\tilde{m}.\phi_A)|\phi_D \approx_s (\nu\tilde{n}.\phi_B)|\phi_D$$

Since $fn(D)\cap(\tilde{m}\cup\tilde{n}) = \emptyset$ and $\phi_{D_1} = \phi_{D_2} = \phi_D$, we get

$$\phi'_1 = \nu\tilde{m}.(\phi_A|\phi_D) \approx_s (\nu\tilde{m}.\phi_A)|\phi_D \approx_s (\nu\tilde{n}.\phi_B)|\phi_D \approx_s \nu\tilde{n}.(\phi_B|\phi_D) = \phi'_2$$

So we have $\nu\tilde{m}.(A|D_1) \approx_s \nu\tilde{n}.(B|D_2)$.
2. If

$$\nu\tilde{m}.(A|D_1) \xrightarrow{\alpha} \nu\tilde{m}.(A'|D_1)$$

We consider the cases of α:
   (a) if $\alpha = \tau$, then by rules Res in Fig. 2 we must have that

$$A|D_1 \xrightarrow{\tau} A'|D_1$$

hence, by rule Par-L,

$$A \xrightarrow{\tau} A'$$

By rules Par-l and Res in Fig. 2, we have

$$\nu\tilde{m}.(\{\widetilde{M}/\tilde{x}\}|A) \xrightarrow{\tau} \nu\tilde{m}.(\{\widetilde{M}/\tilde{x}\}|A').$$

Then there exists some B' s.t.

$$\begin{aligned} \nu\tilde{n}.(\{\widetilde{N}/\tilde{x}\}|B) &\overset{\hat{\tau}}{\Rightarrow} \nu\tilde{n}.(\{\widetilde{N}/\tilde{x}\}|B') \\ &\approx_L \nu\tilde{m}.(\{\widetilde{M}/\tilde{x}\}|A') \end{aligned}$$

So $\nu\tilde{n}.(B|D_2) \overset{\hat{\tau}}{\Rightarrow} \nu\tilde{n}.(B'|D_2)$ and $(\nu\tilde{m}.(A'|D_1), \nu\tilde{n}.(B'|D_2)) \in \mathcal{R}$.

(b) if $\alpha = aM'$, then by rules Res in Fig. 2 we must have that

$$A|D_1 \xrightarrow{aM'} A'|D_1 \quad \text{and} \quad \phi_1' \vdash a,\ \phi_1' \vdash M'$$

hence, by rule Par-L,

$$A \xrightarrow{aM'} A'$$

Since $fn(D) \cap (\tilde{m} \cup \tilde{n}) = \emptyset$, therefore, $\phi_1 \vdash a$ and $\phi_1 \vdash M'$. By rules Par-L and Res in Fig. 2, we have

$$\nu\tilde{m}.(\{\widetilde{M}/\tilde{x}\}|A) \xrightarrow{aM'} \nu\tilde{m}.(\{\widetilde{M}/\tilde{x}\}|A').$$

Then there exist some B', N', b, s.t. $a \asymp_{\{\phi_1,\phi_2\}} b$ and

$$\begin{aligned} \nu\tilde{n}.(\{\widetilde{N}/\tilde{x}\}|B) &\overset{bN'}{\Rightarrow} \nu\tilde{n}.(\{\widetilde{N}/\tilde{x}\}|B') \\ &\approx_L \nu\tilde{m}.(\{\widetilde{M}/\tilde{x}\}|A') \end{aligned}$$

By Proposition 3, there exists a term ζ, such that $M = \zeta\sigma_{\phi_1'}$. Because $\phi_1 \vdash M'$, so $fv(\zeta) \cap (dom(\phi_D) \cup \tilde{x}) = \emptyset$. From $\nu\tilde{m}.\phi_A \approx_s \nu\tilde{n}.\phi_B$ one gets

$$M' \asymp_{\{\nu\tilde{m}.\phi_A, \nu\tilde{n}.\phi_B\}} N'.$$

By Lemma 10(1), we have

$$M' \asymp_{\{\nu\tilde{m}.\phi_A|\phi_D, \nu\tilde{n}.\phi_B|\phi_D\}} N'.$$

Similarly, we can prove

$$a \asymp_{\{\nu\tilde{m}.\phi_A|\phi_D, \nu\tilde{n}.\phi_B|\phi_D\}} b$$

So there exist some B', N', b s.t. $M' \asymp_{\{\phi_1',\phi_2'\}} N'$, $a \asymp_{\{\phi_1',\phi_2'\}} b$, and

$$\nu\tilde{n}.(B|D_2) \overset{bN'}{\Rightarrow} \nu\tilde{n}.(B'|D_2), \quad (\nu\tilde{m}.(A'|D_1), \nu\tilde{n}.(B'|D_2)) \in \mathcal{R}.$$

(c) if $\alpha = \bar{a}x$, this case is similar to the previous one.

3. If $\nu\tilde{m}.(A|D_1) \xrightarrow{\alpha} \nu\tilde{m}.(A|D_1')$ then by rules Res in Fig. 2 we must have that

$$A|D_1 \xrightarrow{\alpha} A|D_1'$$

hence, by rule Par-L,

$$D_1 \xrightarrow{\alpha} D_1'$$

(a) $\alpha = \bar{a}x$ for some a and x. W.l.o.g, assume $D \overset{\text{def}}{=} \nu\tilde{r}.(\bar{\zeta}\langle M'\rangle.C_1|C_2)$ and $\zeta\{\widetilde{M}/\tilde{x}\} = a$, so there exists

$$D' \overset{\text{def}}{=} \nu\tilde{r}.(C_1|C_2|\{M'/x\}), \quad D_1' = D'\{\widetilde{M}/\tilde{x}\}, \quad D_2' = D'\{\widetilde{N}/\tilde{x}\}$$

such that

$$\nu\tilde{n}.(B|D_2) \xrightarrow{\bar{b}x} \nu\tilde{n}.(B|D_2') \text{ and } b = \zeta\{\widetilde{N}/\tilde{x}\}.$$

So

$$a \asymp_{\{\phi_1',\phi_2'\}} b, \text{ and } (\nu\tilde{m}.(A|D_1'), \nu\tilde{n}.(B|D_2')) \in \mathcal{R}.$$

(b) $\alpha = aO_1$ for some a, O_1 and $\phi_1' \vdash O_1$. W.l.o.g, assume

$$D \overset{\text{def}}{=} \nu\tilde{r}.(\zeta(z).C_1|C_2)$$

where $\zeta\{\widetilde{M}/\tilde{x}\} = a$ and $z \notin \tilde{x}$, let

$$D' \overset{\text{def}}{=} \nu\tilde{r}.(C_1|C_2), \quad D_1' = D'\{\widetilde{M}/\tilde{x}\}\{O_1/z\}, \quad D_2' = D'\{\widetilde{N}/\tilde{x}\}\{O_2/z\}$$

where $O_1 \asymp_{\{\phi_1',\phi_2'\}} O_2$.
By Lemma 10, we get

$$\nu\tilde{m}.(A|\{\widetilde{M}/\tilde{x}\}|\{O_1/z\}) \approx_L \nu\tilde{n}.(B|\{\widetilde{M}/\tilde{x}\}|\{O_2/z\})$$

So there exists b s.t. $\nu\tilde{n}.(B|D_2) \xrightarrow{bO_2} \nu\tilde{n}.(B|D_2')$, and $b = \zeta\{\widetilde{N}/\tilde{x}\}$. Therefore,

$$a \asymp_{\{\phi_1',\phi_2'\}} b, \quad (\nu\tilde{m}.(A|D_1'), \nu\tilde{n}.(B|D_2')) \in \mathcal{R}.$$

(c) $\alpha = \tau$. Let

$$D_1 \overset{\text{def}}{=} D\{\widetilde{M}/\tilde{x}\}, \quad D_2 \overset{\text{def}}{=} D\{\widetilde{N}/\tilde{x}\}.$$

Since $D_1 \xrightarrow{\tau} D_1'$, D can perform τ reduction too. So $D \xrightarrow{\tau} D'$ and $D_1' = D'\{\widetilde{N}/\tilde{x}\}$, $D_2' = D'\{\widetilde{N}/\tilde{x}\}$ where $D_2 \xrightarrow{\tau} D_2'$. We get

$$\nu\tilde{n}.(B|D_2) \xrightarrow{\tau} \nu\tilde{n}.(B|D_2')$$

and

$$(\nu\tilde{m}.(A|D_1'), \nu\tilde{n}.(B|D_2')) \in \mathcal{R}.$$

4. If $\nu\tilde{m}.(A|D_1) \xrightarrow{\tau} \nu\tilde{m}.\nu\widetilde{m'}.(A'|D_1')$, where $A \xrightarrow{\bar{a}x} \nu\widetilde{m'}.(A'|\{M/x\})$ and $D_1 \xrightarrow{aM} D_1'$ for some a, x, M. W.l.o.g., assume

$$D \overset{\text{def}}{=} \nu\tilde{r}.(\zeta(z).C_1|C_2),$$

where $\zeta\{\widetilde{M}/\tilde{x}\} = a$ and $z \notin \tilde{x}$, so there exists D' such that

$$D' \overset{\text{def}}{=} \nu\tilde{r}.(C_1|C_2), \text{ and } D_1' = D'\{\widetilde{M}/\tilde{x}\}\{M/z\}, \; D_2' = D'\{\widetilde{N}/\tilde{x}\}\{N/z\}.$$

Because

$$\nu\tilde{m}.(A|\{\widetilde{M}/\tilde{x}\}) \xrightarrow{\bar{a}x} \nu\tilde{m}.\nu\widetilde{m'}.(A|\{M/x\}|\{\widetilde{M}/\tilde{x}\})$$

and

$$\nu\tilde{m}.(A|\{\widetilde{M}/\tilde{x}\}) \approx_L \nu\tilde{n}.(B|\{\widetilde{N}/\tilde{x}\})$$

There exists some B', b s.t. $a \asymp_{\{\phi_1,\phi_2\}} b$ and

$$\begin{aligned} \nu\tilde{n}.(B|\{\widetilde{N}/\tilde{x}\}) &\overset{\bar{b}x}{\Rightarrow} \nu\tilde{n}.\nu\widetilde{n'}.(B'|\{N/x\}|\{\widetilde{N}/\tilde{x}\}) \\ &\approx_L \nu\tilde{m}.\nu\widetilde{m'}.(A'|\{M/x\}|\{\widetilde{M}/\tilde{x}\}) \end{aligned}$$

Since $a \asymp_{\{\phi_1,\phi_2\}} b$ and $a = \zeta\{\widetilde{M}/\tilde{x}\}$, therefore, $b = \zeta\{\widetilde{N}/\tilde{x}\}$. So

$$\nu\tilde{n}.(B|D_2) \overset{\tau}{\Rightarrow} \nu\tilde{n}.(B'|D_2')$$

and

$$(\nu\tilde{m}.(A'|D_1'), \nu\tilde{n}.(B'|D_2')) \in \mathcal{R}.$$

5. If $\nu\tilde{m}.(A|D_1) \xrightarrow{\tau} \nu\tilde{m}.\nu\tilde{r}.(A'|D_1')$, where $A \xrightarrow{aO_1} A'$ and $D_1 \xrightarrow{\bar{a}z} \nu\tilde{r}.(D_1'|\{O_1/x\})$ for some a, z, O_1. W.l.o.g, let $D \overset{\text{def}}{=} \nu\tilde{r}.(\bar{\zeta}\langle O\rangle.C_1|C_2)$, $\zeta\{\widetilde{M}/\tilde{x}\} = a$ and

$$D' \overset{\text{def}}{=} \nu\tilde{r}.(C_1|C_2), \quad D_1' = D'\{\widetilde{M}/\tilde{x}\}, \quad D_2' = D'\{\widetilde{N}/\tilde{x}\}.$$

There exist some O_2, b such that

$$\begin{aligned}
&\nu\tilde{n}.(\{\widetilde{M}/\tilde{x}\}|A) \overset{a\,O_1}{\rightarrow} \nu\tilde{n}.(\{\widetilde{N}/\tilde{x}\}|A')\\
&\nu\tilde{n}.(\{\widetilde{N}/\tilde{x}\}|B) \overset{b\,O_2}{\Rightarrow} \nu\tilde{n}.(\{\widetilde{N}/\tilde{x}\}|B')\\
&\qquad\qquad\qquad \approx_L \nu\tilde{m}.(\{\widetilde{M}/\tilde{x}\}|A')
\end{aligned}$$

where $O_1 \asymp_{\{\phi_1,\phi_2\}} O_2$ and $a \asymp_{\{\phi_1,\phi_2\}} b$.
Since $a \asymp_{\{\phi_1,\phi_2\}} b$ and $a = \zeta\{\widetilde{M}/\tilde{x}\}$, therefore, $b = \zeta\{\widetilde{N}/\tilde{x}\}$.
So

$$\nu\tilde{n}.(B|D_2) \overset{\tau}{\Rightarrow} \nu\tilde{n}.(B'|D_2')$$

and

$$(\nu\tilde{m}.(A'|D_1'), \nu\tilde{n}.(B'|D_2')) \in \mathcal{R}.$$

We can conclude that $\mathcal{R}$ is a *labeled bisimulation*. □

Now we are ready to prove the theorem.

Theorem 5. *For any two closed processes A, B, if $A \approx_L B$, then for any evaluation context $C[\_]$ we have $C[A] \approx_L C[B]$.*

Proof. We need to show the following relation is labeled bisimulation:

$$\mathcal{R} \overset{\text{def}}{=} \approx_L \cup \{(C[A], C[B]) | A \approx_L B \text{ and } C[\_] \text{ is an evaluation context}\}$$

To prove a relation $\mathcal{R}$ is labeled bisimulation, one needs to show that for any $A\mathcal{R}B$ we have

1. $A \approx_s B$.
2. if $A \overset{\tau}{\rightarrow} A'$, then $\exists B'$, s.t. $B \overset{\hat{\tau}}{\Rightarrow} B'$ and $A'\mathcal{R}B'$.
3. if $A \overset{\bar{a}x}{\rightarrow} A'$, then $\exists b, B'$, s.t. $B \overset{\bar{b}x}{\Rightarrow} B'$ with $a \asymp_{\{\phi_A,\phi_B\}} b$ and $A'\mathcal{R}B'$.
4. if $A \overset{aM_1}{\rightarrow} A'$,then $\exists b, B'$ and $\phi_B \vdash M_2$, s.t. $B \overset{b\,M_2}{\Rightarrow} B'$, $M_1 \asymp_{\{\phi_A,\phi_B\}} M_2$, $a \asymp_{\{\phi_A,\phi_B\}} b$ and $A'\mathcal{R}B'$,

It is clear that we only need to consider those pairs $(C[A], C[B])$ for some processes A, B and some evaluation context $C[\_]$ such that $A \approx_L B$. We consider the different cases of $C[\_]$.

1. $C[\_] = \_$ This case is trivial.
2. $C[\_] = \nu n.\_$ for some name n.
   (a) By Lemma 9, we get $\nu n.A \approx_s \nu n.B$.
   (b) If $\nu n.A \overset{\tau}{\rightarrow} A''$, then by inspection of the rules of Fig. 2, we have that $A'' = \nu n.A'$ and $A \overset{\tau}{\rightarrow} A'$. So we have

$$\nu n.A \overset{\tau}{\rightarrow} \nu n.A'.$$

By the fact $A \approx_L B$, there exists some B' such that $B \overset{\hat{\tau}}{\Rightarrow} B'$ and $A' \approx_L B'$. So

$$\nu n.B \overset{\hat{\tau}}{\Rightarrow} \nu n.B' \text{ and } (\nu n.A')\mathcal{R}(\nu n.B').$$

   (c) If $\nu n.A \overset{\bar{a}x}{\rightarrow} A''$ for some a and x, then by rule Res-Out in Fig. 2, we have $A'' = \nu n.A'$, $\phi_{\nu n.A} \vdash a$ and $A \overset{\bar{a}x}{\rightarrow} A'$. So we have

$$\nu n.A \overset{\bar{a}x}{\rightarrow} \nu n.A'.$$

By the fact $A \approx_L B$, there exists some b and B' such that

$$B \overset{\bar{b}x}{\Rightarrow} B',\ a \asymp_{\{\phi_A,\phi_B\}} b \text{ and } A' \approx_L B'.$$

Since $\phi_{\nu n.A} = \nu n.\phi_A \vdash a$, followed by (1) of Lemma 10 we have $\phi_{\nu n.B} = \nu n.\phi_B \vdash b$ and $a \asymp_{\{\phi_{\nu n.A},\phi_{\nu n.B}\}} b$. Therefore,

$$\nu n.B \overset{\bar{b}x}{\Rightarrow} \nu n.B',\ a \asymp_{\{\phi_{\nu n.A},\phi_{\nu n.B}\}} b \text{ and } (\nu n.A')\mathcal{R}(\nu n.B').$$

   (d) The last case is when $\nu n.A \overset{aM}{\rightarrow} A''$, by rule Res-In, we get $\phi_{\nu n.A} \vdash a$ and $\phi_{\nu n.A} \vdash M$, and $A'' = \nu n.A'$ and $A \overset{aM}{\rightarrow} A'$. We have

$$\nu n.A \overset{aM}{\rightarrow} \nu n.A'.$$

By the fact $A \approx_L B$, there exists some b, N and B' such that

$$B \overset{bN}{\Rightarrow} B', \ a \asymp_{\{\phi_A, \phi_B\}} b, \ M \asymp_{\{\phi_A, \phi_B\}} N, \text{ and } A' \approx_L B'.$$

Since $\phi_{\nu n.A} = \nu n.\phi_A$, followed by (1) of Lemma 10 we have

$$\phi_{\nu n.B} = \nu n.\phi_B \vdash b, \ \phi_{\nu n.B} = \nu n.\phi_B \vdash N$$

and

$$a \asymp_{\{\phi_{\nu n.A}, \phi_{\nu n.B}\}} b, \quad M \asymp_{\{\phi_{\nu n.A}, \phi_{\nu n.B}\}} N.$$

Therefore,

$$\nu n.B \overset{bN}{\Rightarrow} \nu n.B', \text{ and } (\nu n.A')\mathcal{R}(\nu n.B').$$

3. $C[\_] = \_|D$ for some D;
   (a) By Lemma 9, we get $A|D \approx_s B|D$.
   (b) If $A|D$ can perform an action α, then by rules in Fig. 2 three cases might happen:
      (i) This action is caused by the same action performed by D;
      (ii) This action is caused by the same action performed by A;
      (iii) $\alpha = \tau$ and is caused by the interaction between A and D;

   We consider these cases one by one.
   i. Assume α is caused by D which means $D \xrightarrow{\alpha} D'$ hence $A|D \xrightarrow{\alpha} A|D'$. Then directly we have

$$B|D \xrightarrow{\alpha} B|D' \text{ and } (A|D')\mathcal{R}(B|D').$$

   ii. Assume α is caused by A, i.e. $A \xrightarrow{\alpha} A'$ hence $A|D \xrightarrow{\alpha} A'|D$. This case is similar to the case where $C[\_] = \nu n.\_$. The proof is the same except that instead of using (1) of Lemma 10 the proof will refer to (2) of Lemma 10.
   iii. When $\alpha = \tau$ and is caused by the interaction between A and D,
      - If $A|D \xrightarrow{\tau} \nu \tilde{n}.(A'|D')$ where

$$D \xrightarrow{\bar{a}x} \nu \tilde{n}.(\{M/x\}|D'), \quad A \xrightarrow{aM} A,$$

and $\tilde{n} \subseteq n(M)$, $fn(M) \cap (bn(A) \cup bn(B)) = \emptyset$. Because $A \approx_L B$, then $M \asymp_{\{\phi_A, \phi_B\}} M$ and there exists some B', $B \overset{aM}{\Rightarrow} B'$ and $A' \approx_L B'$. So $(C'[A'], C'[B']) \in \mathcal{R}$ where $C'[\_] \overset{\text{def}}{=} \nu \tilde{n}.(D'|\_)$.
      - If $A|D \xrightarrow{\tau} \nu \tilde{n}.(A'|D_1)$ where

$$A \xrightarrow{\bar{a}x} \nu \tilde{n}.(\{M/x\}|A'), \quad D \xrightarrow{aM} D_1.$$

and $\tilde{n} \subseteq n(M)$, $fn(M) \cap bn(D_1) = \emptyset$. Because $A \approx_L B$, then there exists some B', such that

$$B \overset{\bar{a}x}{\Rightarrow} \nu \tilde{n}.(\{N/x\}|B'), \quad \nu \tilde{m}.(\{M/x\}|A') \approx_L \nu \tilde{n}.(\{N/x\}|B').$$

Because $D \xrightarrow{aN} D_2$, so $B|D \overset{\hat{\tau}}{\Rightarrow} \nu \tilde{n}.(B'|D_2)$.
If $D \xrightarrow{ax} D'$ then

$$D_1 = D'\{M/x\}, \quad D_2 = D'\{N/x\},$$
$$\nu \tilde{m}.(\{M/x\}|A') \approx_L \nu \tilde{n}.(\{N/x\}|B').$$

By Lemma 12, one can get

$$(\nu \tilde{m}.(D_1|A'), \nu \tilde{n}.(D_2|B')) \in \mathcal{R}$$

We can conclude that $\mathcal{R}$ is a *labeled bisimulation*, and for any two processes A, B, if $A \approx_L B$, then for any evaluation context $C[\_]$ we have $C[A] \approx_L C[B]$. We are done. □

Appendix C. The proof of Theorem 6

The proof of this theorem is more technical than that of Theorem 5. We extend the method of proving the coincidence between barbed congruence and weak bisimilarity of π-calculus by Sangiorgi in his Ph.D thesis [33].

The following lemma is a general result valid for all the equivalences we are aware of. It is originated from X-property proposed by Nicola et al. [13] and then studied by Yuxi Fu as *Bisimulation Lemma* in [18,19].

Lemma 13 (Bisimulation lemma). *If $P \overset{\hat{\tau}}{\Rightarrow} P' \approx_{obs} Q$ and $Q \overset{\hat{\tau}}{\Rightarrow} Q' \approx_{obs} P$ then $P \approx_{obs} Q$.*

Lemma 14. *For any closed plain processes P, Q, if $a(x)|P \approx_{obs} a(x)|Q$ and $a \notin (fn(P) \cup fnQ)$ then $P|R \approx_{obs} Q|R$ where R is any closed plain process.*

Proof. Given context $C[\_] \overset{\text{def}}{=} \_|\bar{a}\langle m\rangle.(\bar{d}|d|R)$ for any plain closed process R and fresh name d. We have $C[a(x)|P] \approx_{obs} C[a(x)|Q]$ from $a(x)|P \approx_{obs} a(x)|Q$, so the following transitions

$$\begin{aligned} C[a(x)|P] &\xrightarrow{\tau} P|\bar{d}|d|R \\ &\xrightarrow{\tau} P|R \end{aligned}$$

must be simulated by

$$\begin{aligned} C[a(x)|Q] &\overset{\tau}{\Rightarrow} Q'|\bar{d}|d|R \\ &\overset{\hat{\tau}}{\Rightarrow} Q'' \approx_{obs} P|R \end{aligned}$$

which can be rearranged

$$\begin{aligned} C[a(x)|Q] &\xrightarrow{\tau} Q|\bar{d}|d|R \\ &\xrightarrow{\tau} Q|R \\ &\overset{\hat{\tau}}{\Rightarrow} Q'' \approx_{obs} P|R \end{aligned}$$

So $Q|R \overset{\hat{\tau}}{\Rightarrow} Q'' \approx_{obs} P|R$. Similarly, one has $P|R \overset{\hat{\tau}}{\Rightarrow} P'' \approx_{obs} Q|R$. Thus $P|R \approx_{obs} Q|R$ by Lemma 13. □

Theorem 6. *For any closed plain processes P, Q, if $P \approx_{obs} Q$, then $P \approx_L Q$.*

Proof. Assume $\mathcal{R}$ is a symmetric relation and defined as follows.

$$\left\{ \begin{array}{l|l} & \widetilde{M} = \{M_1, M_2, \ldots, M_n\}, \widetilde{N} = \{N_1, N_2, \ldots, N_n\}, \\ & \tilde{m} \subseteq fn(\widetilde{M}),\ \tilde{n} \subseteq fn(\widetilde{N}), \\ (\nu\tilde{m}.(P|\{\widetilde{M}/\tilde{x}\}), & \nu\tilde{m}.\{\widetilde{M}/\tilde{x}\} \approx_s \nu\tilde{n}.\{\widetilde{N}/\tilde{x}\},\ \text{and} \\ \nu\tilde{n}.(Q|\{\widetilde{N}/\tilde{x}\})) & c(y)|\nu\tilde{m}.(P|!\bar{a}_1\langle M_1\rangle|!\bar{a}_2\langle M_2\rangle|\cdots|!\bar{a}_n\langle M_n\rangle) \\ & \approx_{obs} \\ & c(y)|\nu\tilde{n}.(Q|!\bar{a}_1\langle N_1\rangle|!\bar{a}_2\langle N_2\rangle|\cdots|!\bar{a}_n\langle N_n\rangle) \\ & \text{for some fresh names } c, a_1, a_2, \ldots, a_n. \end{array} \right\}$$

Now we prove that $\mathcal{R}$ is a *labeled bisimulation*.

To avoid long expressions, we introduce the following abbreviations:

$$\begin{aligned} O_1 &\overset{\text{def}}{=} !\bar{a}_1\langle M_1\rangle|!\bar{a}_2\langle M_2\rangle|\cdots|!\bar{a}_n\langle M_n\rangle \\ O_2 &\overset{\text{def}}{=} !\bar{a}_1\langle N_1\rangle|!\bar{a}_2\langle N_2\rangle|\cdots|\bar{a}_n\langle N_n\rangle \\ \phi_1 &\overset{\text{def}}{=} \nu\tilde{m}.\{\widetilde{M}/\tilde{x}\} \\ \phi_2 &\overset{\text{def}}{=} \nu\tilde{n}.\{\widetilde{N}/\tilde{x}\} \end{aligned}$$

For any $\nu\tilde{m}.(P|\{\widetilde{M}/\tilde{x}\})\mathcal{R}\nu\tilde{n}.(Q|\{\widetilde{N}/\tilde{x}\})$, we have

$$c(y)|\nu\tilde{m}.(P|O_1) \approx_{obs} c(y)|\nu\tilde{n}.(Q|O_2),$$

then by Lemma 14,

$$\nu\tilde{m}.(P|O_1)|R \approx_{obs} \nu\tilde{n}.(Q|O_2)|R$$

for any closed plain process R. We will use this result without explicitly referring to it.

Now we are ready to prove $\mathcal{R}$ is a labeled bisimulation. For each pair

$$(\nu\tilde{m}.(P|\{\widetilde{M}/\tilde{x}\}), \nu\tilde{n}.(Q|\{\widetilde{N}/\tilde{x}\})) \in \mathcal{R}$$

We consider cases of the following transition:

$$\nu\tilde{m}.(P|\{\widetilde{M}/\tilde{x}\}) \xrightarrow{\alpha}$$

1. if $\alpha = \tau$, then the above transition must be caused by $P \xrightarrow{\tau} P'$, therefore,

$$c(y)|\nu\tilde{m}.(P|O_1) \xrightarrow{\tau} c(y)|\nu\tilde{m}.(P'|O_1)$$

where c is fresh. This reduction must be simulated by

$$\begin{aligned} c(y)|\nu\tilde{n}.(Q|O_2) &\overset{\hat{\tau}}{\Rightarrow} c(y)|\nu\tilde{n}.(Q'|O_2) \\ &\approx_{obs} c(y)|\nu\tilde{m}.(P'|O_1) \end{aligned}$$

So if $\nu\tilde{m}.(P|\{\widetilde{M}/\tilde{x}\}) \xrightarrow{\tau} \nu\tilde{m}(P'|\{\widetilde{M}/\tilde{x}\})$, there exists some Q',

$$\nu\tilde{n}.(Q|\{\widetilde{N}/\tilde{x}\}) \overset{\hat{\tau}}{\Rightarrow} \nu\tilde{n}.(Q'|\{\widetilde{N}/\tilde{x}\})$$

and

$$(\nu\tilde{m}.(P'|\{\widetilde{M}/\tilde{x}\}), \nu\tilde{n}.(Q'|\{\widetilde{N}/\tilde{x}\})) \in \mathcal{R}.$$

2. if $\alpha = \bar{a}x$ for some a, x, then

$$\nu\tilde{m}.(P|\{\widetilde{M}/\tilde{x}\}) \xrightarrow{\bar{a}x} \nu\tilde{m}.\nu\widetilde{m'}.(P'|\{M/x\}|\{\widetilde{M}/\tilde{x}\})$$

where $\widetilde{m'} \subseteq fn(M)$. Assume $C[\_]$ be

$$\_|a(x).(\bar{b}|b|!\bar{a}_{n+1}\langle x\rangle|c(y))$$

where b, c, a_{n+1} are fresh. We have

$$\begin{aligned} C[\nu\tilde{m}.(P|O_1)] &\xrightarrow{\tau} \nu\tilde{m}.\nu\widetilde{m'}.(P'|O_1|(\bar{b}|b|!\bar{a}_{n+1}\langle M\rangle|c(y))) \\ &\xrightarrow{\tau} \nu\tilde{m}.\nu\widetilde{m'}.(P'|O_1|!\bar{a}_{n+1}\langle M\rangle|c(y)) \end{aligned}$$

So there must exist some term N and $C[\nu\tilde{n}.(Q|O_2)]$ has to simulate above transition with

$$\begin{aligned} C[\nu\tilde{n}.(Q|O_2)] &\overset{\hat{\tau}}{\Rightarrow} \nu\widetilde{n''}.(Q''|O_2|(\bar{b}|b|!\bar{a}_{n+1}\langle N\rangle|c(y))) \\ &\overset{\hat{\tau}}{\Rightarrow} Q''' \approx_{obs} \nu\tilde{m}.\nu\widetilde{m'}.(P'|O_2|!\bar{a}_{n+1}\langle M\rangle|c(y)) \end{aligned}$$

This reduction can be rearranged as

$$\begin{aligned} C[Q] &\xrightarrow{\tau} \nu\tilde{n}.\nu\widetilde{n'}.(Q'|O_2|(\bar{b}|b|!\bar{a}_{n+1}\langle N\rangle|c(y))) \\ &\xrightarrow{\tau} \nu\tilde{n}.\nu\widetilde{n'}.(Q'|O_2|!\bar{a}_{n+1}\langle N\rangle|c(y)) \\ &\overset{\hat{\tau}}{\Rightarrow} Q''' \approx_{obs} \nu\tilde{m}.\nu\widetilde{m'}.(P'|O_1|!\bar{a}_{n+1}\langle M\rangle|c(y)) \end{aligned}$$

where

$$\nu\tilde{n}.(Q|\{\widetilde{N}/\tilde{x}\}) \xrightarrow{\bar{a}x} \nu\tilde{n}.\nu\widetilde{n'}.(Q'|\{N/x\}|\{\widetilde{N}/\tilde{x}\}).$$

One can get that

$$\nu\tilde{n}.\nu\widetilde{n'}.(Q'|O_2|!\bar{a}_{n+1}\langle N\rangle|c(y)) \overset{\hat{\tau}}{\Rightarrow} Q''' \approx_{obs} \nu\tilde{m}.\nu\widetilde{m'}.(P'|O_1|!\bar{a}_{n+1}\langle M\rangle|c(y))$$

Similarly,

$$\nu\tilde{m}.\nu\widetilde{m'}.(P'|O_1|!\bar{a}_{n+1}\langle M\rangle|c(y)) \overset{\hat{\tau}}{\Rightarrow} P''' \approx_{obs} \nu\tilde{n}.\nu\widetilde{n'}.(Q'|O_2|!\bar{a}_{n+1}\langle N\rangle|c(y))$$

By Lemma 13, we have

$$\nu\tilde{m}.\nu\widetilde{m'}.(P'|O_1|!\bar{a}_{n+1}\langle M\rangle|c(y)) \approx_{obs} \nu\tilde{n}.\nu\widetilde{n'}.(Q'|O_2|!\bar{a}_{n+1}\langle N\rangle|c(y)).$$

Because $c \notin \tilde{m}$, one has

$$\nu\tilde{m}.\nu\widetilde{m'}.(P'|O_1|!\bar{a}_{n+1}\langle M\rangle)|c(y) \approx_{obs} \nu\tilde{n}.\nu\widetilde{n'}.(Q'|O_2|!\bar{a}_{n+1}\langle N\rangle)|c(y)$$

We denote $\nu\tilde{m}.\nu\widetilde{m'}.(\{M/x\}|\{\widetilde{M}/\tilde{x}\})$ with ϕ, and $\nu\tilde{n}.\nu\widetilde{n'}.(\{N/x\}|\{\widetilde{N}/\tilde{x}\})$ with ψ. Now we prove $\phi \approx_s \psi$. Assume $\phi \not\approx_s \psi$, then there muse be at least two terms T_1, T_2 such that

$$(n(T_1) \cup n(T_2)) \cap (bn(\phi) \cup bn(\psi)) = \emptyset, \quad (T_1 = T_2)\phi, \quad (T_1 \neq T_2)\psi.$$

Let $C'[\_]$ be a context as follows

$$\_|a_1(x_1).a_2(x_2).\cdots.a_n(x_n).a_{n+1}(x).\mathsf{if}\ (T_1 = T_2)\ \mathsf{then}\ b$$

So

$$C[\nu\tilde{m}.\nu\widetilde{m'}.(P|O_1|!\bar{a}_{n+1}\langle M\rangle)] \overset{\hat{\tau}}{\Rightarrow} \nu\tilde{m}.\nu\widetilde{m'}.(P'|O_1|!\bar{a}_{n+1}\langle M\rangle|b)\downarrow_b$$

However,

$$C[\nu\tilde{n}.(Q|O_1|!\bar{a}_{n+1}\langle N\rangle)] \overset{\hat{\tau}}{\Rightarrow} Q''\downarrow_b$$

which contradicts to

$$\nu\tilde{m}.\nu\widetilde{m'}.(P'|O_1|!\bar{a}_{n+1}\langle M\rangle)|c(y) \approx_{obs} \nu\tilde{n}.\nu\widetilde{n'}.(Q'|O_2|!\bar{a}_{n+1}\langle N\rangle)|c(y)$$

So there exists Q', N that

$$\nu\tilde{n}.(Q|\{\widetilde{N}/\tilde{x}\}) \xrightarrow{\bar{a}x} \nu\tilde{n}.\nu\widetilde{n'}.(Q'|\{N/x\}|\{\widetilde{N}/\tilde{x}\})$$

and

$$(\nu\tilde{m}.(P'|\{\widetilde{M}/\tilde{x}\}|\{M/x\}), \nu\tilde{n}.(Q'|\{\widetilde{N}/\tilde{x}\}\,|\,\{N/x\})) \in \mathcal{R}$$

3. if $\alpha = aM$ for some a, M, then $\phi_1 \vdash M$, and there exists some P' such that

$$\nu\tilde{m}.(P|\{\widetilde{M}/\tilde{x}\}) \xrightarrow{aM} \nu\tilde{m}.(P'|\{\widetilde{M}/\tilde{x}\})$$

Since $\phi_1 \vdash M$, by Proposition 3 there must exist some term ζ such that $n(\zeta) \cap (\tilde{m} \cup \tilde{n}) = \emptyset$ and $\zeta\{\widetilde{M}/\tilde{x}\} = M$. Assume $C[\_]$ be

$$\_|a_1(x_1).a_2(x_2).\cdots.a_n(x_n).\bar{a}\langle\zeta\rangle.(\bar{b}|b|c(y))$$

where b, c are fresh.

$$\begin{aligned} C[\nu\tilde{m}.(P|O_1)] &\overset{\tau}{\Rightarrow} \nu\tilde{m}.(P'|O_1|\bar{a}\langle M\rangle.(\bar{b}|b|c(y))) \\ &\xrightarrow{\tau} \nu\tilde{m}.(P'|O_1|(\bar{b}|b|c(y))) \\ &\xrightarrow{\tau} \nu\tilde{m}.(P'|O_1|c(y)) \end{aligned}$$

So

$$\begin{aligned} C[\nu\tilde{n}.(Q|O_2)] &\overset{\hat{\tau}}{\Rightarrow} \nu\tilde{n}.(Q''|O_2|\bar{a}\langle N\rangle.(\bar{b}|b|c(y))) \\ &\overset{\hat{\tau}}{\Rightarrow} Q''' \approx_{obs} \nu\tilde{m}.(P'|O_2|c(y)) \end{aligned}$$

where $N = \zeta\{\widetilde{N}/\tilde{x}\}$. We have

$$N \asymp_{\{\phi_1,\phi_2\}} M = \zeta\{\widetilde{M}/\tilde{x}\}$$

This reduction can be rearranged

$$\begin{aligned} C[Q] &\overset{\hat{\tau}}{\Rightarrow} \nu\tilde{n}.(Q|O_2|\bar{a}\langle N\rangle.(\bar{b}|b|c(y))) \\ &\overset{\hat{\tau}}{\Rightarrow} \nu\tilde{n}.(Q'|O_2|(\bar{b}|b|c(y))) \\ &\overset{\hat{\tau}}{\Rightarrow} Q''' \approx_{obs} \nu\tilde{m}.(P'|O_1|c(y)) \end{aligned}$$

where $\nu\tilde{n}.(Q|O_2) \xrightarrow{aN} \nu\tilde{n}.(Q'|O_2)$. One can get

$$\nu\tilde{n}.(Q'|O_2|c(y)) \overset{\hat{\tau}}{\Rightarrow} Q''' \approx_{obs} \nu\tilde{m}.(P'|O_1|c(y))$$

Similarly,

$$\nu\tilde{m}.(P'|O_1|c(y)) \overset{\hat{\tau}}{\Rightarrow} P''' \approx_{obs} \nu\tilde{n}.(Q'|O_2|c(y))$$

By Lemma 13, we have $\nu\tilde{m}.(P'|O_1|c(y)) \approx_{obs} \nu\tilde{n}.(Q'|O_2|c(y))$. Because $c \notin \tilde{m}$, hence $\nu\tilde{m}.(P'|O_1)|c(y) \approx_{obs} \nu\tilde{n}.(Q'|O_2)|c(y)$. There exists Q', N that

$$\nu\tilde{n}.(Q|\{\widetilde{N}/\tilde{x}\}) \xrightarrow{aN} \nu\tilde{n}.(Q'|\{\widetilde{N}/\tilde{x}\})$$

$M \asymp_{\{\phi_1,\phi_2\}} N$, and $(\nu\tilde{m}.(P'|\{\widetilde{M}/\tilde{x}\}, \nu\tilde{n}.(Q'|\{\widetilde{N}/\tilde{x}\}) \in \mathcal{R}$.

We can conclude that $\mathcal{R}$ is a *labeled bisimulation*. Let $\tilde{m} = \emptyset,\ \tilde{n} = \emptyset$ and $\widetilde{M} = \emptyset,\ \widetilde{N} = \emptyset$, if $P|c(y) \approx_{obs} Q|c(y)$ which can be deduced from $P \approx_{obs} Q$, then $P \approx_L Q$. $\square$

References

[1] M. Abadi, B. Blanchet, C. Fournet, Just fast keying in the pi calculus, ACM Transactions on Information and System Security 10 (3) (2007).
[2] M. Abadi, V. Cortier, Deciding knowledge in security protocols under equational theories, Theoretical Computer Science 367 (1–2) (2006) 2–32.
[3] M. Abadi, C. Fournet, Mobile values, new names, and secure communication, in: Proceedings of the 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, ACM, 2001, pp. 104–115.
[4] M. Abadi, A. Gordon, A calculus for cryptographic protocols: the spi calculus, in: Proceedings of 4th ACM Conference on Computer and Communications Security, 1997, pp. 36–47.
[5] M. Abadi, A. Gordon, A calculus for cryptographic protocols: the spi calculus, Information and Computation 148 (1) (1999) 1–70.
[6] M. Backes, M. Maffei, D. Unruh, Zero-knowledge in the applied pi-calculus and automated verification of the direct anonymous attestation protocol, in: IEEE Symposium on Security and Privacy, 2008, pp. 202–215.
[7] J. Baeten, W. Weijland, Process Algebra, Cambridge University Press, New York, USA, 1991.
[8] J. Bergstra, J. Klop, Process algebra for synchronous communication, Information and Control 60 (1984) 109–137.
[9] B. Blanchet, An efficient cryptographic protocol verifier based on prolog rules, in: Proceedings of the 14th IEEE Computer Security Foundations Workshop, IEEE Computer Society, 2001, pp. 82–96.
[10] B. Blanchet, M. Abadi, C. Fournet, Automated verification of selected equivalences for security protocols, in: Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science (LICS'04), IEEE Computer Society, 2005, pp. 331–340.
[11] L. Cardelli, A. Gordon, Mobile ambients, Theoretical Computer Science 240 (1) (2000) 177–213.
[12] D. Chaum, Zero-knowledge undeniable signatures, in: Advances in Cryptology – EUROCRYPT, vol. 473, 1990, pp. 458–464.
[13] R. De Nicola, U. Montanari, F. Vaandrager, Back and forth bisimulations, in: CONCUR'90 Theories of Concurrency: Unification and Extension, 1990, pp. 152–165.
[14] S. Delaune, S. Kremer, M. Ryan, Symbolic bisimulation for the applied pi calculus, Journal of Computer Security 18 (2) (2010) 317–377.
[15] S. Duan, Certificateless undeniable signature scheme, Information Sciences 178 (3) (2008) 742–755.
[16] U. Feige, J. Kilian, Zero knowledge and the chromatic number, in: Proceedings Eleventh Annual IEEE Conference on Computational Complexity, 1996, pp. 278–287.
[17] C. Fournet, M. Abadi, Hiding names: private authentication in the applied pi calculus, in: Mext-NSF-JSPS International Symposium (ISSS'02) on Software Security–Theories and Systems, LNCS, vol. 2609, Springer, 2002, pp. 317–338.
[18] Y. Fu, Variations on mobile processes, Theoretical Computer Science 221 (1–2) (1999) 327–368.
[19] Y. Fu, Bisimulation congruence of chi calculus, Information and Computation 184 (1) (2003) 201–226.
[20] Y. Fu, Fair ambients, Acta Informatica 43 (8) (2007) 535–594.
[21] O. Goldreich, Foundations of Cryptography, Cambridge University Press, 2001.
[22] O. Goldreich, S. Micali, A. Wigderson, Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems, Journal of the ACM (JACM) 38 (3) (1991) 728.
[23] S. Goldwasser, S. Micali, C. Rackoff, The knowledge complexity of interactive proof-systems, in: Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing, ACM, 1985, p. 304.
[24] J. Goubault-Larrecq, C. Palamidessi, A. Troina, A probabilistic applied pi-calculus, Lecture Notes in Computer Science 4807 (2007) 175.
[25] C. Hoare, Communicating sequential processes, Communications of the ACM 21 (8) (1978) 666–677.
[26] S. Kremer, M. Ryan, Analysis of an electronic voting protocol in the applied pi-calculus, in: Proceedings of the 14th European Symposium on Programming (ESOP05), LNCS, vol. 3444, Springer, 2005, pp. 186–200.
[27] Y.E. Lien, A note on transition systems, Information Sciences 10 (4) (1976) 347–362.
[28] G. Lowe, Breaking and fixing the Needham-Schroeder public-key protocol using FDR, Software – Concepts and Tools 17 (3) (1996) 93–102.
[29] M. Merro, M. Hennessy, Bisimulation congruences in safe ambients, in: Proceedings of the 29th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, ACM, 2002, pp. 71–80.
[30] M. Merro, F. Nardelli, Behavioral theory for mobile ambients, Journal of the ACM (JACM) 52 (6) (2005) 961–1023.
[31] R. Milner, A Calculus of Communicating Systems, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1982.
[32] R. Milner, Communicating and Mobile Systems: The π-Calculus, Cambridge University Press, 1999.
[33] D. Sangiorgi, Expressing Mobility in Process Algebras: First-Order and Higher-Order Paradigms, Ph.D. Thesis, University of Edinburgh, 1993.
[34] A. Yasinsac, J. Childs, Formal analysis of modern security protocols, Information Sciences 171 (1–3) (2005) 189–211.