

# On Decidability of the Bisimilarity on Higher-order Processes with Parameterization

Xu Xian (徐贤)

(East China University of Science and Technology)

&

Zhang Wenbo (张文博)

(Shanghai Ocean University)

EXPRESS/SOS 2021

August 23rd, 2021

What is this work about?

The decidability of parameterized higher-order processes.

What is this work about?

The decidability of parameterized higher-order processes.

higher-order = process-passing (no name-passing)

parameterized = a way to promote expressiveness [Lanese et al., 2010]

What is parameterization?

It is composed of **abstraction** and **application** (pursuant to those in lambda-calculus).

What is parameterization?

It is composed of **abstraction** and **application** (pursuant to those in lambda-calculus).

**Two kinds of parameterization extending  $\Pi^m$**

- $\Pi^m$ :  $0 \mid X \mid u(X).P \mid \bar{u}P'.P \mid P \mid P'$

What is parameterization?

It is composed of **abstraction** and **application** (pursuant to those in lambda-calculus).

**Two kinds of parameterization extending  $\Pi^m$**

- $\Pi^m$ :  $0 \mid X \mid u(X).P \mid \bar{u}P'.P \mid P \mid P'$

- Parameterization on **processes**:

abstraction:  $\langle X \rangle P$      ( $X$  is a **process variable**)      $(\lambda X.P)$

application:  $P\langle Q \rangle$

## What is parameterization?

It is composed of **abstraction** and **application** (pursuant to those in lambda-calculus).

### Two kinds of parameterization extending $\Pi^m$

- $\Pi^m$ :  $0 \mid X \mid u(X).P \mid \bar{u}P'.P \mid P \mid P'$

- Parameterization on **processes**:

abstraction:  $\langle X \rangle P$  ( $X$  is a **process variable**)  $(\lambda X.P)$

application:  $P\langle Q \rangle$

- Parameterization on **names**:

abstraction:  $\langle x \rangle P$  ( $x$  is a **name variable**)

application:  $P\langle u \rangle$

## What is parameterization?

It is composed of **abstraction** and **application** (pursuant to those in lambda-calculus).

### Two kinds of parameterization extending $\Pi^m$

- $\Pi^m$ :  $0 \mid X \mid u(X).P \mid \bar{u}P'.P \mid P \mid P'$
- Parameterization on **processes**:
  - abstraction:  $\langle X \rangle P$  ( $X$  is a **process variable**) ( $\lambda X.P$ )
  - application:  $P\langle Q \rangle$
- Parameterization on **names**:
  - abstraction:  $\langle x \rangle P$  ( $x$  is a **name variable**)
  - application:  $P\langle u \rangle$
- $\Pi^{mp}$ :  $\Pi$  extended with both kinds of parameterization



## Main results

- In  $\Pi^{\text{mp}}$ , the strong bisimilarity is decidable.
- For the strong bisimilarity, an axiomatization and a bisimilarity checking algorithm.

## Organization

- DECIDABILITY OF THE STRONG BISIMILARITY IN  $\Pi^{\text{mp}}$
- AXIOMATIZATION
- BISIMILARITY CHECKING ALGORITHM

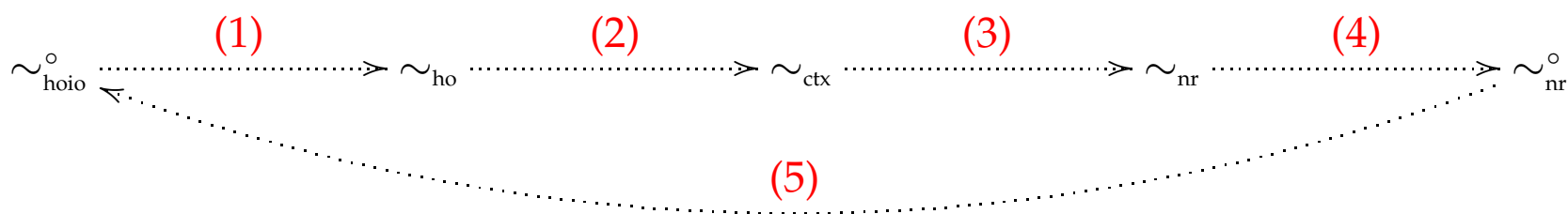
# 1 Decidability of the strong bisimilarity in $\Pi^{\text{mp}}$

## The strong bisimilarities

1. Strong HO-IO bisimilarity  $\sim_{\text{hoio}}^{\circ}$  .
2. Strong HO bisimilarity  $\sim_{\text{ho}}$  .
3. Strong context bisimilarity  $\sim_{\text{ctx}}$  .
4. Strong normal bisimilarity  $\sim_{\text{nr}}$  .
5. Open strong normal bisimilarity  $\sim_{\text{nr}}^{\circ}$  .

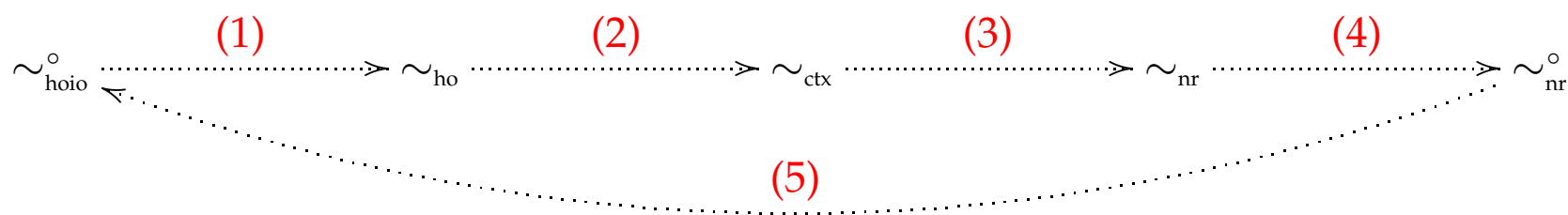
## Main results

1.  $\sim_{\text{hoio}}^{\circ}$  is decidable.
2. All the strong bisimilarities coincide.



## Main results

1.  $\sim_{\text{hoio}}^{\circ}$  is decidable.
2. All the strong bisimilarities coincide.



- ▶ Similar results for  $\Pi^m$  are first given by Lanese et al. [Lanese et al, 2011].

In this work, we adapt the approach of that work to accommodate parameterization.

**Definition 1 (Strong HO-IO bisimilarity).** A symmetric binary relation  $\mathcal{R}$  over  $\Pi^{\text{mp}}$  terms is a strong HO-IO bisimulation, if whenever  $P \mathcal{R} Q$  the following properties hold.

1. If  $P$  is a non-abstraction, then so is  $Q$ .
2. If  $P$  is a process-abstraction  $\langle Y \rangle A$ , then  $Q$  is a process-abstraction  $\langle Y \rangle B$ , and  $A \mathcal{R} B$ .
3. If  $P$  is a name-abstraction  $\langle y \rangle A$ , then  $Q$  is a name-abstraction  $\langle y \rangle B$ , and  $A \mathcal{R} B$ .
4. If  $P \xrightarrow{\bar{a}A} P'$ , then  $Q \xrightarrow{\bar{a}B} Q'$  with  $A \mathcal{R} B$  and  $P' \mathcal{R} Q'$ .
5. If  $P \xrightarrow{a(X)} P'$ , then  $Q \xrightarrow{a(X)} Q'$  and  $P' \mathcal{R} Q'$ .
6. If  $P \equiv X | P'$ , then  $Q \equiv X | Q'$  and  $P' \mathcal{R} Q'$ .
7. If  $P \equiv X \langle A \rangle | P'$ , then  $Q \equiv X \langle B \rangle | Q'$  with  $A \mathcal{R} B$  and  $P' \mathcal{R} Q'$ .
8. If  $P \equiv X \langle d \rangle | P'$ , then  $Q \equiv X \langle d \rangle | Q'$  and  $P' \mathcal{R} Q'$ .

The strong HO-IO bisimilarity,  $\sim_{\text{hoio}}^\circ$ , is the largest strong HO-IO bisimulation.

**Definition 2 (Depth of a term).** We define  $\text{depth}(P)$  as follows.

$P$	$\text{depth}(P)$	
0	0	
$X$	1	
$m(X).P_1$	$\text{depth}(P_1) + 1$	
$\overline{m}(P_1)$	$\text{depth}(P_1) + 1$	
$P_1   P_2$	$\text{depth}(P_1) + \text{depth}(P_2)$	
$\langle X \rangle P_1$	$\text{depth}(P_1) + 1$	
$X \langle P_1 \rangle$	$\text{depth}(P_1) + 1$	
$P_1 \langle P_2 \rangle$	$\text{depth}(P_3 \{P_2/Y\})$	where $P_1$ is $\langle Y \rangle P_3$
$\langle x \rangle P_1$	$\text{depth}(P_1) + 1$	
$X \langle n \rangle$	1	
$P_1 \langle n \rangle$	$\text{depth}(P_3 \{n/y\})$	where $P_1$ is $\langle y \rangle P_3$



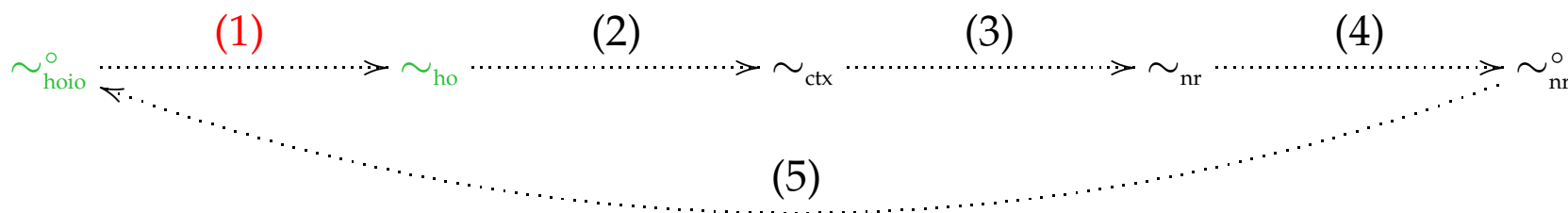
**Lemma 3.**  $\sim_{hoio}^\circ$  is decidable.

*Proof (sketch).* We decide whether  $P \sim_{hoio}^\circ Q$  by **induction on depth( $P$ )**.

1. If  $P$  is an abstraction, then check  $Q$  is also an abstraction accordingly, and continue by induction hypothesis.
2. If  $P$  does an action, then check  $Q$  can simulate, and continue by induction hypothesis.
3. If  $P$  reveals some free variable, say  $X$ , then check  $Q$  also does with  $X$ , and continue by induction hypothesis.

□

# Coincidence between the strong bisimilarities



*Proof (sketch).*

1. **Main difference** between  $\sim_{hoio}^{\circ}$  and  $\sim_{ho}$ :  $\sim_{ho}$  requires closed under substitution and  $\tau$  simulation.
2. Approach: show that the following relation is a strong HO bisimulation.

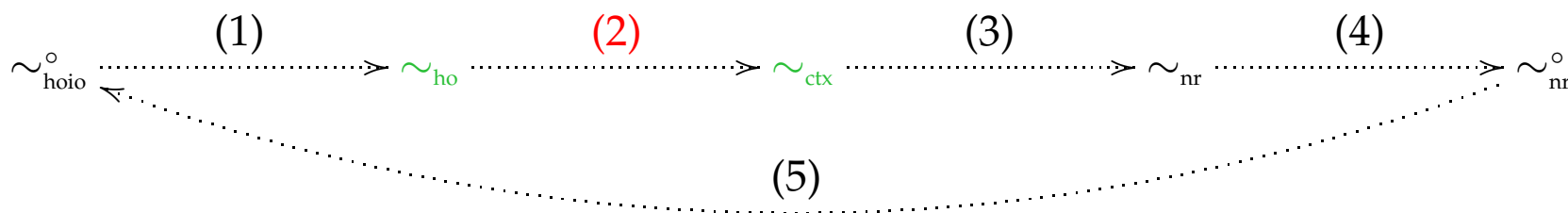
$$\mathcal{R}_1 \stackrel{\text{def}}{=} \{(P, Q) \mid P \sim_{hoio}^{\circ} Q\} \cup \sim_{hoio}^{\circ}$$

3. Technical lemmas:

- (a) If  $P \sim_{hoio}^{\circ} Q$ , then  $P\{R/X\} \sim_{hoio}^{\circ} Q\{R/X\}$ .
- (b) If  $P \sim_{hoio}^{\circ} Q$  and  $P \xrightarrow{\tau} P'$ , then  $Q \xrightarrow{\tau} Q'$  and  $P' \sim_{hoio}^{\circ} Q'$ .

□

# Coincidence between the strong bisimilarities



*Proof (sketch).*

1. **Main difference** between  $\sim_{ho}$  and  $\sim_{ctx}$  : the output simulation.

$\sim_{ho}$  : If  $P \xrightarrow{\bar{a}A} P'$ , then  $Q \xrightarrow{\bar{a}B} Q'$  with  $A \sim_{ho} B$  and  $P' \sim_{ho} Q'$ .

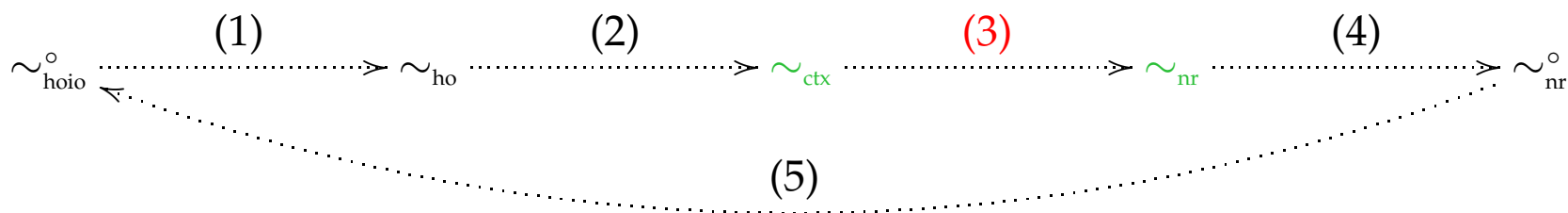
$\sim_{ctx}$  : If  $P \xrightarrow{\bar{a}A} P'$ , then  $Q \xrightarrow{\bar{a}B} Q'$  and for every context  $E$ , it holds that  $E(A) | P' \sim_{ctx} E(B) | Q'$ .

2. Intuition:  $\sim_{ctx}$  requires closure under contexts.

3. Approach: use the congruence property of  $\sim_{ho}$ .

□

# Coincidence between the strong bisimilarities



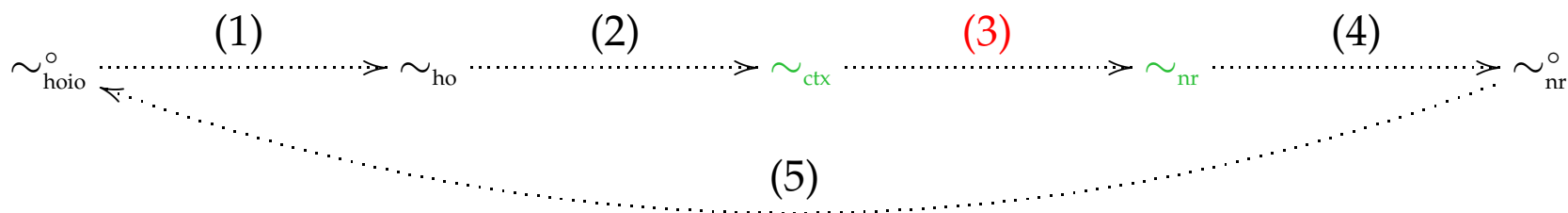
*Proof (sketch).*

The triggers family [Sangiorgi, 1992; Xu, 2013 & 2020]:

$$\textcircled{1} \text{Tr}_m \stackrel{\text{def}}{=} \overline{m}, \quad \textcircled{2} \text{Tr}_m^D \stackrel{\text{def}}{=} \langle Z \rangle \overline{m} Z, \quad \textcircled{3} \text{Tr}_m^{D,d} \stackrel{\text{def}}{=} \langle z \rangle \overline{m} [\langle Z \rangle (Z \langle z \rangle)]$$

□

# Coincidence between the strong bisimilarities



*Proof (sketch).*

The triggers family [Sangiorgi, 1992; Xu, 2013 & 2020]:

$$\textcircled{1} \text{Tr}_m \stackrel{\text{def}}{=} \overline{m}, \quad \textcircled{2} \text{Tr}_m^D \stackrel{\text{def}}{=} \langle Z \rangle \overline{m} Z, \quad \textcircled{3} \text{Tr}_m^{D,d} \stackrel{\text{def}}{=} \langle z \rangle \overline{m} [\langle Z \rangle (Z \langle z \rangle)]$$

1. **Main difference** between  $\sim_{ctx}$  and  $\sim_{nr}$ : closure under substitution and output simulation.

$\sim_{ctx}$ : requires closure for *any processes* for input, and *any contexts* for output.

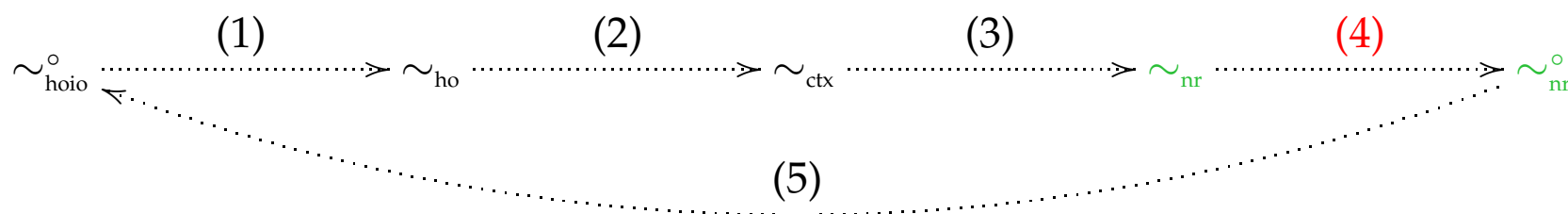
$\sim_{nr}$ : requires closure for *triggers* for input, and *trigger-receiving contexts* for output.

2. Intuition:  $\sim_{nr}$  is a special case of  $\sim_{ctx}$ .

3. Approach: in  $\sim_{ctx}$ , choose special forms of processes or contexts based on triggers.

□

# Coincidence between the strong bisimilarities



*Proof (sketch).*

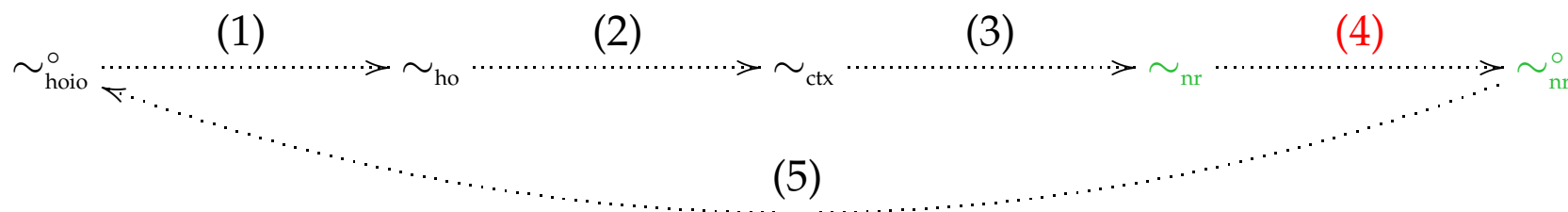
1. **Main difference** between  $\sim_{nr}$  and  $\sim_{nr}^{\circ}$ : closure under substitution and simulation of open terms over free variables.

$\sim_{nr}$ : requires closure under substitution of triggers, while  $\sim_{nr}^{\circ}$  does not.

$\sim_{nr}^{\circ}$ : requires direct matching of free variables, while  $\sim_{nr}$  does not.



# Coincidence between the strong bisimilarities



*Proof (sketch).*

1. **Main difference** between  $\sim_{nr}$  and  $\sim_{nr}^{\circ}$ : closure under substitution and simulation of open terms over free variables.

$\sim_{nr}$ : requires closure under substitution of triggers, while  $\sim_{nr}^{\circ}$  does not.

$\sim_{nr}^{\circ}$ : requires direct matching of free variables, while  $\sim_{nr}$  does not.

2. Intuition: use the closure property of  $\sim_{nr}^{\circ}$  to build a bisimulation.

3. Technical property:  $\sim_{nr}^{\circ}$  is closed under substitution (from its definition). That is,

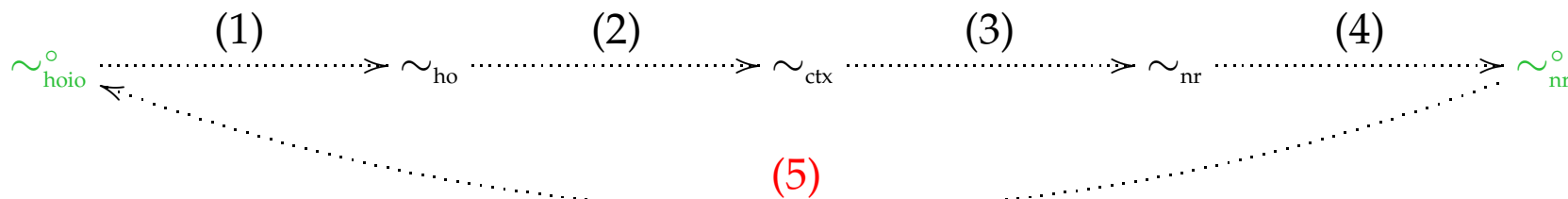
$$P \sim_{nr} Q$$

if and only if

$$P\{\widetilde{\text{Tr}}_{m_1}/\widetilde{X}_1\}\{\widetilde{\text{Tr}}_{m_2}^D/\widetilde{X}_2\}\{\widetilde{\text{Tr}}_{m_3}^{D,d}/\widetilde{X}_3\} \sim_{nr} Q\{\widetilde{\text{Tr}}_{m_1}/\widetilde{X}_1\}\{\widetilde{\text{Tr}}_{m_2}^D/\widetilde{X}_2\}\{\widetilde{\text{Tr}}_{m_3}^{D,d}/\widetilde{X}_3\}$$

□

# Coincidence between the strong bisimilarities



*Proof (sketch).*

1. **Main difference** between  $\sim_{nr}^o$  and  $\sim_{hoio}^o$ : the output simulation and the simulation of an open process with a free variable for process-abstraction. We take a case (the others are similar).

$\sim_{nr}^o$ : If  $P \xrightarrow{\bar{a}A} P'$  in which  $A$  is a process abstraction, then  $Q \xrightarrow{\bar{a}B} Q'$  for process-abstraction  $B$ , and it holds for fresh  $m$  that

$$m(Z).A\langle Z \rangle \mid P' \sim_{nr}^o m(Z).B\langle Z \rangle \mid Q' \quad (\star a)$$

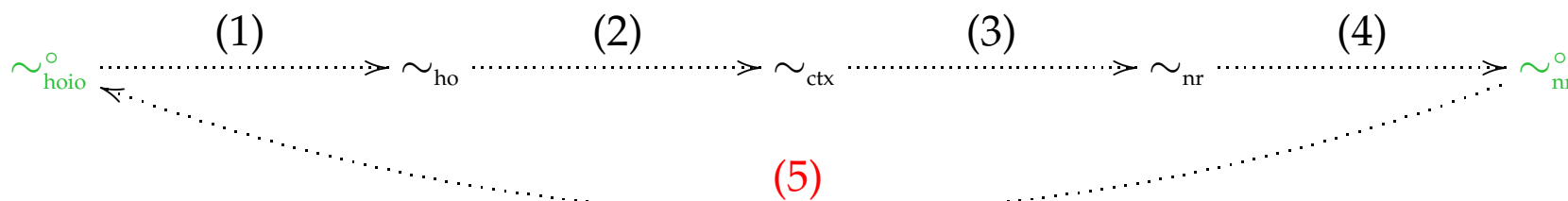
$\sim_{hoio}^o$ : If  $P \xrightarrow{\bar{a}A} P'$ , then  $Q \xrightarrow{\bar{a}B} Q'$  with

$$A \sim_{hoio}^o B \quad \text{and} \quad P' \sim_{hoio}^o Q' \quad (\star b)$$

□



# Coincidence between the strong bisimilarities



*Proof (sketch).*

1. **Main difference** between  $\sim_{\text{nr}}^{\circ}$  and  $\sim_{\text{hoio}}^{\circ}$ : the output simulation and the simulation of an open process with a free variable for process-abstraction. We take a case (the others are similar).

$\sim_{\text{nr}}^{\circ}$ : If  $P \xrightarrow{\bar{a}A} P'$  in which  $A$  is a process abstraction, then  $Q \xrightarrow{\bar{a}B} Q'$  for process-abstraction  $B$ , and it holds for fresh  $m$  that

$$m(Z).A\langle Z \rangle \mid P' \sim_{\text{nr}}^{\circ} m(Z).B\langle Z \rangle \mid Q' \quad (\star a)$$

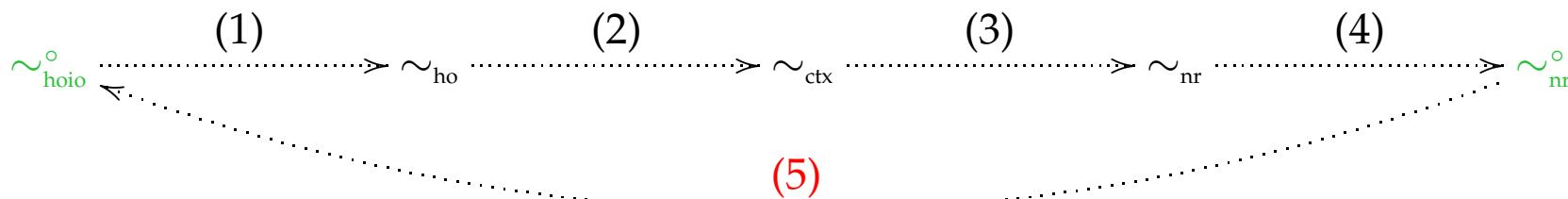
$\sim_{\text{hoio}}^{\circ}$ : If  $P \xrightarrow{\bar{a}A} P'$ , then  $Q \xrightarrow{\bar{a}B} Q'$  with

$$A \sim_{\text{hoio}}^{\circ} B \quad \text{and} \quad P' \sim_{\text{hoio}}^{\circ} Q' \quad (\star b)$$

2. Intuition: use the fresh name to separate the parallel components, so as to extract the two pairs of bisimilar processes.

□

# Coincidence between the strong bisimilarities



*Proof (sketch).*

1. **Main difference** between  $\sim_{\text{nr}}^{\circ}$  and  $\sim_{\text{hoio}}^{\circ}$ : the output simulation and the simulation of an open process with a free variable for process-abstraction. We take a case (the others are similar).

$\sim_{\text{nr}}^{\circ}$ : If  $P \xrightarrow{\bar{a}A} P'$  in which  $A$  is a process abstraction, then  $Q \xrightarrow{\bar{a}B} Q'$  for process-abstraction  $B$ , and it holds for fresh  $m$  that

$$m(Z).A\langle Z \rangle \mid P' \sim_{\text{nr}}^{\circ} m(Z).B\langle Z \rangle \mid Q' \quad (\star a)$$

$\sim_{\text{hoio}}^{\circ}$ : If  $P \xrightarrow{\bar{a}A} P'$ , then  $Q \xrightarrow{\bar{a}B} Q'$  with

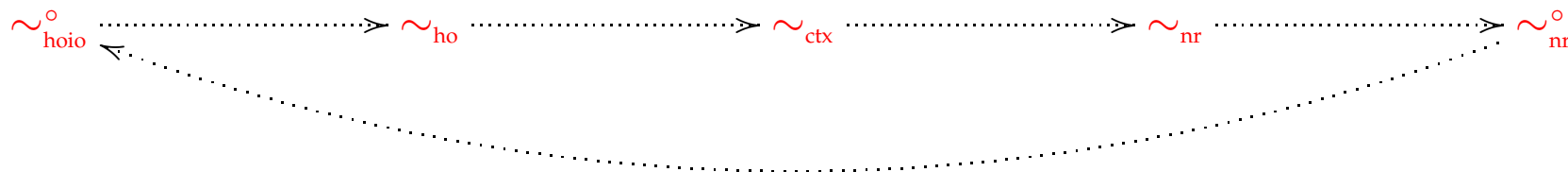
$$A \sim_{\text{hoio}}^{\circ} B \quad \text{and} \quad P' \sim_{\text{hoio}}^{\circ} Q' \quad (\star b)$$

2. Intuition: use the fresh name to separate the parallel components, so as to extract the two pairs of bisimilar processes.
3. Technical lemma: Assume that  $m$  is fresh.

If  $m(Z).A\langle Z \rangle \mid P' \sim_{\text{nr}}^{\circ} m(Z).B\langle Z \rangle \mid Q'$ , then  $A \sim_{\text{nr}}^{\circ} B$  and  $P' \sim_{\text{nr}}^{\circ} Q'$ .

□

# The strong bisimilarities



**Theorem 4.** *All the strong bisimilarities above are **coincident** and **decidable**.*

## 2 Axiomatization

## The axiom system

The axiom system is composed of the following laws.

### 1. Structural congruence.

$$(P \mid Q) \mid R = P \mid (Q \mid R) \qquad P \mid Q = Q \mid P \qquad P \mid 0 = P$$

### 2. Distribution [Lanese et al., 2011].

$$a(X). \left( P \mid \prod_{i=1}^{k-1} a(X).P \right) = \prod_{i=1}^k a(X).P$$

### 3. Parameterization.

$$(\langle X \rangle P) \langle Q \rangle = P \{Q/X\} \qquad (\langle x \rangle P) \langle m \rangle = P \{m/x\}$$

## The axiom system

The axiom system is composed of the following laws.

### 1. Structural congruence.

$$(P \mid Q) \mid R = P \mid (Q \mid R) \qquad P \mid Q = Q \mid P \qquad P \mid 0 = P$$

### 2. Distribution [Lanese et al., 2011].

$$a(X). \left( P \mid \prod_{i=1}^{k-1} a(X).P \right) = \prod_{i=1}^k a(X).P$$

### 3. Parameterization.

$$(\langle X \rangle P) \langle Q \rangle = P \{Q/X\} \qquad (\langle x \rangle P) \langle m \rangle = P \{m/x\}$$

**Normal form**  $P$  is in *normal form* if it cannot be rewritten by the laws for Distribution and Parameterization (left to right). Any  $P$  has a *unique normal form up to*  $\equiv$ , denoted as  $\text{nf}(P)$ .

## Completeness of the axiom system

**Lemma 5 (Completeness).** For any  $P, Q$ , if  $P \sim Q$  then  $P = Q$ , more precisely,  $\text{nf}(P) = \text{nf}(Q)$ .

*Proof (sketch).* Induction on  $\text{depth}(P)$ .

### 1. Key lemmas:

(a) About normal forms:  $P \sim \text{nf}(P)$ .

(b) About input prefix:

If  $a(X).P \sim Q \mid Q'$  ( $Q, Q' \neq 0$ ), then  $a(X).P \sim \prod_{i=1}^k a(X).A$  ( $k > 1$ ) with  $a(X).A$  is in nf.

### 2. A notion of *prime* processes [Lanese et al., 2011; Milner et al., 1993].

(a)  $P$  is *prime* if  $P \neq 0$  and  $P \sim P_1 \mid P_2$  implies  $P_1 \sim 0$  or  $P_2 \sim 0$ .

(b) If  $P \sim \prod_{i=1}^n P_i$  where each  $P_i$  is prime,  $\prod_{i=1}^n P_i$  is called a *prime decomposition* of  $P$ .

### 3. Properties about prime processes.

(a) If  $P$  is a prefixed process in normal form, then  $P$  is prime.

(b) Unique prime decomposition.

If  $P$  has two prime decompositions  $P \sim \prod_{i=1}^n P_i$  and  $P \sim \prod_{j=1}^m Q_j$ , then  $n = m$  and there is a permutation  $\sigma : \{1, 2, \dots, n\} \rightarrow \{1, 2, \dots, n\}$ , such that  $P_i \sim Q_{\sigma(i)}$  for each  $i \in \{1, 2, \dots, n\}$ .

□

### **3 Bisimilarity checking algorithm**



## Core of the algorithm

**Input** Two processes  $P$  and  $Q$ .

### Procedure

1. (Data structure) Representing processes  $P$  and  $Q$  as **trees**.
2. (Transformation) Normalizing the trees.
3. (Comparison) Comparing the trees up-to syntax.

**Output** Whether  $P = Q$  or not.

## Core of the algorithm

**Input** Two processes  $P$  and  $Q$ .

### Procedure

1. (Data structure) Representing processes  $P$  and  $Q$  as **trees**.
2. (Transformation) Normalizing the trees.
  - (a) Rewrite each process by the application rules, if possible.
  - (b) Normalize each process over parallel composition.
  - (c) Apply the distribution law, if possible.
3. (Comparison) Comparing the trees up-to syntax.

**Output** Whether  $P = Q$  or not.

## Tree representation

**Definition 6 (Tree representation).** The tree representation of  $P$  is defined as follows.

- $\text{Tree}(0) = 0[ ]$
- $\text{Tree}(X) = \text{db}(X)[ ]$
- $\text{Tree}(a(X).P) = a[\text{Tree}(P)]$
- $\text{Tree}(\bar{a}(Q)) = a^O[\text{Tree}(Q)]$
- $\text{Tree}(x(X).P) = \text{db}(x)[\text{Tree}(P)]$
- $\text{Tree}(\bar{x}(Q)) = \text{db}(x)^O[\text{Tree}(Q)]$
- $\text{Tree}(\Pi_{i=1}^n P_i) = \Pi_{i=1}^n [\text{Tree}(P_1), \dots, \text{Tree}(P_n)]$
- $\text{Tree}(\langle X \rangle P) = \text{abs}[\text{Tree}(P)]$
- $\text{Tree}(\langle P \rangle Q) = \text{app}[\text{Tree}(P), \text{Tree}(Q)]$
- $\text{Tree}(\langle x \rangle P) = \text{abs}[\text{Tree}(P)]$
- $\text{Tree}(\langle P \rangle n) = \text{app}[\text{Tree}(P), n[ ]]$

**db** assigns uniform indices to variables [De Bruijn, 1972; Lanese et al., 2011].

# Tree normalization

---

Application:  $\text{App}(n_{raw}, ind, n_{eval})$

---

**Input:** Tree nodes  $n_{raw}, n_{eval}$ , an integer  $ind$ .

**Output:** Tree nodes for which application is done.

```

1 if ( $n_{raw}.type == 'var'$  or  $n_{raw}.type == 'inp'$ ) and  $n_{raw}.label == ind$  then
2   |    $n_{raw} = n_{eval}$ ;
3 end if

4 if  $n_{raw}.type == 'out'$  and  $n_{raw}.label == ind^O$  then
5   |    $n_{raw} = n_{eval}$  ;
6   |    $n_{raw}.label = (n_{raw}.label)^O$ ;
7 end if

8 if  $n_{raw}.type == 'inp'$  or  $n_{raw}.type == 'abs'$  then
9   |    $ind = ind + 1$ ;
10 end if

11 for  $i = 1$  to  $n.numChildren$  do
12   |    $\text{App}(n_{raw}.children[i], ind, n_{eval})$ ;
13 end for

```

---

## Tree normalization

---

### Normalization Step 1: NS1( $n$ )

---

**Input:** A tree node  $n$ .

**Output:** Tree node after normalization step 1.

```
1 for  $i = 1$  to  $n.numChildren$  do
2   |   NS1( $n.children[i]$ );
3 end for
4 if  $n.type == 'app'$  then
5   |   if  $n.children[1].type == 'abs'$  then
6     |   App( $n.children[1].children[1]$ , 1,  $n.children[2]$ );
7     |   end if
8 end if
```

---

# Tree normalization

---

## Normalization Step 2: NS2( $n$ )

---

**Input:** A tree node  $n$ .

**Output:** Tree node after normalization step 2.

```

1 for  $i = 1$  to  $n.numChildren$  do NS2( $n.children[i]$ );
2 if  $n.type == 'par'$  then
3    $j = 1$ ;
4   for  $i = 1$  to  $n.numChildren$  do
5     if  $n.children[i].type \neq 'zero'$  then
6        $n.children[j] = n.children[i]$ ;
7        $j = j + 1$ ;
8     end if
9   end for
10   $n.numChildren = j - 1$ ;
11  if  $n.numChildren == 0$  then
12     $n.type = 'zero'$ ;
13  else if  $n.numChildren == 1$  then
14     $n = n.children[1]$ ;
15  end if
16 end if
17 sortChildren( $n$ );

```

---

# Tree normalization

---

## Normalization Step 3: NS3( $n$ )

---

**Input:** A tree node  $n$ .

**Output:** Tree node after normalization step 3.

```

1 for  $i = 1$  to  $n.numChildren$  do NS3( $n.children[i]$ );
2 if  $n.type == 'inp'$  then
3    $p = n.children[1]$ ;
4   if  $p.type == 'par'$  then
5      $smallIndex = -1$ ;  $small = null$ ;  $big = null$ ;
6      $pc1 = p.children[1]$ ;  $pc2 = p.children[p.numChildren]$ ;
7     if  $pc1.type == 'inp'$  and  $pc1.label == n.label$  and  $pc1.children[1] == pc2$  then
8        $small = pc2$ ;  $big = pc1$ ;  $smallIndex = p.numChildren$ ;
9     else if  $pc2.type == 'inp'$  and  $pc2.label == n.label$  and  $pc2.children[1] == pc1$  then
10       $small = pc1$ ;  $big = pc2$ ;  $smallIndex = 1$ ;
11    else return;
12    for  $i = 2$  to  $n.numChildren - 1$  do
13      if  $p.children[i] \neq big$  then return;
14    end for
15     $p.children[smallIndex] = big$ ;
16     $n = n.children[1]$ ;
17  end if
18 end if

```

## Complexity of the algorithm

Let  $n$  be the total number of nodes in the tree representations of the processes under consideration.

The bisimilarity checking algorithm has the following complexity.

**Time**  $O(n \log(n))$ .

**Space**  $O(n)$ .



## 4 Conclusion

1. In presence of parameterization, the strong bisimilarity is still decidable for  $\Pi^{\text{mp}}$ .
2. A complete axiom system for the strong bisimilarity is provided.
3. An algorithm for the bisimilarity checking is designed.

### Future work

1. Expanding the model to allow more modelling capability, e.g., locations or probability, while maintaining the decidability result.
2. Considering the decidability of the weak bisimilarity.

Thank you.