On Bisimulation Theory in Linear Higher-Order π-Calculus

Xian Xu[⋆]

BASICS
Department of Computer Science and Technology
Shanghai Jiao Tong University
800 Dong Chuan Road (200240), Shanghai, P.R. China, Mailbox A0403391
Email: xuxian@sjtu.edu.cn

Abstract. Higher-order process calculi have been receiving much attention in recent years for its significance in both theory and practice. Work on bisimulations has never ceased evolving, typically represented by Thomsen and Sangiorgi for their work on bisimulation theory and encoding to and from first-order process calculi. Fu puts forth linear higher-order π-calculus, and makes improvement to previous work on bisimulation and builds a sound and complete equation system by exploiting linearity of processes, which takes resource sensitiveness into account. In this paper, we establish some recent result on bisimulation theory in linear higher-order π-calculus. By exploiting the properties of linear higher-order processes, we work out two simpler variants than local bisimulation, which is an intuitive observational equivalence. We prove that they both coincide with local bisimilarity. The first variant, called local linear bisimulation, simplifies the matching of higher-order input and higher-order output based on the feature of checking equivalence with some special processes (in input or output) instead of general ones. The second variant, called local linear variant bisimulation, rewrites the first-order bound output clause in local bisimulation in some more suitable form for some application on it, by harnessing the congruence properties. We also mention some future work in the conclusion.

Key words: Bisimulation, Linear, Higher-order, π-Calculus, Process calculi

1 Introduction

Compared with first-order calculi [1] [2] [3], higher-order process calculi excel in that they provide the ability to communicate an integral program, rather than a simple value or a reference to a program. The importance are double-fold. Firstly, theoretically higher-order features offer a broader spectrum of communication capability that can possibly compute more conveniently and efficiently, and the (technical) framework of process calculi is widened. Secondly, practically distributed and mobile computing are increasingly expanding in various forms, in which communication involving higher-order elements can be witnessed in nearly anywhere, such as sending a small application through network or packaging and transmitting a script over several computing nodes in the network and then configuring and running in a remote computer. To some extent the study of higher-order process calculi can enable us to better direct the technology development. To our knowledge, up-to-date research on bisimulation in higher-order process calculi mainly includes the following parts.

CHOCS and plain CHOCS. Thomsen studies two kinds of higher-order CCS, which extends CCS by higher-order communication with processes, that is CHOCS [4] [5] and Plain CHOCS [6] [7], with dynamic and static binding of restriction respectively. The higher-order bisimulation (in CHOCS) and applicative higher-order bisimulation (in Plain CHOCS) are examined. They are quite structural but not intuitive in that communicated and continual processes are separated in comparison [8]. The bisimulations use delayed approach, which prohibits internal moves after an observable action, to counter the technical obstacle in equivalence proving.

Higher-order π-calculus. Sangiorgi studies higher-order π-calculus [9] [8] that extends first-order π-calculus [2] [10] with communication of processes. The bisimulation he puts forth, context bisimulation [8], improves that in Plain CHOCS by considering residual and transmitted processes in the meanwhile. However the style is still delayed. Yet it is proven that the early and late versions of context bisimulations coincide, which is not so obvious in the Ambient calculus. Triggered bisimulation, normal bisimulation are proposed to simplify context bisimulation on its heavy usage of universal quantification. And barbed bisimulation [11] is also considered in higher-order paradigm.

Nevertheless, the work mentioned above is not perfect in several points. For example,

[⋆] The author is supported by The National 973 Project (2003CB317005), The National Nature Science Foundation of China (60473006,60573002), and The BoShiDian Research Fund (20010248033).

- Delayed bisimulations. The bisimulations are all delayed versions. That is weak transitions have no trailing silent moves.
- Computational power. The computational power levels that of Turing machines. This leads to the status that no axiom system is available for higher-order process calculi for a long period.

Fu attempts to settle these problems with linear higher-order π-calculus (LHOPi for short) [12], which abstracts from the practical scene that one program can be used only once in one network application on the client side, such as on-line games and video on demand, by demanding that a same process variable shall never appear simultaneously in concurrent positions, typically the parallel composition and higher-order output. Fu shows that linearity can effectively downscale the computational power, and entail a sound and complete axiom system as well as an algorithm checking the equivalence of two processes. Besides, the bisimulation he uses, called local bisimulation, is a general one, not delayed. The technical difficulty is solved by the bisimulation lemma [12]. And most of the results in [12] hold in general higher-order π-calculus.

Contribution

In this paper, we put forward some recent work on linear process calculi. Our endeavor in this paper is focused on the bisimulation theory of LHOPi. We seek for a simpler expression of local bisimulation. This is achieved by making full use of the properties of linear processes, that is Abstraction Theorem for higher-order input and Concretion Theorem for higher-order output. These theorems are firstly used by Fu to lay the foundation for an axiom system, and we harness them to simplify local bisimulation, by loosening the requirements of bisimulation. For example, we only demand the receiving of a special process in the comparison of higher-order input rather than a general process. So ($\mathcal{R}$ is a certain bisimulation, and we omit some relatively unimportant details).

$$\text{If } P\xrightarrow{a(A)}P', \text{ then } Q\overset{a(A)}{\Longrightarrow}Q' \text{ for some } Q' \text{ and } P'\mathcal{R}Q'$$

is simplified to

$$\text{If } P\xrightarrow{a(\mathbf{c})}P', \text{ where } c \text{ is a fresh name, then } Q\overset{a(\mathbf{c})}{\Longrightarrow}Q' \text{ for some } Q', \text{ and } P' \mathcal{R}\, Q'$$

(where $\mathbf{c}$ abbreviates $c(x).0$ and c is fresh)

And in higher-order output, the demand that the transmitted process should be considered in an arbitrary environment with the residual process is weakened to the holding of the similar property on a special environment. Such special processes can be infinite, but we merely need one of them for checking bisimulation. Thus we usually say '*a*' instead of '*every/any*' in a simulating clause. So

$$\text{If } P\xrightarrow{(\widetilde{x})\overline{a}A}P' \text{ then } Q\overset{(\widetilde{y})\overline{a}B}{\Longrightarrow}Q' \text{ for some } \widetilde{y}, B, Q', \text{ and for every process } E[X] \text{ it holds that } (\widetilde{x})(E[A]|P')\ \mathcal{R}\ (\widetilde{y})(E[B]|Q')$$

is simplified to

$$\text{If } P\xrightarrow{(\widetilde{x})\overline{a}A}P', \text{ then } Q\overset{(\widetilde{y})\overline{a}B}{\Longrightarrow}Q' \text{ for some } \widetilde{y}, B, Q'. \text{ And for a process } E[X] \equiv \overline{c}.(X+d), \text{ where } c, d \text{ are fresh names, it holds that } (\widetilde{x})(E[A]|P')\ \mathcal{R}\ (\widetilde{y})(E[B]|Q')$$

The obtained bisimulation after the modification above is called local linear bisimulation ($\approx_{ll}$ for the corresponding bisimilarity). We show that Abstraction Theorem and Concretion Theorem also hold on local linear bisimilarity, which leads to the coincidence between local bisimilarity and local linear bisimilarity. The proofs are non-trivial.

Then we examine the first-order bound output, whose corresponding clause in local bisimulation is

$$\text{If } P\xrightarrow{\overline{a}(x)}P', \text{ then } Q\overset{\overline{a}(x)}{\Longrightarrow}Q' \text{ for some } Q', \text{ and for every process } O,\ (x)(O|P')\ \mathcal{R}\ (x)(O|Q')$$

We show in an informal way that this cannot be simplified. However, to gain a handy manipulation, we define a variant on first-order bound output clause by using the congruence properties, and obtained the following clause.

$$\text{If } P\xrightarrow{\overline{a}(x)}P', \text{ then } Q\overset{\overline{a}(x)}{\Longrightarrow}Q' \text{ for some } Q', \text{ and for all processes } O_1 \text{ and } O_2 \text{ such that } O_1\ \mathcal{R}\ O_2, (x)(O_1|P')\ \mathcal{R}\ (x)(O_2|Q')$$

The bisimulation gained after this adjustment is called local linear variant bisimulation ($\approx_{ll}^{v}$ for the corresponding bisimilarity). We show that local linear variant bisimilarity coincides with local linear bisimilarity.
Organization. The paper is organized as follows. Section 2 gives an introduction to linear higher-order π-calculus by Fu [12]. We define the syntax, operational semantics, local bisimulation, and equivalence and congruence properties of local bisimilarity. Critical results, which describe the relationship between the equivalence of the prefixed processes and the equivalence of the continuations and lay foundation for axiom system for linear higher-order π-calculus are also presented. In section 3, we define several variants, local linear bisimulation and local linear variant bisimulation, which simplify yet are coincident to the original bisimulation, that is local bisimulation (in the largest sense). The detailed proofs are given. We also make some discussion on the possibility of simplifying first-order bound output clause in local bisimulation. Section 4 concludes our work before some future work is discussed.

2 Linear higher-order π-calculus

In this section, we give an introduction to LHOPi [12].

2.1 Syntax

Linear higher-order π (LHOPi) processes (or simply processes) are denoted by capital letters $(A, B, \ldots, E, F, \ldots, P, Q, \ldots)$. They are defined by the following abstract grammar. $fv(\cdot)$ denotes free process variables.

$$P := 0 \mid X \mid a(x).P \mid \overline{a}x.P \mid a(X).P \mid \overline{a}Q.P \mid P|Q \mid (x)P \mid [x{=}y]P \mid P{+}Q$$

Most of them have their usual meanings. The linearity is guaranteed by demanding $fv(P) \cap fv(Q) = \emptyset$ in $\overline{a}Q.P$ and $P|Q$.

We have derived prefixes: $\tau.P \triangleq (m)(m(x)|\overline{m}m.P), m$ fresh and $\overline{a}(x).P \triangleq (x)\overline{a}x.P$. The first is silent action and the latter is first-order bound output. Most conventions are standard. $fn(P_1, ..., P_n)$, $bn(P_1, ..., P_n)$, $n(P_1, ..., P_n)$; $fv(P_1, ..., P_n)$, $bv(P_1, ..., P_n)$, $v(P_1, ..., P_n)$ denote free names, bound names, names; free variables, bound variables and variables in processes $P_1, ..., P_n$, respectively. We generally focus on closed processes, which contain no free variables. Sometimes we use "(closed)" before processes to indicate this. "process expressions" indicate general processes. Name substitution on processes $P\{y/x\}$ and higher-order (process or variable) substitution on processes $P\{Q/X\}$ are defined structurally on processes as usual. We demand that higher-order substitutions do not produce an non-linear processes. σ and Σ denote name substitution and higher-order substitution respectively. $\widetilde{x}$ denotes a finite set of names, that is $x_1, x_2, ..., x_n$. $E[X_1, X_2, ..., X_n]$ stands for the process with (at most) a series of process variables occurring in it. We write $E[A_1, A_2, ..., A_n]$ for $E[X_1, X_2, ..., X_n]\{A_1/X_1, A_2/X_2, ..., A_n/X_n\}$. We usually omit the process 0. Some abbreviations are: a for $a(x).0$; $\overline{a}$ for $\overline{a}x.0$; τ for $\tau.0$. I_a is defined as $a(X).X$. A fresh name (variable) is a name (variable) that does not occur in the processes under consideration. We assume structural equivalence on processes, like that in [13].

Contexts are processes with holes for processes. It is important that process behavior has some invariance under various contexts. We define three kinds of contexts below. Note context $C[\cdot]$ is different from $E[X]$ in that the latter should not let name capturing occur whereas the former does not take care of this.
Contexts: $[\cdot]$ is a context; If $C[\cdot]$ is a context, then $a(x).C[\cdot]$, $\overline{a}x.C[\cdot]$, $\tau.C[\cdot]$, $a(X).C[\cdot]$, $\overline{a}A.C[\cdot]$, $(x)C[\cdot]$, $P|C[\cdot]$ and $[x{=}y]C[\cdot]$ are contexts.
Full contexts: A context is a full context; If $C[\cdot]$ is a full context, then $\overline{a}[C[\cdot]].P$ and $C[\cdot]{+}P$ are full contexts.
Local contexts: A full context of the form $(\widetilde{x})([\cdot]|O)$. The more usual form of a local context is ($\widetilde{x}, \widetilde{c}$ are all pairwise distinct)$(x_1) \cdots (x_n)(\overline{c_1}x_1| \cdots |\overline{c_n}x_n|[\cdot])$ (or $(\widetilde{x})(\widetilde{\overline{c}x}|[\cdot])$).

2.2 Semantics

For the labelled transition system, we need a concept $cp(P, X)$ of a process variable, indicating the process variables locating at the concurrent positions of X, such as parallel composition and higher-order output. It is due to Fu [12] and routine, so we omit the formal definition. the semantics of LHOPi is given in Figure 1. Symmetric rules are omitted. We use $\alpha, \beta, \lambda, ...$ for actions.

The transition rules are mostly self-explained. In higher-order input, the received process shall not break the linearity of processes, which is why we demand $fv(A) \cap cp(P, X) = \emptyset$. Two actions α, β are said to be *complementary* if they can form a (first-order or higher-order) communication. $\Longrightarrow$ is the reflexive transitive closure of silent actions, and $\overset{\lambda}{\Longrightarrow}$ is $\Longrightarrow \overset{\lambda}{\longrightarrow} \Longrightarrow$. $\overset{\hat{\lambda}}{\Longrightarrow}$ is $\Longrightarrow$ when λ is τ and $\overset{\lambda}{\Longrightarrow}$ otherwise.

$$\frac{}{a(x).P\xrightarrow{a(y)}P\{y/x\}}\quad \frac{}{\overline{a}x.P\xrightarrow{\overline{a}x}P}\quad \frac{fv(A)\cap cp(P,X)=\emptyset}{a(X).P\xrightarrow{a(A)}P\{A/X\}}\quad \frac{}{\overline{a}A.P\xrightarrow{\overline{a}A}P}\quad \frac{P\xrightarrow{\lambda}P'}{P+Q\xrightarrow{\lambda}P'}\quad \frac{P\xrightarrow{\lambda}P'}{[x=x]P\xrightarrow{\lambda}P'}$$

$$\frac{P\xrightarrow{\lambda}P'}{P|Q\xrightarrow{\lambda}P'|Q}bn(\lambda)\cap fn(Q)=\emptyset\quad \frac{P\xrightarrow{a(x)}P',Q\xrightarrow{\overline{a}x}Q'}{P|Q\xrightarrow{\tau}P'|Q'}\quad \frac{P\xrightarrow{a(x)}P',Q\xrightarrow{\overline{a}(x)}Q'}{P|Q\xrightarrow{\tau}(x)(P'|Q')}\quad \frac{P\xrightarrow{a(A)}P',Q\xrightarrow{(\widetilde{x})\overline{a}[A]}Q'}{P|\xrightarrow{\tau}(\widetilde{x})(P'\mid Q')}$$

$$\frac{P\xrightarrow{\lambda}P'}{(x)P\xrightarrow{\lambda}(x)P'}x\notin n(\lambda)\quad \frac{P\xrightarrow{\overline{a}x}P'}{(x)P\xrightarrow{\overline{a}(x)}P'}x\neq a\quad \frac{P\xrightarrow{(\widetilde{x})\overline{a}[A]}P'}{(y)P\xrightarrow{(y)(\widetilde{x})\overline{a}[A]}P'}y\in fn(A)-\{\widetilde{x},a\}$$

Fig. 1. LTS of LHOPi

The following two lemmas ensure that the LTS preserves linearity of processes. Their proofs are structure or transition induction.

Lemma 1. *Suppose $E[X],F[Y]$ are LHOPi processes with at most process variable X or Y. If $fv(F)\cap cp(E,X)=\emptyset$, then $E[F[Y]]$ is also a LHOPi process.*

Lemma 2. *Suppose P,A are LHOPi processes. It holds that: (i) If $P\xrightarrow{\lambda}P'$, where λ is a first-order action or τ, then P' is also a LHOPi process; (ii) If $P\xrightarrow{a(A)}P'$, then P' is also a LHOPi process; (iii) If $P\xrightarrow{\overline{a}A}P'$, then P',A are LHOPi processes too.*

The following lemmas state the properties of LTS concerning substitutions. Their proofs are basically transition inductions.

Lemma 3. *If P is a LHOPi process and $P\xrightarrow{\lambda}P'$ then $P\sigma\xrightarrow{\lambda\sigma}P'\sigma$.*

Lemma 4. *If P is a LHOPi process and $fv(P)=\{X_1,X_2,...,X_n\}$. And $P_1,P_2,...,P_n$ are LHOPi process. If $P\xrightarrow{\lambda}P'$ then $P\{P_1/X_1,P_2/X_2,...,P_n/X_n\}\xrightarrow{\lambda\{P_1/X_1,P_2/X_2,...,P_n/X_n\}}P'\{P_1/X_1,P_2/X_2,...,P_n/X_n\}$.*

Lemma 5. *If P is a LHOPi process and $fv(P)=\{X_1,X_2,...,X_n\}$. And $b_1,b_2,...,b_n$ are fresh names. If $P\{b_1/X_1,b_2/X_2,...,b_n/X_n\}\xrightarrow{\lambda\{b_1/X_1,b_2/X_2,...,b_n/X_n\}}P'\{b_1/X_1,b_2/X_2,...,b_n/X_n\}$, then $P\xrightarrow{\lambda}P'$.*

2.3 Bisimulation

A binary relation $\mathcal{R}$ on processes is closed under substitution of names if for each substitution σ, $(P\sigma,Q\sigma)\in\mathcal{R}$ whenever $(P,Q)\in\mathcal{R}$. A relation closed under substitution on process variables can be defined similarly. We first give the higher-order structural equivalence [12], which is essentially from Thomsen's applicative higher-order bisimilarity [6].

Definition 1 (Structural equivalence). *A symmetric binary relation $\mathcal{R}$ on processes is a structural bisimulation if it is closed under substitution of names and whenever $P\mathcal{R}Q$ the following holds:*

- *If $P\xrightarrow{\lambda}P'$, where λ is a silent action, first-order input, first-order output, first-order bound output, or higher-order input. Then $Q\xrightarrow{\lambda}Q'$ for some Q', and $P'\mathcal{R}Q'$;*
- *If $P\xrightarrow{(\widetilde{x})\overline{a}A}P'$, then some B,Q' exist s.t. $Q\xrightarrow{(\widetilde{x})\overline{a}B}Q'$, $P'\mathcal{R}Q'$, and $A\mathcal{R}B$.*

Two processes P,Q are structural equivalent, written $P\sim Q$, if there exists a structural bisimulation $\mathcal{R}$ s.t. $P\mathcal{R}Q$.

$\sim$ is both an equivalence and a congruence.

Definition 2 (Local bisimulation). *A symmetric binary relation $\mathcal{R}$ on (closed) processes is a local bisimulation, if it is closed under substitution of names, and whenever $P\mathcal{R}Q$, the following properties hold:*

1. *If $P\xrightarrow{\tau}P'$, then $Q\Longrightarrow Q'$ for some Q' and $P'\mathcal{R}Q'$;*
2. *If $P\xrightarrow{a(x)}P'$, then $Q\overset{a(x)}{\Longrightarrow}Q'$ for some Q' and $P'\mathcal{R}Q'$;*
3. *If $P\xrightarrow{\overline{a}x}P'$, then $Q\overset{\overline{a}x}{\Longrightarrow}Q'$ for some Q' and $P'\mathcal{R}Q'$;*
4. *If $P\xrightarrow{\overline{a}(x)}P'$, then $Q\overset{\overline{a}(x)}{\Longrightarrow}Q'$ for some Q', and for every process O, $(x)(O|P')\ \mathcal{R}\ (x)(O|Q')$;*
5. *If $P\xrightarrow{a(A)}P'$, then $Q\overset{a(A)}{\Longrightarrow}Q'$ for some Q' and $P'\mathcal{R}Q'$;*

6. *If* $P \xrightarrow{(\widetilde{x})\overline{a}A} P'$ *then* $Q \overset{(\widetilde{y})\overline{a}B}{\Longrightarrow} Q'$ *for some* $\widetilde{y}, B, Q'$*, and for every process* $E[X]$ *s.t.* $\widetilde{x}\widetilde{y} \cap fn(E) = \emptyset$ *it holds that* $(\widetilde{x})(E[A]|P') \mathrel{\mathcal{R}} (\widetilde{y})(E[B]|Q')$.

We say P *is local bisimilar to* Q*, written* $P \approx_l Q$ *(*$\approx_l$ *is local bisimilarity), if there exists a local bisimulation* $\mathcal{R}$ *s.t.* $P\mathcal{R}Q$.

The bisimulation is on closed processes, and can be extended to open processes in the following standard way. Clauses $4, 6$ are in a late style, whereas 5 is in an early style. Corresponding early cases for $4, 6$ and late case for 5 can be defined. Moreover, it is proven that the corresponding early (or late) case is equivalent to the late (or early) case [12]. Local bisimilarity is an equivalence and congruence relation (excluding choice operator), and what's more, as showed by Fu, it is an observed bisimulation [12] that is a general bisimulation satisfying the least requirements to be qualified for an observational equivalence and somewhat like barbed bisimulation [14] [11] in that it is closed under contexts, barb preserving and reduction closed. Up-to technique can be defined on local bisimilarity. For example local bisimulation up-to $\sim$ can be defined by replacing $\mathcal{R}$ with $\sim \mathcal{R} \sim$ in the clauses.

The congruence under local contexts is as the following theorem specifies. $\simeq_l$ is built from $\approx_l$ *a la* Milner's standard congruence-construction approach [1].

Theorem 1. *Suppose* $(\widetilde{z})(\widetilde{\overline{c}z}|P) \simeq_l (\widetilde{z})(\widetilde{\overline{c}z}|Q)$ *where* $\widetilde{c}$ *are pairwise distinct, then* $(\widetilde{z})(\widetilde{\overline{c}z}|C[P]) \simeq_l (\widetilde{z})(\widetilde{\overline{c}z}|C[Q])$*, for every full context* $C[\cdot]$ *that has no name collision on* $\widetilde{c}, \widetilde{z}$.

The next three theorems clarify the relationship between the equivalence of prefixed processes and the equivalence of continual processes.

Theorem 2 (Localization). *Suppose* $a \notin \widetilde{c}, x \notin \widetilde{z}$*,* $\widetilde{c}(fresh)$ *are pairwise distinct, so are* $\widetilde{z}$*. Then the following equations are equivalent: (i)* $(\widetilde{z})(\widetilde{\overline{c}z}|\overline{a}(x).P) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|\overline{a}(x).Q)$*; (ii)* $(\widetilde{z})(\widetilde{\overline{c}z}|(x)(\overline{b}x|P)) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|(x)(\overline{b}x|Q))$ *for a fresh name* b*; (iii)* $(\widetilde{z})(\widetilde{\overline{c}z}|(x)(E|P)) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|(x)(E|Q))$ *for every process* E.

Theorem 3 (Abstraction). *Suppose* $x \notin \widetilde{z}$*,* $\widetilde{c}(fresh)$ *are pairwise distinct, so are* $\widetilde{z}$*. Then the following equations are equivalent: (i)* $(\widetilde{z})(\widetilde{\overline{c}z}|a(X).P) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|a(X).Q)$*; (ii)* $(\widetilde{z})(\widetilde{\overline{c}z}|P\{I_b/X\}) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|Q\{I_b/X\})$ *for a fresh name* b*; (iii)* $(\widetilde{z})(\widetilde{\overline{c}z}|P\{b/X\}) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|Q\{b/X\})$ *for a fresh name* b*; (iv)* $(\widetilde{z})(\widetilde{\overline{c}z}|P\{E/X\}) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|Q\{E/X\})$ *for every process* E.

Theorem 4 (Concretion). *Suppose* $x \notin \widetilde{z}$*,* $\widetilde{c}(fresh)$ *are pairwise distinct, so are* $\widetilde{z}$*. Then the following equations are equivalent: (i)* $(\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{x})\overline{a}[A].P) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{y})\overline{a}[B].Q)$ *for some name* a*; (ii)* $(\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{x})(\overline{b}[A]|P)) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{y})(\overline{b}[B]|Q))$ *for a fresh name* b*; (iii)* $(\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{x})(\overline{d}.(A+e)|P)) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{y})(\overline{d}.(B+e)|Q))$ *for fresh names* d, e*; (iv)* $(\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{x})(E[A]|P)) \approx_l (\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{y})(E[B]|Q))$ *for every process* $E[X]$.

3 Local linear bisimulation

We will put forth a variant of local bisimulation, called local linear bisimulation, which simplifies the former by harnessing the special properties of linear processes. Such a variant renders easy the related study, such as axiomatization and logical characterization. The proof of the coincidence between the variant and the original bisimilarity is basically a deep exploiting of two theorems, Theorem 3 and Theorem 4.

Definition 3 (Local linear bisimulation). *A symmetric binary relation* $\mathcal{R}$ *on (closed) processes is a local linear bisimulation, if it is closed under substitution of names, and whenever* $P \mathrel{\mathcal{R}} Q$*, the following properties hold:*

1. *If* $P \xrightarrow{\tau} P'$*, then* $Q \Longrightarrow Q'$ *for some* Q'*, and* $P' \mathrel{\mathcal{R}} Q'$*;*
2. *If* $P \xrightarrow{a(x)} P'$*, then* $Q \overset{a(x)}{\Longrightarrow} Q'$ *for some* Q'*, and* $P' \mathrel{\mathcal{R}} Q'$*;*
3. *If* $P \xrightarrow{\overline{a}x} P'$*, then* $Q \overset{\overline{a}x}{\Longrightarrow} Q'$ *for some* Q'*, and* $P' \mathrel{\mathcal{R}} Q'$*;*
4. *If* $P \xrightarrow{\overline{a}(x)} P'$*, then* $Q \overset{\overline{a}(x)}{\Longrightarrow} Q'$ *for some* Q'*, and for every process* O*,* $(x)(O|P') \mathrel{\mathcal{R}} (x)(O|Q')$.
5. *If* $P \xrightarrow{a(\mathbf{c})} P'$*, where* c *is a fresh name, then* $Q \overset{a(\mathbf{c})}{\Longrightarrow} Q'$ *for some* Q'*, and* $P' \mathrel{\mathcal{R}} Q'$*;*
6. *If* $P \xrightarrow{(\widetilde{x})\overline{a}A} P'$*, then* $Q \overset{(\widetilde{y})\overline{a}B}{\Longrightarrow} Q'$ *for some* $\widetilde{y}, B, Q'$*. And for a process* $E[X]$ *of the form* $\overline{c}.(X+d)$*, where* c, d *are fresh names, it holds that* $(\widetilde{x})(E[A]|P') \mathrel{\mathcal{R}} (\widetilde{y})(E[B]|Q')$*, that is* $(\widetilde{x})(\overline{c}.(A+d)|P') \mathrel{\mathcal{R}} (\widetilde{y})(\overline{c}.(B+d)|Q')$.

We say P *is local linear bisimilar to* Q*, written* $P \approx_{ll} Q$*, if there exists some local linear bisimulation* $\mathcal{R}$ *such that* $P\mathcal{R}Q$*. Hence* $\approx_{ll}$ *is the largest local linear bisimulation.*

Note the difference from local bisimulation in higher-order input and higher-order output, which borrows some insight into the bisimulation feature on linear higher-order processes and down-scales general higher-order analysis. Though Theorem 2 looks like Theorem 3 and Theorem 4 in a sense, it differs in that it has no essential down-scaling effect, as will be seen. It is clear that $\sim \subseteq \approx_l \subseteq \approx_{ll}$. Local linear bisimulation up-to $\sim$ can be defined in the standard way.

3.1 Characterizing local linear bisimulation

In this section, we examine the properties of local linear bisimulation.

Lemma 6 (Bisimulation Lemma). *Suppose P, Q are processes. If $P\Longrightarrow\cdot \approx_{ll} Q$ and $Q\Longrightarrow\cdot \approx_{ll} P$, then $P \approx_{ll} Q$.*

Lemma 7. *Suppose P, Q are processes and $\mathcal{R}$ a local linear bisimulation. If $P\Longrightarrow \cdot\ \mathcal{R}\ Q$ and $Q\Longrightarrow \cdot\ \mathcal{R}\ P$, then $P \approx_{ll} Q$.*

The two lemmas above serve as a basis for the lemmas henceforth, and the proof is straightforward. Next are two lemmas forming the basis of equivalence property of $\approx_{ll}$, and the second also contributes to the Concretion Theorem (Theorem 7).

Lemma 8. *Suppose O, P, Q are processes. If $(x)(a.O|P) \approx_{ll} (x)(a.O|Q)$ for a fresh name a, then $(x)(O|P) \approx_{ll} (x)(O|Q)$.*

Proof. As a is fresh, $(x)(P|a.R)\xrightarrow{a}(x)(P|R)$ must be simulated by $(x)(Q|a.R)\Longrightarrow(x)(Q_1|a.R)\xrightarrow{a}(x)(Q_1|R)\Longrightarrow Q' \approx_{ll} (x)(P|R)$, which can be rewritten as $(x)(Q|a.R)\xrightarrow{a}(x)(Q|R)\Longrightarrow Q' \approx_{ll} (x)(P|R)$. Similarly we have that $(x)(P|R)\Longrightarrow P' \approx_{ll} (x)(Q|R)$ for some P'. So by Bisimulation Lemma (Lemma 6), we have $(x)(P|R) \approx_{ll} (x)(Q|R)$. □

Lemma 9. *Suppose A, B, P, Q are processes, and a, c, d are fresh names. If $(\widetilde{x})(\overline{a}[A]|P) \approx_{ll} (\widetilde{y})(\overline{a}[B]|Q)$, then $(\widetilde{x})(\overline{c}.(A{+}d)|P) \approx_{ll} (\widetilde{y})(\overline{c}.(B{+}d)|Q)$.*

Proof. Suppose $(\widetilde{x})(\overline{a}[A]|P) \approx_{ll} (\widetilde{y})(\overline{a}[B]|Q)$ for a fresh name a. Then $(\widetilde{x})(\overline{a}[A]|P)\xrightarrow{(\widetilde{x_1})\overline{a}[A]}(\widetilde{x_2})P$, where $\widetilde{x_1}\widetilde{x_2}$ is $\widetilde{x}$. As a is fresh, this must be simulated by (for some Q_1, Q')

$$(\widetilde{y})(\overline{a}[B]|Q)\Longrightarrow(\widetilde{y})(\overline{a}[B]|Q_1)\xrightarrow{(\widetilde{y_1})\overline{a}[B]}(\widetilde{y_2})Q_1\Longrightarrow(\widetilde{y_2})Q'$$

, where $\widetilde{y_1}\widetilde{y_2}$ is $\widetilde{y}$, such that $(\widetilde{y_1})(G[B]|(\widetilde{y_2})Q') \approx_{ll} (\widetilde{x_1})(G[A]|(\widetilde{x_2})P)$, for a process $G[X] \triangleq \overline{c}.(X{+}d)$ (c, d are fresh). By α-conversion and structure equivalence, it can be rewritten as $(\widetilde{y})(G[B]|Q') \approx_{ll} (\widetilde{x})(G[A]|P)$.

It follows from the simulating transition sequence that $Q\Longrightarrow Q_1\Longrightarrow Q'$, so we have $(\widetilde{y})(G[B]|Q)\Longrightarrow\cdot \approx_{ll} (\widetilde{x})(G[A]|P)$, where the dot is the process $(\widetilde{y})(G[B]|Q')$. Similarly we know that $(\widetilde{x})(G[A]|P)\Longrightarrow\cdot \approx_{ll} (\widetilde{y})(G[B]|Q)$. By Bisimulation Lemma (Lemma 6) we have $(\widetilde{x})(G[A]|P) \approx_{ll} (\widetilde{y})(G[B]|Q)$, which is exactly $(\widetilde{x})(\overline{c}.(A{+}d)|P) \approx_{ll} (\widetilde{y})(\overline{c}.(B{+}d)|Q)$. □

Lemma 10. *$\approx_{ll}$ is an equivalence relation.*

Proof. We need to take advantage of Lemma 8, Lemma 9 and Bisimulation Lemma (Lemma 6 or Lemma 7). The proof is quite routine after taking this into consideration. □

Theorem 5. *$\approx_{ll}$ is a congruence relation on all the calculus operators except the choice operator.*

Proof. We use a similar approach to that in [12] [8]. That is, define the transition closure of a designed relation saying the desired properties of the bisimilarity, as $\mathcal{S}_0 \triangleq\approx_{ll}$,
$\mathcal{S}_{i+1} \triangleq \left\{ \begin{array}{l} (\tau.P, \tau.Q), (a(x).P, a(x).Q), (\overline{a}x.P, \overline{a}x.Q), (a(X).P, a(X).Q), \\ (\overline{a}A.P, \overline{a}A.Q), (P\,|\,R, Q\,|\,R), ((x)P, (x)Q) \end{array} \middle| \, P\,\mathcal{S}_i\,Q \right\}$. And $\mathcal{S} \triangleq \bigcup_{i\in\omega} \mathcal{S}_i$.

And we then show that $\mathcal{S}$ is a local linear bisimulation up-to $\sim$. The details are routine and we skip them here. □

The pattern of the proof of Theorem 5 is like that in [12], [8] or even [6]. Also note an alternative approach for proving congruence property in higher-order process calculi in [15], which is considered more uniform and general. Related work can be found in [16] [17] [18].

Lemma 11. *Suppose $E[X], A$ are processes and $E[A]\xrightarrow{\alpha}E'[A']$, where A contributes in the action. Then $E[I_a]\xrightarrow{a(A)}E''[A]\xrightarrow{\alpha}E'[A']$, for a fresh name a and some E''.*

Proof. Routine by induction on the derivation height of $E[A]\xrightarrow{\alpha}E'[A']$. Note the linearity plays an important part in the proof, since in general $E[I_a]$ has to make several input actions to reach the same state as from $E[A]$ because X can appear in several (concurrent) positions. Also note the choice operator that results in the necessity of E''. □

Lemma 12. *Suppose P is a process and actions α, β are complementary. If $P\xrightarrow{\alpha}P_1, P\xrightarrow{\beta}P_2$, and $P\xrightarrow{\alpha}P_1\xrightarrow{\beta}P_2'$, for some P_2', or $P\xrightarrow{\beta}P_2\xrightarrow{\alpha}P_1'$, for some P_1', and α, β are not from the same sub-process of P, then $P\xrightarrow{\tau}P_3$, for some P_3.*

Moreover, we have similar result on weak transitions. If $P\overset{\alpha}{\Longrightarrow}P_1, P\overset{\beta}{\Longrightarrow}P_2$, and $P\overset{\alpha}{\Longrightarrow}P_1\overset{\beta}{\Longrightarrow}P_2'$, for some P_2', or $P\overset{\beta}{\Longrightarrow}P_2\overset{\alpha}{\Longrightarrow}P_1'$, for some P_1', and α, β are not from the same sub-process of P, then $P\overset{\tau}{\Longrightarrow}P_3$, for some P_3.

Proof. By induction on the derivation height of $P\xrightarrow{\alpha}P_1$ and $P\xrightarrow{\beta}P_2$. It is a case analysis of the form of P, based on the operational semantics. It is a routine check. The weak case can be derived from the strong case, tackling the internal action sequence with some care. □

Remark. This lemma states an operational property. One can refine P_3 to be $(\widetilde{z})P_2'$ or $(\widetilde{z})P_1'$ for some local names $\widetilde{z}$, which is empty when the communication involves no local names. We will apply this lemma in, for example, the proof of the Abstraction theorem of $\approx_{ll}$.

Theorem 6 (Abstraction). *Suppose $\widetilde{c}$ are pairwise distinct fresh names, and $\widetilde{z}$ are pairwise distinct. Then the following equations are equivalent:*
(i) $(\widetilde{z})(\widetilde{\overline{c}z}|a(X).P) \approx_{ll} (\widetilde{z})(\widetilde{\overline{c}z}|a(X).Q)$ for some name a; (ii) $(\widetilde{z})(\widetilde{\overline{c}z}|P\{I_b/X\}) \approx_{ll} (\widetilde{z})(\widetilde{\overline{c}z}|Q\{I_b/X\})$ for a fresh name b; (iii) $(\widetilde{z})(\widetilde{\overline{c}z}|P\{b/X\}) \approx_{ll} (\widetilde{z})(\widetilde{\overline{c}z}|Q\{b/X\})$ for a fresh name b; (iv) $(\widetilde{z})(\widetilde{\overline{c}z}|P\{R/X\}) \approx_{ll} (\widetilde{z})(\widetilde{\overline{c}z}|Q\{R/X\})$ for every process R.
Or sometimes we just need the special case:
(i) $a(X).P \approx_{ll} a(X).Q$ for some name a; (ii) $P\{I_b/X\} \approx_{ll} Q\{I_b/X\}$ for a fresh name b; (iii) $P\{b/X\} \approx_{ll} Q\{b/X\}$ for a fresh name b; (iv) $P\{R/X\} \approx_{ll} Q\{R/X\}$ for every process R.

Proof. The proof concentrates on the special cases. The general one is not far from the proof here.

$(i) \Leftrightarrow (iii)$ is easy by definition.

$$\left.\begin{array}{l}(ii) \Leftrightarrow (iv)\\(iii) \Leftrightarrow (iv)\end{array}\right\}\begin{array}{l}\text{These two cases are similar in style, and note}\\(a)(E\{I_a/X\}|\overline{a}[A]) \sim (a)(E\{a/X\}|\overline{a}.A).\end{array}$$

So we simply consider $(iii) \Leftrightarrow (iv)$ here. Since $(iv) \Rightarrow (iii)$ is obvious, we cope with $(iii) \Rightarrow (iv)$. We define the following relation:

$$\mathcal{R} \triangleq \{(P\{R/X\}, Q\{R/X\}) \mid \begin{array}{l}P\{b/X\} \approx_{ll} Q\{b/X\}\\ b \text{ is fresh, } R \text{ is a process}\end{array}\} \cup \approx_{ll}$$

First we show that $\mathcal{R}$ is closed under name substitution. Suppose $P\{R/X\}\ \mathcal{R}\ Q\{R/X\}$ for $P\{b/X\} \approx_{ll} Q\{b/X\}$. Let σ be a substitution on names and d be a fresh name (not in $n(P,Q)$ and σ). We have $P\sigma\{d/X\} \approx_{ll} Q\sigma\{d/X\}$, since $\approx_{ll}$ is closed under substitution of names. Then we know $P\sigma\{R\sigma/X\}\ \mathcal{R}\ Q\sigma\{R\sigma/X\})$.

Secondly we show $\mathcal{R}$ is a local linear bisimulation up-to $\sim$. Suppose $P\{R/X\}\xrightarrow{\alpha}P'$. Note α is of the form $a(\mathbf{d})$ in higher-order input. There are several cases to analyze.

- The action α is caused by a copy of R, that is $R\xrightarrow{\alpha}R'$ for some R'. Note substitution on process variables should avoid name capturing. So $P\{R/X\}\xrightarrow{\alpha}P_1\{R'/X\} \equiv P'$. Now we have the following reasoning (for some P_1, Q_1):

$$P\{R/X\} \xrightarrow{\alpha} P_1\{R'/X\}$$

$$\begin{tikzcd}P\{b/X\} \arrow[r, "b(x)"] \arrow[d, dash, dotted, "\approx_{ll}"'] & P_1\{0/X\} \arrow[d, dash, dotted, "\approx_{ll}"] \\ Q\{b/X\} \arrow[r, Rightarrow, "b(x)"] & Q_1\{0/X\}\end{tikzcd}$$

$$Q\{R/X\} \overset{\alpha}{\Longrightarrow} Q_1\{R'/X\}$$

And obviously, one can get, by a simple reasoning, $P_1\{d/X\} \approx_{ll} Q_1\{d/X\}$, for some fresh name d. Hence $P_1\{R'/X\}\ \mathcal{R}\ Q_1\{R'/X\}$.

- The action α is caused by P, that is $P\{R/X\}\xrightarrow{\alpha}P_1\{R/X\} \equiv P'$. Now we have the following reasoning (for some P_1, Q_1):

$$\begin{tikzcd}
P\{R/X\} \arrow[r, "\alpha"] & P_1\{R/X\}
\end{tikzcd}$$

$$\begin{tikzcd}
P\{b/X\} \arrow[r, "\alpha"] \arrow[d, dash, dotted, "\approx_{ll}"'] & P_1\{b/X\} \arrow[d, dash, dotted] \\
Q\{b/X\} \arrow[r, Rightarrow, "\hat{\alpha}"] & Q_1\{b/X\}
\end{tikzcd}$$

$$\begin{tikzcd}
Q\{R/X\} \arrow[r, Rightarrow, "\hat{\alpha}"] & Q_1\{R/X\}
\end{tikzcd}$$

  If α is first-order input, output or higher-order input, the result is straightforward. If α is first-order bound output or higher-order output, some (similar) minor manipulation is needed. We take first-order bound output as the example. Suppose α is $\overline{u}(v)$, in this case, we have, for every process O, and some $P'_1[X] \triangleq (v)(O|P_1[X]), Q'_1[X] \triangleq (v)(O|Q_1[X])$:

$$\begin{tikzcd}
(v)(O|P_1\{b/X\}) \arrow[r, dash, dotted, "\approx_{ll}"] \arrow[d, dash, dotted, "\equiv"'] & (v)(O|Q_1\{b/X\}) \arrow[d, dash, dotted, "\equiv"] \\
P'_1\{b/X\} & Q'_1\{b/X\}
\end{tikzcd}$$

  Then we know $P'_1\{R/X\}\ \mathcal{R}\ Q'_1\{R/X\}$. In summary, $P\{R/X\}\xrightarrow{\alpha}P_1\{R/X\}$ can be simulated by $Q\{R/X\}\overset{\hat{\alpha}}{\Longrightarrow}Q_1\{R/X\}$.
- The action α is τ, and is caused by a communication between P and R. This is the most involved case and has totally six sub-cases. We examine them below.
  - $P\xrightarrow{u(v)}P'$, $R\xrightarrow{\overline{u}(v)}R'$, and $P\{R/X\}\xrightarrow{\tau}(v)(P'\{R'/X\})$.
    We have the following reasoning (for some P'', Q'', Q'). Note the upper row is simulated by the lower row, and b is fresh.

$$\begin{tikzcd}
P\{b/X\} \arrow[r, "b(x)"] \arrow[d, dash, dotted, "\approx_{ll}"'] & P''\{0/X\} \arrow[r, "u(v)"] \arrow[d, dash, dotted, "\approx_{ll}"] & P'\{0/X\} \arrow[d, dash, dotted, "\approx_{ll}"] \\
Q\{b/X\} \arrow[r, Rightarrow, "b(x)"] & Q''\{0/X\} \arrow[r, Rightarrow, "u(v)"] & Q'\{0/X\}
\end{tikzcd}$$

    Moreover, from the premise we also have

$$\begin{tikzcd}
P\{b/X\} \arrow[r, "u(v)"] \arrow[d, dash, dotted, "\approx_{ll}"'] & P'\{b/X\} \arrow[d, dash, dotted, "\approx_{ll}"] \\
Q\{b/X\} \arrow[r, Rightarrow, "u(v)"] & Q'\{b/X\}
\end{tikzcd}$$

    So to summarize a little, we can have (because $\overset{u(v)}{\Longrightarrow}$ comes from Q)

$$\begin{gathered}
Q\{R/X\}\overset{u(v)}{\Longrightarrow}Q'\{R/X\}, \\
Q\{R/X\}\overset{\overline{u}(v)}{\Longrightarrow}Q''\{R'/X\}\overset{u(v)}{\Longrightarrow}Q'\{R'/X\}, \\
P'\{b/X\} \approx_{ll} Q'\{b/X\}.
\end{gathered}$$

    It follows from this, Lemma 12 and (congruence) property of $\approx_{ll}$ that

$$\begin{gathered}
Q\{R/X\}\overset{\tau}{\Longrightarrow}(v)(Q'\{R'/X\}), \\
(v)P'\{b/X\} \approx_{ll} (v)Q'\{b/X\}.
\end{gathered}$$

    Now define $P'''[X] \triangleq (v)P'[X], Q'''[X] \triangleq (v)Q'[X]$ so that

$$P'''\{R'/X\} \triangleq (v)(P'\{R'/X\}), \quad Q'''\{R'/X\} \triangleq (v)(Q'\{R'/X\}).$$

Thus we have in summary

$$\begin{gathered}P\{R/X\}\xrightarrow{\tau}P'''\{R'/X\},\\ Q\{R/X\}\xRightarrow{\tau}Q'''\{R'/X\},\\ P'''\{b/X\}\approx_{ll}Q'''\{b/X\}.\end{gathered}$$

Hence it follows that

$$P'''\{R'/X\}\ \mathcal{R}\ Q'''\{R'/X\}.$$

- $P\xrightarrow{u(v)}P'$, $R\xrightarrow{\overline{u}v}R'$, and $P\{R/X\}\xrightarrow{\tau}(P'\{R'/X\})$. This case is similar to the last case.
- $P\xrightarrow{\overline{u}(v)}P'$, $R\xrightarrow{u(v)}R'$, and $P\{R/X\}\xrightarrow{\tau}(v)(P'\{R'/X\})$.
  We have the following reasoning (for some P'', Q'', Q'). Note the upper row is simulated by the lower row, and b is fresh.

$$\begin{tikzcd}P\{b/X\} \arrow[r, "b(x)"] \arrow[d, dash, dotted, "\approx_{ll}"'] & P''\{0/X\} \arrow[r, "\overline{u}(v)"] \arrow[d, dash, dotted, "\approx_{ll}"] & P'\{0/X\} \arrow[d, dash, dotted] \\ Q\{b/X\} \arrow[r, Rightarrow, "b(x)"] & Q''\{0/X\} \arrow[r, Rightarrow, "\overline{u}(v)"] & Q'\{0/X\}\end{tikzcd}$$

And for every process O, $(v)(O|P'\{0/X\})\approx_{ll}(v)(O|Q'\{0/X\})$. Moreover, from the premise we also have

$$\begin{tikzcd}P\{b/X\} \arrow[r, "\overline{u}(v)"] \arrow[d, dash, dotted, "\approx_{ll}"'] & P'\{b/X\} \arrow[d, dash, dotted] \\ Q\{b/X\} \arrow[r, Rightarrow, "\overline{u}(v)"] & Q'\{b/X\}\end{tikzcd}$$

And for every process O, $(v)(O|P'\{b/X\})\approx_{ll}(v)(O|Q'\{b/X\})$. So to summarize a little, we can have

$$\begin{gathered}Q\{R/X\}\xRightarrow{\overline{u}(v)}Q'\{R/X\},\\ Q\{R/X\}\xRightarrow{u(v)}Q''\{R'/X\}\xRightarrow{\overline{u}(v)}Q'\{R'/X\},\\ (v)(O|P'\{b/X\})\approx_{ll}(v)(O|Q'\{b/X\}),\ \text{for every } O.\end{gathered}$$

It follows from this, Lemma 12 and taking O as 0 (null process) that

$$\begin{gathered}Q\{R/X\}\xRightarrow{\tau}(v)(Q'\{R'/X\}),\\ (v)P'\{b/X\}\approx_{ll}(v)Q'\{b/X\}.\end{gathered}$$

Now define $P'''[X]\triangleq(v)P'[X], Q'''[X]\triangleq(v)Q'[X]$ so that

$$P'''\{R'/X\}\triangleq(v)(P'\{R'/X\}),\quad Q'''\{R'/X\}\triangleq(v)(Q'\{R'/X\}).$$

Thus we have in summary

$$\begin{gathered}P\{R/X\}\xrightarrow{\tau}P'''\{R'/X\},\\ Q\{R/X\}\xRightarrow{\tau}Q'''\{R'/X\},\\ P'''\{b/X\}\approx_{ll}Q'''\{b/X\}.\end{gathered}$$

Hence it follows that $P'''\{R'/X\}\ \mathcal{R}\ Q'''\{R'/X\}$.

- $P\xrightarrow{\overline{u}v}P'$, $R\xrightarrow{u(v)}R'$, and $P\{R/X\}\xrightarrow{\tau}(P'\{R'/X\})$. This case is similar to the last case.
- $P\xrightarrow{u(A)}P'$, $R\xrightarrow{(\widetilde{z})\overline{u}[A]}R'$, and $P\{R/X\}\xrightarrow{\tau}(\widetilde{z})(P'\{R'/X\})$. This case is somewhat similar to the first case. We have the following reasoning (for some P'', Q'', Q_1, P_1). Note the upper row is simulated by the lower row, and b is fresh. Suppose c is fresh.

$$\begin{tikzcd}P\{b/X\} \arrow[r, "b(x)"] \arrow[d, dash, dotted, "\approx_{ll}"'] & P''\{0/X\} \arrow[r, "u(\mathbf{c})"] \arrow[d, dash, dotted, "\approx_{ll}"] & P_1\{0/X\} \arrow[d, dash, dotted, "\approx_{ll}"] \\ Q\{b/X\} \arrow[r, Rightarrow, "b(x)"] & Q''\{0/X\} \arrow[r, Rightarrow, "u(\mathbf{c})"] & Q_1\{0/X\}\end{tikzcd}$$

where $P_1 \equiv P'\{\mathbf{c}/A\}$. Moreover, from the premise we also have

$$\begin{tikzcd}
P\{b/X\} \arrow[r, "u(\mathbf{c})"] \arrow[d, dotted, no head, "\approx_{ll}"'] & P_1\{b/X\} \arrow[d, dotted, no head, "\approx_{ll}"] \\
Q\{b/X\} \arrow[r, Rightarrow, "u(\mathbf{c})"] & Q_1\{b/X\}
\end{tikzcd}$$

So to summarize a little, we can have

$$\begin{gathered}
Q\{R/X\} \xRightarrow{u(\mathbf{c})} Q_1\{R/X\}, \\
Q\{R/X\} \xRightarrow{(\widetilde{z})\overline{u}[A]} Q''\{R'/X\} \xRightarrow{u(\mathbf{c})} Q_1\{R'/X\}, \\
P_1\{b/X\} \approx_{ll} Q_1\{b/X\}.
\end{gathered}$$

Then we have (for some Q')

$$\begin{gathered}
Q\{R/X\} \xRightarrow{u(A)} Q'\{R/X\}, \\
Q\{R/X\} \xRightarrow{(\widetilde{z})\overline{u}[A]} Q''\{R'/X\} \xRightarrow{u(A)} Q'\{R'/X\}, \\
P'\{\mathbf{c}/A\}\{b/X\} \approx_{ll} Q'\{\mathbf{c}/A\}\{b/X\},
\end{gathered}$$

where $Q_1 \equiv Q'\{\mathbf{c}/A\}$. It follows from this, Lemma 12

$$Q\{R/X\} \xRightarrow{\tau} (\widetilde{z})(Q'\{R'/X\}). \tag{1}$$

We define some $P_1'[Y]$ and $Q_1'[Y]$ (Y is different from X) such that

$$\begin{gathered}
P_1'\{\mathbf{c}/Y\} \equiv P_1, P_1'\{A/Y\} \equiv P'; \\
Q_1'\{\mathbf{c}/Y\} \equiv Q_1, Q_1'\{A/Y\} \equiv Q'.
\end{gathered}$$

Then $P_1'\{\mathbf{c}/Y\}\{b/X\} \approx_{ll} Q_1'\{\mathbf{c}/Y\}\{b/X\}$. Since Y and X are different, we know that $P_1'\{b/X\}\{\mathbf{c}/Y\} \approx_{ll} Q_1'\{b/X\}\{\mathbf{c}/Y\}$. Now by the definition (structure of $\mathcal{R}$) we know $P_1'\{b/X\}\{A/Y\} \;\mathcal{R}\; Q_1'\{b/X\}\{A/Y\}$. That is $P_1'\{A/Y\}\{b/X\} \;\mathcal{R}\; Q_1'\{A/Y\}\{b/X\}$. And this is exactly $P'\{b/X\} \;\mathcal{R}\; Q'\{b/X\}$. Again by the definition (of $\mathcal{R}$) we have in any case $P'\{b'/X\} \approx_{ll} Q'\{b'/X\}$, for a fresh name b'. By the (congruence) property of $\approx_{ll}$ it follows that

$$(\widetilde{z})P'\{b'/X\} \approx_{ll} (\widetilde{z})Q'\{b'/X\}. \tag{2}$$

Now define $P'''[X] \triangleq (\widetilde{z})P'[X], Q'''[X] \triangleq (\widetilde{z})Q'[X]$ so that

$$P'''\{R'/X\} \triangleq (\widetilde{z})(P'\{R'/X\}), \quad Q'''\{R'/X\} \triangleq (\widetilde{z})(Q'\{R'/X\}).$$

Thus we have in summary

$$\begin{gathered}
P\{R/X\} \xrightarrow{\tau} P'''\{R'/X\}, \\
Q\{R/X\} \xRightarrow{\tau} Q'''\{R'/X\}, \quad \text{by (1)}, \\
P'''\{b'/X\} \approx_{ll} Q'''\{b'/X\}, \quad \text{by (2)}.
\end{gathered}$$

Hence it follows that $P'''\{R'/X\} \;\mathcal{R}\; Q'''\{R'/X\}$.

- $P \xrightarrow{(\widetilde{z})\overline{u}[A]} P'$, $R \xrightarrow{u(A)} R'$, and $P\{R/X\} \xrightarrow{\tau} (\widetilde{z})(P'\{R'/X\})$.
  We have the following reasoning (for some $\widetilde{z'}, B, P'', Q'', Q'$). Note the upper row is simulated by the lower row, and b is fresh.

$$\begin{tikzcd}
P\{b/X\} \arrow[r, "b(x)"] \arrow[d, dotted, no head, "\approx_{ll}"'] & P''\{0/X\} \arrow[r, "(\widetilde{z})\overline{u}[A]"] \arrow[d, dotted, no head, "\approx_{ll}"] & P'\{0/X\} \arrow[d, dotted, no head] \\
Q\{b/X\} \arrow[r, Rightarrow, "b(x)"] & Q''\{0/X\} \arrow[r, Rightarrow, "(\widetilde{z'})\overline{u}[B]"] & Q'\{0/X\}
\end{tikzcd}$$

And for a process $E[X] \equiv \overline{c}.(X{+}d)$ (c, d are fresh), $(\widetilde{z})(E[A]|P'\{0/X\}) \approx_{ll} (\widetilde{z'})(E[B]|Q'\{0/X\})$. Moreover, from the premise we also have:

$$\begin{tikzcd}
P\{b/X\} \arrow[r, "(\widetilde{z})\overline{u}[A]"] \arrow[d, dotted, no head, "\approx_{ll}"'] & P'\{b/X\} \arrow[d, dotted, no head] \\
Q\{b/X\} \arrow[r, Rightarrow, "(\widetilde{z'})\overline{u}[B]"] & Q'\{b/X\}
\end{tikzcd}$$

And for a process $E[X] \equiv \overline{c}.(X{+}d)$ (c, d are fresh), $(\widetilde{z})(E[A]|P'\{b/X\}) \approx_{ll} (\widetilde{z'})(E[B]|Q'\{b/X\})$. So to summarize a little, we can have (for some R'' such that $R'' \equiv R'\{B/A\}$)

$$Q\{R/X\} \stackrel{(\widetilde{z'})\overline{u}[B]}{\Longrightarrow} Q'\{R/X\}, \tag{3}$$

$$Q\{R/X\} \stackrel{u(B)}{\Longrightarrow} Q''\{R''/X\} \stackrel{(\widetilde{z'})\overline{u}[B]}{\Longrightarrow} Q'\{R''/X\}, \tag{4}$$

$$(\widetilde{z})(\overline{c}.(A{+}d)|P'\{b/X\}) \approx_{ll} (\widetilde{z'})(\overline{c}.(B{+}d)|Q'\{b/X\}). \tag{5}$$

It follows from (3), (4), and Lemma 12 that

$$Q\{R/X\} \stackrel{\tau}{\Longrightarrow} (\widetilde{z'})(Q'\{R''/X\}). \tag{6}$$

By Concretion Theorem (Theorem 7) and (5), $(\widetilde{z})(G[A]|P'\{b/X\}) \approx_{ll} (\widetilde{z'})(G[B]|Q'\{b/X\})$, for every $G[Y]$. It is easy to know that there exists some $R'''[Y]$ such that

$$R'''\{A/Y\} \equiv R', \quad R'''\{B/Y\} \equiv R''.$$

Now choose $G[Y] \equiv \overline{b}.(R'''[Y]{+}d)$, then we get

$$(\widetilde{z})(\overline{b}.(R'''\{A/Y\}{+}d)|P'\{b/X\}) \approx_{ll} (\widetilde{z'})(\overline{b}.(R'''\{B/Y\}{+}d)|Q'\{b/X\}),$$

that is $(\widetilde{z})(\overline{b}.(R'{+}d)|P'\{b/X\}) \approx_{ll} (\widetilde{z'})(\overline{b}.(R''{+}d)|Q'\{b/X\})$. Thus by Lemma 14, we have $(\widetilde{z})(P'\{R'/X\}) \approx_{ll} (\widetilde{z'})(Q'\{R''/X\})$. Hence $(\widetilde{z})(P'\{R'/X\})\ \mathcal{R}\ (\widetilde{z'})(Q'\{R''/X\})$. Taking (6) into consideration, this closes the simulation.

By here the proof is completed. □

Below we consider the Concretion Theorem, before whose proof we give some auxiliary lemmas.

Lemma 13. *Suppose a, b are fresh names, $E[X]$ is an arbitrary process with at most process variable X, and A is a process. We have the following properties:*

1. *If $(\widetilde{x})E[A] \xrightarrow{\lambda} P$, where A takes part in the action, then $(\widetilde{x})(E[a]|\overline{a}.(A{+}b)) \stackrel{\lambda}{\Longrightarrow} P'$ for some P', and $P \sim P'$;*
2. *The converse. If $(\widetilde{x})(E[a]|\overline{a}.(A{+}b)) \stackrel{\lambda}{\Longrightarrow} P'$ and a, b do not appear in P' (and λ either), then we have $(\widetilde{x})E[A] \stackrel{\lambda}{\Longrightarrow} P$ for some P, and $P' \sim P$.*

Proof. The proof is essentially the same as Lemma 22 in [12] and note linearity plays an essential part, so we omit the detail. □

Lemma 14. *Suppose $a, b, \widetilde{c}$ are all fresh names, and $E[X], F[X]$ are processes. If $(\widetilde{x})(\widetilde{\overline{c}x}|(\widetilde{y})(\overline{a}.(A{+}b)|E[a])) \approx_{ll} (\widetilde{x})(\widetilde{\overline{c}x}|(\widetilde{z})(\overline{a}.(B{+}b)|F[a]))$, then $(\widetilde{x})(\widetilde{\overline{c}x}|(\widetilde{y})E[A]) \approx_{ll} (\widetilde{x})(\widetilde{\overline{c}x}|(\widetilde{z})F[B])$. Or we just need the special case. If $(\widetilde{y})(\overline{a}.(A{+}b)|E[a]) \approx_{ll} (\widetilde{z})(\overline{a}.(B{+}b)|F[a])$, then $(\widetilde{y})E[A] \approx_{ll} (\widetilde{z})F[B]$.*

Proof. Since handling of local contexts is somewhat regular [12], the proof focuses on the contents in the local environment. It is just proving the special case.

We define a relation $\mathcal{R}$ as follows:
$\mathcal{R} \triangleq \{((\widetilde{y})E[A], (\widetilde{z})F[B]) \mid (\widetilde{y})(\overline{a}.(A{+}b)|E[a]) \approx_{ll} (\widetilde{z})(\overline{a}.(B{+}b)|F[a]), a, b$ are fresh$\} \cup \approx_{ll}$. We show that $\mathcal{R}$ is a local linear bisimulation up-to $\sim$. It is easy to show that $\mathcal{R}$ is closed under substitution of names, so this is skipped. Suppose $(\widetilde{y})E[A]\mathcal{R}(\widetilde{z})F[B]$, and $(\widetilde{y})E[A] \xrightarrow{\lambda} P$. We have the following analysis.

- A does not take part in the action λ. Then we know that there exists $E_1[X]$ such that $P \equiv (\widetilde{y})E_1[A]$. By this we have $(\widetilde{y})(\overline{a}.(A{+}b)|E[a]) \xrightarrow{\lambda} (\widetilde{y})(\overline{a}.(A{+}b)|E_1[a])$. From the premise, we have the next simulation for some $F_1[X]$: $(\widetilde{z})(\overline{a}.(B{+}b)|F[a]) \stackrel{\lambda}{\Longrightarrow} (\widetilde{z})(\overline{a}.(B{+}b)|F_1[a])$, which is the only possibility because a, b are fresh. Therefore, we know that $(\widetilde{z})F[B] \stackrel{\lambda}{\Longrightarrow} (\widetilde{z})F_1[B]$.
  - λ is a silent action, first-order input, output or higher-order input. This case is direct. We have $(\widetilde{z})(\overline{a}.(B{+}b)|F_1[a]) \approx_{ll} (\widetilde{y})(\overline{a}.(A{+}b)|E_1[a])$. Then $(\widetilde{y})E_1[A]\ \mathcal{R}\ (\widetilde{z})F_1[B]$.
  - λ is a first-order bound output or higher-order output. This case is a little complicated. We take the first-order bound output as example, the higher-order output case is similar. Suppose λ is $\overline{u}(v)$. We have for every process O, $(v)(O|(\widetilde{y})(\overline{a}.(A{+}b)|E_1[a])) \approx_{ll} (v)(O|(\widetilde{z})(\overline{a}.(B{+}b)|F_1[a]))$, which results in $(\widetilde{yv})(\overline{a}.(A{+}b)|(E_1[a]|O)) \approx_{ll} (\widetilde{zv})(\overline{a}.(B{+}b)|(F_1[a]|O))$, thanks to α-conversion. Define $E_1'[X] \triangleq E_1[X]|O$, $\quad F_1'[X] \triangleq F_1[X]|O$. So we have $(\widetilde{yv})(\overline{a}.(A{+}b)|E_1'[a]) \approx_{ll} (\widetilde{zv})(\overline{a}.(B{+}b)|F_1'[a])$, and $(v)(O|(\widetilde{y})E_1[A]) \sim (\widetilde{yv})(E_1[A]|O) \equiv (\widetilde{yv})(E_1'[A])$, also $(v)(O|(\widetilde{z})F_1[B]) \sim (\widetilde{zv})(F_1[B]|O) \equiv (\widetilde{zv})(F_1'[B])$. Now we know $(\widetilde{yv})(E_1'[A])\ \mathcal{R}\ (\widetilde{zv})(F_1'[B])$.

- A is involved in the action λ. Then by Lemma 13, $(\widetilde{y})(\overline{a}.(A{+}b)|E[a])\overset{\lambda}{\Longrightarrow}P_1$, for some P_1, and $P_1 \sim P$. From the premise, we know that $(\widetilde{z})(\overline{a}.(B{+}b)|F[a])\overset{\lambda'}{\Longrightarrow}Q_1$, for some Q_1. A simple analysis can tell us that neither the fresh name a nor b shall appear in Q_1. So again by Lemma 13, we have $(\widetilde{z})F[B]\overset{\lambda'}{\Longrightarrow}Q$, for some Q, and $Q \sim Q_1$.
  - λ is a silent action, first-order input, output or higher-order input. In this case, λ' is just λ. We have immediately $P \sim P_1 \approx_{ll} Q_1 \sim Q$.
  - λ is a first-order bound output or higher-order output. We take the higher-order output as example, the first-order bound output case is similar. Suppose λ is $(\widetilde{v})\overline{u}[H]$ and λ' is $(\widetilde{v'})\overline{u}[K]$. We have for a process $G[X] \equiv \overline{d}.(X{+}e)$ (d, e are fresh): $(\widetilde{v})(G[H]|P_1) \approx_{ll} (\widetilde{v'})(G[K]|Q_1)$. Since $(\widetilde{v})(G[H]|P) \sim (\widetilde{v})(G[H]|P_1), (\widetilde{v'})(G[K]|Q_1) \sim (\widetilde{v'})(G[K]|Q)$, we are finished.

Hence $\mathcal{R}$ is a local linear bisimulation up-to $\sim$. □

Theorem 7 (Concretion). *Suppose $\widetilde{c}$ are pairwise distinct fresh names, and $\widetilde{z}$ are pairwise distinct. Then the following equations are equivalent:*
(i) $(\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{x})\overline{a}[A].P) \approx_{ll} (\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{y})\overline{a}[B].Q)$ for some name a; (ii) $(\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{x})(\overline{b}[A]|P) \approx_{ll} (\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{y})(\overline{b}[B]|Q)$ for a fresh name b; (iii) $(\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{x})(\overline{c}.(A{+}d)|P) \approx_{ll} (\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{y})(\overline{c}.(B{+}d)|Q))$ for fresh names c, d;
(iv) $(\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{x})(E[A]|P) \approx_{ll} (\widetilde{z})(\widetilde{\overline{c}z}|(\widetilde{y})(E[B]|Q)$ for every process $E[X]$.
Or we just need the special case:
(i) $(\widetilde{x})\overline{a}[A].P \approx_{ll} (\widetilde{y})\overline{a}[B].Q$ for some name a; (ii) $(\widetilde{x})(\overline{b}[A]|P) \approx_{ll} (\widetilde{y})(\overline{b}[B]|Q)$ for a fresh name b;
(iii) $(\widetilde{x})(\overline{c}.(A{+}d)|P) \approx_{ll} (\widetilde{y})(\overline{c}.(B{+}d)|Q)$ for fresh names c, d; (iv) $(\widetilde{x})(E[A]|P) \approx_{ll} (\widetilde{y})(E[B]|Q)$ for every process $E[X]$.

Proof. The proof as usual focuses on the contents in the local environment. We prove this theorem in the following strategy:

$$\begin{tikzcd}
1 \arrow[r, Leftrightarrow, "definition"] & 3 \arrow[rr, Rightarrow, "lemma\ 14"] & & 4 \arrow[dl, Rightarrow, "direct"] \\
& & 2 \arrow[ul, Rightarrow, "lemma\ 9"] &
\end{tikzcd}$$

Most is straightforward. Something worth noting is the applying of Lemma 14 in $3 \Rightarrow 4$. Since $\approx_{ll}$ is closed under parallel composition, we have, for every process $E[X]$: $(\widetilde{x})(\overline{c}.(A{+}d)|P)|E[c] \approx_{ll} (\widetilde{y})(\overline{c}.(B{+}d)|Q)|E[c]$, which can be equivalently transformed to $(\widetilde{x})(\overline{c}.(A{+}d)|(P|E[c])) \approx_{ll} (\widetilde{y})(\overline{c}.(B{+}d)|(Q|E[c]))$. By defining $E'[X] \triangleq P|E[X], E''[X] \triangleq Q|E[X]$, we have $(\widetilde{x})(\overline{c}.(A{+}d)|E'[c]) \approx_{ll} (\widetilde{y})(\overline{c}.(B{+}d)|E''[c])$. Then by Lemma 14, $(\widetilde{x})(E'[A]) \approx_{ll} (\widetilde{y})(E''[B])$, which is exactly $(\widetilde{x})(P|E[A]) \approx_{ll} (\widetilde{y})(Q|E[B])$. To summary (by commutativity of $\approx_{ll}$) we have $(\widetilde{x})(E[A]|P) \approx_{ll} (\widetilde{y})(E[B]|Q)$, for every process E. We are done. □

3.2 Coincidence with local bisimilarity

Below is the important theorem for local linear bisimilarity. It constitutes the main result of this paper.

Theorem 8. *Local linear bisimilarity coincides with local bisimilarity, that is $\approx_l = \approx_{ll}$.*

Proof. A routine checking based on the definition of the two bisimulations, by taking the following two theorems into consideration: (i) Abstraction Theorem (Theorem 6); (ii) Concretion Theorem (Theorem 7). We prove $\approx_l = \approx_{ll}$ in two steps.
"$\subseteq$"". This direction is straightforward. Because by the definitions (examining each clause in the definitions), every local bisimulation is a local linear bisimulation. That is, local linear bisimilarity is not less than local bisimilarity.
"$\supseteq$"". We show that $\mathcal{R} \triangleq \{(P, Q) \mid P \approx_{ll} Q,\ P, Q$ are LHOPi processes$\} \cup \approx_l$ is a local bisimulation. One has to examine the clauses in the definition of local bisimulation one by one. The most difficult cases are higher-order input and output. Below we analyze each of them. Suppose $P\mathcal{R}Q$ because $P \approx_{ll} Q$.

$$\left.\begin{array}{l} (i).\ P\xrightarrow{\tau}P';\ (ii).\ P\xrightarrow{a(x)}P' \\ (iii).\ P\xrightarrow{\overline{a}x}P';\ (iv).\ P\xrightarrow{\overline{a}(x)}P' \end{array}\right\}$$ These cases are not hard, since the simulation clauses in $\approx_{ll}$ are stating the same things as those in local bisimulation.

(v). $P\xrightarrow{a(A)}P'$. Clearly we can define $P''[X]$ so that $P''\{A/X\} \equiv P'$. Then $P\xrightarrow{a(b)}P''\{b/X\}$, for a fresh name b. Since $P \approx_{ll} Q$, we have $Q\overset{a(b)}{\Longrightarrow}Q''\{b/X\}$ for some Q'', and thus $Q\overset{a(A)}{\Longrightarrow}Q''\{A/X\} \triangleq Q'$, meanwhile $P''\{b/X\} \approx_{ll} Q''\{b/X\}$. Then by Abstraction Theorem on $\approx_{ll}$ (Theorem 6), $P' \equiv P''\{A/X\} \approx_{ll} Q''\{A/X\} \equiv Q'$, which leads to $P'\ \mathcal{R}\ Q'$.

(vi). $P\xrightarrow{(\widetilde{x})\overline{a}[A]}P'$. Because $P \approx_{ll} Q$, we know that there exist some $\widetilde{y}, B, Q'$ such that $Q\overset{(\widetilde{y})\overline{a}[B]}{\Longrightarrow}Q'$, and for a process $E[X] \triangleq \overline{c}.(X{+}d)$ (c, d are fresh), $(\widetilde{x})(E[A]|P') \approx_{ll} (\widetilde{y})(E[B]|Q')$. That is $(\widetilde{x})(\overline{c}.(A{+}d)|P') \approx_{ll}$

$(\widetilde{y})(\overline{c}.(B+d)|Q')$. Then by Concretion Theorem on $\approx_{ll}$ (Theorem 7), we have, for every process $G[X]$ (with no name collision) $(\widetilde{x})(G[A]|P') \approx_{ll} (\widetilde{y})(G[B]|Q')$, which is exactly what we need to close the simulation, that is $(\widetilde{x})(G[A]|P')\ \mathcal{R}\ (\widetilde{y})(G[B]|Q')$. Now the proof is completed. □

3.3 On first-order bound output

What if we try simplifying the clause of first-order bound output in local bisimulation using Localization Theorem (Theorem 2), in the way like what we have done for higher-order actions? Will it be effective as in the higher-order output? Considering the characteristic of linear higher-order processes and the essence of Localization Theorem, our answer is NO. We have the following points.

- If we try to use *(ii)* in Localization Theorem to simplify the local bisimulation, that is, in the simulation step a special process $\overline{b}x$ (b is fresh) rather than an arbitrary process O is required, the obtained bisimulation (local linear bisimulation with this modification on first-order bound output clause, we denote it by "LLN bisimulation") may not even have the corresponding Localization Theorem, because the *(iii)* cannot be reached under a simplified simulation condition. So one cannot recover the original local bisimilarity.
- In the "LLN bisimulation" , the first-order bound output cannot be eliminated in simulation, because the simulation result says the same thing as before the simulation, which may cause loop definition. One shall avoid this anytime.
- Apart from the special process $\overline{b}x$, which contributes nothing to simplification, no other special process is known to exist to replace the arbitrary process O without loss of generality. We tend to believe no such process exist.

Although the first-order bound output clause in local bisimulation cannot be simplified, it can be rewritten in a form that eases discussion. In other words, the 'simplification' here is in the sense that it can provide some simple means in tackling local bisimulation in the case of first-order bound output.

Definition 4. *A symmetric binary relation $\mathcal{R}$ on (closed) processes is a local linear variant bisimulation, if it is closed under substitution of names, and whenever $P\ \mathcal{R}\ Q$, the following properties hold:*

1. *If $P \xrightarrow{\tau} P'$, then $Q \Longrightarrow Q'$ for some Q', and $P'\ \mathcal{R}\ Q'$;*
2. *If $P \xrightarrow{a(x)} P'$, then $Q \overset{a(x)}{\Longrightarrow} Q'$ for some Q', and $P'\ \mathcal{R}\ Q'$;*
3. *If $P \xrightarrow{\overline{a}x} P'$, then $Q \overset{\overline{a}x}{\Longrightarrow} Q'$ for some Q', and $P'\ \mathcal{R}\ Q'$;*
4. *If $P \xrightarrow{\overline{a}(x)} P'$, then $Q \overset{\overline{a}(x)}{\Longrightarrow} Q'$ for some Q', and for all processes O_1 and O_2 such that $O_1\ \mathcal{R}\ O_2$, it holds that $(x)(O_1|P')\ \mathcal{R}\ (x)(O_2|Q')$;*
5. *If $P \xrightarrow{a(\mathbf{c})} P'$, where c is a fresh name, then $Q \overset{a(\mathbf{c})}{\Longrightarrow} Q'$ for some Q', and $P'\ \mathcal{R}\ Q'$;*
6. *If $P \xrightarrow{(\widetilde{x})\overline{a}A} P'$, then $Q \overset{(\widetilde{y})\overline{a}B}{\Longrightarrow} Q'$ for some $\widetilde{y}, B, Q'$. And for a process $E[X]$ of the form $\overline{c}.(X+d)$, where c, d are fresh names, it holds that $(\widetilde{x})(E[A]|P')\ \mathcal{R}\ (\widetilde{y})(E[B]|Q')$.*

We say P is local linear variant bisimilar to Q, written $P \approx^{v}_{ll} Q$, if there exists some local linear variant bisimulation $\mathcal{R}$ such that $P\ \mathcal{R}\ Q$.

It can be shown, in a fashion similar to that of $\approx_{ll}$, that $\approx^{v}_{ll}$ is an equivalence relation and a congruence.

Theorem 9. *$\approx^{v}_{ll}$ is an equivalence relation and a congruence relation on all the calculus operators except the choice operator.*

Theorem 10. *Local linear variant bisimilarity coincides with local linear bisimilarity, that is $\approx_{ll} = \approx^{v}_{ll}$.*

Proof. We focus on the first-order bound output case in two bisimulations, since that is where the difference in the definitions lies. We recall the two clauses in each definition.

1. In $\approx_{ll}$: If $P \xrightarrow{\overline{a}(x)} P'$, then $Q \overset{\overline{a}(x)}{\Longrightarrow} Q'$ for some Q', and for every process O, $(x)(O|P') \approx_{ll} (x)(O|Q')$.
2. In $\approx^{v}_{ll}$: If $P \xrightarrow{\overline{a}(x)} P'$, then $Q \overset{\overline{a}(x)}{\Longrightarrow} Q'$ for some Q', and for all processes O_1 and O_2 such that $O_1 \approx^{v}_{ll} O_2$, $(x)(O_1|P') \approx^{v}_{ll} (x)(O_2|Q')$.

" $\subseteq$ ". This case is straightforward, since 1 is a special case of 2, by choosing $O_2 \equiv O_1$.
" $\supseteq$ ". This case is a little complex, in that it needs to exploit the congruence properties of $\approx^{v}_{ll}$ (Theorem 9), specifically the closure under parallel composition and restriction. We can define a relation: $\mathcal{R} \triangleq \{(P,Q) \mid P \approx^{v}_{ll} Q\} \cup \approx^{v}_{ll}$, and show $\mathcal{R}$ is a local linear bisimulation. In the first-order bound output case, we have $(x)(O_1|P') \approx^{v}_{ll} (x)(O_2|Q') \approx^{v}_{ll} (x)(O_1|Q')$, thanks to the congruence properties. Now we conclude that $\approx_{ll} = \approx^{v}_{ll}$. □

Remark. Notice that all the bisimulations we define till now are of late style in first-order bound output and higher-order output. We can define (respectively) the early ones on these clauses accordingly. Using a similar approach to that in [12], one can readily prove that the early versions coincide with the corresponding late versions. We will take advantage of this fact in the future work on logical characterization, as will be mentioned in the conclusion.

4 Conclusion

In this paper, starting from previous work on bisimulation theory of higher-order process calculi, we arrive at a recent result on bisimulation theory in linear higher-order π-calculus, which is proposed to reduce the power of higher-order calculi so that an equation system is guaranteed. Local bisimulation is an intuitively reasonable observational equivalence enjoying such characteristics as closure under substitution, equivalence, and congruence. By exploiting the properties of linear processes, we design two variants, which simplify local bisimulation and are coincident on bisimilarities. The first variant, called local linear bisimulation, simplifies the higher-order input and higher-order output simulation steps in local bisimulation through examining the essence in Abstraction Theorem and Concretion Theorem. The coincidence proof is non-trivial and new. The second variant, called local linear variant bisimulation, adjusts the first-order bound output in local bisimulation to make it more appropriate for some analysis like axiomatization and logical characterization, by making use of the congruence properties.

Future work

Some future work based on the result in this paper can be addressed. We mention several of them below.

- Recursion. Our calculus here is free of recursion operator or fix-point operator. This can provide us a complete axiom system, as shown in [12]. Albeit the inclusion of recursion would grant the calculus the power of Turing machines, it can enrich the diversity of the behavior of processes and the description capability, especially in cooperation with restriction and possibly relabelling which we do not include here either. We think that the inclusion of recursion would not shatter the main result in this paper, that is the simplification can still be obtained through a similar technical routine. The difference worth noticing is in the proof concerning process structures, where one shall not use induction on process structure any more, but induction on derivation height instead, because the recursion can increase the complexity of a process during transitions.
- Logical characterization. Another immediate yet important task starting off from the bisimulation theory in this paper is to achieve a logical characterization of local bisimulation, which can complement the algebraic theory and enable practical modeling and verification using LHOPi. Related work on logically characterizing bisimulations in higher-order process calculi is [19], where strong context bisimulation in higher-order π-calculus is characterized, and [20], where weak context bisimulation in higher-order π-calculus is characterized. The framework is likewise. We summarize a little the rough pattern of logical characterization.
  1. Target bisimulation in some process calculus;
  2. Its variant(s) tailored for logical characterization;
  3. The variant's coincidence with the original bisimulation on bisimilarities;
  4. The approximation of the variant bisimilarity using a chain of "bisimulations";
  5. The (modal) logic for characterizing the variant bisimulation;
  6. The characteristic formulas for aiding the proof of the characterization theorem;
  7. The characterization theorem, that is the coincidence between the bisimilarity and logical equivalence.

  A direct logical characterization of local bisimulation is possible by the results in [19] [20]. However since we are dealing with linear processes, the bisimulation is expected to enjoy a simpler form of logical characterization. The work in this paper indeed provides the simplification of local bisimulation. That is we can characterize local linear variant bisimulation instead. We recall that the simplification resides in higher-order input and output, but first-order bound output clause cannot be simplified, though some more desirable form is available. The existence of first-order bound output results in the necessity of using constructive implication in the logic.

  In summary, we can see that the results in this paper has settled several parts in the pattern above of logical characterization. The next task is to exercise the design of logic, for which we utilize related results and technique in [19] [20], where the main contribution is the constructive implication operator ($\Rightarrow$) that is used to specifies the property of a function process, which is a process with process variables appearing in it. For example, $\vDash P[X] : \phi \Rightarrow \phi'$ means that when inputted with a process R satisfying

ϕ, the obtained process $P\{R/X\}$ shall satisfy ϕ'. Moreover, to accomplish the logical characterization, one has to reformulate the calculus under a new framework to tailor the processes to be suitable for a logical description. And the reformulation must be equivalent to the original calculus in the sense of bisimulation. After all the preparation, we have to go through a number of technical steps to arrive at the characterization theorem that relates logical equivalence to bisimulation equivalence. The task is not so trivial. For now we think our logic may be composed of three parts:

1. Traditional parts from modal logic for first-order mobile processes, like those in [21];
2. Parts on constructive implication or something alike to handle first-order bound output;
3. Parts concerning the higher-order input and output based on the simplification in local linear variant bisimulation.

We work on this task in [22]. In order to prepare for future work and make the bisimulation theory more complete, we also adopt the traditional approach to approximate local bisimilarity in the appendix A.

References

1. Milner, R.: Communication and Concurrency. Prentice Hall (1989)
2. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes (parts i and ii). Information and Computation **100** (1992) 1–77
3. Sangiorgi, D., Walker, D.: The Pi-calculus: a Theory of Mobile Processes. Cambridge Universtity Press (2001)
4. Thomsen, B.: A calculus of higher order communication systems. In: Proceedings of POPL'89. (1989) 143–154
5. Thomsen, B.: Calculi for Higher Order Communicating Systems. Phd thesis, Department of Computing, Imperial College (1990)
6. Thomsen, B.: Plain chocs, a second generation calculus for higher-order processes. Acta Informatica **30** (1993) 1–59
7. Thomsen, B.: A theory of higher order communication systems. Information and Computation **116** (1995) 38–57
8. Sangiorgi, D.: Bisimulation for higher-order process calculi. Information and Computation **131(2)** (1996) 141–178 Preliminary version in proceedings PROCOMET'94 (IFIP Working Conference on Programming Concepts, Methods and Calculi), North Holland, 1994.
9. Sangiorgi, D.: Expressing Mobility in Process Algebras: First-order and Higher-order Paradigms. Phd thesis, University of Edinburgh (1992)
10. Sangiorgi, D.: A theory of bisimulation for π-calculus. Acta Informatica **33(1)** (1996) 69–97 An extended abstract in the proceedings of CONCUR '93, LNCS 715.
11. Sangiorgi, D., Walker, D.: On barbed equivalences in pi-calculus. In: Proceedings of CONCUR'01. Volume 2154 of LNCS. (2001) 292–304 Proceedings of CONCUR'01.
12. Fu, Y.: Checking equivalence for higher order processes. SJTU BASICS (2005)
13. Milner, R.: Functions as processes. Journal of Mathematical Structures in Computer Science **2(2)** (1992) 119–141 Research Report 1154, INRIA, Sofia Antipolis, 1990.
14. Milner, R., Sangiorgi, D.: Barbed bisimulation. In: Proceedings 19-the International Colloquium on Automata, Languages and Programming (ICALP '92). Volume 623 of LNCS., Springer Verlag (1992)
15. Baldamus, M., Frauenstein, T.: Congruence proofs for weak bisimulation equivalences on higher-order process calculi. Technical Report Report 95-21,, Berlin University of Technology, Computer Science Department (1995)
16. Astesiano, E., Giovini, A., Reggio, G.: Generalized bisimulation on relational specifications. In: Proceedings of Theoretical Aspects of Computer Science. Volume 294 of LNCS. (1988) 207–226
17. Ferreiram, W., Henessy, M., Jeffrey, A.: A theory of weak bisimulation for core cml. In: Proceedings of Functional Programming. (1996) 201–212
18. Frauenstein, T., Baldamus, M., Glas, R.: Congruence proofs for weak bisimulation on higher-order processes: Results for typed ω-order calculi. Technical Report 96-19, Berlin University of Technology, Computer Science Department (1996) Precursor report: M. Baldamus and T. Frauenstein. Congruence Proofs for Weak Bisimulation Equivalences on Higher-order Process Calculi, 1995.
19. Amadio, R., Dam, M.: Reasoning about higher-order processes. In: TAPSOFT95. Volume 915 of LNCS. (1995) 202–216
20. Baldamus, M., Dingel, J.: Modal characterization of weak bisimulation for higher-order processes. In: TAPSOFT97. Volume 1214 of LNCS. (97) 285–296
21. Milner, R., Parrow, J., D.Walker: Modal logics for mobile processes. Theoretical Computer Science **114(1)** (1993) 149–171 In 2nd CONCUR, volume 527, LNCS, pages 45-60, 1991.
22. Xu, X.: A logical characterization of local bisimulation in linear higher-order π-calculus. Technical report, BASICS Lab, SJTU (2007)

A Approximating local bisimilarity

In this section, we define a descending chain of “bisimulation” equivalence relations (indexed by ordinal k) to approximate the local linear variant bisimilarity. We include this as an extra credit on bisimulation theory to make it more complete, and moreover, it can serve as a basis for further work such as a logical characterization.

Below is the definition of the function $\mathcal{F}$ and a family of relations $\approx^k$ ($k \le \omega$ is an ordinal). We use $k, l..., \lambda, \kappa...$ for ordinals, I, J for index sets, and ω is the first transfinite ordinal.

Definition 5. *Define the function $\mathcal{F} : Pr_0^2 \to Pr_0^2$ as below: $P\ \mathcal{F}(\mathcal{R})\ Q$ if $\mathcal{F}(\mathcal{R})$ is closed under substitution of names and the following properties hold:*

1. *If $P \xrightarrow{\tau} P'$, then $Q \Longrightarrow Q'$ for some Q', and $P'\ \mathcal{R}\ Q'$;*
2. *If $P \xrightarrow{a(x)} P'$, then $Q \overset{a(x)}{\Longrightarrow} Q'$ for some Q', and $P'\ \mathcal{R}\ Q'$;*
3. *If $P \xrightarrow{\overline{a}x} P'$, then $Q \overset{\overline{a}x}{\Longrightarrow} Q'$ for some Q', and $P'\ \mathcal{R}\ Q'$;*
4. *If $P \xrightarrow{\overline{a}(x)} P'$, then $Q \overset{\overline{a}(x)}{\Longrightarrow} Q'$ for some Q', and for all (closed) processes O_1 and O_2 such that $O_1\ \mathcal{R}\ O_2$, $(x)(O_1|P')\ \mathcal{R}\ (x)(O_2|Q')$.*
5. *If $P \xrightarrow{a(\mathbf{c})} P'$, where c is a fresh name, then $Q \overset{a(\mathbf{c})}{\Longrightarrow} Q'$ for some Q', and $P'\ \mathcal{R}\ Q'$;*
6. *If $P \xrightarrow{(\widetilde{x})\overline{a}A} P'$, then $Q \overset{(\widetilde{y})\overline{a}B}{\Longrightarrow} Q'$ for some $\widetilde{y}, B, Q'$. And for a process $E[X]$ of the form $\overline{c}.(X{+}d)$, where c, d are fresh names, it holds that $(\widetilde{x})(E[A]|P')\ \mathcal{R}\ (\widetilde{y})(E[B]|Q')$.*

And vice versa.

Now the hierarchy $\approx^k$ ($k < \omega$ is an ordinal) can be defined as (λ is a transfinite ordinal):

$$\begin{aligned} \approx^0 &= Pr_0^2 \\ \approx^{k+1} &= \mathcal{F}(\approx^k) \\ \approx^\lambda &= \textstyle\bigcap_{k<\lambda} \approx^k \end{aligned}$$

The relation $\approx^k$ can be extended to open processes in the usual fashion.

Based on the definition above, we have the following two important propositions.

Proposition 1 ($\mathcal{F}$ properties). *Suppose $2^{Pr_0^2}$ is a complete lattice with the order of set inclusion on it. Then the following properties hold:*

1. *$\mathcal{F}$ is monotone. That is If $k < k'$, then $\approx^{k'} \subseteq \approx^k$.*
2. *$\mathcal{R}$ is a bisimulation iff $\mathcal{R} \subseteq \mathcal{F}(\mathcal{R})$;*
3. *If $\{X_i\}_{i\in I}$ is a codirected set, then $\mathcal{F}(\bigcap_{i\in I} X_i) = \bigcap_{i\in I} \mathcal{F}(X_i)$;*
4. *The greatest bisimulation $\approx_{ll}^v$ exists and it coincides with $\approx^\omega$. That is $\approx^\omega = \approx_{ll}^v$.*

Proof. The proof is of the traditional style like that in [19] [20] [1] [21]. As an example, we focus on 4 to show that $\approx^\omega = \approx_{ll}^v$. The existence of $\approx^\omega$ is not hard.

"$\supseteq$". By induction on the ordinal $k < \omega$. When $k = 0$, it is obvious that $\approx_{ll}^v \subseteq \approx^0$. Now suppose $\approx_{ll}^v \subseteq \approx^k$, we show that $\approx_{ll}^v \subseteq \approx^{k+1}$. Suppose $P \approx_{ll}^v Q$. We have the following analysis:

1. If $P \xrightarrow{\tau} P'$, then $Q \Longrightarrow Q'$ for some Q', and $P' \approx_{ll}^v Q'$;
2. If $P \xrightarrow{a(x)} P'$, then $Q \overset{a(x)}{\Longrightarrow} Q'$ for some Q', and $P' \approx_{ll}^v Q'$;
3. If $P \xrightarrow{\overline{a}x} P'$, then $Q \overset{\overline{a}x}{\Longrightarrow} Q'$ for some Q', and $P' \approx_{ll}^v Q'$;
4. If $P \xrightarrow{\overline{a}(x)} P'$, then $Q \overset{\overline{a}(x)}{\Longrightarrow} Q'$ for some Q', and for all (closed) processes O_1 and O_2 such that $O_1 \approx_{ll}^v O_2$, $(x)(O_1|P') \approx_{ll}^v (x)(O_2|Q')$.
5. If $P \xrightarrow{a(\mathbf{c})} P'$, where c is a fresh name, then $Q \overset{a(\mathbf{c})}{\Longrightarrow} Q'$ for some Q', and $P' \approx_{ll}^v Q'$;
6. If $P \xrightarrow{(\widetilde{x})\overline{a}A} P'$, then $Q \overset{(\widetilde{y})\overline{a}B}{\Longrightarrow} Q'$ for some $\widetilde{y}, B, Q'$. And for a process $E[X]$ of the form $\overline{c}.(X{+}d)$, where c, d are fresh names, it holds that $(\widetilde{x})(E[A]|P') \approx_{ll}^v (\widetilde{y})(E[B]|Q')$.

This suffices to show that $\approx_{ll}^v \subseteq \approx^{k+1}$, by induction hypothesis.

"$\subseteq$". We show that $\approx^\omega$ is a local linear variant bisimulation (through definition checking). Suppose $P \approx^\omega Q$, then for every $(k+1) < \omega$, $P \approx^{k+1} Q$. Thus we have the analysis below:

1. If $P \xrightarrow{\tau} P'$, then $Q \Longrightarrow Q'$ for some Q', and $P' \approx^k Q'$;
2. If $P \xrightarrow{a(x)} P'$, then $Q \overset{a(x)}{\Longrightarrow} Q'$ for some Q', and $P' \approx^k Q'$;
3. If $P \xrightarrow{\overline{a}x} P'$, then $Q \overset{\overline{a}x}{\Longrightarrow} Q'$ for some Q', and $P' \approx^k Q'$;
4. If $P \xrightarrow{\overline{a}(x)} P'$, then $Q \overset{\overline{a}(x)}{\Longrightarrow} Q'$ for some Q', and for all (closed) processes O_1 and O_2 such that $O_1 \approx^k O_2$, $(x)(O_1|P') \approx^k (x)(O_2|Q')$.

5. If $P \xrightarrow{a(\mathbf{c})} P'$, where c is a fresh name, then $Q \overset{a(\mathbf{c})}{\Longrightarrow} Q'$ for some Q', and $P' \approx^k Q'$;
6. If $P \xrightarrow{(\widetilde{x})\overline{a}A} P'$, then $Q \overset{(\widetilde{y})\overline{a}B}{\Longrightarrow} Q'$ for some $\widetilde{y}, B, Q'$. And for a process $E[X]$ of the form $\overline{c}.(X{+}d)$, where c, d are fresh names, it holds that $(\widetilde{x})(E[A]|P') \approx^k (\widetilde{y})(E[B]|Q')$.

In any case, P can be matched by Q, and it holds for every $k+1$. By this and a standard argument on ordinals, we conclude that $\approx^\omega$ is a local linear variant bisimulation.

Another is on the congruence property of $\approx^k$.

Proposition 2 (Congruence of $\approx^k$). *The relation $\approx^k$ $(k \leq \omega)$ is a congruence with respect to all the operators in the calculus except the choice operator. That is, suppose $P_i \approx^k Q_i$ $(i = 1, 2)$ and $P \approx^k Q$, then*

$$\begin{array}{l}
\tau.P \approx^k \tau.Q \\
c(x).P \approx^k c(x).Q, \quad \overline{c}x.P \approx^k \overline{c}x.Q \\
c(X).P \approx^k c(X).Q, \quad \overline{c}P_1.P_2 \approx^k \overline{c}Q_1.Q_2 \\
(x)P \approx^k (x)Q \\
P_1|P_2 \approx^k Q_1|Q_2
\end{array}$$

Proof. Routine check using similar approach to that in the proof of the congruence of $\approx^v_{ll}$.