数理解析研究所講究録
976巻 1997年 146-168

Semantic Characterizations for Reachability and Trace Equivalence in a Linear Logic-Based Process Calculus (Preliminary Report)

(岡田 光弘) (照井 一成)
Mitsuhiro Okada Kazushige Terui
Department of Philosophy, Keio University

Abstract

We give semantic characterizations for reachability and trace equivalence in a version of asynchronous process calculus based on linear logic.

Usually the *reachability relation* in linear logic-based process calculi is characterized by the logical notion of provability, which is in turn characterized by model-theoretic semantics such as phase semantics. We introduce considerably simplified phase models, which we call *naive phase models*, and show that reachability is also characterized by the completeness with respect to the naive phase models.

On the other hand, logical provability does not provide any satisfactory notion of equivalence on processes. We consider the *trace equivalence* (Hoare[7]) on our process calculus and introduce certain algebraic models, which we call *trace models*. Then the trace equivalence is characterized by the completeness with respect to the trace models.

§1 Introduction

We investigate a version of asynchronous process calculus based on linear logic. In our framework, formulas are identified with processes and inference rules are identified with actions in terms of "message passing"-based process calculi. Then a bottom-up proof construction of a formula A is naturally interpreted as a computation of the process A (cf. §2).

Under these identifications, various notions which have been discussed in the framework of process calculi are brought into logical study. This paper attempts to give a logical analysis to these new notions from process calculi in the framework of traditonal model-theoretic semantics.

Usually the logical notion of provability captures the *reachability relation* from the inputs to the outputs. On the other hand, the provability is characterized by logical semantics, eg. phase semantics, via the completeness theorem in the traditional framework of logic. However, the usual logical semantics complete for full linear logic, eg. phase semantics, is of rather abstract nature; such a semantics interpretes a formula on a certain model-theoretic domain, but the interpretation of a formula is usually so complicated that one could hardly catch any intuitive meaning of a formula from such a semantics. For example, phase semantics requires a certain closure condition to be complete with respect to full linear logic, and requires a formula to be interpreted by a certain "closed" set, called a *fact* (cf. §3.1), which makes the intuitive meaning of the formula ambiguous.

Our theory of process calculus uses only a very restricted fragment, essentially Horn fragment of linear logic. Hence there is a possibility to obtain simpler semantics that is complete with respect to the framgent. In §3.2, we introduce a simplified semantics for the Horn fragment, called *naive phase semantics*, which is obtained from phase semantics by dropping the closure condition. Naive phase semantics gives more intuitive meaning of formula than the original phase semantics. Our first main result says (in §3.2);

1. The reachability relation is characterized by the completeness with respect to the naive phase models.

The next problem which we address in this paper is to characterize certain notion of equivalence on processes in the traditional framework of model-theoretic semantics. The identification of formulas with processes naturally leads us to the following question; what is an appropriate notion of equivalence on formulas/processes from the viewpoint of process calculi? One might expect that *logical equivalence*, defined in terms of logical provability, provides such an adequate notion of equivalence; A and B are *logically equivalent* if $A \vdash B$ and $B \vdash A$ are provable in linear logic. Logical equivalence is, however, too coarse in a sense and too fine in another sense to be an adeqate notion of equivalence on processes. Consider two processes $\alpha \multimap \beta \multimap \gamma$ and $\beta \multimap \alpha \multimap \gamma$. $\alpha \multimap \beta \multimap \gamma$ intuitively means "first receive α, then receive β and send γ", while $\beta \multimap \alpha \multimap \gamma$ intuitively means "first receive β, then receive α and send γ". So they behave quite differently, whereas the logical equivalence identifies them. On the other hand, it is reasonable to think that $(\alpha \multimap \beta) \otimes (\gamma \multimap \delta)$ and $\alpha \multimap (\beta \otimes (\gamma \multimap \delta)) \& \gamma \multimap ((\alpha \multimap \beta) \otimes \delta))$ are equivalent with respect to their behavior, whereas they are not logically equivalent. Here $\alpha \multimap A$ means "receive α and invoke A", $\alpha \otimes A$ means "send α and invoke A", $A \,\&\, B$ means "choose A or B". (See Kobayashi&Yonezawa[11] for a slightly different view of logical equivalence.)

The leading principle to find an adequate notion of equivalence is that processes should be equivalent if they are indistinguishable by an external observer. What makes two processes equivalent or distinct is their observable behavior. Under this principle, various notions of equivalence have been proposed in the literature (cf. van Glabbeek[29][30]). Among those, we deal with *trace equivalence* in this paper.

Trace equivalence, presented by Hoare[7], is known to be one of the simplest notion of equivalence; roughly, A and B are trace equivalent if they can perform the same set of sequences of observable actions. Trace equivalence provides a better notion of equivalence on processes than that of logical equivalence from the observational point of view. As a matter of fact, it is easily shown that $\alpha \multimap \beta \multimap \gamma$ and $\beta \multimap \alpha \multimap \gamma$ are not trace equivalent, and that $(\alpha \multimap \beta) \otimes (\gamma \multimap \delta)$ and $\alpha \multimap (\beta \otimes (\gamma \multimap \delta)) \& \gamma \multimap ((\alpha \multimap \beta) \otimes \delta))$ are trace equivalent. Trace equivalence is sometimes considered to be too weak to identify processes in the sense that it identifies too many processes. In particular, it possibly identifies a deadlocking process with one that does not deadlock (see Example 3(2) in §4.1). Nevertheless it is of significance as the basis for other equivalence notions of processes. Trace equivalence provides a simple formalization of our basic intuition that processes are equivalent if they are observationally indistiguishable, and any equivalence naturally defined on the basis of this intuition can be seen as a refinement of trace equivalence.

In §4.1 we consider the trace equivalence in our framework, defined in terms of observable behavior of processes, and in §4.2 we introduce certain algebraic models, which we call *trace models*. Trace models are defined in the traditional frame work of algebraic semantics. Then our second main result says (in §4.2);

2. Trace equivalence is characterized by the completeness with respect to the trace models.

For the completeness proof we use the technique similar to the phase-semantic completeness proof.

In this preliminary report, we only deal with the systems that can be developed in propositional fragment of linear logic.

§2 Syntax and Operational Semantics

Through this paper, we consider the following correspondence between the logical notions and the notions from the theory of "message passing"-based process calculi;

Table 1

propositional variables	=	tokens or messages
logical connectives	=	action names
inference rules	=	transition rules
formulas	=	processes
sequents	=	process configurations
bottom-up proof construction	=	computation

We identify a propositional variable with a *token* or a *message*, and each logical connective symbol with an *action name*. Then the operational meaning of an action, namely the transition rule determining the behavior of the action, is described in terms of a logical inference rule corresponding to the logical connective associated to the action. A formula constructed from propositional variables and logical connectives is viewed as a *process* and a sequent (in the sequent-calculus formulation of logic) is viewed as a *process configuration*. A logical inference is interpreted as a state-transition by reading them bottom-up, thus, eg. a logical inference of the form

$$\frac{A, B, \Gamma \vdash}{A \otimes B, \Gamma \vdash} \otimes$$

is read as "state $A \otimes B, \Gamma$ transforms to state A, B, Γ by Parallel action $\otimes$". Then a bottom-up proof construction for a sequent "$\Gamma \vdash$" corresponds to a computation starting from a process configuration "$\Gamma \vdash$".

We introduce the system $\mathcal{S}$, a version of asynchronous concurrent process calculus based on linear logic proof search, that is essentially a subsystem of the system considered by Okada[20][21]. $\mathcal{S}$ is based on left one-sided sequent calculus. However, nothing important is missing for theoretical issues compared with process calculi based on two-sided (classical) sequent calculi such as Andreoi&Pareschi[2]'s LO and Kobayashi&Yonezawa[10][12]'s ACL, although two-sided formulation would be convenient for practical issues like the logic programming languages design.

Let us begin by defining the language $\mathcal{L}(\mathcal{S})$ of our system $\mathcal{S}$. We presuppose that a set $\mathcal{P}$ of propositional variables is given. As mentioned before, logical constants and connectives can be naturally interpreted by *actions* in our process calculus. In the following definition, we give the action names corresponding to the outermost logical connectives. Their precise operational meanings will be given below in terms of logical inference rules of linear logic.

Definition 1 The language $\mathcal{L}(\mathcal{S})$ is defined as follows;

1. If $\alpha \in \mathcal{P}$, then $\alpha \in \mathcal{L}(\mathcal{S})$ (Token or Message).
2. $\mathbf{1} \in \mathcal{L}(\mathcal{S})$ (Suicide-action).
3. If $A, B \in \mathcal{L}(\mathcal{S})$, then $A \otimes B \in \mathcal{L}(\mathcal{S})$ (Parallel-action), in particular if $\alpha \in \mathcal{P}$, then $\alpha \otimes B$ is called a Sending-action
4. If $\alpha_1, \ldots, \alpha_n \in \mathcal{P} (n \geq 1)$ and $B \in \mathcal{L}(\mathcal{S})$, then $\alpha_1 \otimes \cdots \otimes \alpha_n \multimap B \in \mathcal{L}(\mathcal{S})$ (Receiving-action).
5. If $A, B \in \mathcal{L}(\mathcal{S})$, then $A \,\&\, B \in \mathcal{L}(\mathcal{S})$ (Choice-action).
6. If $A \in \mathcal{L}(\mathcal{S})$, then $!A \in \mathcal{L}(\mathcal{S})$ (Bang-action).

Thus our language $\mathcal{L}(\mathcal{S})$ is a subset of that of the usual intuitionistic linear logic; $\mathcal{L}(\mathcal{S})$ lacks $\top, \mathbf{0}, \bot, \oplus$, and implications are restricted to the Horn implications in $\mathcal{L}(\mathcal{S})$ that require antecedents to be of the form $\alpha_1 \otimes \cdots \otimes \alpha_n$ for $\alpha_i \in \mathcal{P}$.

Roughly speaking, the formulas in $\mathcal{L}(\mathcal{S})$ correspond to the processes in CCS[16] and π-calculus[19][18] in the way described in Table 2.

Table 2: Correspondence between formulas in $\mathcal{S}$ and processes in CCS and π-calculus

	Parallel	Sending	Receiving	Choice	Bang (Replication)
$\mathcal{S}$	$A \otimes B$	$\alpha \otimes B$	$\alpha \multimap B$	$A \,\&\, B$	$!A$
CCS and π-calculus	$A \mid B$	$\overline{\alpha}.B$	$\alpha.B$	$A + B$	$!A$

It should be noted, however, that there are some serious differences between them; $\mathcal{S}$ is an asynchronous calculus in the sense explained later whereas CCS and π-calculus are synchronous, and $\mathcal{S}$ is based on proof-theoretic notions whereas CCS and π-calculus are based on algebraic notions. (An asynchronous version of π-calculus was also introduced in Honda[9] on the basis of algebraic notions.)

A finite multiset of formulas in $\mathcal{L}(\mathcal{S})$ is called a *process configuration*. A sequent of $\mathcal{S}$ of the form $\Gamma \vdash$ is identified with a process configuration Γ. In the sequel, $\alpha, \beta, \ldots$ range over $\mathcal{P}$, $A, B, \ldots$ range over $\mathcal{L}(\mathcal{S})$, and $\Gamma, \Delta, \ldots$ range over the process configurations of $\mathcal{L}(\mathcal{S})$. $\vec{\alpha}, \vec{\beta}, \ldots$ range over the finite sequences of propositional variables. If $\vec{\alpha} = \alpha_1, \ldots, \alpha_n$, then $\otimes\vec{\alpha}$ stands for $\alpha_1 \otimes \cdots \otimes \alpha_n$. In particular, if $\vec{\alpha_i}$ is the empty sequence, then $\otimes\vec{\alpha}$ stands for $\mathbf{1}$. We write A^n to denote a formula $\underbrace{A \otimes \cdots \otimes A}_{n \text{ times}}$. A formula of the form $!A$ is called a *modal formula*.

The inference rules of $\mathcal{S}$, which corresponds to the transition rules in process calculi (when read bottom-up), are essentially those of linear logic restricted to our language $\mathcal{L}(\mathcal{S})$. It should be noted that Γ below is considered as a *multiset*, hence the exchange rule is implicit.

- Parallel Action ($\otimes$)
  $$\frac{A, B, \Gamma \vdash}{A \otimes B, \Gamma \vdash} \otimes$$
  (Parallel action $A \otimes B$ invokes processes A and B in parallel.) A special case of this action is the Sending Action
  $$\frac{\alpha, B, \Gamma \vdash}{\alpha \otimes B, \Gamma \vdash} \otimes$$
  (Sending action $\alpha \otimes B$ sends a token α and invokes B.)

- Receiving Action ($\multimap$)
  $$\frac{(\vec{\alpha} \vdash \otimes\vec{\alpha}) \quad A, \Gamma \vdash}{\vec{\alpha}, \otimes\vec{\alpha} \multimap A, \Gamma \vdash} \multimap$$
  where $\vec{\alpha}$ denotes $\alpha_1, \ldots, \alpha_n (n \geq 1)$. (Receiving action $\otimes\vec{\alpha} \multimap A$ receives tokens $\vec{\alpha}$ from the environment and invokes A.) We treat this rule/action as if it had only one premise. This convention is justified by the fact that the left premise of this rule is always provable, (hence the proof construction of the left premise terminates immediately.)

- Choice Action (&)
  $$\frac{A, \Gamma \vdash}{A \,\&\, B, \Gamma \vdash} \,\& \qquad \frac{B, \Gamma \vdash}{A \,\&\, B, \Gamma \vdash} \,\&$$
  (Choice action $A \,\&\, B$ chooses either A or B, and invokes it.)

- Suicide Action ($\mathbf{1}$)
  $$\frac{\Gamma \vdash}{\mathbf{1}, \Gamma \vdash} \mathbf{1}$$
  (Suicide action $\mathbf{1}$ terminates itself.)

- Bang Action (!)
  $$\frac{!A, A, \Gamma \vdash}{!A, \Gamma \vdash} !$$
  (Bang action $!A$ produces a copy A and invokes it.)

Note that the above Bang ! rule is slightly different from Girard[5]'s original bang (modality) rules; ours is derivable from Girard's, but not *vice versa*. Our bang ! is sometimes called *Milner's*

bang because it behaves like the one Milner introduced in [18] for his theory of π-calculus. It can be shown that these two versions of bang ! are equivalent up to *reachability* and *trace equivalence* defined later (in this section and in §4.1).

Listed below are some useful derived rules in $\mathcal{S}$.

(1)

$$\frac{\dfrac{A, \Gamma \vdash}{\alpha, \alpha \multimap A, \Gamma \vdash}}{\alpha, \alpha \multimap A \& \beta \multimap B, \Gamma \vdash} \qquad \frac{\dfrac{B, \Gamma \vdash}{\beta, \beta \multimap B, \Gamma \vdash}}{\beta, \alpha \multimap A \& \beta \multimap B, \Gamma \vdash}$$

The process $\alpha \multimap A \& \beta \multimap B$ selects A or B depending on α or β which the process receives.

(2)

$$\frac{\dfrac{\vec{\beta}, !(\otimes\vec{\alpha} \multimap \otimes\vec{\beta}), \Gamma, \vdash}{\vec{\alpha}, \otimes\vec{\alpha} \multimap \otimes\vec{\beta}, !(\otimes\vec{\alpha} \multimap \otimes\vec{\beta}), \Gamma, \vdash}}{\vec{\alpha}, !(\otimes\vec{\alpha} \multimap \otimes\vec{\beta}), \Gamma, \vdash}$$

The process $!(\otimes\vec{\alpha} \multimap \otimes\vec{\beta})$ transforms tokens $\vec{\alpha}$ into $\vec{\beta}$, while the process itself remains unchanged.

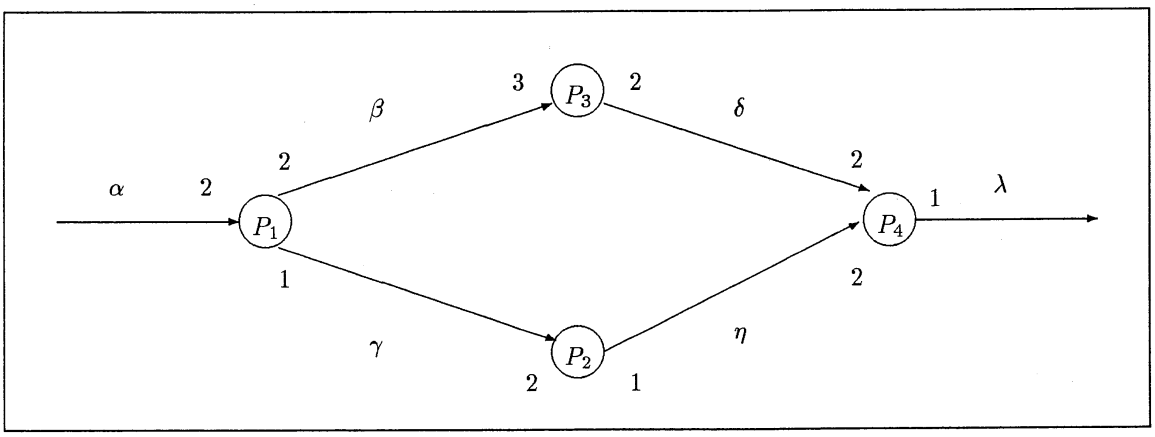
(3)

$$\frac{\dfrac{A, B, \Gamma \vdash}{\alpha, A, \alpha \multimap B, \Gamma \vdash}}{\alpha \otimes A, \alpha \multimap B, \Gamma \vdash}$$

The sender $\alpha \otimes A$ passes a message α to the receiver $\alpha \multimap B$. Note that this communication occurs asynchronously in the sense that the sender can send a message without synchronizing with the receiver. This is the most important difference from synchronous concurrent process calcului such as Milner[16]'s CCS, Hoare[8]'s CSP, and so on, and for this reason our $\mathcal{S}$ is said to be an *asynchronous concurrent process calculus* (also cf. Honda[9]).

Example 1 Consider the detafolw diagram below;

Figure 1: A dataflow diagram



Here, α,β, γ, δ, η and λ denote channels in the above dataflow network. Process P_1 is a process to receive two messages (tokens) from the channel α and to produce two tokens to channel β and

one token to channel γ concurrently. This is specified by $!(\alpha \otimes \alpha \multimap \beta \otimes \beta \otimes \gamma)$ in $\mathcal{S}$. We abbreviate this as $!(\alpha^2 \multimap \beta^2 \otimes \gamma)$. The bang (modality) ! means that this process can be repeated infinitely many times. P_2 is waiting for three tokens through channel β then sending two tokens through channel δ. P_2 is specified by $!(\beta^3 \multimap \delta^2)$. P_3 is waiting for two tokens through channel γ then sending one token through channel η. P_3 is specified by $!(\gamma^2 \multimap \eta)$. P_4 receives two tokens from channel δ and two from η concurrently, then produces one output token through channel λ. P_4 is written as $!(\delta^2 \otimes \eta^2 \multimap \lambda)$.

Then the whole network is described as Γ, where

$$\Gamma \equiv !(\alpha^2 \multimap \beta^2 \otimes \gamma), !(\beta^3 \multimap \delta^2), !(\gamma^2 \multimap \eta), !(\delta^2 \otimes \eta^2 \multimap \lambda).$$

Now consider an initial channel state m, say $\alpha^2, \beta, \gamma^5$. This means that the network is started with channel state m, i.e., two tokens at channel α, one token at channel β and five tokens at channel γ.

By using derived rule (2) above, we observe that the following is a derivation in $\mathcal{S}$;

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\lambda, \gamma^2, \Gamma \vdash}{\delta^2, \eta^2, \gamma^2, \Gamma \vdash}}{\delta^2, \eta, \gamma^4, \Gamma \vdash}}{\delta^2, \gamma^6, \Gamma \vdash}}{\beta^3, \gamma^6, \Gamma \vdash}}{\alpha^2, \beta, \gamma^5, \Gamma \vdash}$$

Let us denote by $n(\equiv \lambda, \gamma^2)$ the channel state in which there are one token at channel λ and two tokens at channel γ. Then the above derivation expresses that channel state n is reachable from channel state m under specification Γ. The channel state λ, η is also reachable from m under Γ, but the state λ^2 is not. ■

Example 2 If we incorporate such an infinitary expression as $\&_{i \in I} A_i$, where I denotes an arbitrary index set, into the language, and add an inference rule

$$\dfrac{A_j, \Gamma \vdash}{\&_{i \in I} A_i, \Gamma \vdash},$$

where $j \in I$, then we can express *value passing* between two processes in this extended system. Assume that $\mathcal{P}$ includes propositional variables of the form α_i where i is a natural number. Let us write $\alpha(i)$ to denote a_i. Nonatomic formulas are treated as if they were first-order formulas; we write $A(i)$ to indicate some occurrences of subscript i attached to propositional variables in A. We abbreviate a process $\&_{i \in N} \alpha(i) \multimap B(i)$ where N is the set of natural numbers by $\alpha(x) \multimap B(x)$. Then we see

$$\dfrac{\dfrac{\dfrac{A, B(n), \Gamma \vdash}{\alpha(n), A, \alpha(n) \multimap B(n), \Gamma \vdash}}{\alpha(n), A, \alpha(x) \multimap B(x), \Gamma \vdash}}{\alpha(n) \otimes A, \alpha(x) \multimap B(x), \Gamma \vdash}$$

is a derivation in this extended system. This expresses that the sender $\alpha(n) \otimes A$ passes value n to the receiver $\alpha(x) \multimap B(x)$ through channel α.

All results shown in this paper would still hold by this extension. Later we shall introduce this infinitary & formally in §4.1. ■

If

$$\dfrac{\Gamma_2 \vdash}{\Gamma_1 \vdash}$$

is an instance of an inference rule of $\mathcal{S}$, then the pair of Γ_1 and Γ_2 is called a *transition* (and we denote the transition relation by $\Gamma_1 \longrightarrow \Gamma_2$). A (finite or infinite) sequence $\Gamma_0, \Gamma_1, \ldots, \Gamma_i, \ldots$ of process configurations is a *transition sequence* if for each i a transition relation $\Gamma_{i-1} \longrightarrow \Gamma_i$ holds. The transitive reflexive closure of $\longrightarrow$ is written as $\longrightarrow^*$. Hence $\Gamma \longrightarrow^* \Gamma'$ means that there exists a finite transition sequence from Γ to Γ'.

Let $\vec{\alpha}$ be $\alpha_1, \ldots, \alpha_m (m \geq 0)$, $\vec{\beta}$ be $\beta_1, \ldots, \beta_n (n \geq 0)$ and Γ be a process configuration, i.e. a sequence of formulas of $\mathcal{L}(\mathcal{S})$. Then we say $\vec{\beta}$ is *reachable* from $\vec{\alpha}$ under Γ if $\vec{\alpha}, \Gamma \longrightarrow^* \vec{\beta}, !\Sigma$ for some sequence $!\Sigma$ of modal formulas.

Proposition 1 *The following are equivalent;*
(1) $\vec{\beta}$ is reachable from $\vec{\alpha}$ under Γ;
(2) $\vec{\alpha}, \Gamma \vdash \otimes\vec{\beta}$ is provable in classical full linear logic;
(3) $\vec{\alpha}, \Gamma \vdash \otimes\vec{\beta}$ is provable in intuitionistic full linear logic.
(See Girard[6] for the precise definition of classical and intuitionistic linear logic.)

Proof. Consider the following subsystem $\mathcal{S}'$ of linear logic;

Axiom: $\vec{\alpha} \vdash \otimes\vec{\alpha}$

Inference rules:

$$\frac{A, B, \Gamma \vdash \otimes\vec{\beta}}{A \otimes B, \Gamma \vdash \otimes\vec{\beta}} \otimes \qquad \frac{\vec{\alpha} \vdash \otimes\vec{\alpha} \quad A, \Gamma \vdash \otimes\vec{\beta}}{\vec{\alpha}, \otimes\vec{\alpha} \multimap A, \Gamma \vdash \otimes\vec{\beta}} \multimap$$

$$\frac{A, \Gamma \vdash \otimes\vec{\beta}}{A \,\&\, B, \Gamma \vdash \otimes\vec{\beta}} \,\&\qquad \frac{B, \Gamma \vdash \otimes\vec{\beta}}{A \,\&\, B, \Gamma \vdash \otimes\vec{\beta}} \,\&\qquad \frac{\Gamma \vdash \otimes\vec{\beta}}{\mathbf{1}, \Gamma \vdash \otimes\vec{\beta}} \mathbf{1}$$

$$\frac{\Gamma \vdash \otimes\vec{\beta}}{!A, \Gamma \vdash \otimes\vec{\beta}} !W \qquad \frac{!A, !A, \Gamma \vdash \otimes\vec{\beta}}{!A, \Gamma \vdash \otimes\vec{\beta}} !C \qquad \frac{A, \Gamma \vdash \otimes\vec{\beta}}{!A, \Gamma \vdash \otimes\vec{\beta}} !D$$

As easily shown, a sequent of the form $\Gamma \vdash \otimes\vec{\beta}$, where Γ is a process configuration of $\mathcal{S}$, is provable in $\mathcal{S}'$ iff it is provable in classical full linear logic iff it is provable in intuitionistic full linear logic. We can easily transform a finite transition sequence of $\mathcal{S}$ into a proof in $\mathcal{S}'$, and *vice versa*. ■

The above Proposition shows that the logical notion of provability characterizes reachability.

§3 Naive Phase Semantics Characterizing Reachability

§3.1 Preliminary Remark on Intuitionistic Phase Semantics

This subsection is devoted to a brief introduction to intuitionistic phase semantics, as a preliminary to the next subsection.

Phase semantics, originally introduced by Girard[5], is a standard model-theoretic semantics for (classical) linear logic. After the publication of [5], its intuitionistic versions are investigated by several authors, eg. Abrusci[1], Okada[22], Sambin[27], with slight differences in their definitions. Here we introduce a version of intuitionistic phase semantics, following Okada[22]. As proved in [22], the semantics completely characterizes provability in intuitionistic linear logic. In the light of Proposition 1 in §2, which says that provability in intuitionistic linear logic characterizes reachability in S, it is immediate that satisfiability in intuitionistic phase semantics also characterize reachability in S. However, intuitionistic phase semantics usually requires a certain closure operator to interprete formulas, which causes difficulty in understanding the intuitive meaning of formulas via the semantics.

We shall introduce naive phase semantics, i.e., phase semantics without any closure condition, in §3.2. That is obtained by modifying intuitionistic phase semantics defined below. A variant of

intuitionistic phase semantics is also used for the canonical model construction for completeness of the trace models in §4.4.

Definition 2 An *intuitionistic phase space* $(M, D, \bot)$ consists of a commutative monoid M, a subset D (called the *facts*) of the powersets of M and $\bot \in D$ that satisfies

(P1) D is closed under arbitrary $\bigcap$; in particular $M \in D$,

(P2) If $X \subseteq M$ and $Y \in D$, then $X \multimap Y \in D$

where $\multimap$ is defined by $X \multimap Y = \{y | \forall x \in X xy \in Y\}$ for any $X, Y \subseteq M$. We also define XY as $\{xy | x \in X, y \in Y\}$ and X^C as $\bigcap\{Y \in D | X \subseteq Y\}$ (the smallest fact that includes X).

Then, we can define $\mathbf{1} = \{1\}^C$ (1 stands for the unit element of M), $\top = M$, $\mathbf{0} = \emptyset^C$, and for any facts X, Y,

- $X \otimes Y = (XY)^C$,
- $X \,\&\, Y = X \cap Y$,
- $X \oplus Y = (X \cup Y)^C$,
- $X^\bot = X \multimap \bot$.

Among the basic properties of intuitionistic phase spaces, we see the following;

- For any facts X, Y and Z, $X \otimes Y = Y \otimes X$, $X \otimes (Y \otimes Z) = (X \otimes Y) \otimes Z$, $\mathbf{1} \otimes X = X$;
- $X \otimes Y \subseteq Z$ iff $Y \subseteq X \multimap Z$;
- $X \multimap (Y \multimap Z) = Y \multimap (X \multimap Z)$, $\mathbf{1} \multimap X = X$;
- $X \otimes (Y \oplus Z) = (X \otimes Y) \oplus (X \otimes Z)$, $X \multimap Y \,\&\, Z = (X \multimap Y) \,\&\, (X \multimap Z)$;
- $X \otimes (Y \,\&\, Z) \subseteq (X \otimes Y) \,\&\, (X \otimes Z)$, but the reverse does not hold in general.

Associativity of $\otimes$ is nontrivial, but follows from the observation that $X^C Y^C \subseteq (XY)^C$ for any $X, Y \subseteq M$. Note that $(D, \&, \oplus, \mathbf{0}, \multimap, \otimes, \mathbf{1})$ forms an *IL-algebra* in the sense of Troelstra[28].

A *classical phase space* is a special intuitionic phase space in which D consists of all X's such that $X = X^{\bot\bot}$.

The following definition is analogous to that of *enriched* (classical) phase spaces in Lafont[13] (cf. also Girard[6]). If M is an intuitionistic phase space, then $J(M) = \{x \in \mathbf{1} | x \in \{xx\}^C\}$ is a submonoid of M. An *enriched intuitionistic phase space* is an intuitionistic phase space M endowed with a submonoid K of $J(M)$ (not necessary to be a fact).

For any fact X of enriched intuitionistic phase space, define

- $!X = (X \cap K)^C$.

The following are some basic properties of modality ! (cf. Girard[5][6], Lafont[13]);

- For any facts X and Y, $!X \subseteq X$, $!X \otimes !X = !X$, $!X \subset \mathbf{1}$; if $!Y \subseteq X$, then $!Y \subseteq !X$;
- $!(X \,\&\, Y) = !X \otimes !Y$.

An *intuitionistic phase model* is given by an (enriched) intuitionistic phase space and an *interpretation* which maps each atom α to a fact α^* of M. Then any formula A is interpreted by a fact A^* along the above definitions, and $\Gamma \equiv A_1, \ldots, A_n$ is interpreted by $\Gamma^* = A_1^* \otimes \cdots \otimes A_n^*$. We say that A is *satisfied* in M if $1 \in A^*$, and that $\Gamma \vdash C$ is *satisfied* in M if $\Gamma^* \subseteq C^*$.

Theorem 1 *Let $\Gamma \vdash C$ be a sequent in intuitionistic linear logic. Then $\Gamma \vdash C$ is provable in intuitionistic linear logic if and only if it is satisfied in every intuitionistic phase model.*

Proof. See Okada[22]. ■

Combined with Proposition 1 in §2, we obtain;

Corollary 1 *Let Γ be a process configuration of $\mathcal{S}$. Then $\vec{\beta}$ is reachable from $\vec{\alpha}$ under Γ if and only if $\vec{\alpha}, \Gamma \vdash \otimes\vec{\beta}$ is satisfied in every intuitionistic phase model.*

§3.2 Naive Phase Semantics

As mentioned in the previous subsection, reachability in $\mathcal{S}$ is characterized by satisfiability in intuitionistic phase semantics (Corollary 1). But it would be a shortcoming of phase semantics that phase semantics heavily relies on some closure condition in order to be complete for its corresoponding syntax; for example, $X \otimes Y$ should be interpreted by $(XY)^{\perp\perp}$ in classical phase spaces and by $(XY)^C$ in intuitionistic phase spaces. However, from the viewpoint of practical applications of linear logic such as logic programming, process calculus and formal linguistics, one could hardly catch any intuitive meaning of these closure conditions. Hence it would be preferable to dispense with any closure condition. Such a phase semantics without closure condition is sometimes called a *naive phase semantics.*

Naive phase semantics is sound for full intuitionistic linear logic, but fails to be complete for the following obvious reasons;

1. The distributive law between & and $\oplus$ holds for every naive phase model, but it cannot be proved in linear logic.
2. Phase semantics requires that **1** be interpreted by the smallest fact including 1, the monoid unit. In a naive phase model, however, such a fact would be $\{1\}$, that is too poor to be an interpretation of **1**; any formula of the form $!A$ would collapse into **1** or **0**, since $!A$ must be interpreted by a subset of the interpretation of **1**.

Hence it is an interesting question to what extent of subsystems of linear logic one can obtain the completeness with respect to naive phase semantics.

There are several completeness results on the naive phase semantics for certain very restricted subsystems of linear logic, especially for Lambek Calculus[14], which is essentially $(\otimes, \multimap)$-fragment of noncommutative intuitionistic linear logic, and its related systems. Buszkowski[4] proved that Lambek Calculus and some systems related to it are complete with respect to the naive phase models (the *generalized standard models* or *GS-models*, in his terminology). Pentus[26] proved that the naive phase models based on *free* semigroups, called the *language models*, are sufficient to be complete for Lambek Calculus. Okada&Terui[23] showed that the *finite* naive phase models are sufficient to be complete for Lambek Calculus and some related systems, hence that these systems have the finite model property with respect to the naive phase models.

In this subsection, we shall introduce the system $\mathcal{S}_1$ by restricting $\mathcal{S}$ in such a way that **1** does not occur and ! only occurs as an outermost connective. Then we prove that reachability in $\mathcal{S}_1$ is characterized by the completeness with respect to the naive phase models (Theorem 2 below). $\mathcal{S}_1$ has enough expressive power to represent a wide range of message-passing based communication networks, hence our result in this Section would be useful in practical applications.

Definition 3 The language $\mathcal{L}(\mathcal{S}_1)$ is defined as follows;

- If A is a formula in $\mathcal{L}(\mathcal{S})$ that contains neither **1** nor !, then A and $!A$ are formulas in $L(\mathcal{S}_1)$.

The inference rules of $\mathcal{S}_1$ is the same as $\mathcal{S}$ (but restricted to $\mathcal{L}(\mathcal{S}_1)$).

In this Section, we write a process configuration in the form $!\Gamma, \Delta$, where all modal formulas in the sequent are indicated by $!\Gamma$.

Definition 4 A *naive phase model* M is an intuitionistic phase model (not enriched) in which the facts D consist of all subsets of M.

A naive phase model does not need the closure operation ^{C}; $X \otimes Y$ is simply interpreted by XY and each atomic formula is interpreted by *any* subset of M. Since D plays no role and $\perp$ does not have to be specified (because our language does not contain $\perp$), we can say that a naive phase model is simply a commutative monoid M with an interpretation $*$ that maps each atomic formula α to a subset α^* of M. Bang !(modality) cannot be interpreted directly in a naive phase

model for the reason discussed above. Thus we do not require a naive phase model to be enriched, rather, we interprete modal formulas as if they were axioms.

By a Γ-*model* we mean a naive phase model in which $1 \in A^*$ holds for each A occuring in Γ, namely a naive phase model in which Γ is true.

Proposition 2 (Soundness) *Let* $!\Gamma, \Delta$ *be a process configuration of* $\mathcal{S}_1$. *If* $\vec{\beta}$ *is reachable from* $\vec{\alpha}$ *under* $!\Gamma, \Delta$, *then* $(\vec{\alpha}, \Delta)^* \subseteq (\otimes\vec{\beta})^*$ *in every* Γ*-model.*

Proof. If $\vec{\beta}$ is reachable from $\vec{\alpha}$ under $!\Gamma, \Delta$, then there is a transition sequence

$$\vec{\alpha}, !\Gamma, \Delta \equiv !\Gamma, \Sigma_0 \longrightarrow !\Gamma, \Sigma_1 \longrightarrow \cdots \longrightarrow !\Gamma, \Sigma_n \equiv \vec{\beta}, !\Gamma.$$

(Note that the total amount of modal formulas does not change through transitions, by our restriction on $\mathcal{L}(\mathcal{S}_1)$.) Then it is easily shown that $\Sigma_{i-1}^* \subseteq \Sigma_i^*$ for each $1 \leq i \leq n$ in every Γ-model.■

§3.3 Completeness of Naive Phase Semantics

To show the reverse of Proposition 2, namely, completeness, we exploit the completeness proof method investigated by Okada&Terui[23].

First let us give some ideas informally. Given a process configuration $!\alpha \multimap \beta \otimes \gamma, \alpha$, we have the following transition sequence (derivation);

$$\dfrac{\dfrac{\dfrac{\vdots}{!\alpha \multimap \beta \otimes \gamma,\ \beta,\ \gamma \vdash}}{!\alpha \multimap \beta \otimes \gamma,\ \beta \otimes \gamma \vdash}}{\dfrac{!\alpha \multimap \beta \otimes \gamma,\ \alpha \multimap \beta \otimes \gamma,\ \alpha \vdash}{!\alpha \multimap \beta \otimes \gamma,\ \alpha \vdash}}$$

We would like to give a concrete model to this transition sequence and to give a concrete interpretation in the model to each formula occurring in the transition sequence. It is natural to construct a model based on *preconditions* of processes. What we mean by the term precondition is illustrated in the following transition sequence, where each formula B is *labelled* like $a : B$ with a expressing a precondition of B;

$$\dfrac{\dfrac{\dfrac{\vdots}{!\alpha \multimap \beta \otimes \gamma,\ \sqrt[1]{\alpha} : \beta,\ \sqrt[r]{\alpha} : \gamma \vdash}}{!\alpha \multimap \beta \otimes \gamma,\ \alpha : \beta \otimes \gamma \vdash}}{\dfrac{!\alpha \multimap \beta \otimes \gamma,\ 1 : \alpha \multimap \beta \otimes \gamma,\ \alpha : \alpha \vdash}{!\alpha \multimap \beta \otimes \gamma,\ \alpha : \alpha \vdash}}$$

- α occurs in the initial process configuration, hence α itself is a precondition of α.
- We do not consider preconditions for modal formulas.
- $\alpha \multimap \beta \otimes \gamma$ has the empty precondition denoted by 1 above because this can be produced freely by Bang action.
- $\beta \otimes \gamma$ emerges from two processes $\alpha \multimap \beta \otimes \gamma$ and α, which have preconditions 1 and α, respectively. Hence $1\alpha \equiv \alpha$ is a precondition of $\beta \otimes \gamma$.
- $\beta \otimes \gamma$ splits into β and γ. Let us consider $\sqrt[1]{\alpha}$ (the *left-half* of α) to be a precondition of β, and $\sqrt[r]{\alpha}$ (the *right-half* of α) to be a precondition of γ.

The labels express the preconditions which have a natural monoid-structure, thus, $1\alpha \equiv \alpha$ and $\sqrt[1]{\alpha} \cdot \sqrt[r]{\alpha} = \alpha$. We can construct a naive phase model from the labels occuring in the above transition sequence, in which

if $a:B$ occurs in the transition sequence, then $a \in B^*$.

Hence this model can be seen as a direct representation of the above transition sequence. This model is indeed a $(\alpha \multimap \beta \otimes \gamma)$-model, because $1 : \alpha \multimap \beta \otimes \gamma$ occurs in the sequence, hence $1 \in (\alpha \multimap \beta \otimes \gamma)^*$.

By a construction like the above, we can obtain a countermodel for the completeness proof. Suppose that $\vec{\beta}$ is not reachable from $\vec{\alpha}$ under $!\Gamma, \Delta$. Then we can construct a naive phase model in which $\vec{\alpha}\Delta \in (\vec{\alpha}, \Delta)^*$ and $\vec{\alpha}\Delta \notin \otimes\vec{\beta}^*$. The resulting phase model is indeed a Γ-model, hence we obtain the completeness.

Let us begin the proof by giving the precise definition of the labels. Our labels are obtained by modifying the terms of the system ND introduced by Buszkowski[4], which was used in his proof of completeness for Lambek Calculus with respect to GS-models. See also Pankrat'ev[24] for another use of the system ND. We modify ND-terms by adding the unit label 1 with convention $a1 \equiv 1a \equiv a$, and by imposing commutativity $ab \equiv ba$ on the labels.

Definition 5 The *labels* L and the *simple labels* $\overline{L} \subset L$ are defined as follows;

1. 1 is a simple label.
2. Each formula in $\mathcal{L}(\mathcal{S}_1)$ is a simple label.
3. if a is a label and A is a formula of the form $B \otimes C$, then $\sqrt[l]{a}_A$ and $\sqrt[r]{a}_A$ are simple labels.
4. if a and b are labels, then ab is a label.

As a convention, we identify $a_1 a_2 \cdots a_n$ with any of its permutations. Moreover, we assume that $a1 \equiv 1a \equiv a$ for any label a. For example, $b\sqrt[l]{a1}_A \equiv b\sqrt[l]{a}_A \equiv \sqrt[l]{a}_A b$.

Now we define a reduction relation on the labels.

Definition 6 For any labels a, a', b and any formula A, if a contains as sublabel $\sqrt[l]{b}_A \sqrt[r]{b}_A$ and a' results from a by replacing one occurrence of $\sqrt[l]{b}_A \sqrt[r]{b}_A$ by b, then we say that a *reduces* to a', denoted by $a \mapsto a'$. We denote the reflexive, transitive closure of the relation $\mapsto$ by $\mapsto^*$.

Lemma 1 *The relation* $\mapsto^*$ *on* $\mathcal{L}_1$ *is confluent and terminating.*

Proof. This was essentially due to Buszkowski[4]. See also Pankrat'ev[24]. ■

As a corollary, each label a has a unique *normal form* denoted by $a^\bullet$. Write $a \bullet b$ to denote $(ab)^\bullet$. Then we can easily derive associativity of $\bullet$ from the above lemma.

A *labelled formula* is a formula equipped with a label in normal form (write $a:A$ for a label a and a formula A). A *labelled process configuration* is of the form $!B_1, \ldots, !B_m, a_1:A_1, \ldots, a_n:A_n$, where each non-modal formula A_i is labelled by a label a_i, whereas each modal formula $!B_j$ is not labelled. If $\Delta \equiv A_1, \ldots, A_n$, then $a_1:A_1, \ldots, a_n:A_n$ is sometimes abbreviated by $a_1 \bullet \cdots \bullet a_n : \Delta$; eg., if $\sqrt[l]{b}_C : A$ and $\sqrt[r]{b}_C : B$, then $\sqrt[l]{b}_C : A$, $\sqrt[r]{b}_C : B$ is abbreviated by $b : A, B$.

The inference rules of $\mathcal{S}_1$ are extended to those for labelled sequents, as follows;

$$\frac{!\Gamma, \sqrt[l]{a}_{A\otimes B}:A, \sqrt[r]{a}_{A\otimes B}:B, c:\Delta \vdash}{!\Gamma, a:A\otimes B, c:\Delta \vdash} \qquad \frac{!\Gamma, a_1 \bullet \cdots \bullet a_n \bullet b:B, c:\Delta \vdash}{!\Gamma, a_1:\alpha_1, \ldots, a_n:\alpha_n, b:\alpha_1 \otimes \cdots \alpha_n \multimap B, c:\Delta \vdash}$$

$$\frac{!\Gamma, a:A, c:\Delta \vdash}{!\Gamma, a:A \,\&\, B, c:\Delta \vdash} \qquad \frac{!\Gamma, a:B, c:\Delta \vdash}{!\Gamma, a:A \,\&\, B, c:\Delta \vdash} \qquad \frac{!\Gamma, !A, 1:A, c:\Delta \vdash}{!\Gamma, !A, c:\Delta \vdash}$$

Note that if

$$\frac{!\Gamma, b:\Delta_2 \vdash}{!\Gamma, a:\Delta_1 \vdash}$$

is an instance of one of the above inference rules, then b is identical with a.

Let Δ be $A_1, \ldots, A_n$. A *proof search tree* $\mathcal{T}(!\Gamma, \Delta)$ is a rooted tree where a labelled process configuration of $\mathcal{S}_1$is assigned to each node, constructed as follows;

(1) Process configuration $!\Gamma, A_1 : A_1, \ldots, A_n : A_n$ (A_i is labelled by A_i itself) is assigned to the root;

(2) When $!\Gamma, a:\Delta'$ is assigned to a node,

(i) if there is no $!\Gamma, a:\Sigma$ such that $!\Gamma, a:\Delta' \longrightarrow !\Gamma, a:\Sigma$, then this node is a leaf of $\mathcal{T}(!\Gamma, \Delta)$;

(ii) otherwise, all sequents of the form $!\Gamma, a : \Sigma$ such that $!\Gamma, a : \Delta' \longrightarrow !\Gamma, a : \Sigma$, are the assignments of the children nodes of this node (with assignment $!\Gamma, a:\Delta'$).

Note that $\mathcal{T}(!\Gamma, \Delta)$ includes all transition sequences starting from $!\Gamma, A_1 : A_1, \ldots, A_n : A_n$ as the assignments. Let

$$\mathcal{T}^*(!\Gamma, \Delta) = \{b:\Sigma \mid !\Gamma, b:\Sigma, c:\Pi \text{ is a node of } \mathcal{T}(!\Gamma, \Delta) \text{ for some } c:\Pi\}.$$

We say that a label b *occurs* in $\mathcal{T}^*(!\Gamma, \Delta)$ if $b:\Sigma \in \mathcal{T}^*(!\Gamma, \Delta)$ for some Σ.

The basic properties of labelled proof search trees are expressed in Lemma 2 and Corollary 2, which play key roles in Lemma 3 and Lemma 4 below. To show Lemma 2, we need two Sublemmas (Sublemma 1 and Sublemma 2), which we state without proofs.

For each label a, we define a sequence $O(a)$ of labelled formulas as follows;

1. $O(1) = \phi$ (the empty sequence);
2. $O(A) = A : A$ if A is a formula;
3. $O(\sqrt[l]{b}_{B\otimes C}) = \sqrt[l]{b}_{B\otimes C} : B$, $O(\sqrt[r]{b}_{B\otimes C}) = \sqrt[r]{b}_{B\otimes C} : C$;
4. $O(p_1 \cdots p_n) = O(p_1), \ldots, O(p_n)$ where each p_i is a simple label.

Sublemma 1 *Let* $b_1:\Sigma_1, \ldots, b_n:\Sigma_n \in \mathcal{T}^*(!\Gamma, \Delta)$. *Then*
(i) $O(b_1), \ldots, O(b_n) \in \mathcal{T}^*(!\Gamma, \Delta)$;
(ii) $!\Gamma, O(b_i) \longrightarrow^* !\Gamma, b_i:\Sigma_i$ *for each* i.

Sublemma 2 *Let* $b_1, \ldots, b_n$ *be labels occurring in* $\mathcal{T}^*(!\Gamma, \Delta)$. *If* $O(b_1 \bullet \cdots \bullet b_n) \in \mathcal{T}^*(!\Gamma, \Delta)$ *then* $!\Gamma, O(b_1 \bullet \cdots \bullet b_n) \longrightarrow^* !\Gamma, O(b_1), \ldots, O(b_n)$.

Lemma 2 *If* $b_i : \Sigma_i \in \mathcal{T}^*(!\Gamma, \Delta)$ *for each* $1 \leq i \leq n$ *and* $b_1 \bullet \cdots \bullet b_n : \Pi \in \mathcal{T}^*(!\Gamma, \Delta)$, *then* $b_1:\Sigma_1, \cdots, b_n:\Sigma_n \in \mathcal{T}^*(!\Gamma, \Delta)$.

Proof. By Sublemma 1(i), $O(b_1 \bullet \cdots \bullet b_n) \in \mathcal{T}^*(!\Gamma, \Delta)$, hence $O(b_1), \ldots, O(b_n) \in \mathcal{T}^*(!\Gamma, \Delta)$ by Sublemma 2. Since $O(b_i) \longrightarrow^* b_i:\Sigma_i$ by Sublemma 1(ii), it easily follows that $b_1:\Sigma_1, \cdots, b_n:\Sigma_n \in \mathcal{T}^*(!\Gamma, \Delta)$. ■

Corollary 2 *If each of* a, b, c *and* $a \bullet b \bullet c$ *occur in* $\mathcal{T}^*(!\Gamma, \Delta)$, *then* $a \bullet b$, $b \bullet c$ *and* $a \bullet c$ *also occur in* $\mathcal{T}^*(!\Gamma, \Delta)$.

Proof. By definition $a:\Sigma_1 \in \mathcal{T}^*(!\Gamma, \Delta)$, $b:\Sigma_2 \in \mathcal{T}^*(!\Gamma, \Delta)$, $c:\Sigma_3 \in \mathcal{T}^*(!\Gamma, \Delta)$ and $a \bullet b \bullet c:\Pi \in \mathcal{T}^*(!\Gamma, \Delta)$ for some $\Sigma_1, \Sigma_2, \Sigma_3$ and Π. Hence by Lemma 2 $a:\Sigma_1, b:\Sigma_2, c:\Sigma_3 \in \mathcal{T}^*(!\Gamma, \Delta)$. Then Corollary 2 follows by definition. ■

Given a proof search tree $\mathcal{T}(!\Gamma, \Delta)$ defined above, we construct a naive phase model $\mathcal{M} \equiv \mathcal{M}(!\Gamma, \Delta)$. In the sequel, $\mathcal{T}$ stands for $\mathcal{T}(!\Gamma, \Delta)$ and $\mathcal{T}^*$ stands for $\mathcal{T}^*(!\Gamma, \Delta)$.

$\mathcal{M}$ consists of a commutative monoid (also denoted by $\mathcal{M}$) and an interpretation $*$ defined as follows;

- $\mathcal{M} = \{a \in L | a \text{ occurs in } \mathcal{T}^*\} \cup \{\surd\}$, where $\surd$ is a distinguished propositional variable not occuring in $\mathcal{T}^*$.

We assume that 1 is always in $\mathcal{M}$. Note that every $a \in \mathcal{M}$ is a label in normal form.

- For $a, b \in \mathcal{M}$, $a \cdot b = \begin{cases} a \bullet b & \text{if } a \bullet b \text{ occurs in } \mathcal{T}^*; \\ \surd & \text{otherwise.} \end{cases}$

In particular, $a \cdot \surd = \surd$ for any $a \in \mathcal{M}$.

- For each α, $\alpha^* = \{b | b{:}\alpha \in \mathcal{T}^*\} \cup \{\surd\}$

Lemma 3 *$(\mathcal{M}, \cdot, 1)$ is actually a commutative monoid.*

Proof. Almost immediate. Only nontrivial is associativity $(a \cdot b) \cdot c = a \cdot (b \cdot c)$. If $a \bullet b \bullet c$ occurs in $\mathcal{T}^*$, then by Corollary 2, $a \bullet b$ and $b \bullet c$ occur in $\mathcal{T}^*$. Hence $(a \cdot b) \cdot c = (a \bullet b) \bullet c = a \bullet (b \bullet c) = a \cdot (b \cdot c)$. If $a \bullet b \bullet c$ does not occur in $\mathcal{T}^*$, then $(a \cdot b) \cdot c = \surd = a \cdot (b \cdot c)$. ■

Lemma 4 *For any formula B, (i) if $b{:}B \in \mathcal{T}^*$, then $b \in B^*$, and (ii) $\surd \in B^*$.*

Proof. (ii) is obvious. Here we only prove (i) by induction on the complexity of B.

(Case 1) B is an atomic formula. Immediate by definition.

(Case 2) $B \equiv C \otimes D$.

Assume $b : C \otimes D \in \mathcal{T}^*$. Then $\sqrt[1]{b}_{C \otimes D} : C, \sqrt[r]{b}_{C \otimes D} : D \in \mathcal{T}^*$. By induction hypothesis, $\sqrt[1]{b}_{C \otimes D} \in C^*$ and $\sqrt[r]{b}_{C \otimes D} \in D^*$. Hence $b = \sqrt[1]{b}_{C \otimes D} \cdot \sqrt[r]{b}_{C \otimes D} \in C^* \otimes D^*$.

(Case 3) $B \equiv \otimes\vec{\alpha} \multimap D$, where $\vec{\alpha} = \alpha_1, \ldots, \alpha_n$.

Assume $b{:}C \multimap D \in \mathcal{T}^*$. It suffices to show that for any $c \in \otimes\vec{\alpha}^*$, $c \cdot b \in D^*$. If $c \cdot b = \surd$, then by induction hypothesis (ii) $\surd \in D^*$. Hence we may assume that $c \bullet b$ occurs in $\mathcal{T}^*$. By definition, $c \in \otimes\vec{\alpha}^*$ means that there are labels $c_1, \ldots, c_n$ such that $c_1 \bullet \cdots \bullet c_n \equiv c$ and $c_i{:}\alpha_i \in \mathcal{T}^*$ for each c_i. Hence by Lemma 2, $!\Gamma_0, c_1{:}\alpha_1, \ldots, c_n{:}\alpha_n, b{:}\otimes\vec{\alpha} \multimap D, d{:}\Sigma \vdash$ is an assignment of a node of $\mathcal{T}$ for some $d{:}\Sigma$. Hence,

$$\frac{!\Gamma_0, c \bullet b{:}D, d{:}\Sigma \vdash}{!\Gamma_0, c_1{:}\alpha_1, \ldots, c_n{:}\alpha_n, b{:}\otimes\vec{\alpha} \multimap D, d{:}\Sigma \vdash}$$

Therefore $c \bullet b{:}D \in \mathcal{T}^*$, and by induction hypothesis, $c \bullet b \in D^*$.

(Case 4) $B \equiv C \& D$. Obvious. ■

Finally we obtain;

Theorem 2 *Let $!\Gamma_0, \Delta_0$ be a process configuration of $\mathcal{S}_1$. Then $\vec{\beta}$ is reachable from $\vec{\alpha}$ under $!\Gamma_0, \Delta_0$ if and only if $(\vec{\alpha}, \Delta_0)^* \subseteq (\otimes\vec{\beta})^*$ in every Γ_0-model.*

Proof. The only-if part is Proposition 2. To show the reverse, suppose that $\vec{\beta}$ is not reachable from $\vec{\alpha}$ under $!\Gamma_0, \Delta_0$. Let $!\Gamma_0 \equiv !G_1, \ldots, !G_k$, $\Delta_0 \equiv D_1, \ldots, D_l$ and $\vec{\beta} \equiv \beta_1, \ldots, \beta_m$.

By the above construction we get a proof search tree $\mathcal{T}_0 \equiv \mathcal{T}(!\Gamma_0, \vec{\alpha}, \Delta_0)$ and a naive phase model $\mathcal{M}_0 \equiv \mathcal{M}(!\Gamma_0, \vec{\alpha}, \Delta_0)$. We claim the following;

(1) $\mathcal{M}_0$ constructed above is a Γ_0-model.

(2) In $\mathcal{M}_0$, label $D_1 D_2 \cdots D_l$ is in $(\vec{\alpha}, \Delta_0)^*$.

(3) In $\mathcal{M}_0$, label $D_1 D_2 \cdots D_l$ is not in $\otimes\vec{\beta}^*$.

As for (1), $1 : G_i \in \mathcal{T}^*{}_0$ for each $!G_i$ in $!\Gamma_0$. Hence by Lemma 4, $1 \in G_i^*$. (2) also follows from Lemma 4. As for (3), by assumption $!\Gamma_0, a : \vec{\beta} \notin \mathcal{T}_0$, where $a \equiv D_1 \cdots D_l$. Hence, it easily follows by Lemma 2 that there are no labels $a_1, \ldots, a_n$ such that $a_i \in \beta_i^*$ for each i and $a_1 \bullet \cdots \bullet a_n \equiv D_1 D_2 \cdots D_l$. ■

§4 Algebraic Semantics Characterizing Trace Equivalence

§4.1 Trace Equivalence

In this Section we introduce the notion of trace equivalence (Hoare[7]) in our system of process calculus, and give the characterization of the equivalence by means of model-theoretic semantics.

We introduce the system $\mathcal{S}_2$ in which implications are restrictied to the ones of the form $\alpha \multimap B$ and two inference rules that express *observable actions* are added. These observable actions are not inference rules of linear logic, but it enables us to estimate observable effects of processes in a precise manner. Then we define the notion of *trace* and *trace equivalence* on processes (or process configurations) in system $\mathcal{S}_2$ in terms of these observable actions.

We also introduce the system $\overline{\mathcal{S}_2}$ which has the infinitary & expressions. $\overline{\mathcal{S}_2}$ can express, for example, value passing between processes (See Example 2 in §2).

Trace equivalence is a simple and intuitive notion, but has certain shortcomings. Among them, it is often pointed out (cf. van Glabbeek [29], Milner[17]) that it identifies too many processes, in particular it possibly identifies a deadlocking process with one that does not deadlock. We shall briefly mention this point in Example 3(2). Nevertheless, trace equivalence deserves careful analysis, because it can be seen as the basis for other equivalences finer than this equivalence.

A relationship between the notion of trace and that of reachability is established in Proposition 3.

Definition 7 The language $\mathcal{L}(\mathcal{S}_2)$ of $\mathcal{S}_2$ is obtained by restricting $\mathcal{L}(\mathcal{S})$ so that if a formula in $\mathcal{L}(\mathcal{S}_2)$ contains $A \multimap B$ as subformula, then A is a propositional variable α.

$\mathcal{S}_2$ has the following two actions in addition. These are called *observable actions*, while the actions described in §2 is called *silent actions*, since those actions are completely taken inside the system, and an external observer outside the system cannot observe them.

- Input Action (α)
  $$\frac{P, \Gamma \vdash}{\alpha \multimap P, \Gamma \vdash}\ \alpha$$
  (Input action α gets a token α from the outside of the system. This action is understood to be always possible no matter what the environment is.)

- Output Action ($\overline{\alpha}$)
  $$\frac{\Gamma \vdash}{\alpha, \Gamma \vdash}\ \overline{\alpha}$$
  (Output action $\overline{\alpha}$ throws away a token α to the outside of the environment.)

Of course, observable actions are not logical inference rules at all. The point of introducing these actions is that it enables us to *observe* process behavior from the outside of the system, and by means of these actions we can define the notion of trace eqivalence.

We also introduce system $\overline{\mathcal{S}_2}$, which extends $\mathcal{S}_2$ with infinitary & described in §2 Example 2.

Definition 8 The language $\mathcal{L}(\overline{\mathcal{S}_2})$ is defined as follows;

1. if $\alpha \in \mathcal{P}$, then $\alpha \in \mathcal{L}(\overline{\mathcal{S}_2})$;
2. if $A_i \in \mathcal{L}(\mathcal{S}_2)$, (i.e., A_i contains no $\&_{j \in J}$) for each $i \in I$, where I denotes an arbitrary index set, then $\&_{i \in I} A_i \in \mathcal{L}(\overline{\mathcal{S}_2})$;
3. if $\alpha \in \mathcal{P}$, $A, B \in \mathcal{L}(\overline{\mathcal{S}_2})$, then $\alpha \multimap A$, $A \otimes B$, $A \,\&\, B$ and $!A$ are in $\mathcal{L}(\overline{\mathcal{S}_2})$.

$\overline{\mathcal{S}_2}$ has the following inference rule in addition to those of $\mathcal{S}_2$;

$$\frac{A_j, \Gamma \vdash}{\&_{i \in I} A_i, \Gamma \vdash},$$

where $j \in I$.

It is clear that $\overline{\mathcal{S}_2}$ is a conservative extension of $\mathcal{S}_2$. All results stated below hold both for $\mathcal{S}_2$ and for $\overline{\mathcal{S}_2}$.

Let Act be $\{\alpha | \alpha \in \mathcal{P}\} \cup \{\overline{\alpha} | \alpha \in \mathcal{P}\}$ and Act^* be the set of all finite sequences over Act. In particular, the empty sequence is in Act^* and denoted by 1. For $t \equiv p_1 \ldots p_n \in Act^*$, we define $len(t) = n$. In the sequel, $s, t, u, \ldots$ range over Act^*.

Now the transition relation $\longrightarrow$, defined in §2, is reformulated for the *labelled transition relation* as follows;

Definition 9

- $\Gamma \xrightarrow{p} \Delta$ if
  $\dfrac{\Delta \vdash}{\Gamma \vdash} p$
  is an instance of an inference rule of $\mathcal{S}_2$ with p indicating the action name corresponding to the inference.
- $\Gamma \longrightarrow^* \Delta$ if $\Gamma \xrightarrow{p_1} \cdots \xrightarrow{p_m} \Delta$ (possibly $m = 0$) where each p_i is a silent action.

For each $t \in Act^*$, we define a binary relation $\stackrel{t}{\Rightarrow}$ on process configurations by induction on $len(t)$, as follows;

- If $len(t) = 0$ then $t \equiv 1$. We define $\Gamma \stackrel{1}{\Rightarrow} \Delta$ by $\Gamma \longrightarrow^* \Delta$;
- If t is of the form pt', where $p \in Act$, then $\Gamma \stackrel{t}{\Rightarrow} \Delta$ holds whenever there is a Γ' such that $\Gamma \longrightarrow^* \xrightarrow{p} \longrightarrow^* \Gamma' \stackrel{t'}{\Rightarrow} \Delta$.

Let Γ be a process configuration of $\mathcal{S}_2$. If $\Gamma \stackrel{t}{\Rightarrow} \Gamma'$ for some Γ', we say that $t \in Act^*$ is a *trace* of Γ and write $\Gamma \stackrel{t}{\Rightarrow}$. Define $tr(\Gamma) = \{t | \Gamma \stackrel{t}{\Rightarrow}\}$. Then Γ and Δ are said to be *trace equivalent* if $tr(\Gamma) = tr(\Delta)$.

Example 3 Consider the processes below;

(1) $\alpha \multimap \beta \,\&\, \alpha \multimap \mathbf{1}$ $\stackrel{\alpha}{\Rightarrow}$ β $\stackrel{\overline{\beta}}{\Rightarrow}$ $\emptyset$
$\stackrel{\alpha}{\Rightarrow}$ $\emptyset$

$tr(\alpha \multimap \beta \,\&\, \alpha \multimap \mathbf{1}) = \{1, \alpha, \alpha\overline{\beta}\} = tr(\alpha \multimap \beta)$. Hence $\alpha \multimap \beta \,\&\, \alpha \multimap \mathbf{1}$ is trace equivalent to $\alpha \multimap \beta$.

(2) $(!\alpha) \,\&\, \alpha$ $\stackrel{\overline{\alpha}}{\Rightarrow}$ $!\alpha$ $\stackrel{\overline{\alpha}}{\Rightarrow}$ $!\alpha \stackrel{\overline{\alpha}}{\Rightarrow} \cdots$
$\stackrel{\overline{\alpha}}{\Rightarrow}$ $\emptyset$

This process is trace equivalent to $!\alpha$. This exemplifies a drawback of trace equivalence; $(!\alpha)\,\&\,\alpha$ may deadlock whereas $!\alpha$ never deadlock, but they are taken to be the same if we adopt trace equivalence.

(3)

$$\begin{aligned} tr(\alpha \otimes (\beta \,\&\, \gamma)) &= \{1, \overline{\alpha}, \overline{\alpha}\overline{\beta}, \overline{\alpha\gamma}, \overline{\beta}, \overline{\beta}\overline{\alpha}, \overline{\gamma}, \overline{\gamma\alpha}\} \\ &= tr(\alpha \otimes \beta \,\&\, \alpha \otimes \gamma). \end{aligned}$$

Hence, $\alpha \otimes (\beta \,\&\, \gamma)$ is trace equivalent to $\alpha \otimes \beta \,\&\, \alpha \otimes \gamma$. ■

Given $t \in Act^*$, let $Inp(t)$ be the multiset $\{\alpha | \alpha \in t$ and α is an input action$\}$ and let $Out(t)$ be the multiset $\{\alpha | \overline{\alpha} \in t$ and $\overline{\alpha}$ is an output action$\}$. The following Proposition shows the relationship between the notion of reachability and that of trace;

Proposition 3 *If $\Gamma \stackrel{t}{\Rightarrow} !\Xi$ where $!\Xi$ consists of modal formulas, then $Out(t)$ is reachable from $Inp(t)$ under Γ. Conversely, if $\vec{\beta}$ is reachable from $\vec{\alpha}$ under Γ, then there are $t \in Act^*$ and $!\Xi$, such that $\Gamma \stackrel{t}{\Rightarrow} !\Xi$, $Inp(t) = \vec{\alpha}$ and $Out(t) = \vec{\beta}$.*

§4.2 Trace Models

Our next purpose is to characterize trace equivalence by means of model-theoretic semantics. To this end, we introduce an algebraic model, called a *trace model*, and show soundness and completeness for trace equivalence with respect to the trace models.

Definition 10 A *trace algebra* $< \mathcal{D}, \bigwedge, \mathbf{1}, \otimes, \multimap, \mathcal{A} >$ consists of the following;

- $\mathcal{A} \subseteq \mathcal{D}$
- $< \mathcal{D}, \bigwedge, \mathbf{1} >$ is a complete meet semilattice with maximal element $\mathbf{1}$. We define a partial order $\leq$ on $\mathcal{D}$ by $p \leq q \stackrel{def}{\Leftrightarrow} p \wedge q = p$.
- $< \mathcal{D}, \otimes, \mathbf{1} >$ is a commutative monoid.
- $\multimap : \mathcal{A} \times \mathcal{D} \longrightarrow \mathcal{D}$. We write $a \multimap p$ to denote $\multimap(a, p)$ (in the sequel we assume that $a \in \mathcal{A}$ when we write $a \multimap p$).
- $a \otimes (a \multimap p) \leq p$, $p \otimes (a \multimap q) \leq a \multimap (p \otimes q)$.
- $\otimes$ *distributes over* $\bigwedge$, i.e., $\bigwedge_{i \in I} q \otimes p_i = q \otimes \bigwedge_{i \in I} p_i$.
- $\multimap$ *distributes over* $\bigwedge$, i.e., $\bigwedge_{i \in I} a \multimap p_i = a \multimap \bigwedge_{i \in I} p_i$
- *The expansion law* (cf. Milner[16][17]) holds, i.e., $(a \multimap p) \otimes (b \multimap q) = a \multimap (p \otimes (b \multimap q)) \wedge b \multimap ((a \multimap p) \otimes q)$.

In a trace algebra bang operator ! is defined by $!p = \bigwedge_{i \in N} p^i$, where N is the set of natural numbers and p^i denotes $\underbrace{p \otimes \cdots \otimes p}_{i \text{ times}}$.

The following are easily derived in a trace algebra.

- If $p \leq q$ then $r \otimes p \leq r \otimes q$ and $a \multimap p \leq a \multimap q$.
- $p \otimes q \leq p$.
- $!p \leq p$, $!p \otimes !p = !p$. If $!p \leq q$ then $!p \leq !q$.
- $!(p \wedge q) = !p \otimes !q$.

Definition 11 A *trace model* is a trace algebra with an *interpretation* * which maps $\alpha \in \mathcal{P}$ into $\alpha^* \in \mathcal{A}(\subseteq \mathcal{D})$. In a trace model, nonatomic formulas and process configurations are interpreted as follows;

- $(A \otimes B)^* = A^* \otimes B^*$;
- $(\alpha \multimap B)^* = \alpha^* \multimap B^*$;
- $(A \,\&\, B)^* = A^* \wedge B^*$; $(\&_{i \in I} A_i)^* = \bigwedge_{i \in I} A_i^*$;
- $(!A)^* = !(A^*)$;
- $(A_1, \ldots, A_n)^* = A_1 \otimes \cdots \otimes A_n$, in particular the empty process configuration is interpreted by $\mathbf{1}$.

Remark that $!A$ has the same interpretation as $\&_{i \in N} A^i$. This reflects the syntactic observation that $tr(!A) = tr(\&_{i \in N} A^i)$ (cf. Lemma 6).

The trace models characterize trace equivalence, in the form of completeness theorem below;

Theorem 3 *$tr(A) = tr(B)$ if and only if $A^* = B^*$ in every trace model.*

We prove the "only-if" part (soundness) in §4.3 (Corollary 3) and the "if" part (completeness) in §4.4 (Corollary 4).

§4.3 Soundness of Trace Models

Through this subsection we fix a trace model $\mathcal{D} =< \mathcal{D}, \bigwedge, \mathbf{1}, \otimes, \multimap, \mathcal{A} >$. We cannot use the usual induction on the length of proof to show the soundness of trace models, since we deal with possibly infinite proof constructions that do not reach any axiom. Instead, the proof below proceeds as follows;

1. Assign $[t] \in \mathcal{D}$ to each $t \in Act^*$;
2. Define the *observation value* of A by $\bigwedge_{A \stackrel{t}{\Rightarrow}} [t]$;
3. Show that $[A] = A^*$ for any $A \in \mathcal{L}(\mathcal{S}_2)$.

Soundness easily follows from 3. It should be noted that A^* above is the interpretation of A inductively defined along Definition 11, while $[A]$ is completely determined by the traces of A; to determine $[A]$, one does not have to know *what A exactly is*. It is sufficient to know its observable behavior, i.e., its traces.

First we inductively define a ternary relation $\_ \lhd \_ \circ \_ \subseteq Act^* \times Act^* \times Act^*$ as follows;

- $1 \lhd 1 \circ 1$.
- If $u \lhd s \circ t$, then $\alpha u \lhd \alpha s \circ t$, $\alpha u \lhd s \circ \alpha t$, $u \lhd \alpha s \circ \overline{\alpha} t$, and $u \lhd \overline{\alpha} s \circ \alpha t$ for any α.

Then we define $F(s,t) \subseteq Act^*$ by $F(s,t) = \{u \in Act^* | u \lhd s \circ t\}$.

Lemma 5 *$\Gamma, \Delta \stackrel{u}{\Rightarrow}$ if and only if $u \in F(s,t)$, $\Gamma \stackrel{s}{\Rightarrow}$ and $\Delta \stackrel{t}{\Rightarrow}$ for some s, t.*

Proof. "If" part: By induction on the generation of $u \lhd s \circ t$. We treat only the case when $u' \lhd \overline{\alpha} s' \circ \alpha t'$ is derived from $u' \lhd s' \circ t'$, $\Gamma \stackrel{\overline{\alpha} s'}{\Rightarrow}$ and $\Delta \stackrel{\alpha t'}{\Rightarrow}$. Then there are some Γ' and Δ' such that $\Gamma, \Delta \longrightarrow^* \Gamma', \Delta'$ and $\Gamma' \stackrel{s'}{\Rightarrow}$ and $\Delta' \stackrel{t'}{\Rightarrow}$ hold. By induction hypothesis, $\Gamma', \Delta' \stackrel{u'}{\Rightarrow}$ holds, therefore $\Gamma, \Delta \longrightarrow^* \Gamma', \Delta' \stackrel{u'}{\Rightarrow}$ holds.

"Only-if" part: $\Gamma, \Delta \stackrel{u}{\Rightarrow}$ means that there is a finite transition sequence

$$\Gamma, \Delta \equiv \Gamma_0, \Delta_0 \xrightarrow{p_1} \Gamma_1, \Delta_1 \xrightarrow{p_2} \cdots \xrightarrow{p_n} \Gamma_n, \Delta_n$$

and $\Gamma_0, \Delta_0 \stackrel{u}{\Rightarrow} \Gamma_n, \Delta_n$ holds, where for each $1 \leq i \leq n$ one of the following holds;

(1) $\Gamma_{i-1} \xrightarrow{p_i} \Gamma_i$ and $\Delta_{i-1} \equiv \Delta_i$;
(2) $\Gamma_{i-1} \equiv \Gamma_i$ and $\Delta_{i-1} \xrightarrow{p_i} \Delta_i$;
(3) $\Gamma_{i-1} \not\equiv \Gamma_i$ and $\Delta_{i-1} \not\equiv \Delta_i$.

We define $s_i \in Act^*$ and $t_i \in Act^*$ by induction on i. If $i = 0$, then $s_i \equiv t_i \equiv 1$.

When (1) holds for $i \geq 1$, then $t_i \equiv t_{i-1}$. If p_i is an observable action, then $s_i \equiv s_{i-1} p_i$; otherwise $s_i \equiv s_{i-1}$.

When (2) holds for $i \geq 1$, similar to the previous case.

When (3) holds for $i \geq 1$, then p_i must be a receiving action, and either

(3a) $\Gamma_{i-1} \xrightarrow{\alpha} \Gamma_i$ and $\Delta_{i-1} \xrightarrow{\overline{\alpha}} \Delta$ holds for some α, or

(3b) $\Gamma_{i-1} \xrightarrow{\overline{\alpha}} \Gamma_i$ and $\Delta_{i-1} \xrightarrow{\alpha} \Delta$ holds for some α.

If (3a) is the case, then $s_i \equiv s_{i-1}\alpha$ and $t_i \equiv t_{i-1}\overline{\alpha}$. If (3b) is the case, then $s_i \equiv s_{i-1}\overline{\alpha}$ and $t_i \equiv t_{i-1}\alpha$.

By the above construction, we see that $u \in F(s_n, t_n)$, $\Gamma \stackrel{s_n}{\Rightarrow}$ and $\Delta \stackrel{t_n}{\Rightarrow}$. ■

Lemma 6 *$!A \stackrel{u}{\Rightarrow}$ if and only if $A^n \stackrel{u}{\Rightarrow}$ for some $n \in N$.*

Proof. Obvious. ■

Remark: This lemma shows that, in $\overline{\mathcal{S}_2}$, our bang !A is equivalent to $\&_{i \in N} A^n$ up to traces.

Definition 12 For $t \in Act^*$, we define $[t] \in D$ as follows;

- $[1] = \mathbf{1}(\in \mathcal{D})$;
- $[\overline{\alpha}t] = \alpha^* \otimes [t]$;
- $[\alpha t] = \alpha^* \multimap [t]$.

The *observation value* of A is defined by

- $[A] = \bigwedge_{A \stackrel{t}{\Rightarrow}} [t]$.

Lemma 7 $\bigwedge_{u \in F(s,t)} [u] = [s] \otimes [t]$.

Proof. By induction on $len(s) + len(t)$. Since other cases are similar, we only treat the case when s is of the form $\alpha s'$ and t is of the form $\overline{\alpha}t'$.

$$\begin{aligned}
\bigwedge_{u \in F(\alpha s', \overline{\alpha} t')} [u] &= \bigwedge_{u' \in F(s', \overline{\alpha} t')} [\alpha u'] \wedge \bigwedge_{u' \in F(\alpha s', t')} [\overline{\alpha} u'] \wedge \bigwedge_{u' \in F(s', t')} [u'] \\
&= \bigwedge_{u' \in F(s', \overline{\alpha} t')} \alpha \multimap [u'] \wedge \bigwedge_{u' \in F(\alpha s', t')} \alpha \otimes [u'] \wedge \bigwedge_{u' \in F(s', t')} [u'] \\
&= \alpha \multimap \bigwedge_{u' \in F(s', \overline{\alpha} t')} [u'] \wedge \alpha \otimes \bigwedge_{u' \in F(\alpha s', t')} [u'] \wedge \bigwedge_{u' \in F(s', t')} [u'] \\
&= \alpha \multimap ([s'] \otimes [\overline{\alpha} t']) \wedge \alpha \otimes ([\alpha s'] \otimes [t']) \wedge ([s'] \otimes [t']) \\
&= \alpha \multimap ([s'] \otimes (\alpha \otimes [t'])) \wedge \alpha \otimes ((\alpha \multimap [s']) \otimes [t']) \wedge ([s'] \otimes [t']) \\
&= \alpha \otimes ((\alpha \multimap [s']) \otimes [t']) \\
&= [\alpha s'] \otimes [\overline{\alpha} t'].
\end{aligned}$$

The expansion law is needed in the case when s is of the form $\alpha s'$ and t is of the form $\beta t'$. ■

Lemma 8 $[P^n] = [P]^n$.

Proof. Obvious. ■

Now we obtain the main proposition with the help of the above Lemmas.

Proposition 4 *In every trace model,* $[A] = A^*$.

Proof. By induction on the complexity of A.
Case 1) A is a propositional variable. Obvious.
Case 2) A is of the form $B \otimes C$.

$$\begin{aligned}
[B \otimes C] &= \textstyle\bigwedge_{B \otimes C \stackrel{u}{\Rightarrow}} [u] = \bigwedge_{B, C \stackrel{u}{\Rightarrow}} [u] = \bigwedge_{B \stackrel{s}{\Rightarrow}} \bigwedge_{C \stackrel{t}{\Rightarrow}} \bigwedge_{u \in F(s,t)} [u] && \text{(by Lemma 5)} \\
&= \textstyle\bigwedge_{B \stackrel{s}{\Rightarrow}} \bigwedge_{C \stackrel{t}{\Rightarrow}} [s] \otimes [t] && \text{(by Lemma 7)} \\
&= \textstyle\bigwedge_{B \stackrel{s}{\Rightarrow}} [s] \otimes \bigwedge_{C \stackrel{t}{\Rightarrow}} [t] \\
&= [B] \otimes [C] = B^* \otimes C^* && \text{(by induction hypothesis)}
\end{aligned}$$

Case 3) A is of the form $!B$. By Lemma 6 and Lemma 8.
Case 4) A is of the form $B \,\&\, C$. Obvious. ■

Soundness is almost immediate if we take into consideration that the observation value of a formula is completely determined by its traces.

Corollary 3 (Soundness) *If* $tr(A) = tr(B)$ *then* $A^* = B^*$ *in every trace model.*

§4.4 Completeness of Trace Models

The proof of completeness employs the usual canonical model construction for (phase semantic) completeness proof (cf. Okada[22]). We construct a syntactic (canonical) model of $\overline{\mathcal{S}_2}$ in which $\Gamma \in A^*$ implies $tr(\Gamma) \supseteq tr(A)$ (Proposition 5). Then the completeness (Corollary 4) directly follows. It is trivial that for any formulas A in $\mathcal{L}(\mathcal{S}_2)$, $tr(A)$ in system $\mathcal{S}_2$ is equivalent to $tr(A)$ in system $\overline{\mathcal{S}_2}$, hence we have the completeness for $\mathcal{S}_2$ at the same time.

For each $t \in Act^*$, we define $\widetilde{t} \in \mathcal{L}(\overline{\mathcal{S}_2})$ as follows;

- $\widetilde{1} = \mathbf{1}(\in \mathcal{L}(\overline{\mathcal{S}_2}))$;
- $\widetilde{\overline{\alpha}t} = \alpha \otimes \widetilde{t}$;
- $\widetilde{\alpha t} = \alpha \multimap \widetilde{t}$.

We define a phase space-like structure $\mathcal{M}$ as follows;

- Let $\mathcal{M}$ be the free commutative monoid generated by all formulas in $\mathcal{L}(\overline{\mathcal{S}_2})$.
- For each $t \in Act^*$, define $[\![t]\!] = \{\Gamma | \Gamma \stackrel{t}{\Rightarrow}\}$, and for each $A \in \mathcal{L}(\overline{\mathcal{S}_2})$, $[\![A]\!] = \bigcap_{A \stackrel{t}{\Rightarrow}} [\![t]\!]$.
- $\mathcal{D}$ is defined in such a way that $X \in \mathcal{D}$ if and only if X is of the form $\bigcap_{i \in \Lambda} [\![t_i]\!]$ where $t_i \in Act^*$ for each $i \in \Lambda$. Then clearly $[\![A]\!] \in \mathcal{D}$ for any $A \in \mathcal{L}(\overline{\mathcal{S}_2})$.
- $\mathcal{A} = \{[\![\alpha]\!] | \alpha \in \mathcal{P}\}$.

It is easily seen that $[\![A]\!] = \{\Gamma | tr(\Gamma) \supseteq tr(A)\}$ and $[\![\widehat{t}]\!] = [\![t]\!]$. Moreover, we observe that

($\sharp$) for any $X \in \mathcal{D}$, if $\Gamma \in X$ and $tr(\Delta) \supseteq tr(\Gamma)$ then $\Delta \in X$.

On this structure, we can define phase-semantic operations & and $\otimes$ as in §3.1; let $X \subseteq \mathcal{M}$, $Y \subseteq \mathcal{M}$, then

- $X \,\&\, Y = X \cap Y$; $\&_{i \in I} X_i = \bigcap_{i \in I} X_i$;
- $X \otimes Y = (XY)^C = \bigcap \{[\![t]\!] | XY \subseteq [\![t]\!]\}$, where $XY = \{\Gamma, \Delta | \Gamma \in X, \Delta \in Y\}$.

The interpretation of $\multimap$ is modified; let $X \in \mathcal{A}$ and $Y \subseteq \mathcal{M}$, then X is of the form $[\![\alpha]\!]$ (such an α is uniquely determined), and we define

- $[\![\alpha]\!] \multimap Y = \bigcap \{[\![\alpha t]\!] | Y \subseteq [\![t]\!]\}$.

$!X$ is defined by $\bigcap_{i \in N} X^i$ for each $X \subseteq \mathcal{M}$. $\mathbf{1} = [\![\mathbf{1}]\!] = \{\Gamma | \Gamma \stackrel{1}{\Rightarrow}\} = \mathcal{M}$. Finally we define an interpretation * by $\alpha^* = [\![\alpha]\!]$ for each $\alpha \in \mathcal{P}$. It is obvious that $A^* \in \mathcal{D}$ for each $A \in \mathcal{L}(\overline{\mathcal{S}_2})$.

Then we show the following;

Proposition 5 *(i)* $\mathcal{D} \equiv < \mathcal{D}, \bigcap, \mathbf{1}, \otimes, \multimap, \mathcal{A} >$ *and* * *defined above form a trace model.*
(ii) For every formula $A \in \mathcal{L}(\overline{\mathcal{S}_2})$, $A \in A^* \subseteq [\![A]\!]$.

From this, the completeness easily follows;

Corollary 4 (Completeness) *If* $A^* = B^*$ *in every trace model, then* $tr(A) = tr(B)$.

Proof of Completeness. By Proposition 5(i), $A^* = B^*$ holds in $\mathcal{D}$ defined above. By Proposition 5(ii), $A \in A^*$, hence $A \in B^*$, which in turn implies $A \in [\![B]\!]$ by Proposition 5(ii), that means $tr(A) \supseteq tr(B)$. $tr(A) \subseteq tr(B)$ is shown similarly. ■

Proof of Proposition 5. First we show (ii) by induction on the complexity of A.

(Case 1) A is a propositional vairable. By definition.

(Case 2) A is of the form $\alpha \multimap B$. First we prove that $\alpha \multimap B \in (\alpha \multimap B)^* = [\![\alpha]\!] \multimap B^* = \bigcap\{[\![\alpha t]\!] | B^* \subseteq [\![t]\!]\}$. It suffices to show that $\alpha \multimap B \in [\![\alpha t]\!]$ for each t such that $B^* \subseteq [\![t]\!]$. Since $B \in B^*$ by induction hypothesis, $B \in [\![t]\!]$, that means $B \stackrel{t}{\Rightarrow}$. Hence $\alpha \multimap B \stackrel{\alpha t}{\Rightarrow}$. Therefore $\alpha \multimap B \in [\![\alpha t]\!]$.

To prove $(\alpha \multimap B)^* \subseteq [\![\alpha \multimap B]\!]$, assume $\Gamma \in (\alpha \multimap B)^*$. Then $\Gamma \in [\![\alpha t]\!]$ for any $[\![t]\!] \supseteq B^*$. In particular, $\Gamma \in [\![\alpha t]\!]$ for any $[\![t]\!] \supseteq [\![B]\!]$, since $B^* \subseteq [\![B]\!]$ by induction hypothesis. Therefore, $\Gamma \in [\![\alpha t]\!]$ for any $B \stackrel{t}{\Rightarrow}$. Then it follows that $\Gamma \in [\![\alpha \multimap B]\!]$ because $[\![\alpha \multimap B]\!] = \bigcap_{\alpha \multimap B \stackrel{u}{\Rightarrow}} [\![u]\!] = \bigcap_{B \stackrel{t}{\Rightarrow}} [\![\alpha t]\!]$.

(Case 3) A is of the form $B \otimes C$. By induction hypothesis, $B \in B^*$ and $C \in C^*$. Hence $B, C \in B^*C^*$. It is immediate that $B \otimes C \in (B^*C^*)^C = B^* \otimes C^*$ by $tr(B, C) = tr(B \otimes C)$ and $(\sharp)$.

On the other hand, $B^*C^* \subseteq [\![B]\!][\![C]\!]$, since $B^* \subseteq [\![B]\!]$ and $C^* \subseteq [\![C]\!]$ by induction hypothesis. $[\![B]\!][\![C]\!] \subseteq [\![B \otimes C]\!]$ is shown by the observation that if $tr(\Gamma) \supseteq tr(B)$ and $tr(\Delta) \supseteq tr(C)$, then $tr(\Gamma, \Delta) \supseteq tr(B, C) = tr(B \otimes C)$. Finally we infer $B^* \otimes C^* \subseteq [\![B \otimes C]\!]$ from $B^*C^* \subseteq [\![B \otimes C]\!]$.

(Case 4) A is of the form $B \,\&\, C$. This is a special case of (Case 6) below.

(Case 5) A is of the form $!B$. By induction hypothesis, $B \in B^*$. Since $tr(!B) \supseteq tr(B)$, $!B \in B^*$. Hence we see that

$$\underbrace{!B, \ldots, !B}_{n \text{ times}} \in \underbrace{B^* \otimes \cdots \otimes B^*}_{n \text{ times}}$$

for each $n \in N$. On the other hand, it can be easily shown that $tr(!B) = tr(\underbrace{!B, \ldots, !B}_{n \text{ times}})$. Therefore, by $(\sharp)$, $!B \in (B^*)^n$ for each n, thus $!B \in !B^*$.

To show $!B^* = \bigcap_{n \in N} B^{*n} \subseteq [\![!B]\!] = \bigcap_{!B \stackrel{t}{\Rightarrow}} [\![t]\!]$, suppose $\Gamma \in \bigcap_{n \in N} B^{*n}$ and $!B \stackrel{t}{\Rightarrow}$. Then, using induction hypothesis, we can prove that $\Gamma \in [\![B^n]\!]$ for each n. On the other hand, by Lemma 6, $!B \stackrel{t}{\Rightarrow}$ if and only if $B^n \stackrel{t}{\Rightarrow}$ for some $n \in N$. Hence $\Gamma \in [\![t]\!]$ and we conclude $!B^* \subseteq [\![!B]\!]$.

(Case 6) A is of the form $\&_{i \in I} B_i$. By induction hypothesis, $B_i \in B_i^*$ for each $i \in I$. Since $tr(\&_{i \in I} B_i) \supseteq tr(B_i)$, we see by $(\sharp)$ that $\&_{i \in I} B_i \in B_i^*$ for each $i \in I$. Therefore, $\&_{i \in I} B_i \in \bigcap_{i \in I} B_i^*$.

$\&_{i \in I} B_i^* \subseteq [\![\&_{i \in I} B_i]\!]$ is almost immediate from induction hypothesis.

Now let us show (i). First note that we can obtain more strong result than (i); $A^* = [\![A]\!]$ holds for each $A \in \mathcal{L}(\overline{S_2})$. Suppose $\Gamma \in [\![A]\!]$, then $tr(\Gamma) \supseteq tr(A)$. By (i) and $(\sharp)$ we obtain that $\Gamma \in A^*$.

Here we only show the expansion law and the distribution law between $\otimes$ and $\bigcap$. The other laws are shown similary. Note that each element in $\mathcal{A}$ is of the form $[\![\alpha]\!]$ and each element in $\mathcal{D}$ is of the form $\bigcap_{i \in I} [\![t_i]\!] = \bigcap_{i \in I} [\![\widetilde{t_i}]\!]$. The expansion law is proved as follows;

$$\begin{aligned}
&([\![\alpha]\!] \multimap \textstyle\bigcap_{i \in I} [\![t_i]\!]) \otimes ([\![\beta]\!] \multimap \bigcap_{j \in J} [\![s_j]\!]) = [\![(\alpha \multimap \&_{i \in I} \widetilde{t_i}) \otimes (\beta \multimap \&_{j \in J} \widetilde{s_j})]\!] \\
&= [\![\alpha \multimap (\&_{i \in I} \widetilde{t_i} \otimes (\beta \multimap \&_{j \in J} \widetilde{s_j})) \,\&\, \beta \multimap ((\alpha \multimap \&_{i \in I} \widetilde{t_i}) \otimes \&_{j \in J} \widetilde{s_j})]\!] \quad (*) \\
&= [\![\alpha]\!] \multimap ((\textstyle\bigcap_{i \in I} [\![t_i]\!] \otimes ([\![\beta]\!] \multimap \bigcap_{j \in J} [\![s_j]\!])) \cap [\![\beta]\!] \multimap (([\![\alpha]\!] \multimap \textstyle\bigcap_{i \in I} [\![t_i]\!]) \otimes \bigcap_{j \in J} [\![s_j]\!])
\end{aligned}$$

In the above, (*) is due to the observation that

$$tr((\alpha \multimap B) \otimes (\beta \multimap C)) = tr(\alpha \multimap (B \otimes (\beta \multimap C)) \,\&\, \beta \multimap ((\alpha \multimap B) \otimes C)).$$

The distribution law is shown as follows;

$$\begin{aligned}
\bigcap_{k \in K} (\bigcap_{i \in I} [\![t_i]\!] \otimes \bigcap_{j \in J} [\![s_{kj}]\!]) &= \bigcap_{k \in K} ([\![\&_{i \in I} \widetilde{t_i} \otimes \&_{j \in J} \widetilde{s_{kj}}]\!]) \\
&= [\![\&_{i \in I} \widetilde{t_i} \otimes \&_{(k,j) \in K \times J} \widetilde{s_{kj}}]\!] \quad (**) \\
&= \bigcap_{i \in I} [\![t_i]\!] \otimes \bigcap_{k \in K} \bigcap_{j \in J} [\![s_{kj}]\!]
\end{aligned}$$

For (**), observe that

$\bigcup_{k\in K} tr(\&_{i\in I}\widetilde{t_i} \otimes \&_{j\in J}\widetilde{s_{kj}}) = tr(\&_{i\in I}\widetilde{t_i} \otimes \&_{(k,j)\in K\times J}\widetilde{s_{kj}})$.

This completes the proof of Proposition 5. ■

§5 Conclusion and Future Works

So far we have seen how traditional logical analyses, especially model-theoretic semantics, contribute to characterize new notions from the discipline of process calculi.

We have obtained a simple semantics characterizing the reachability. One interesting point can be found in the completeness proof of naive phase semantics in §3.3. Our naive phase model construction for the completeness proof gives each process configuration Γ a naive phase model $\mathcal{M}(\Gamma)$ that directly represents the precondition structure in the computation starting from Γ; in $\mathcal{M}(\Gamma)$, $a \in A^*$ holds whenever a is a precondition of A. We also considered *splitted* preconditions of the form $\sqrt[l]{b}_B$, which represents a half of precondition b of B. It is the use of these splitted preconditions that allows us to drop the closure condition and obtain the completeness with respect to the naive phase models. This model construction suggests that our naive phase models provide somewhat direct interpretation of processes. A further study is needed to clarify this point.

The second part of this paper is an attempt to apply the traditional model-theoretic technique to characterization of the new equivalence notions of processes that come from the theory of process calculi. Trace equivalence, which we have characterized by trace models in §4, is so coarse that it would be of little use in practice. Nevertheless, it is important as the basis for further study on equivalences on processes/formulas; the notion of trace equivalence well-captures a primitive intuition that processes/formulas should be identified on the ground of their observable behavior. There is a wide range of equivalence notions naturally defined in terms of process behavior, among which trace equivalence is not only the coarsest, but also the most general in the sense that each equivalence can be seen as a refinement of trace equivalence (cf. van Glabbeek[29], Milner[17]). Therefore, it is expected, on the basis of our result in §4, to obtain model-theoretic characterizations for finer equivalences than trace equivalence, such as *failure equivalence* (Brookes,Hoare&Roscoe[3]), *simulation equivalence* (cf. Park[25]) and *bisimulation equivalence* (Milner[15]).

Our system is too restricted to develop a full theory of process calculus. An important mechanism which our system lacks is the *guarding*. By guarding a process (or a process configuration) with respect to some messages $\vec{\alpha}$, we can inhibit communications of $\vec{\alpha}$ between the outside of the scope of guarding operator and the inside of the scope. This, roughly, corresponds to the existential quantifier in terms of logic. However, incorporation of guarding mechanism will undoubtedly complicate the situation around the notion of observation and observable actions. Hence it still remains uncertain whether our results hold for richer fragments that include a kind of guarding operator in syntax.

Besides, the first author introduced in [20] the *mobile linear logic* to express *mobile message passing* in a linear logic-based process calculus, corresponding to Milner's extension of CCS to π-calculus[19], a mobile version and higher order mobile version of CCS. We leave these invesigation on extended systems to future works.

References

[1] V. Michele Abrusci. Non-commutative intuitionistic linear propositional logic. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, Vol. 36, pp. 297–318, 1990.

[2] Jean-Marc Andreoli and Remo Pareschi. Linear objects: Logical processes with built-in inheritance. *New Generation Computing*, Vol. 9, pp. 445–473, 1991.

[3] S. D. Brookes, C. A. R. Hoare, and A. W. Roscoe. A theory of communicating sequential processes. *JACM*, Vol. 31, No. 3, pp. 560–599, 1984.

[4] W. Buszkowski. Completeness results for Lambek syntactic calculus. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, Vol. 32, pp. 13–28, 1986.

[5] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, Vol. 50, pp. 1–102, 1987.

[6] Jean-Yves Girard. Linear logic: Its syntax and semantics. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, pp. 1–42. Cambridge University Press, 1995. Proceedings of the Workshop on Linear Logic, Ithaca, New York, June 1993.

[7] C. A. R. Hoare. Communicating sequential processes. In R. M. McKeag and A. M. Macnaghten, editors, *On the construction of programs – an advanced course*, pp. 229–254. Cambridge University Press, 1980.

[8] C. A. R. Hoare. *Communicating Sequential Processes.* Prentice-Hall, Englewood Cliffs, NJ, 1985.

[9] K. Honda and M. Tokoro. An object calculus for asynchronous communication. In *Proceedings of ECOOP'91*, pp. 133–147. Springer-Verlag LNCS 512, 1991.

[10] Naoki Kobayashi. Asynchronous communication model based on linear logic. Master's thesis, Department of Information Science, University of Tokyo, March 1992.

[11] Naoki Kobayashi and Akinori Yonezawa. Logical, testing, and observation equivalence for for processes in a linear logic programming. preprint (presented at Linear Logic Workshop, Cornell University), 1993.

[12] Naoki Kobayashi and Akinori Yonezawa. Asynchronous communication model based on linear logic. *Formal Aspects of Computing*, Vol. 7, pp. 113–149, 1995. Short version appeared in Joint International Conference and Symposium on Logic Programming, Washington, DC, November 1992, Workshop on Linear Logic and Logic Programming.

[13] Yves Lafont. The finite model property for various fragments of linear logic. *Journal of Symbolic Logic*, to appear. Available by ftp anonymous on iml.univ-mrs.fr, in /pub/lafont.

[14] J. Lambek. The mathematics of sentence structure. *American Mathematical Monthly*, Vol. 65, No. 3, pp. 154–170, 1958.

[15] R. Milner. Calculi for synchrony and asynchrony. *Theoretical Computer Science*, Vol. 25, pp. 267–310, 1983.

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

[17] R. Milner. Operational and algebraic semantics of concurrent processes. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pp. 1201–1242. Elsevier Science Publishers B.V. (North-Holland), 1990.

[18] R. Milner. Functions as processes. *Mathematical Structure in Computer Science*, Vol. 2, No. 2, pp. 119–146, 1992.

[19] R. Milner, J.G. Parrow, and D.J. Walker. A calculus of mobile processes. *Information and Computation*, Vol. 100, No. 1, pp. 1–77, 1992.

[20] Mitsuhiro Okada. Mobile linear logic as a framework for asynchronous and synchronous mobile communication calculi. Technical Report, Concordia University, February 1993.

[21] Mitsuhiro Okada. A brief look at the linear logic from the proof search paradigm. In *Introduction to Linear Logic (the lecture note of the introduction course at Linear96, Tokyo Meeting).* 1996.

[22] Mitsuhiro Okada. Phase semantics for higher order completeness, cut-elimination and normalization proofs (extended abstract). In J.-Y. Girard, M. Okada, and A. Scedrov, editors, *ENTCS (Electronic Notes of Theoretical Computer Science) Vol.3: A Special Issue on the Linear Logic 96, Tokyo Meeting.* Elsevier-ENTCS, 1996. An earlier version is available by ftp anonymous on iml.univ-mrs.fr, in pub/okada.

[23] Mitsuhiro Okada and Kazushige Terui. Naive phase semantics for some fragments of intuitionistic linear logic. Manuscript, available by ftp anonymous on abelard.flet.mita.keio.ac.jp, in pub/Papers/Terui, 1996.

[24] Nikolai Pankrat'ev. On the completeness of the Lambek calculus with respect to relativized relational semantics. *Journal of Logic, Language, and Information*, Vol. 3, pp. 233–246, 1994.

[25] D. M. R. Park. Concurrency and automata on infinite sequences. In P. Deussen, editor, *Proceedings of 5th GI Conference*, pp. 167–183. Springer-Verlag LNCS 104, 1981.

[26] M. Pentus. Language completeness of the Lambek calculus. In *proc. of LICS 94*, pp. 487–496. IEEE, 1994.

[27] Giovanni Sambin. The semantics of pretopologies. In K. Došen and P. Schröder-Heister, editors, *Substructural logics*, pp. 293–307. Oxford University Press, 1994.

[28] Anne S. Troelstra. *Lectures on Linear Logic.* CSLI Lecture Notes 29, Center for the Study of Language and Information, Stanford, California, 1992.

[29] R. J. van Glabbeek. The linear time - branching time spectrum. In J. C. M. Baeten and J. W. Klop, editors, *Proceedings CONCUR '90, Amsterdam*, pp. 278–297. Springer-Verlag LNCS 458, 1990.

[30] R. J. van Glabbeek. The linear time - branching time spectrum II the semantics of sequential processes with silent moves. In E. Best, editor, *Proceedings CONCUR '93, Hildesheim*, pp. 66–81. Springer-Verlag LNCS 715, 1993. Preliminary version available at ftp://Boole.stanford.edu/pub/spectrum.A4.ps.gz.