プログラミング 13−4
(1997. 5. 21)

並行プロセスの非インターリーブ的テスティング

村上 昌己

岡山大学 工学部

〒700 岡山市 津島中 3-1-1

e-mail: murakami@momo.it.okayama-u.ac.jp

Abstract

本稿では、ラベル付き遷移システムを用いた操作的意味論をもつπ計算の試験等価性について述べる. 本稿で提案するプロセスの試験システムはHennessyらによって提案された試験システムの拡張であり、プロセスのテスタに枝分かれ構造を持つ半順序構造を用いるという点が新たな点である. 試験の成功/失敗の基準は、従来のmust/may成功を用いる. これによって従来の半順序構造を用いた意味論に基づく等価性よりも粗であり、かつインターリービングによって生ずる入力prefix演算についての合同性の問題を持たない等価性を定義することが可能となる.

Non-interleaving Testing of Concurrent Processes

Masaki Murakami

Faculty of Engineering, Okayama University

3-1-1 Tsushima-naka, Okayama 700 JAPAN

e-mail: murakami@momo.it.okayama-u.ac.jp

Abstract

This paper presents a new testing equivalence of mobile processes that are defined as a subset of $\pi-$calculus that has the operational semantics defined with the labeled transition system. The testing system presented in this paper is an extension of the testing system that was introduced by Hennessy. We extend the system by adopting partially ordered events with branching structure of processes as the tester. We redefine *must* and *may* equivalences using the new testing system. The equivalence relation defined by the testing is coarser than the equivalence relation that was defined using partial order without the problem of congruence wrt input prefix caused by interleaving.

1 はじめに

Milnerらによって提案され、近年並行計算の形式的モデルとして注目を集めているπ計算[MPW]は、やはりMilnerによって提案されたCCS[Mil89]を拡張したモデルである．π計算は、CCSの特徴を受けつぐと同時に、名前渡し(name passing)という拡張された機能をもち、実行中にプロセス間の通信リンクを動的に生成することができる等、CCSに比べて柔軟で多彩な表現力を備えたモデルとなっている．このような並行計算のモデルについて研究される場合、特に研究の対象となるのが、プロセスの等価性の定義である．CCSでは、双模倣性を基にした等価性として、強等価性、観測合同性などの等価性が研究されており、見掛け上の振舞いの等しさと整合性をもち、かつ合同関係である(すなわち同じと見なされた二つのプロセスは、任意のコンテクストで同じはたらきをする．)ような関係が得られている．

π計算においても、同様に双模倣性に基づく等価性の研究が、提案当初から進められてきた．しかしながら、CCSに名前渡しをつけ加えることによって得られるπ計算の体系に、そのまま従来のCCSにおける相模倣性の定義を拡張したものをあてはめた場合、強等価性すらprefixingについて合同関係にならないという現象が起る．すなわち$a|b$と$ab+ba$を従来の相模倣性に基づいて等しいとしてしまうと、これらの前にある動作を付けくわえたプロセスで双模倣等価にならないものが存在する．このことはさらに、CCSでは並列合成と非決定性の間を関係つける展開規則(expansion law)が成立していたが、π計算ではそれが同様な形では成立しないという現象をも導く．このようにCCSを単純に名前渡しの機能をつけ加えただけでは、従来のCCSでのいくつかの好ましい性質が失われてしまうという問題が生じた．

これらの問題を解決するために、いくつかの研究が報告されている．筆者は先に[Mur97]において、入力プレフィックスの合同性が満足されない原因について、プロセスの操作的意味論が各プロセスの動作列をインターリーブした意味論に基いていることに起因することを指摘した．

その上で、プロセスの実行時に発生する事象の因果性を表現する半順序構造とプロセスの分岐構造を同時に表現する方法を提案し、これによってプロセスの形式的意味を定める方法について述べた．しかしながら、先に述べた意味論は本質的にインターリーブ方式の木構造を半順序構造に拡張した場合と同様な情報量を持ち、半順序を用いた双模倣等価に相当する細かい等価関係を与えるものであった．

インターリーブ方式における双模倣等価は、プロセスの同値分類としては細か過ぎ、より粗な等価関係である失敗集合意味論に基づく等価関係で十分であることが知られている．同様な現象は、半順序構造を用いた場合にも発生する．例えば$(a.b.c+a.c.d)$というプロセスと$a.(b.c+b.d)$というプロセスは、失敗集合意味論及びそれと同等な試験等価性のもとでは、同一視されるが、双模倣等価のもとでは区別されてしまう．[Mur97]において提案した半順序構造を用いた等価性でも、同様にこれらのプロセスを区別してしまう．しかし、外部からの観測に忠実で合同な等価関係を定めるという立場からは、これらのプロセスを区別する必要は無いと考えられる．

本稿では、[Mur97]で提案した、プロセスの分岐構造を表現する半順序構造を、プロセスのテスタとして用いることによって[Henn88, Henn92]の試験システムを拡張する．この試験システムを用いて等価性を定義する．これによって、上記の入力プレフィックスの合同性についての問題を含まず、かつ$(a.b.c+a.c.d)$と$a.(b.c+b.d)$を同一視できる等価性を得ることができる．

2 π−計算

2.1 構文

本稿では[MPW]で提案されたπ−計算のうち、空(nill), プレフィックス(prefix), 和(sum), 並列合成(composition), 制限(hiding)及び不動点演

算 (fixpoint) から構成されるサブセットを対象とする.

$\mathcal{N}$ を名前の集合とする. $x, y \in \mathcal{N}$とするとき Act は以下のように 内部動作 (silent action) τ, $x(y)$ のような入力動作 (input actions) , $\overline{x}y$のような自由出力動作 (free output actions), $\overline{x}(y)$ などの束縛出力動作 (bound output actions) から構成される動作 (action) の集合である.

プロセスの集合は以下に定義される通りである. $E, F, \ldots$ (又は $E(X), F(X), \ldots$) をプロセス変数 X を含むプロセス式 (*expressions*) とする.

nill $\mathbf{0}$

prefix $\overline{x}y.E, \quad x(y).E, \quad \tau.E$

sum $E + F$

composition $E|F$

hiding $(x)E$

本稿では replication '!' は扱わない. ただし再帰的プロセスを扱うため不動点演算子を導入する.

fixpoint $\mathbf{fix}X.E(X)$

また、本稿では *match* 演算子 '$[x = y]P$' も対象とはしない. 何故ならば、この演算子はインターリーブ的手法による展開規則の妥当性を保持するために導入されたものと考えられるからである.

プロセス式全体の集合を $\mathcal{E}$ であらわす. プロセス (*processes*) $P, Q, R, \ldots$ とは、プロセス変数を含まないプロセス式である. プロセス全体の集合を $\mathcal{P}$と表記する.

各演算子の直観的な意味は、並列合成以外は [MPW] で提案されたものと概ね同様である. $\mathbf{0}$ は、全く動作しないプロセスをあらわす. $\overline{x}y.P$ は、最初に名前 y を x というポートを用いて出力し、以降は P と同様に動作する. $\tau.P$ はCCS の場合と同様である. $x(z).P$は、最初に名前 y を x という名前のポートを用いて読み込み、以降は $P\{y/z\}$ と同様に動作する. ここで $P\{y/z\}$ は P の中に出現するすべての z を y で置き換えたプロセスである[1]. $P+Q$ は P 又は Q のように動作する. $(x)P$ は P と同様に動作するが、 x を周囲の環境との通信の subject に用いることはない. $\mathbf{fix}X.E(X)$ は $E(\mathbf{fix}X.E(X))$ と同様に動作する.

本稿で提案する等価性は、$a|b$ と $a.b + b.a$ を区別する立場から、$P|Q$ については通常とは少し異なった意味を与える. $P|Q$ は、P と Q の並列実行である. したがって、$P|Q$ は、P と Q をインターリーブしたようにも動作しうるし、又 P中の動作と Q 中の動作は同時に発生することもある.

本稿では構文的等価性 "$\equiv$" を通常と同じく以下のように定義する. ふたつのプロセス P と Q について $P \equiv Q$ であるとは:

1. P から Q へ α−変換可能のとき、 $P \equiv Q$.
2. $P|0 \equiv P$, $P|Q \equiv Q|P$, $P|(Q|R) \equiv (P|Q)|R$
3. $(x)0 \equiv 0$, $(x)(y)P \equiv (y)(x)P$
4. x が P で束縛された名前であるとき $(x)(P|Q) \equiv P|(x)Q$

"自由な名前", "束縛された名前" などの呼び方は通常通り用いて、$fn(P), bn(P), fn(\alpha)$, $bn(\alpha)$ 等のように表記する. またα に出現する名前の集合を $name(\alpha)$ であらわす. 本稿で対象とするプロセスは、自由な名前として出現する名前の集合と束縛された名前として出現する名前の集合は、互いに素となるように常に適切にα−変換されているものと仮定する.

2.2 並行ラベル付き遷移システム

本稿では、π−計算の意味論として、[MPW] で提案されたラベル付き遷移システムを基本に、並行に実行できる動作は同時に発生することも認めたラベル付き遷移システムを導入する.

プロセス P, P'、動作α について、 $P \xrightarrow{\alpha} P'$ で、[MPW] で導入されたラベル付き遷移システムによって

[1]本稿では束縛された名前の衝突は発生しないよう、適切に名前の付け換えが行なわれているものとする. 例えば、$y(z).\bar{z}a.(z)P$ は α−変換を用いて $y(z).\bar{z}a.(z')P$ とする.

$$\frac{P \xrightarrow{\alpha} P'}{P \xrightarrow{\{\!|\alpha|\!\}} P'}\ \text{multi} \qquad \frac{P \xrightarrow{\tilde{\alpha}} P'}{P+Q \xrightarrow{\tilde{\alpha}} P'}\ \text{Sum} \qquad \frac{P \xrightarrow{\tilde{\alpha}} P' \quad \forall \alpha \in \tilde{\alpha}\ x \notin \mathrm{name}(\alpha)}{(x)P \xrightarrow{\{\!|\tilde{z}(x)|\!\} \uplus \tilde{\alpha} \setminus \tilde{z}x} P'}\ \text{open}$$

$$\frac{P \xrightarrow{\tilde{\alpha}} P'}{P|Q \xrightarrow{\tilde{\alpha}} P'|Q}\ \text{parallel} \qquad \frac{P \xrightarrow{\tilde{\alpha}_1} P' \quad Q \xrightarrow{\tilde{\alpha}_2} Q' \quad \tilde{\alpha}_1 \| \tilde{\alpha}_2 \vdash_X \tilde{\alpha}}{P|Q \xrightarrow{\tilde{\alpha}} (X)(P'|Q')}\ \text{com} \qquad \frac{P \xrightarrow{\tilde{\alpha}} P' \quad \forall \alpha \in \tilde{\alpha} x \notin \mathrm{name}(\alpha)}{(x)P \xrightarrow{\tilde{\alpha}} (x)P'}\ \text{hide}$$

図. 1

「P から α によって P' に遷移可能」であることをあらわす. 以下では動作のマルチ集合を $\tilde{\alpha}$ のように '$\sim$' をつけたギリシャ文字であらわす. マルチ集合$\tilde{\alpha}_1, \tilde{\alpha}_2$ の直和を $\tilde{\alpha}_1 \oplus \tilde{\alpha}_2$ であらわす. α がマルチ集合 $\tilde{\alpha}$ に出現することを、$\alpha \in \tilde{\alpha}$ と表記する. また$\tilde{\alpha}_1$ から 動作 α をひとつ取り除いて得られるマルチ集合を $\tilde{\alpha}_1 \setminus \alpha$ であらわす. また要素が$\alpha_1, \alpha_2, \ldots$ のマルチ集合を $\{\!|\alpha_1, \alpha_2, \ldots|\!\}$ であらわす. また $\tilde{\alpha} \oplus \{\!|\alpha|\!\}$ を $\tilde{\alpha} \oplus \alpha$ と略記する. また $\mathrm{fn}(\tilde{\alpha}), \mathrm{bn}(\tilde{\alpha}), \mathrm{name}(\tilde{\alpha})$ は、それぞれ$\tilde{\alpha}$に出現する自由な名前、束縛された名前、又は名前全体の集合をあらすのに用いる.

定義 2.1 動作のマルチ集合 $\tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\alpha}$及び名前の集合 X について、四項関係 $\tilde{\alpha}_1 \| \tilde{\alpha}_2 \vdash_X \tilde{\alpha}$ を以下のように定める.

1. $\tilde{\alpha}_1$又は$\tilde{\alpha}_2$に出現する束縛出力 $x(y)$ について、yは $\tilde{\alpha}_1$ 及び$\tilde{\alpha}_2$の他の要素には出現しないとき、$\tilde{\alpha}_1 \| \tilde{\alpha}_2 \vdash_\emptyset \tilde{\alpha}_1 \oplus \tilde{\alpha}_2$.

以下では $\tilde{\alpha}_1 \| \tilde{\alpha}_2 \vdash_X \tilde{\alpha}$ とする.

2. $\tilde{\alpha}_1 \oplus \bar{x}y \| \tilde{\alpha}_2 \oplus x(y) \vdash_X \tilde{\alpha} \oplus \tau$
3. $\tilde{\alpha}_1 \oplus x(y) \| \tilde{\alpha}_2 \oplus \bar{x}y \vdash_X \tilde{\alpha} \oplus \tau$.

さらに以下では $y \notin X$ とする.

4. $\tilde{\alpha}_1 \oplus \bar{x}(y) \| \tilde{\alpha}_2 \oplus x(y) \vdash_{X \cup \{y\}} \tilde{\alpha} \oplus \tau$
   ただし $y \notin \mathrm{name}(\tilde{\alpha}_1) \oplus \mathrm{name}(\tilde{\alpha}_2)$.
5. $\tilde{\alpha}_1 \oplus x(y) \| \tilde{\alpha}_2 \oplus \bar{x}(y) \vdash_{X \cup \{y\}} \tilde{\alpha} \oplus \tau$.
   ただし $y \notin \mathrm{name}(\tilde{\alpha}_1) \oplus \mathrm{name}(\tilde{\alpha}_2)$.

定義 2.2 プロセス P, P', 動作のマルチ集合 $\tilde{\alpha}$ について、$P \xrightarrow{\tilde{\alpha}} P'$ は、図 1 に示す規則によって導かれる最小の関係である.

直観的には $P \xrightarrow{\tilde{\alpha}} P'$であるとは、$P$ は $\tilde{\alpha}$ の要素をすべて同時に実行して P' に遷移することができることを意味している. このとき$\tilde{\alpha}$は、必ずしも Pで最初に実行可能な動作をすべて含んでいる訳ではない. この点は [Oda] で提案されたラベル付き遷移システムとは異なっている.

3 Dual Bracket Structure

3.1 DBS の定義

この節では、本稿でテスタとして採用する DBS (*dual bracket structures*) について述べる. DBS とは半順序のついた事象 (event) の集りであり、同時にプロセスの分岐構造を表現している. プロセスの分岐構造は、"$[\![,]\!]$" と "$\langle,\rangle$" という二種類の括弧構造によって表現される. 基本的な規則は以下の通りである.

- 事象 a と b について、両方を同時に囲む最も内側の括弧が $[\![,]\!]$ であった場合、a と b の両方が (両者に順序が無ければ) 並行又は (両者の間に順序があれば) 逐次的に起こる. 一方だけが起こってもう一方が起こらないということはない.
- 両者を囲む最も内側の括弧が $\langle,\rangle$ であった場合、高々一方だけが起こる. a と b の両方とも起こるということはない.

例えば CCS 風に書くと $a.(b+c)$ というような形のプロセスは $a \prec b$ かつ $a \prec c$ で $[\![a, \langle b, c\rangle]\!]$ というここで b と c の両方を囲む括弧が $\langle,\rangle$ となっていることで、b と c は排他的に実行されることをあらわしている. 一方 $a.b + a.c$ のようなプロセスは $\langle [\![a, b]\!], [\![a, c]\!] \rangle$ のように表される. ここ

で二つの a を囲む括弧は $\langle,\rangle$ であり、どちらか一方の a だけが実行されることがわかる.

以下ではDBSの形式的定義について述べる.

S を各元が Act の元でラベル付けされた半順序集合とする. Sは直観的には各動作のオカレンスの集合を意味する. Sの各元を事象(event)と呼ぶ.

定義 3.1 Sを 半順序 $\prec$ を持つ事象の集合とする. $\mathcal{D}_S$ は S から定まる集合で、以下の条件を満足するものであるとする.

- $S_1 \subset S$とするとき、$(\langle,\rangle, S) \in \mathcal{D}_S$ かつ $([\![,]\!], S) \in \mathcal{D}_S$
- $D \subset \mathcal{D}_S$ のとき $(\langle,\rangle, D) \in \mathcal{D}_S$ かつ $([\![,]\!], D) \in \mathcal{D}_S$

言い換えると、$\delta \in \mathcal{D}_S$ とするとき、δは括弧の対 ($\langle,\rangle$ 又は $[\![,]\!]$) と 幾つかの $S \cup \mathcal{D}_S$の元の対からなる.

例えば $a,b,c \in S$のとき、$([\![,]\!], \{a, (\langle,\rangle, \{b,c\})\}) \in \mathcal{D}_S$ である. 以下ではこれを読み易さのため $[\![a, \langle b,c\rangle]\!]$ のように表記する. 以下では混乱が無い限り S を省略し $\mathcal{D}_S$ を $\mathcal{D}$ のように表記する.

定義 3.2 $S_1 \subset S \cup \mathcal{D}_S$ とする. $\delta \in \mathcal{D}_S$, $\lessdot$, $\in$, $\prec$ $(\subset S \cup \mathcal{D}_S \times \mathcal{D}_S)$ 及び $\in^*$ $(\subset S \times \mathcal{D}_S)$ を以下おように定義する.

1. $\delta = (\langle,\rangle, S_1)$ かつ $s \in S_1$ のとき $s \lessdot \delta$.
2. $\delta = ([\![,]\!], S_1)$ かつ $s \in S_1$, のとき $s \in \delta$.
3. $s \in \delta$ 又は $s \lessdot \delta$ のとき $s \prec \delta$.
4. 関係 $\in^* \subset S \times \mathcal{D}_S$ は以下のように定められる.
   (a) $s \prec \delta$ かつ $s \in S$ のとき $s \in^* \delta$.
   (b) $\delta' \prec \delta$ かつ $s \in^* \delta'$ のとき $s \in^* \delta$.

定義 3.3 $s_1, s_2 \in^* \delta$ $(s_1, s_2 \in S)$ について、s_1 と s_2両方を同時に囲む括弧が "$[\![,]\!]$" であったとき、s_1 と s_2はδ で両立であるという. 一方、s_1 と s_2 両方を同時に囲む括弧が "$\langle,\rangle$" であったとき、s_1 と s_2はδで排他であるという.

3.2 DBS の遷移システム

本稿では DBS をテスタとして用いるため、DBS が動作を行なって遷移する様子を表現するラベル付き遷移システムを定義する. DBS のラベル付き遷移システムは、[Mur97] で提案されたDBS の導出 (derivative) の概念を使って定義することができる.

定義 3.4 事象の集合$\hat{\alpha}$ が DBS δ の初期値集合であるとは、任意の $s_1, s_2 \in \hat{\alpha}$ について $s_i \in^* \delta (i = 1,2)$ かつ任意の $s \in^* \delta$について $s \not\prec s_1 (s \not\prec s_2)$ でありかつ s_1, s_2は δ で両立であることをいう.

定義 3.5 δ を DBS とする. δ の *sub-DBS* の集合 $sub(\delta)$ とは、すべての $\delta' \in sub(\delta)$ について以下の条件のいずれかを満足するような最大の集合である.

1. δ' は δであるか、
2. $\delta = \langle \delta_1, \ldots, \delta_i, \ldots \rangle$ であって、ある i について $\delta' \in sub(\delta_i)$ であるか
3. $\delta = [\![\delta_1, \ldots, \delta_i, \ldots]\!]$ かつ$\delta' = [\![\delta'_1, \ldots, \delta'_i, \ldots]\!]$ で、すべての i について $\delta'_i \in sub(\delta_i)$ かつ $\forall s, s' \in^* \delta (s \in^* \delta' \wedge s \prec s' \Rightarrow s' \in^* \delta')$ であるか又は、
4. δ は、ある $S_1 = \{s_1, s_2, \ldots, s_i, \ldots\} \subset S$ について $\delta = ([\![,]\!], S_1)$ ($\delta = [\![s_1, s_2, \ldots, s_i, \ldots]\!]$) であり、$\delta'$ はある $S'_1 \subset S_1$ について$\delta' = ([\![,]\!], S'_1)$ $(= [\![s'_1, s'_2, \ldots, s'_j, \ldots]\!])$ かつ $\forall s'_j \in \delta', \forall s_i \in \delta, (s'_j \prec s_i \Rightarrow s_i \in \delta')$ である.

DBS の集合Δについて、$\delta (\in \Delta)$ が Δ で極大であるとは、すべての $\delta' \in \Delta$ について $\delta \in sub(\delta')$ ならば $\delta' = \delta$ であることをいう.

定義 3.6 δ を DBS、$\hat{\alpha}$ を δの初期値集合とする. Δ_δ を以下のような条件を充たすすべての δ_0 の集合とする.

1. $\delta_0 \in sub(\delta)$ であり
2. すべての $s \in^* \delta_0$ について $s \notin \hat{\alpha}$ かつ、

3. すべての $s \in^* \delta_0$ と すべての1 $s' \in \hat{\alpha}$に‘ついて、s と s' は δで両立である.

δ' が δ の $\hat{\alpha}$ による導出 (derivative) であるとは、δ' が集合 Δ_δ で極大であることをいう.

δ' が δ の $\hat{\alpha}$ のよる導出であるとき、$\delta \xrightarrow{\hat{\alpha}} \delta'$と表記する.

4 試験システムと試験等価性

定義 4.1 名前の集合$\mathcal{N}$について、Pを$\mathcal{N}$上で定義されたプロセス、δ を $S_{\mathcal{N}\cup\{\omega\}}$ の上で定義された DBS とする. $P \parallel \delta$ を P のδ による試験システムという. このときδ を P のテスタという.

事象の集合 $\hat{\alpha}$ に出現する事象のτ以外のラベルを重複も含めてすべて集めたマルチ集合を $labels(\hat{\alpha})$ であらわす. また動作のマルチ集合 $\tilde{\alpha}$ から、τの出現をすべてとり除いたものを $\tilde{\alpha}\downarrow$ と表記する.

定義 4.2 試験システム $P \parallel \delta$の計算の集合 $\mathrm{Comp}(P,\delta)$ とは、次のような系列の集合である.

$$P_0 \parallel \delta_0 \to P_1 \parallel \delta_1 \to \cdots \to P_i \parallel \delta_i \to \cdots$$

ここで $P_0 \parallel \delta_0 = P \parallel \delta$ かつ各 i について、次のいずれかが成立する.

- $P_i \xmapsto{\tilde{\alpha}} P_{i+1}$、$\delta_i \xrightarrow{\hat{\alpha}} \delta_{i+1}$かつ $labels(\hat{\alpha}) = \tilde{\alpha}\downarrow$ であるか、
- $P_i = P_{i+1}$ かつ $labels(\hat{\alpha}) = \{\tau\}$ であるか、又は
- $\delta_i = \delta_{i+1}$ かつ $\tilde{\alpha}\downarrow = \{\!|\}$ である.

ここで $\{\!|\}$ は、空なマルチ集合をあらわす. ここで試験システムの計算については、τによる遷移とτをラベルに持つ事象の対応は問題にしていないので、以下で定まる等価性はτを考慮しない等価性であることに注意されたい. この点でも [Henn92] とは異なるものとなっている.

定義 4. 3 $P_0 \parallel \delta_0 \to \cdots \to P_i \parallel \delta_i \to \cdots$ が成功する計算であるとは、ある i についてδ_i の初期値集合にωをラベルとして持つ事象が含まれることをいう.

定義 4.4 $\mathrm{Comp}(P,\delta)$ のすべての元が成功する計算であるとき、P **must** δ と表記する. また $\mathrm{Comp}(P,\delta)$ に成功する計算が少なくともひとつ含まれるとき、P **may** δと表記する.

これによって [Henn88] の場合と同様に $P \sqsubseteq_{\mathbf{must}} Q$、$P \sqsubseteq_{\mathbf{may}} Q$、$P \sqsubseteq Q$、$P =_{\mathbf{may}} Q$、$P =_{\mathbf{must}} Q$、$P = Q$ といった順序関係、等価関係を定義することができる.

このようにして定義された等価関係 $=$ は、[Mur97] で提案したたものに対して、$a|b$ と $ab + ba$ を区別してこれらの入力プレフィックスについての合同性の問題を回避しているという点では同様であるが、τ を無視するという点と $(a.b.c + a.c.d)$ と $a.(b.c + b.d)$ を同一視できるという点でより粗な等価関係となっている.

謝辞: 岡山大学山崎進教授及び小田幸弘君他知能情報処理工学研究室の皆様には、有益な議論と研究上の御支援をいただきました. 記して感謝します.

参考文献

[Henn88] Hennessy M., "Algebraic Theory of Processes", *The MIT Press*(1988)

[Henn92] Hennessy M., Concurrent Testing of Processes, CONCUR'92, Lecture Notes in Computer Science 630 (1992) pp. 94-107

[MPW] Milner R., J. Parrow and D. Walker, A Calculus of Morbile Processes, Part1, Part2, LFCS Report Series, ECS-LFCS-89-85 and 86, University of Edinburgh, (1989)

[Mil89] Milner R. , *Communication and Concurrency*, Prentice Hall (1989)

[Mur97] Murakami M. , A Non-Interleaving Semantics of Mobile Processes, submitted to FORTE/PSTV 97 (1997) (`http://www.momo.it.okayama-u.ac.jp/~murakami/dvi/non-inter.dvi`)

[Oda] Oda, Y. and M. Murakami, Multi-action π-calculus, RIMS Workshop in Computing, Concurrency Theory and Applications '96, (1996)