Electronic Notes in Theoretical Computer Science 85 No. 1 (2003)
URL: http://www.elsevier.nl/locate/entcs/volume85.html 14 pages

Intensionality of Modal Logic for Robust Ambient Calculus

Taolue Chen [1,2]

State Key Laboratory of Novel Software Technology
Nanjing University
Nanjing, P.R.China

Tingting Han [3]

State Key Laboratory of Novel Software Technology
Nanjing University
Nanjing, P.R.China

Jian Lu [4]

State Key Laboratory of Novel Software Technology
Nanjing University
Nanjing, P.R.China

Abstract

Computation with mobility becomes a novel distributed computation paradigm with the development of network technology. The calculus of Mobile Ambient is a widely studied formal mechanism for describing both mobile computing and mobile computation. To improve this process calculus, Robust Mobile Ambient is introduced by providing co-actions. This paper deals with the modal logic for finite fragment of this calculus without communication primitives. In details, we compare standard behavioral equivalence with the equivalence induced by the logic ($=_L$) and with the structural congruence. As a result, we prove the intensionality of the logic and at the same time, an axiomatization for $=_L$ is given.
Key words: Concurrent theory, Process algebra, Mobile ambient, Modal logic, Intensionality

[1] Supported by NNSFC (No.60273034, No.60233010), the 863 Hi-Tech Project (No.2002AA116010), the 973 Program of China (No.2002CB312002)
[2] Email: ctl@softlab.nju.edu.cn
[3] Email: hantt@softlab.nju.edu.cn
[4] Email: lj@nju.edu.cn

1 Introduction

Computation with mobility is a kind of new distributed computation paradigm with the development of network technology. There are two distinct areas of work in mobility: mobile computing and mobile computation ([2]), the former concerns computation that is carried out in mobile devices based on wireless network while the latter concerns mobile code based on Web, e.g. applet, agent, etc. The calculus of mobile ambient ([2], MA in short), introduced by Cardelli and Gordon, is a formalism which can describe both of the two aspects of mobility within a single framework, and is one of the hottest topics in the area of mobile process calculi after π-calculus ([14]).

However, in [12], Levi and Sangiorgi have pointed out that there exists a kind of dangerous interference in the original ambient calculus, which shows some deficiency of original design for MA. To settle this problem, a new calculus called Mobile Safe Ambient ([12], SA in short) is presented in order to eliminate this kind of interferences by adding the corresponding co-action primitive to the original action primitives of MA. However, as pointed out by Guan et al ([7][8]), the co-actions introduced in SA are very vulnerable to tampering in that some malicious third parties may easily consume them. Thus a variant of SA, called Robust Ambient (ROAM in short), is introduced, which can explicitly name the ambient that can participate in the reduction and eliminate grave interference problem, too. At the same time, it allows a finer grained and more natural access control.

Recently, because of the deficiency of using algebra method to model and describe related properties of systems, e.g. mobility, safety, a lot of research has focused on modal logic of MA. Modal logic (temporal logic especially) is thought as a good compromise between description convenience and abstraction. In addition, many modal logics support useful computational applications, such as model-checking ([3]). In [3][5], Cardelli and Gordon introduce a (branch time) temporal logic called ambient logic. In contrast to traditional process logic, such as the well-known Hennessy-Milner logic ([10]), it has strong expressing power while only provides few modal operators; the ambient logic has also been advocated as a foundation of query language for semi-structured data ([1][5]). A lot of research has given insight into some properties of ambient logic itself and related model checking algorithms, such as [6][9][11][16].

However, within our knowledge, the research on modal logic for ROAM has not been reported. One work of this paper is to adapt the related work of Cardelli et al on MA to ROAM. We provide a modal logic for this important calculi, which is significant to utilize the control ability of ROAM better and model checking related problems.

For a logic system, extensionality and intensionality is a fundamental question, especially for process logic, which can tell us whether the logic fits the intended semantics of the calculus or model on which the logic is defined. A

logic, defined on the terms of a calculus or of a model, is extensional if it can only separate terms that have different behaviors: processes with the same behaviors satisfy the same sets of formulas; the logic is intensional if it can separate terms on the basis of their internal structure, even though their behaviors are the same ([16]). In the research on traditional modal logic, the topic is well understood, such as Hennessy-Milner logic for CCS ([13]), the discussion can be found in [10][13]. In [16], Sangiorgi deals with this problem in the setting of a fragment of MA. However in ROAM, it is not immediately clear whether his result can be adapted, which is mainly due to two reasons. First, in [16], the calculus has no restriction operators. Correspondingly, modality which is used to handle restriction has not been considered in the logic. Second, the co-action is introduced in ROAM while it is not in MA, thus is not considered in [16]. The main work of this paper is to study the extensionality and intensionality of modal logic for the finite fragment of pure ROAM. Since in contrast, the conclusion for the former is obvious, we focus on the latter. In detail, we compare three relations on process: the equivalence induced by the ambient logic $=_L$, behavioral equivalence and structural congruence. It is worth pointing out that the calculus studied in this paper is so-called pure calculus, i.e. it does not contain communication primitives. Our starting point lies in that on one hand, in theory this kind of calculus has adequate expressing power, and can encode synchronous π-calculus ([7]); on the other hand, using existing conclusion, it is not difficult to extend our results. We defer detailed explanations to Section 5; at the same time, we exclude recursive or replicate operator in the calculus, due to a lack of mature method dealing with recursive operator in modal logic. In the rest of this paper, except pointing out otherwise, the ROAM refers to finite fragment of pure calculus.

The rest of this paper is organized as follows: Section 2 introduces the syntax and semantics of ROAM in brief and presents the modal logic for it; Section 3 gives the co-inductive characterization for $=_L$ by introducing two congruence relations; Section 4 compares three relations of process, and proves the intensionality of the logic. As a byproduct, it also gives the axiomatization for $=_L$; Section 5 concludes the paper and discusses the related work.

2 ROAM and Modal Logic

In this section, we review the syntax and semantics for ROAM in brief and will present the modal logic.

2.1 Syntax and Semantics

2.1.1 Syntax

We assume that $\mathcal{N}$ is countable name set and is ranged over by m, n. Π is process set (in general, ranged by P, Q) and is recursively defined as follows:

$$P ::= 0|(\nu n)P|P|Q|M.P|n[P]$$

where, M is element of capability sets (denoted by CAP) which is recursively defined as:

$$M ::= in\langle n\rangle|out\langle n\rangle|open\langle n\rangle|\overline{in}\langle n\rangle|\overline{out}\langle n\rangle|\overline{open}$$

where, $n \in \mathcal{N}$. Due to space restriction, we refer reader to [7][8] for the intended meaning of these operators. As in common process calculi, $(\nu n)P$ introduces the distinction of bound names and free names. In common, we use $fn(P)$ to denote the set of free names appearing in process P; for capability M, every name appearing in it is a free name. A process P is called closed if $fn(P) = \emptyset$. In general, we denote that P is a process expression by $P : \Pi$. Note that the well-known α-convention is used implicitly so as to avoid name capturing.

2.1.2 Reduction Semantics

The reduction relation $\rightarrow$ of processes is given by the following reduction rules:

(R-in)

$$n[\overline{in}\langle m\rangle.P_1|P_2]|m[in\langle n\rangle.Q_1|Q_2] \rightarrow n[P_1|P_2|m[Q_1|Q_2]]$$

(R-out)

$$n[\overline{out}\langle m\rangle.P_1|P_2|m[out\langle n\rangle.Q_1|Q_2]] \rightarrow n[P_1|P_2]|m[Q_1|Q_2]$$

(R-open)

$$open\langle n\rangle.P|n[\overline{open}.Q_1|Q_2] \rightarrow P|Q_1|Q_2$$

(R-Par)

$$\frac{P \rightarrow P'}{P|Q \rightarrow P'|Q}$$

(R-Res)

$$\frac{P \rightarrow P'}{(\nu n)P \rightarrow (\nu n)P'}$$

(R-Amb)

$$\frac{P \rightarrow P'}{n[P] \rightarrow n[P']}$$

(R-$\equiv$)

$$\frac{P \equiv P' \quad P' \rightarrow P'' \quad P'' \equiv P'''}{P \rightarrow P'''}$$

As usual, pure reduction semantics can't describe process behavior completely, so we introduce structural congruence, denoted by $\equiv$, which is defined

as the smallest binary relation satisfying the following rules:

(Struct Ref)	P	$\equiv$	P	
(Struct Symm)	Q	$\equiv$	P	if $P \equiv Q$
(Struct Trans)	P	$\equiv$	R	if $P \equiv Q$ and $Q \equiv R$
(Struct Res)	$(\nu n)P$	$\equiv$	$(\nu n)Q$	if $P \equiv Q$
(Struct Par)	$P\|R$	$\equiv$	$Q\|R$	if $P \equiv Q$
(Struct Amb)	$m[P]$	$\equiv$	$m[Q]$	if $P \equiv Q$
(Struct Act)	$M.P$	$\equiv$	$M.Q$	if $P \equiv Q$
(Struct Par Comm)	$P\|Q$	$\equiv$	$Q\|P$	
(Struct Par Assoc)	$(P\|Q)\|R$	$\equiv$	$P\|(Q\|R)$	
(Struct Res Res)	$(\nu n)(\nu m)P$	$\equiv$	$(\nu m)(\nu n)P$	
(Struct Res Par)	$(\nu n)(P\|Q)$	$\equiv$	$P\|(\nu n)Q$	if $n \notin fn(P)$
(Struct Res Amb)	$(\nu n)m[P]$	$\equiv$	$m[(\nu n)P]$	if $n \neq m$
(Struct Zero Par)	$P\|0$	$\equiv$	P	
(Struct Zero Res)	$(\nu n)0$	$\equiv$	0	

we also refer reader to [7][8] for intuition explanation. In general, we use $\Rightarrow$ to denote the reflective and transitive closure of $\rightarrow$.

2.1.3 Behavior equivalence relation

In the research of equation theory for ROAM, the notion of behavior equivalence can be formalized in two ways: one is the context equivalence used in [7] which is based on testing, the other is barbed congruence based on context ([15]). In fact, in the context of ROAM, the latter is stricter than the former, and has clearer intuition sense and neater proof. Due to these advantages, we regard it as our definition for behavior equivalence. Further more, due to Lemma 2.5 and results of Section 4 in this paper, in contrast to using context equivalence in [7] as the definition for $\approx$, our conclusion is stronger.

Definition 2.1(i) For name n, $P \downarrow_n$ if there exists $\tilde{p}, M, P_1$, such that $M \in \{in\langle n\rangle, \overline{in}\langle n\rangle, out\langle n\rangle, \overline{open}|n \in \mathcal{N}\}$, $(fn(M) \cup \{n\}) \cap \{\tilde{p}\} = \emptyset$, and $P \equiv (\nu\tilde{p})(n[M.P_1|P_2]|P_3)$.

(ii) $P \Downarrow_n$ if there exists P' such that $P \Rightarrow P'$ and $P' \downarrow_n$.

Definition 2.2 (**Barbed bisimulation**)

Barbed bisimulation is the largest symmetric relation on Π, denoted by $\dot{\approx}$,

which satisfies: if $P \dot{\approx} Q$, then

(i) If $P \Rightarrow P'$, then Q' exits, such that $Q \Rightarrow Q'$ and $P' \dot{\approx} Q'$.

(ii) For any name n, $P \Downarrow_n$ if and only if $Q \Downarrow_n$.

Definition 2.3 (**Barbed congruence**)

Process P, Q are barbed congruence, denoted by $P \approx Q$, if for any context $C[.]$, $C[P] \dot{\approx} C[Q]$. Where, the definition for context is standard in literature.

Definition 2.4 (**Context equivalence**, c.f. [7])

Process P, Q are context equivalence, denoted by $P \approx_c Q$, if for any context $C[.]$ and ambient name n, $C[P] \Downarrow_n \Leftrightarrow C[Q] \Downarrow_n$.

The following lemma can be derived directly from definition:

Lemma 2.5 $\approx \subseteq \approx_c$.

2.2 Modal logic

In this section, based on the results of [3][4], we introduce a modal logic and refer the reader to above literatures for intended meanings.

2.2.1 Logic formula

We assume countable infinite variable set $\mathcal{V}$, in general, $\mathcal{V} = \{x, y, z, \cdots\}$. Note that it is different from name set $\mathcal{N}$ for process expression. In general, we assume that $\mathcal{V} \cap \mathcal{N} = \emptyset$ and η ranges over $\mathcal{V}$ or name set $\mathcal{N}$. The well-formed formula of modal logic is defined by BNF as follows:

$$A, B ::= T | \neg A | A \vee B | 0 | A | B | A \triangleright B | \eta[A] | A@\eta | \eta \circledR A | A \oslash \eta | \forall x.A | \diamond A$$

We denote the set of all logic formulas by $\emptyset$. Note that for a formula A, the notion of bound names is of no sense, but because of the existence of $\forall x.A$, we must introduce the notion of bound variable, which coincides with the traditional definition. In this paper, we denote the free variable and bound variable of formula A by $fv(A)$ and $bv(A)$ respectively. A formula is closed if $fv(A) = \emptyset$.

It is worth noting that the logic introduced in this section is actually identical to the original *Ambient Logic*([3][4]), since the difference in operational semantics on ROAM is all hidden by the temporal diamond operator $\diamond$.

2.2.2 Satisfy relation

In general, the semantics of modal logic can be defined by satisfy relation $P \models A$ as follows, where A is closed.

$$\begin{aligned}
P &\models \quad T \\
P &\models \quad \neg A \quad \stackrel{def}{=} \neg(P \models A) \\
P &\models A \vee B \stackrel{def}{=} P \models A \vee P \models B \\
P &\models \quad 0 \quad \stackrel{def}{=} P \equiv 0
\end{aligned}$$

$$
\begin{aligned}
P \models\ & A|B \stackrel{def}{=} \exists P_1, P_2.P \equiv P_1|P_2 \wedge P_1 \models A \wedge P_2 \models B \\
P \models\ & A \triangleright B \stackrel{def}{=} \forall R.R \models A \Rightarrow R|P \models B \\
P \models\ & n[A] \stackrel{def}{=} \exists P_1.P \equiv n[P_1] \wedge P_1 \models A \\
P \models\ & A@n \stackrel{def}{=} n[P] \models A \\
P \models\ & n®A \stackrel{def}{=} \exists P_1.P \equiv (\nu n)P_1 \wedge P_1 \models A \\
P \models\ & A \oslash n \stackrel{def}{=} (\nu n)P \models A \\
P \models\ & \forall x.A \stackrel{def}{=} \forall n.n \in \mathcal{N} \Rightarrow P \models A\{n/x\} \\
P \models\ & \diamond A \stackrel{def}{=} \exists P_1.P \Rightarrow P_1 \wedge P_1 \models A
\end{aligned}
$$

As in [3], we have:

Lemma 2.6 *If $P \equiv Q$ and $P \models A$, then $Q \models A$.*

Definition 2.7 (**Equivalence induced by logic** $=_L$) For any process P, Q, $P =_L Q$, if for any closed formula A, $P \models A$ if and only if $Q \models A$.

For convenience, we introduce the some dual modalities. They also appear in standard modal logic.

$$
\begin{aligned}
A \propto B &\stackrel{def}{=} \neg(A \triangleright \neg B) \\
\circ A &\stackrel{def}{=} \neg \diamond \neg A \\
\exists x.A &\stackrel{def}{=} \neg(\forall x.\neg A) \\
A \wedge B &\stackrel{def}{=} \neg(\neg A \vee \neg B)
\end{aligned}
$$

In intuition, $P \models A \propto B$, if there exists process R, $R \models A$, $P|R \models B$; $P \models \circ A$, if for any process P', $P \Rightarrow P'$, $P' \models A$. The other two dual modalities are the standard forms in first order logic, we omit the explanations.

3 Co-inductive Characterization for $=_L$

As mentioned above, we need to compare three equivalence relations on process. However, if we use the definition for $=_L$ directly, it is difficult to attain our goal. Under this circumstance, we wish to give the co-inductive characterization for $=_L$. In this section, inspired by [16], we will give two bisimulation relations which coincide with $=_L$.

3.1 Intension Bisimulation

Definition 3.1 Let P be a process, then

(i) $P \xrightarrow{\mu} P'$, if $P \equiv \mu.P_1|P_2$ and $P' \equiv P_1|P_2$, where $\mu \in CAP$.

(ii) $P \stackrel{\mu}{\Rightarrow} P'$, if $P \Rightarrow \stackrel{\mu}{\rightarrow} \Rightarrow P'$.

Definition 3.2 (**Intension bisimulation**) Intension bisimulation, denoted by $\cong$, is the largest symmetric relation on processes satisfying the following conditions: if $P \cong Q$, then

(i) If $P \equiv P_1|P_2$, then there exists Q_1, Q_2, such that $Q \equiv Q_1|Q_2$ and $P_i \cong Q_i$, where $i = 1, 2$.

(ii) If $P \equiv 0$, then $Q \equiv 0$.

(iii) If $P \rightarrow P'$, then Q' exists, such that $Q \Rightarrow Q'$ and $P' \cong Q'$.

(iv) If $P \stackrel{\mu}{\rightarrow} P'$, then Q' exists, such that $Q \stackrel{\mu}{\Rightarrow} Q'$ and $P' \cong Q'$, where $\mu \in CAP$.

(v) If $P \equiv n[P']$, then Q' exists, such that $Q \equiv n[Q']$ and $P' \cong Q'$.

(vi) If $P \equiv (\nu n)P'$, then Q' exists, such that $Q \equiv (\nu n)Q'$ and $P' \cong Q'$.

In order to prove the congruence property of intension bisimulation, we introduce another characterization denoted by $\cong'$ for $=_L$.

Definition 3.3 (**Variant of intension bisimulation**) Variant of intension bisimulation, denoted by $\cong'$, is the largest symmetric relation on processes satisfying the following conditions: if $P \cong' Q$, then

(i) If $P = P_1|P_2$, then there exists Q_1, Q_2, such that $Q \equiv Q_1|Q_2$ and $P_i \cong' Q_i$, where $i = 1, 2$.

(ii) If $P = 0$, then $Q \equiv 0$.

(iii) If $P = \mu.P'$, then Q' exists, such that $Q \stackrel{\mu}{\Rightarrow} Q'$ and $P' \cong' Q'$, where $\mu \in CAP$.

(iv) If $P = n[P']$, then Q' exists, such that $Q \equiv n[Q']$ and $P' \cong' Q'$.

(v) If $P = (\nu n)P'$, then Q' exists, such that $Q \equiv (\nu n)Q'$ and $P' \cong' Q'$.

In the below, we will prove that the two relations coincide. We use the same way as [16]. First, we give some lemmas whose proofs are trivial. Due to space restriction, we omit the details.

Lemma 3.4 *If $P \cong' Q$, then for any context $C[.]$, $C[P] \cong' C[Q]$.*

Proof. Induction on the structure of $C[.]$. □

Lemma 3.5 *If $P \equiv P' \cong' P''$, then $P \cong' P''$.*

Proof. Induction on the derivation of $P \equiv P'$. □

Lemma 3.6 *If $P \cong' Q$ and $P \rightarrow P'$, then there exists Q', such that $Q \Rightarrow Q'$ and $P' \cong' Q'$.*

Proof. Induction on the derivation of $P \rightarrow P'$. □

Based on the above work, it is easy to show the following theorem.

Theorem 3.7 *$\cong = \cong'$, and they are congruence relations.*

3.2 Co-inductive Characterization

In this section, we will show that the intention bisimulation introduced in Section 3.1 coincides with $=_L$.

Theorem 3.8 *If $P \cong Q$, then $P =_L Q$.*

Proof. It is enough to show that if $P \cong Q$, then $P \models A$ if and only if $Q \models A$, which can be shown by induction on the structure of formula A. □

It is rather difficult to prove the converse of Theorem 3.8. In standard modal logic, there are abound temporal modalities. However, in our logic system, we don't introduce corresponding modality for each capability, which may make it difficult to use some standard techniques (e.g. the proof for Hennessy-Milner logic characterizes the strong bisimulation of CCS process [12][15]). To settle this problem, we define this kind of modality in our logic first, and then standard techniques can be used. The following definition is taken from [16].

Definition 3.9(i) Process P is **non-trivial**, if $\neg(P \equiv 0)$.

(ii) Process P is **single threaded** , if for some P', $P \equiv P'$, the outmost operator of P' is not parallel $|$.

Obviously, we have

(i) Process P is **non-trivial** if and only if $\neg(P \models 0)$.

(ii) Process P is **single threaded**, if and only if $P \models 1comp \stackrel{def}{=} \neg(\neg 0|\neg 0)$.

Definition 3.10(i) $1proc \stackrel{def}{=} 1comp \wedge \neg\exists x.x[T]$.

(ii) $\langle\overline{open}\rangle.A \stackrel{def}{=} \forall x.((1proc \propto \diamond\neg x[T] \wedge A)@x) \wedge 1proc.$

(iii) $\langle open\langle n\rangle\rangle.A \stackrel{def}{=} \forall x.(n[x[0]|\langle\overline{open}\rangle] \triangleright \diamond(x[0]|A)) \wedge 1proc.$

(iv) $\langle\overline{in}\langle n\rangle\rangle.A \stackrel{def}{=} \forall x((n[1proc] \propto \diamond x[n[T]|A])@x) \wedge 1proc.$

(v) $\langle in\langle n\rangle\rangle.A \stackrel{def}{=} \forall x.((n[\langle\overline{in}\langle x\rangle\rangle] \triangleright \diamond n[x[A]])@x) \wedge 1proc.$

(vi) $\langle\overline{out}\langle n\rangle\rangle.A \stackrel{def}{=} \forall x.(n[1proc] \propto \diamond(x[T]|n[A])@x) \wedge 1proc.$

(vii) $\langle out\langle n\rangle\rangle.A \stackrel{def}{=} \forall x.((\diamond(x[A]|n[\overline{out}\langle n\rangle]@n@x) \wedge 1proc.$

Lemma 3.11 *$P \models \langle\mu\rangle.A$ if and only if there exists P', P'' such that $P \equiv \mu.P'$, $P' \Rightarrow P''$ and $P'' \models A$, where $\mu \in CAP$.*

Proof. (*Sketch*) Due to space restriction, we only choose a pair of cases to prove, e.g. $\overline{open}.A$, and $\langle open\langle n\rangle\rangle.A$. Define

$$\langle\overline{open}\rangle \stackrel{def}{=} \forall x.((1proc \propto \diamond\neg x[T])@x) \wedge 1proc$$

It is enough to show that $P \models \langle\overline{open}\rangle$ if and only if there exits P' such that $P \equiv \overline{open}.P'$.

By the definition of satisfy relation, we have: if $P \models 1proc$, then P is single threaded and is not an ambient, so P is only structure congruence to 0, $\overline{open}.P'$, $open\langle n\rangle.P'$, $\overline{in}\langle n\rangle.P'$, $in\langle n\rangle.P'$, $\overline{out}\langle n\rangle.P'$, $out\langle n\rangle.P'$. Think of the remainder of formulas, we have: for any x, there exists Q such that Q is single threaded and not an ambient, and $x[P]|Q$ can reduce to an ambient containing no x, so we can conclude that there exists P' such that $P \equiv \overline{open}.P'$. In the same way, define $\langle open\langle n\rangle\rangle \stackrel{def}{=} n[x[0]|\langle\overline{open}\rangle] \propto \diamond(x[0]|T) \wedge 1proc$. It is enough to show $P \models \langle open\langle n\rangle\rangle$ if and only if there exists P', such that $P \equiv open\langle n\rangle.P'$. By the definition of satisfy relation, we have: if $P \models 1proc$, then P is single threaded and not an ambient, so P can only be structure congruence to 0, $\overline{open}.P'$, $open\langle n\rangle.P'$, $\overline{in}\langle n\rangle.P'$, $in\langle n\rangle.P'$, $\overline{out}\langle n\rangle.P'$, $out\langle n\rangle.P'$. Think of the remainder of formula, we have: there exists Q, $Q \models \langle\overline{open}\rangle$, and $P|n[x[0]|Q]$ can reduce to a process R such that $R \models x[0]$, and then we can conclude that $P \equiv open\langle n\rangle.P'$ for some P'.

The remainder of the theorem can be proved in the same way, here, we give some important definitions used by the proof.

$\langle\overline{in}\langle n\rangle\rangle \stackrel{def}{=} (n[1proc] \propto \diamond x[n[T]|A])@x \wedge 1proc.$

$\langle in\langle n\rangle\rangle \stackrel{def}{=} (n[\langle\overline{in}\langle x\rangle\rangle] \propto \diamond n[x[A]])@x \wedge 1proc.$

$\langle\overline{out}\langle n\rangle\rangle \stackrel{def}{=} (n[1proc] \propto \diamond x[T]|n[A])@x \wedge 1proc.$

$\langle out\langle n\rangle\rangle \stackrel{def}{=} (\diamond(x[A]|n[\overline{out}\langle x\rangle]))@n@x \wedge 1proc$

□

Based on the above work, we can apply some standard proof techniques ([13]).

Definition 3.12 (1) $\cong'_0 = \Pi \times \Pi$.

(2) $P \cong'_{i+1} Q$ iff

(i) If $P = P_1|P_2$, then there exists Q_1, Q_2, such that $Q \equiv Q_1|Q_2$ and $P_1 \cong'_k Q_1$, $P_2 \cong'_l Q_2$, where $i = max(k, l)$.

(ii) If $P = 0$, then $Q \equiv 0$.

(iii) If $P = \mu.P'$, then there exists Q', such that $Q \stackrel{\mu}{\Rightarrow} Q'$ and $P' \cong'_i Q'$, where $\mu \in CAP$.

(iv) If $P = n[P']$, then there exists Q', such that $Q \equiv n[Q']$ and $P' \cong'_i Q'$.

(v) If $P = (\nu n)P'$, then there exists Q', such that $Q \equiv (\nu n)Q'$ and $P' \cong'_i Q'$

(3) $\cong'_i = \bigcap_{j<i} \cong'_j$.

The following lemma can be proved easily:

Lemma 3.13 $\cong' = \bigcap_j \cong'_j$.

Due to Definition 3.10, we can use some special logic formula to discriminate processes. we will abuse the notion of formula a little which is defined as $A, B ::= 0|\langle\mu\rangle.A|A|B|\eta[A]|\eta®A$. Note that the formula can be derived from the original modal logic in Section 2. So we don't extend the syntax of modal

logic.

Definition 3.14 (1) The depth of formula A can be defined recursively:

$$\begin{cases} dep(0) & = 0 \\ dep(\langle\mu\rangle.A) & = dep(A)+1 \quad \mu \in CAP \\ dep(A|B) & = max(dep(A), dep(B))+1 \\ dep(\eta[A]) & = dep(A)+1 \\ dep(\eta \circledR A) & = dep(A)+1 \end{cases}$$

(2) $PL_k \stackrel{def}{=} \{A | dep(A) \leq k\}$

Lemma 3.15 *For any $A \in PL_k$, if $P \models A \Leftrightarrow Q \models A$, then $P \cong'_k Q$.*

Proof. Induction on k. □

Theorem 3.16 *If $P =_L Q$, then $P \cong Q$.*

Proof. Note that $PL \stackrel{def}{=} \bigcup_k PL_k$. By Lemma 3.13 and Lemma 3.15, it can be easily obtained. □

By Theorem 3.7, Theorem 3.8 and Theorem 3.16, we can conclude that intension bisimulation characterizes $=_L$.

4 Comparison of Three Relations

In this section, we will compare three relations on process: the equivalence induced by the ambient logic $=_L$, behavioral equivalence $\approx$ and structural congruence $\equiv$. Since the relation of $\approx$ and $=_L$ is easily derived, we focus the discussion on the relation between $=_L$ and $\equiv$.

Theorem 4.1 $=_L \subset \approx$.

Proof. It is easy to prove that $\cong \subseteq \approx$. By Theorem 3.16, $=_L \subseteq \approx$. Think of $P = in\langle n\rangle.in\langle m\rangle$ and $Q = in\langle n\rangle | in\langle m\rangle$, obviously, $P \neq_L Q$, however, $P \approx Q$, so the inclusion relation is strict. □

In the below, we will discuss the relation between $=_L$ and $\equiv$.

Lemma 4.2 *If $P \cong Q$, then $Prefix(P) = Prefix(Q)$, where $Prefix(P)$ is the number of prefix in process P.*

Proof. By induction on the structure of P, it is easy to show that $Prefix(P) > Prefix(Q)$ will lead to contradiction. We omit the details. □

The intuition of above lemma lies in that in the definition of $\cong$, the weak bisimulation $\stackrel{\mu}{\Rightarrow}$ can only be $\stackrel{\mu}{\rightarrow}$, namely in essence it is a strong bisimulation. Strictly speaking, we have the following lemma, which can be derived from Lemma 4.2 directly.

Lemma 4.3 *If $P \cong Q$ and $P \xrightarrow{\mu} P'$, then Q' exists, such that $Q \xrightarrow{\mu} Q'$ and $P' \cong Q'$, where $\mu \in CAP$.*

Lemma 4.4 *If $P \cong Q$, then $P \equiv Q$.*

Proof. Note that by Theorem 3.7, we have $P \cong' Q$. Apply induction on the structure of P, we only give the proof for one case. Suppose $P = \mu.P'$, by definition, there exists Q', Q'', Q''', such that $Q \Rightarrow Q' \equiv \mu.Q''$, $Q'' \Rightarrow Q''', Q''' \cong' P'$. By Lemma 4.3, we have $Q = Q'$ and $Q'' = Q'''$. Then by induction hypothesis, the proposition is correct. □

Theorem 4.5 $=_L = \equiv$

Proof. It is enough to show that $\cong = \equiv$. Obviously, $\equiv \subseteq \cong$ and by Lemma 4.4, $\cong \subseteq \equiv$. The proof is complete. □

Now, we can conclude that $\equiv = =_L \subset \approx$, which shows that the modal logic introduced in this paper is an intensional logic. As a byproduct, we give an axiomatization for $=_L$, which is the structural congruence rules in Section 2.1.

5 Conclusion and Related Work

We summarize our main contributions as follows: in the setting of finite fragment of pure ROAM, a modal (temporal) logic is introduced; at the same time, three relations on process, that is, the equivalence induced by the ambient logic $=_L$, behavior equivalence $\approx$ and structural congruence $\equiv$ are compared. We conclude that $\equiv = =_L \subset \approx$. This result tells us that the logic introduced in this paper is intensional, which is just what we want. In the proof for this result, we find that although the logic only contains a few modalities, the standard modality corresponding to process action in common process logic (e.g. Hennessy-Milner Logic) can be defined in the system, which shows the strong expressing power of our logic system. Besides that, the axiomatization for $=_L$ is given as a byproduct. The proof in this paper follows the pattern of [16], however, we focus on handling restriction operator and co-actions. In contrast to [16], some definitions are neater due to the introduction of co-actions, and the restriction operator needs careful attentions. The main result of this paper strengthens the findings of Sangiorgi ([16]) significantly, and shows the robustness against variations of the mobile ambient calculus.

The related works on this paper are mainly [3][4][7][8][9][16]. [7][8] give detailed discussion on type system and algebra theory of ROAM. [3][4] introduce modal logic called ambient logic in the setting of mobile ambient. Because the form of this logic has noting to do with concrete actions (capability), our modal logic has the same form as theirs. [16] deals with the intensionality and extensionality of ambient logic while [9] deals with the property and axiomatization of $=_L$ for two Turing complete sub-calculus. The main idea and method are inspired by [16], however, different from it, the robust ambient calculus introduces co-actions, which brings the difficulties for defining extended

modalities. We settled this problem in the setting of ROAM; further more, in [16], the restriction operator is not considered, we study this operator based on [4], which can extend the result of [16] to the full finite calculus. Besides that, due to the similarity of ROAM and SA, our results can be easily adapted to the setting of safe ambient calculus (SA [12]).

As mentioned in Section 1, we don't introduce communication primitives. Due to the similarity of communication mechanism in ROAM and MA, we can apply the method used in [16] and obtain the similar result. Since the goal of this paper is to settle the difficulties induced by the restriction operator and co-actions, we do not report the trivial extension.

In the setting of SA, Merro and Hennessy also give a characterization of barbed congruence ([11]), since the SA and ROAM both have co-actions, it is sensible to compare their techniques and results with ours. And the work in [11] is in the **weak** setting, there is an obvious question that how much of the work in this paper can also be carried over to that setting. We leave it as our future work. Besides that, [3][6] introduce model-checking algorithm in the setting of ambient logic, which can be adapted to modal logic discussed in this paper directly. However, how to improve the efficiency of the algorithm according to the character of our logic is worth studying further.

Acknowledgement

The authors thank the three referees for many constructive suggestions and comments.

References

[1] L.Cardelli. Semistructured computations. Proc.7th Intl. Workshop on Data Base Programming Languages, 1999.

[2] L.Cardelli, A.Gordon. Mobile ambients. Theoretical Computer Science, 240(2000), pp.177-213, 2000.

[3] L.Cardelli, A.Gordon. Anytime, Anywhere: Modal logics for mobile ambients. POPL'2000, pp.365-377. ACM Press, 2000.

[4] L.Cardelli, A.Gordon. Logical properties of name restriction. Typed Lambda Calculi and Applications, volume 2044 of LNCS, Springer-Verlag, 2001.

[5] L.Cardelli, G.Ghelli. A query language for semistructured data based on the ambient logic ESOP'2001, volume 2028 of LNCS, pp.1-22. Springer-Verlag, 2001.

[6] W.Charatonik, J.Talbot. The decidability of model checking mobile ambients. 15th Annual Conf of European Association for Computer Science Logic. Volume 2142 of LNCS, pp.339-354. Springer-Verlag, 2001.

[7] X.Guan. Type system and algebraic theory of Robust Ambiens, Phd thesis, ShangHai Jiaotong University, 4,2002.

[8] X.Guan, Y.Yang, J.You. Typing evolving ambients. Information Processing Letters, 80(5), pp.265-270, Nov.2001.

[9] D.Hirschkoff, E.Lozes, D.Sangiorgi. Separability, expressiveness, and decidability in the ambient logic. LNCS, Springer-Verlag, 2002.

[10] M.Hennessy, R.Milner. Algebraic laws for nondeterminism and concurrency. Journal of the ACM, 32:137-161, 1985.

[11] M.Merro and M.Hennessy. Bisimulation Congruence in Safe Ambients. In Proc.POPL'02, ACM Press, 2002.

[12] F.Levi, D.Sangiorgi. Controlling interference in ambients. In Proc. POPL'00, pp.352-364, Boston, Massachusetts, 2000.

[13] R.Milner. Communication and Concurrency, Prentice Hall, 1989.

[14] R.Milner, J.Parrow, D.Walker. A Calculus of Mobile Process, part I/II. Journal of Information and Computation, 100:1-77,Sept.1992.

[15] R.Milner, D.Sangiorgi. Barbed bisimulation. In W.Kuich, editor, Proceedings of ICALP'92, volume 623 of LNCS, pp.685-695, Springer-Verlag, 1992.

[16] D.Sangiorgi. Extensionality and intensionality of the ambient logic. POPL'2001, pp.4-13, ACM Press, 2001.