構造化通信を基礎とした プログラミング言語の簡約意味論

久保　誠

1．はじめに

近年，コンピュータネットワーク社会では，ネットワーク上の通信を基礎とするプログラミングは，非常に重要である。ネットワークプロトコル記述，サーバー／クライアント間の通信モデル，WWWにおける動的ロボット検索モデルなどは，並行プログラムであるが，複数のプロセス間の相互的で複雑な通信の実行が重要な要素となる。そのような背景において，通信を基礎とするソフトウェア記述のためのプログラミング言語および形式モデルが多く提案されてきたが，基本通信プリミティブを単独あるいは組み合わせた形で記述して行くことになる。それらは固定した一度かぎりの相互作用（call-returnやメッセージ渡し）を表現する通信プリミティブはあるものの，2者間の一連の通信による相互通信をひとまとまりのものとして構造化するための構成子が与えられていないということである。つまり，サーバー・クライアント間の通信セッションのような一連のやりとりを表現する方法は，構文上関係のない通信の集まりとしてしか記述できない。これらは複雑なネットワークアプリケーションにおいて，構造化手法の欠如により可読性の低さや誤動作の原因となる。特に，複数の相手との連続的な相互通信が混在する場合，それぞれを別個の構造に属するものとして抽象・記述する方法がないことは，抽象化手法としてのみならず，通信パターンの整合性の検証などの実際的な目的のためにも大きな問題となる。そこで必要なことは，複雑な通信構造を持つ並行プログラムの抽象化手法の提供と，様々な正当性の検証の基盤の提供である。そのために，一番基礎通信プリミティブの選択，構造化手法として用意すべき構造の2点が重要になる。

すでに上記に述べたことは，［4］において，通信に基づくプログラミング（communication-based programming）のための構造化プリミティブの提案および

［6］においてそのプリミティブに対する型システムを提案した。これらの論文では，きわめて単純な基礎プリミティブを構造化手法により組み合わせることで，複雑な相互作用にもとづく並行プログラムを見通しよく構成することが可能になることを示す。これらの構成子の動作は単純かつ厳密な操作意味論によって定義され，従来の様々な通信プリミティブは，これらの構成子の組み合わせとして容易に表現される。さらに，型システムにおいては，プロセスにおける相互作用パターンの一致の自動検証を ML［10］にみられるような型推論アルゴリズムを用いて可能となることを示し，構造化手法において重要な枠割を担っていることを示している。

本論文では，上記の論文で示していなかった構造化プリミティブの操作意味論の詳細，および，その上で成り立つ観測あるいは停止性の概念に基づかない等価理論の定式化を提供する。まず，構造化プリミティブの構文定義をあたえ，その操作的意味論を形式的にあたえる。つぎに，主題である操作的意味論での簡約を基にした等価性の提案とそれが有する基本的結果を示し証明する。最後に，構造化プリミティブの型との関連性を示す。

2．構文と操作的意味論

2．1　基本概念

構造化プリミティブの特徴は以下のとおりである。

1．構造化のための基本概念はセッション

セッションは，二者間の連続かつ相互的な通信であり，可能であれば，分岐や再帰を可能にする。そして，通信記述の抽象的構造の単位にもなる。セッションによる通信は，チャネルと呼ばれるセッション毎に決まったポートを用いて行う。チャネルは各セッションの開始時に生成され，全ての通信はチャネルを用いて行われる。

2．3つの基礎通信プリミティブは，値渡し，ラベル分岐，デレゲーション

生成されたチャネルを使用したセッションにおける2者間の通信は，三種類の基本的な通信プリミティブで行われる。

データ送信/受信　　$k![e_1,\cdots,e_n];P$　　$k?(x_1,\cdots,x_n)$ **in** P

ラベル選択/分岐　　$k \lhd l;P$　　$k \rhd \{l_1 : P_1 \| \cdots \| l\mathrm{n} : P\mathrm{n}\}$

チャネル送信／受信　throw $k[k']\,;P$　　catch k (k') in P

3．プリミティブに対する型付けは，構造化手法の欠かすことの出来ない要素

2．2　構文定義

本節では，構造化プリミティブを形式的に定義する。

まず，構文定義に用いられる基本的な構文要素として，名前は $a, b, \cdots$，チャネルは $k, k', \cdots$，変数は $x, y, \cdots$，定数（名前，整数，真偽値を含む）は $c, c', \cdots$，式（定数を含む）は $e, e', \cdots$，ラベルは $l, l', \cdots$，プロセス変数は $X, Y, \cdots$，プロセスは $P, Q, \cdots$で表現される。また，$u, u', \cdots$は名前あるいはチャネルを示すものとする。

構造化プリミティブを含む言語の構文は以下のとおりである。

P	::=	request $a(k)$ in P	セッション開始
	|	accept $a(k)$ in P	セッション開始
	|	$k![e_1,\cdots,e_n]\,;P$	データ送信
	|	$k?(x_1,\cdots,x_n)$ in P	データ受信
	|	$k \lhd l\,;P$	ラベル選択
	|	$k \rhd \{l_1 : P_1 \| \cdots \| l_n : P_n\}$	ラベル分岐
	|	throw $k[k']\,;P$	チャネル送信
	|	catch $k(k')$ in P	チャネル受信
	|	if e then P else Q	条件分岐
	|	$P_1 \mid P_2$	並行合成
	|	Inact	不活性プロセス
	|	$(\nu u)P$	名前/チャネル制約
	|	def D in P	再帰
	|	$X[\tilde{e}\tilde{k}]$	プロセス変数
D	::=	$X_1(\tilde{x}_1,\tilde{k}_1)=P_1$ and $\cdots$ and $X_n(\tilde{x}_n,\tilde{k}_n)=P_n$	再帰のための定義

並行合成における｜が結合でもっとも弱く，残りは同等である。括弧 () は，対応する自由な出現を束縛する。また，標準的な同時置換 $P\,[\tilde{c}/\tilde{x}]$を使用する。プロセスPの自由な名前，チャネル，変数，プロセス変数の集合は，標準的な方法で定義され，それぞれ $\mathrm{fn}(P)$, $\mathrm{fc}(P)$, $\mathrm{fv}(P)$, $\mathrm{fpv}(P)$ であらわす。α 等価性は $\equiv_\alpha$であらわす。また，$\mathrm{fu}(P)=\mathrm{fc}(P)\cup\mathrm{fn}(P)$である。自由な変数もしくは自由なチャネルのないプロセスをプログラムと呼ぶ。また，文脈 C, C',…は通常通り以下のように定義される。

C ::= [] | (νu) C | (C|P) | (P|C) | k?($x_1,\cdots,x_n$) in C

同値関係は，文脈および置換に関して閉じている場合は，合同関係として定義される。

2．3　操作的意味論

構文定義の次に，動作を形式的に表現する操作的意味論を定義する。

まず，操作的意味論を定義する前に構造同値性を定義する。

定義　構造合同関係≡

1．$P \equiv P'$　if $P \equiv_\alpha P'$

2．$P \mid Q \equiv Q \mid P$　　$(P \mid Q) \mid R \equiv P \mid (Q \mid R)$　　$P \mid \mathbf{inact} \equiv P$

3．$(\nu uu)\ P \equiv (\nu u)P$　　$(\nu uu')\ P \equiv (\nu u'u)P$　　$(\nu u)\mathbf{inact} \equiv \mathbf{inact}$

$(\nu u)\ P \mid Q \equiv (\nu u)(P \mid Q)$　　$u \notin \mathrm{fu}(Q)$

$(\nu u)\mathbf{def}\ D\ \mathrm{in}\ P \equiv \mathbf{def}\ D\ \mathbf{in}\ (\nu u)P$　　$u \notin \mathrm{fu}(D)$

4．$(\mathbf{def}\ D\ \mathbf{in}\ P) \mid Q \equiv \mathbf{def}\ D\ \mathbf{in}\ (P \mid Q)$　　if $\mathrm{fpv}(D) \cap \mathrm{fpv}(Q) = \emptyset$

5．$\mathbf{def}\ D\ \mathbf{in}\ (\mathbf{def}\ D'\ \mathbf{in}\ P) \equiv \mathbf{def}\ D\ \mathbf{and}\ D'\ \mathbf{in}\ P$

if $\mathrm{fpv}(D) \cap \mathrm{fpv}(D') = \emptyset$

次に，操作的意味論について簡約関係を使って定義する。

定義　簡約関係→

簡約関係→は，P→Qの形で表し，以下の条件を満たすプロセス間の最小な関係である。

[Link]　$(\mathbf{accept}\ a(k)\ \mathbf{in}\ P_1) \mid (\mathbf{request}\ a(k)\ \mathbf{in}\ P_2) \rightarrow (\nu k)(P_1 \mid P_2)$

[Com]　$k![\tilde{e}]\,;P_1 \mid \mathrm{k}?(\tilde{x})\ \mathrm{in}\ P_2 \rightarrow P_1 \mid P_2[\tilde{c}/\tilde{x}]$　　$(\tilde{e} \downarrow \tilde{c})$

[Label]　$(k \lhd l_i\,;P) \mid (k \rhd \{l_1 : P_1 \parallel \cdots \parallel l_n : P_n\}) \rightarrow P \mid P_i$　$(1 \leqq i \leqq \mathrm{n})$

[Pass]　$\mathbf{throw}\ k[k']\,;P_1 \mid \mathbf{catch}\ k(k'')\ \mathrm{in}\ P_2 \rightarrow P_1 \mid P_2[k'/k'']$

[IF 1]　$\mathbf{if}\ e\ \mathbf{then}\ P_1\ \mathbf{else}\ P_2 \rightarrow P_1$　　$(e \downarrow \mathbf{true})$

[IF 2]　$\mathbf{if}\ e\ \mathbf{then}\ P_1\ \mathbf{else}\ P_2 \rightarrow P_2$　　$(e \downarrow \mathbf{false})$

[DEF] **def** D **in** $(X[\tilde{e}\tilde{k}] \mid Q) \rightarrow$ **def** D **in** $(P[\tilde{c}/\tilde{x}] \mid Q)$

$(\tilde{e} \downarrow \tilde{c}\ ,\ X(\tilde{x}\tilde{k}) = P \quad \in D)$

[Scop] $P \rightarrow P' \Rightarrow (\nu u)P \rightarrow (\nu u)P'$

[Par] $P \rightarrow P' \quad \Rightarrow \quad P \mid Q \rightarrow P' \mid Q$

[Str] $P \equiv P'$ **and** $P' \rightarrow Q'$ **and** $Q' \equiv Q \quad \Rightarrow \quad P \rightarrow Q$

式の評価関係↓は，標準的な方法で定義されているとする。また，簡約関係→の反射推移閉包を$\rightarrow^*$であらわす。**複数簡約**$\rightarrow\rightarrow$は，$\rightarrow^* \cup \equiv$と定義される。

規則のうち，注意すべき項目のみあげる。規則［Link］は，相互通信の結果，新しいチャネル（k）を生成していることをしめしている。規則［Label］は分岐の一つ（P_i）を選択し，それ以外の分岐は捨てられることをしめしている。

最後に，前述した二つのプロセスがどのように簡約されるかを示す。**inact** は省略されていることに注意。

request a (k) **in** k?(x) **in** $k![x+1]$ | **accept** $a(k)$**in** k![1] ; $k?(y)$ **in** P

$\rightarrow (\nu k)(k?(x)$ **in** $k![x+1]$ | $k![1]$;$k?(y)$ **in** $P)$

$\rightarrow (\nu k)(k![1+1]$ | $k?(y)$ **in** P $) \rightarrow P\,[2/y]$

次に，観測に基づく動作の等価性を定義する。プロセスラベル pl は，以下の構文におって与えられる。

pl ::= τ | ～ab | ab | ～a(b)

（b）は，束縛的出現である。ラベル pl に（束縛されて／自由に）出現する名前の集合を，(BN(pl) / FN(pl)) と書く。また，ラベル pl がPに対して，BN(pl) ∩FN(P) $=\phi$という条件を満たすとき，**適切**（relevant）であるという。以下に観測の概念を表現するラベルつき遷移系を与える。

定義　ラベルつき遷移系

ラベル遷移系$\rightarrow_s$は閉式間に以下の規則で定義される最小の関係である。ただしPARでは，BN(pl) ∩ FN(R) $=\phi$，RESではa $\notin$ FN(pl) ∪ BN(pl)，OPENで

はa ≠ bとする。

IN：$k?(\tilde{x})$ in $P \xrightarrow{ab}_s P\,[\tilde{b}/\tilde{x}]$

OUT：$k![\tilde{e}];\ P \xrightarrow{\bar{a}b}_s P$

COM：$k![\tilde{e}];\ P_1 \mid k?(\tilde{x})$ in $P_2 \xrightarrow{\tau}_s P_1 \mid P_2[\tilde{c}/\tilde{x}]$

$(\tilde{e} \downarrow \tilde{c})$

REP：def D in $(X[\tilde{e}\tilde{k}] \mid Q) \xrightarrow{\tau}_s$ def D in $(P[\tilde{c}/\tilde{x}] \mid Q)$

$(\tilde{e} \downarrow \tilde{c},\ X(\tilde{x}\tilde{k}) = P \in D)$

PAR：$P \xrightarrow{pl}_s P' \Rightarrow P \mid Q \xrightarrow{pl}_s P' \mid Q$

RES: $P \xrightarrow{pl}_s P' \Rightarrow (\nu u)P \xrightarrow{pl}_s (\nu u)\,P'$

OPEN: $P \xrightarrow{\bar{a}<b>}_s P' \Rightarrow (\nu u)\,P \xrightarrow{\bar{a}<b>}_s P'$

STR: $P \equiv P'$ and $P' \to Q'$ and $Q' \equiv Q \Rightarrow P \to Q$

$\Rightarrow_s$ は，$pl = \tau$ ならば，$\to\to$を，他の場合は$\to\to \xrightarrow{pl}_s \to\to$を示す。

INは，同期相互作用に適合している。観測において同期が必須であるという概念に基づいている。続けて強相互模倣性（strong bisimilarity）と弱相互模倣性（weak bisimilarity）の定義を与える。Rは項上の2項関係を表現している。

定義　強相互模倣関係

通信プリミティブの閉式上の対称な関係Rは，以下の条件を満たしたとき，強相互模倣関係（strong bisimilation）とよばれる。すなわち，もしPRQであり，に対して適切なplに関して$P \xrightarrow{pl}_s P'$ならば，いつでもあるQ'があって$Q \xrightarrow{pl}_s Q'$かつP' RQ'。

このような関係で最大のものが（標準的に）存在し，これは強相互模倣性（Strong bisimilarity）とよばれ，$\sim_s$と書かれる。次に弱相互模倣性（Weak bisimilarity）を定義する。

定義　弱相互模倣関係

通信プリミティブの閉式上の対称な関係Rは，以下の条件を満たしたとき，弱相

互模倣関係（weak bisimilation）とよばれる。すなわち，もしPRQであり，それに対して適切なplに関してP $\xrightarrow{pl}_s$ P' ならば，いつでもあるQ'があって$Q \overset{pl}{\Rightarrow}_s Q'$かつ$P'$ RQ'。

このような関係で最大のものが（標準的に）存在し，これは弱相互模倣性（Weak bisimilarity）とよばれ，$=_s$と書かれる。

ラベル付遷移関係や観測可能性の概念を用いる限りでは，多種多様な目的に即した意味論を定義できる。これは，規範的な意味理論を与えることはできないことを意味している。ここで停止性の概念を用いても同様である。そこで，観測可能概念に基づかない規範的な意味理論を構築することが重要であり，そのためには項の等価性を中心に議論する前に，等価式が等式理論によって生成されることを示すことで，様々の異なった等価性を要領よく扱うことができるようになる。

定義　理論

理論（Theory）とは，P＝Qという形をした式が少なくとも以下の公理と推論規則によって作られる等式理論である。

(1)P≡Q のとき　P=Q

(2)P=Q ⇒ Q=P

(3)P=Q, Q=R ⇒ P=R

(4)P=Q ⇒ P | R = Q | R

(5)P=Q ⇒ R | P = R | Q

(6)P=Q ⇒ (νu)P = (νu)Q

(7)P=Q ⇒ **catch** $k(k'')$**in** P = **catch** k(k'') ***in*** Q

(8)P=Q ⇒ R ; P = R ; Q

(9)P=Q ⇒ P [c/x] = Q [c/x]

(10)P=Q ⇒ $k?(\tilde{x})$ **in** P = $k?(\tilde{x})$ **in** Q

以下にいくつかの記法を導入する。理論をT, T'で示す。上の公理と規則以外，なにも付け加えない理論を$T_{\equiv}$とする。Tによって P=Qが導出されるとき，T ⊢ P=Qと書く。また導出する理論が文脈により明らかなときは単にP=Qと書く。また，

そうでない場合はT $\nvdash$ P=Q と書く。ある与えられた等式の集合EをTに加えたものをE+Tと書く。またE+$T_{\equiv}$をE+と略記する。T+T'は上の公理と規則に2つの理論の等式の集合を加えたものである（ただし規則の和をとらない）Iをインデックス集合とすると，これを理論の任意の族に拡張したものを$\Sigma\{T_i|i\in I$と書く。理論Tから導出される関係を|T|と書く。ある理論の族が与えられたとき，最大，最小,極大,極小な理論とは，それらに対応する合同関係について最大，最小，極大，極小なものとする。また，もし|T|$\subset$|T'| であれば，TをT'の部分理論といい，もしこの包含関係が真包含関係ならば，TをT'の真部分理論とよぶ。ある理論が全ての項を等価にしない場合を無矛盾という。逆に無矛盾でない場合，その理論は矛盾しているという。

3．型システム

この節では，観測で示されたラベルとの関連する型システムの定義と概要を述べることにする。

3．1　通信型

構造化手法を提案するのは，従来の通信プリミティブを超える複雑な通信構造を明示的に記述できるようにする。しかし，複雑になればなるほど全体の通信に関わる動作を捉えたり，正しいプログラムを記述することが難しくなる。そこで，通信動作に対し，型付けする規則を単純な記法を使って実現することがよいと思われる。そのために，まず相互通信の双対性を表現するような型に関する記法を導入する。

定義　通信型

型変数は$t, t', \cdots$と記述し，ソート変数はs, s'と記述し，ソートは，S, S'と記述し，型はα, βと記述すると，ソートや型は以下のように定義される。

$S=$ nat | bool | $\left\langle \alpha, \overline{\alpha} \right\rangle$ | s | $\mu s.\ S$

$\alpha=$ $\downarrow[S];\alpha$ | $\downarrow[\alpha];\beta$ | $\&\{l_1:\alpha_1, l_n:\alpha_n\}$ | 1 | $\bot$

$\uparrow[S];\alpha$ | $\uparrow[\alpha];\beta$ | $\oplus\{l_1:\alpha_1, l_n:\alpha_n\}$ | t | $\mu t.\alpha$

また，型αの補型$\bar{\alpha}$（$\bot$の補型は定義に現れない）は，以下のように定義される。

$\overline{\uparrow[\tilde{S}];\alpha} = \downarrow[\tilde{S}];\bar{\alpha}$　$\overline{\oplus\{l_i:\alpha_i\}} = \&\{l_i:\bar{\alpha}_i\}$　$\overline{\uparrow[\alpha];\beta} = \downarrow[\alpha];\bar{\beta}$　$\bar{1} = 1$　$\overline{\mu t.\alpha} = \mu t.\bar{\alpha}$

$\overline{\downarrow[\tilde{S}];\alpha} = \uparrow[\tilde{S}];\bar{\alpha}$　$\overline{\&\{l_i:\alpha_i\}} = \oplus\{l_i:\bar{\alpha}_i\}$　$\overline{\downarrow[\alpha];\beta} = \uparrow[\alpha];\bar{\beta}$　$\bar{t} = t$

ソーティング（Sorting）は，名前あるいは変数からソートへの有限部分写像，タイピング（Typing）は，チャネルから型への有限部分写像，ベーシス（Basis）は，プロセス変数からソートと型の列への有限部分写像である。ソーティングをΓ，Γ'，タイピングをΔ，Δ'，ベーシスをΘ，Θ'であらわす。型とソートは，標準的な方法で対応する木構造に表現し，その構造で等価性を与える。

ソートの$<\alpha, \bar{\alpha}>$は，名前に関連付けられた相互通信に関する二つの双対構造をあらわしている。その一つはacceptで始まる通信動作，もう一つは，requestで始まる通信動作に関連付けられている。一方，型は，チャネルを通じて行われた通信の抽象になっている。ここで，αが定義されているならば，$\bar{\bar{\alpha}} = \alpha$である。⊥は，与えられた名前にこれ以上関連付けられない状態になる場合の特殊な型である。

以下に定義するタイピングの集合への部分代数の妥当性は，型システムで双対性の検証のために使用されるために重要である。

定義2　型代数における妥当性

タイピングΔ_0，Δ_1が妥当である（$\Delta_0 \approx \Delta_1$と書く）とは，

全ての$k \in \mathrm{dom}(\Delta_0) \cap \mathrm{dom}(\Delta_1)$に対して$\Delta_0(k) = \Delta_1(k)$である。

$\Delta_0 \approx \Delta_1$であるとき，$\Delta_0$と$\Delta_1$の合成（$\Delta_0 \circ \Delta_1$と書く）は，以下のような条件を満たすタイピングである。

$(\Delta_0 \circ \Delta_1)$(k) は，

(1)$k \in dom(\Delta_0) \cap dom(\Delta_1)$　ならば，⊥

(2)$k \in dom(\Delta_0) \setminus dom(\Delta_{i+1 \bmod 2})$　$(i \in \{0,1\})$　ならば，$\Delta_i(k)$

(3)それ以外は,定義されない。

妥当性（Compatibility）は，各共有チャネルkに対し双対な通信動作が関連付けられていることを意味しており，チャネルkを使った通信がエラーなく実行されることを保証するものである。合成したときに，kに対する型が⊥になるのは，kでのそれ以上の通信動作への関連を終わらせるためである。なぜなら，⊥は補型をもたないからである。また部分演算（$\circ$）は，部分的に交換的かつ結合的である。

3．2　型システム

型システムの主をなすシーケント（sequent）は，$\Theta;\Gamma \vdash P \succ \Delta$の形をし，「環境$\Theta$；$\Gamma$のもとで，プロセスPはタイピング$\Delta$をもつ」と読む。ソーティング$\Gamma$は，Pの自由な名前に対する動作を特定し，タイピングΔは，その自由なチャネルでのPの通信動作を特定している。Pがプログラムであるときは，ΘとΔはともに空集合になる。

タイピング，または，ソーティングを与えたとき（以下Φであらわす）に，$\Phi \cdot s:p$ は，$s \notin \mathrm{dom}(\Phi)$ という条件において，$\Phi \cup \{s:p\}$となる。$\Phi, s:p$ は，$s \notin \mathrm{dom}(\Phi)$，または，$\Phi(s) = p$ という条件でのみ$\Phi \cup \{s:p\}$となる。$\Phi \setminus s$は，$\Phi(s)$ が定義されているならばΦから$s:\Phi(s)$ を取り除いた結果である。また算術演算，論理演算に関する型推論規則はMLなどの従来の型推論システムと同様の規則が与えられていると仮定する。そのシーケントは，$\Gamma \vdash e \succ a$ という形を持ち，$\Gamma \vdash e \succ \mathrm{S}$かつ，$e \downarrow c$ならば，$\Gamma \vdash c \succ \mathrm{S}$という性質を有する。いよいよ，型システムの定義をする。

定義3　型システムは，以下の公理および推論規則により定義される。

$$[\mathrm{ACC}]\ \frac{\Theta;\Gamma \vdash P \succ \Delta \cdot k : a}{\Theta;\Gamma, a:<a, \overline{a}> \vdash \textbf{accept}\ a(k)\ \textbf{in}\ P \succ \Delta}$$

$$[\mathrm{REQ}]\ \frac{\Theta;\Gamma \vdash P \succ \Delta \cdot k : \overline{a}}{\Theta;\Gamma, a:<a, \overline{a}> \vdash \textbf{request}\ a(k)\ \textbf{in}\ P \succ \Delta}$$

$$[\mathrm{SEND}]\ \frac{\Gamma \vdash \tilde{e} \succ \tilde{S} \qquad \Theta;\Gamma \vdash P \succ \Delta \cdot k : a}{\Theta;\Gamma \vdash k![\tilde{e}];P \succ \Delta \cdot k:\uparrow[\tilde{S}];a}$$

$$[\mathrm{RCV}]\ \frac{\Theta;\Gamma \cdot \tilde{x}:\tilde{S} \vdash P \succ \Delta \cdot k:a}{\Theta;\Gamma \vdash k?(\tilde{x})\ \ \textbf{in}\ \ P \succ \Delta \cdot k:\downarrow[\tilde{S}];a}$$

$$[\mathrm{BR}]\ \frac{\Theta;\Gamma \vdash P_1 \succ \Delta \cdot k:a_1 \quad \cdots \quad \Theta;\Gamma \vdash P_n \succ \Delta \cdot k:a_n}{\Theta;\Gamma \vdash k \rhd \{l_1:P_1 \,\|\, \cdots \,\|\, l_n:P_n\} \succ \Delta \cdot k:\&\{l_1:a_1 \cdots, l_n:a_n\}}$$

$$[\mathrm{SEL}]\ \frac{\Theta;\Gamma \vdash P \succ \Delta \cdot k:a_j}{\Theta;\Gamma \vdash k \lhd l_j;\ P \succ \Delta \cdot k:\oplus\{l_1:a_1, \cdots, l_n:a_n\}}\ (1 \leq j \leq n)$$

$$[\mathrm{THR}]\ \frac{\Theta;\Gamma \vdash P \succ \Delta \cdot k:\beta}{\Theta;\Gamma \vdash \ \mathbf{throw}\ \ k[k'];P \succ \Delta \cdot k:\uparrow[a];\beta \cdot k':a}$$

$$[\mathrm{CAT}]\ \frac{\Theta;\Gamma \vdash P \succ \Delta \cdot k:\beta \cdot k':a}{\Theta;\Gamma \vdash \ \mathbf{catch}\ \ k(k')\ \ \mathbf{in}\ \ P \succ \Delta \cdot k:\downarrow[a];\beta}$$

$$[\mathrm{CONC}]\ \frac{\Theta;\Gamma \vdash P \succ \Delta \qquad \Theta;\Gamma \vdash Q \succ \Delta'}{\Theta;\Gamma \vdash P \mid Q \succ \Delta \circ \Delta'}\ (\Delta \approx \Delta')$$

$$[\mathrm{IF}]\ \frac{\Gamma \vdash e \succ \mathrm{bool} \quad \Theta;\Gamma \vdash P \succ \Delta \quad \Theta;\Gamma \vdash Q \succ \Delta'}{\Theta;\Gamma \vdash \mathbf{if}\ \ e\ \ \mathbf{then}\ \ P\ \ \mathbf{else}\ \ Q \succ \Delta}$$

$$[\mathrm{NRES}]\ \frac{\Theta;\Gamma \cdot a:S \vdash P \succ \Delta}{\Theta;\Gamma \vdash (\nu a)P \succ \Delta} \qquad [\mathrm{CRES}]\ \frac{\Theta;\Gamma \vdash P \succ \Delta \cdot k \cdot \bot}{\Theta;\Gamma \vdash (\nu k)P \succ \Delta}$$

$$[\mathrm{INACT}]\quad \Theta;\Gamma \vdash \mathbf{inact} \succ \Delta$$

$$[\mathrm{VAR}]\ \frac{\Gamma \vdash \tilde{e} \succ \tilde{S}}{\Theta,\ X:\tilde{S}\,\tilde{a};\Gamma \vdash X[\tilde{e}\tilde{k}] \succ \Delta \cdot \tilde{k}:\ \tilde{a}}$$

$$[\mathrm{DEF}]\ \frac{\Theta;\Gamma \cdot \tilde{x}:\ \tilde{S} \vdash P \succ \tilde{k}:\ \tilde{a} \qquad \Theta;\Gamma \vdash Q \succ \Delta'}{\Theta;\Gamma \vdash \mathrm{def}\ \ X(\tilde{x}\tilde{k}) = P\ \ \mathbf{in}\ \ Q \succ \Delta}\ (\Theta(X) = \tilde{S}\,\tilde{a})$$

ここで，［INACT］と［VAR］でのΔの値域は，1および⊥のみである。単純化のために［DEF］は，単一の再帰に限定されている。ただし，これを多重再帰に拡張することは簡単に出来る。

もし，型システムからΘ;Γ |- P ≻ Δを導出できるとき，PはΘ;Γ下でΔに型付け可能である，あるいは，Pは型つけ可能であるという。

以下に，型システムに関わる補足をあげる。

(1)型システムで，⊢ の左側は共有な名前や変数のために存在し，右側は2者間でのみ共有されるチャネルに関してのみ存在する。特にチャネルに対し違ったソーティングがなされる点が従来のソーティングに関するシステムと違う点である。例えば，チャネル上であるときは整数を送信したり，真偽値を送信したりすることがありえる。それにもかかわらず，型代数により，Δの妥当性を検証する過程で，この種の通信エラーをさけることができる。

(2) [THR] は，チャネルk'に対してaによって表現される動作が，実はk'を「catch」するプロセスによって実行されていることを示している。ここで，k'はΔにもPにも自由に出現していないことに注意。これは，規則中でタイピングに関わる記法の「・」で結合されている点からわかる。以降k'を使用した通信全てを知るために$k':a$は右側に付加されていく。一方，[CAT] では，チャネルk'を受け取ったプロセスが受け取る前の通信動作に足されるようになっている。仮定から結論に読むとわかるように，[THR] と [CAT] でk'を [THR] から [CAT] に "throw" しているように見えるのである。

(3)型システムの規則の単純さは，セッションの明示的な構文の構造に有用である。しかし，単純で取り扱いやすいからといって，得られる情報がないわけではなく，システムにより作られた型抽象はプログラムの通信に関わる動作を示していることがわかる。これは[4] [6]にプログラム例を交えて説明されているので参照のこと。

3．3　型システムの基本性質

型システムの基本的な構文的性質を与える。その前に，以下の記法を与える。k-プロセスとは，kを宛先として前置したプロセスのことであり，例えば $k![e]; P$ や **catch** $k(k')$**in** P のようなプロセスのことである。k-簡約子（redex）とは，$|$によって並行合成された二つのk-プロセスの対である。例えば，$(k![e]; P \mid k?(x)\textbf{in}\ Q)$ がそうである。Pがエラーであるとは，$P = \textbf{def}\ D\ \textbf{in}\ (\nu\, u)(P' \mid R)$ の形をしており，かつ，P'はあるkに対して，k-簡約子でない二つのk-プロセスであるか，三つ以上のk-プロセスのどちらかの形を$|$による並行合成したプロセスであることと同義である。

このような記法のもとで，以下の基本かつ重要な性質があたえられる。

定理

1．（構造合同性かでの不変性）

$\Theta;\Gamma \vdash P \succ \Delta$ かつ $P \equiv Q$ ならば，$\Theta;\Gamma \vdash Q \succ \Delta$

2．（主簡約定理）

$\Theta;\Gamma \vdash P \succ \Delta$ かつ $P \rightarrow^* Q$ ならば，$\Theta;\Gamma \vdash Q \succ \Delta$

3．（実行時エラーの回避）

型付け可能なプログラムは，エラーなプロセスに簡約されない

（証明）証明は，厳密には型システムの導出木および構造合同性の定義規則および簡約の各規則と帰納法を利用することで証明可能。証明の詳細は，［6］を参照

この性質は，明らかに型システムで型づけされたプログラムは，実行時にエラーを出さないという，型つけシステムの基本で最も重要な性質を満たすことがわかる。

4．簡約意味論

本節では，2.で提案した操作的意味論に対し，簡約関係と等式推論のみに基づく等式理論の定式化を示すことである。まず，簡約意味論の中心となる簡約閉包性を定義し，健全な理論を提案する上で必要な不作用項と健全性を定義・証明する。これらの理論の補題および定理などの基本的な成果は，［15］に述べられている並行プロセス計算における簡約意味論を基に通信プリミティブに適用する形で示している。

4．1　簡約閉包性

並行計算において，状態（state）の概念は重要である。項が簡約しているうちに，状態が変化し，それに伴って意味が変わってしまう並行プロセス計算の世界で，「等価」の概念は，2つの等価な項が状態遷移した後に等価な状態に行きつくことを意味すると考えられる。つまり，もし2つの項が等しく，かつ，そのうちの1つの項が簡約によって別の項にいきつくなら，もう1つの項も（関係する等価性の範囲で）等価な項に行き着くことができなければならない。この概念を簡約閉包性とよばれ，以下のように定義される。

定義　簡約閉包性（reduction-closure property）

理論Tは，次のことがいつでも成り立つとき，簡約について閉じている（reduction-closed），あるいは,簡約閉包性（reduction-closure property）を持つと

いう,$T \vdash P=Q$ かつ $P \twoheadrightarrow Q$ならば，ある Q'に関し，$Q \twoheadrightarrow Q'$かつ $T \vdash P'=Q'$

ある理論がβ等価性を公理として含むならば，明らかにこの理論は簡約に対し閉じている。このように簡約閉包性はβ等価性を状態のある計算の枠組みに一般化したものと考えられる。なお，ここで→でなく→→を用いること,すなわち基本的な意味論のレベルで簡約回数を取捨することは一般的な考え方である。また，上の定義より，C[]を任意の文脈とすると,Tが簡約閉包性を持つならば，またそのときに限り,いつでも $T \vdash P=Q$かつ $C[P] \twoheadrightarrow P'$ ならば,ある Q'が存在して，$C[Q] \twoheadrightarrow Q'$かつ $T \vdash P'=Q'$が成り立つ。この命題は上の定義の１つの基本的な側面である「文脈による項の動作のテスト」という概念を明確にしている。すなわち,簡約閉包は，等価とされる二つの項の片方の項を任意の文脈にいれて簡約を行なわせ，それがある項にたどりついた場合，同じ文脈にもう片方の項をいれたものも等価性の限りで同じ項にたどりつくという動作テストを行なっているとも考えられる。以降で簡約について閉じている理論を簡単のために簡約理論とよぶことにする。
補題ののちに，簡約理論の重要な性質を示す。

補題　連鎖補題

T'をある理論の族の和$\Sigma\{T_i\}_{i\in I}$とし，$T' \vdash P=Q$とすると，ある$j_i \in I$, $0 \leqq i \leqq n$が存在し,次のような等式の鎖に分解できる。

$$T_{j0} \vdash P=R_0, \cdots, T_{jk} \vdash R_{k-1}=R_k, \cdots, T_{jn} \vdash R_{n-1}=Q$$

証明　理論の定義中の各規則の帰納法から（証明終わり）

命題　全ての $i \in I$ について，T_iは簡約について閉じているとする。そのとき$\Sigma\{T_i\}_{i\in I}$も簡約について閉じている。

証明　$T' \stackrel{def}{=} \Sigma\{T_i\}_{i\in I}$とする。連鎖補題より，等式の連鎖を得る。今$P \twoheadrightarrow P'$とすると，帰納法の仮定によりある$R'_0$が存在し，$R \twoheadrightarrow R'_0$かつ $T_{j0} \vdash P'=R'_0$が成り立つ。このようにして，同様な連鎖が形成されそれが成り立つような$R_k \twoheadrightarrow R'_k$かつQ

→→Q'が存在する。これに理論の定義の(3)を適用すると，T' ⊢ P'=Q'が得られる　（証明終わり）

簡約閉包性は，理論の内部的整合性に対する条件として与えられる。しかし命題は，最大の無矛盾な簡約理論の存在を意味しているわけではない。いくつかの簡約理論は，無矛盾ではあるが意味の持たない等式を導出する。よって，簡約閉包性のみでは，理論に満足な制約を与えるのは不十分である。通常の観測可能性や停止性を導入することなしに定式化するためには，λ計算の有意な理論（sensible theories）における意味のない式（不定式）の同一視をすることで，矛盾を導く可能性のある「不健全な」等式を内的にふるい落とすことができるという性質を利用できる。そこで，並行計算における「無意味性」の概念を用いるために，不作用性（insensitivity）を導入する。

4．2　不作用性と健全な理論

不作用性はそれを取り囲んでいる文脈にすくなくとも構文的にはなんの影響を与えない項である。この概念は弱い意味論の枠組で，項の操作的無意味性のための十分条件を与える。AN(P) は，アクティブな名前の集合，つまり通信可能なチャネルや名前の集合を現しているとすると，不作用項は，次のように定義される。

定義　不作用項（insensitive term）

閉式Pは，どんなP→→P'であるようなP'について，AN(P') =ϕが成り立つとき，不作用であるという。開式Pはどんな投じた代入σに関してもPσが不作用であれば，不作用項と定義される。このような不作用項の集合をInsと書く。

言い替えれば，もしどんな部分項も外部と相互作用を行なわない場合，すなわち外部の項の部分項と可簡約項の対を作りえないとき，その項は不作用である。また，不作用項は構造合同性，複数簡約，および，名前から名前への代入について閉じている。次の命題は，不作用項は実際に「なにもしない」ことを述べている。ここで，簡約文脈（reduction context）を，以下のように定義している。

Cr ::= [ ] | (νu)Cr | (Cr | P) | (P | Cr)

Cr, Cr'...は簡約文脈を示す。これを用い，文脈の一般簡約（generic reduction of a context）とは，Cr→→Cr' ⇔ ∀P. Cr[P] →→ Cr'[P]で与えられる。Cr→Cr'のとき，文脈に挿入されたプロセスと相互作用を行なうことなく，前者の文脈とは独立に後者に簡約できる。この概念を用い，以下の命題が容易に証明できる。

命題　Pが不作用であって，Cr→→Rならば，ある不作用であるP'について，R≡Cr'[P']と書け，Cr→→Cr'かつP→→P'である。

この命題は，もし不作用項が簡約文脈に挿入されたとし，その後どのような簡約が行なわれたとしても，全体の項の変化には，全く関与していないことを示している。また，上の不作用項の性質が簡約関係と文脈のみを基礎にして与えられていることに注意すること。つまり，不作用項の動作的無意味性は，主観的な観測の概念から独立に導かれている。これから導入する意味論の基本概念は，不作用項の同一視を行なうことである。上の命題は，この同一視が意味論的に自然であることを主張している。以降では，不作用項を投下にする無矛盾な簡約理論を健全である（sound）とよぶことにする。

命題　$T=_s$は健全である。

証明　簡約閉包性・無矛盾性については，相互模倣性の定義より明らか。あとは，不作用項同士の関係の反射的閉包が弱相互模倣性であることを証明すれば，不作用項の同一視を$T=_s$で行なっていることになり，健全であることが証明できる。しかし，不作用であるPからは，$P \xrightarrow{\tau} sP'$という遷移のみしかおきえないので，不作用項の定義より，全てのP'はまた不作用となる。よって$T=_s$は健全な理論である（証明おわり）

一方で，$T\sim_s$は，簡約閉包性を持つが，健全ではない。

4．3　本来的な観測可能性

健全な理論に関する最も重要な事実は，健全性が自動的に観測可能性を導くこと

である。この観測可能性は，動作的に意義がある規範的な意味論の存在へとつながる。どんな健全な理論においても，$T \nvdash P=Q$ であるとき，この組を非両立的である(incompatible)といいP#Qと書く。

この非両立な組は，観測可能性を導入する上で必要な概念である。以下のような観測可能性に関する基本定理がある。定理中の$\stackrel{\uparrow a}{\rightarrow}$は，観測のうち値に考慮しないラベル付遷移とみなす。

定理　観測可能定理

ある健全な理論Tについて，$T \vdash P=Q$とすると，$P \stackrel{\uparrow a}{\rightarrow} P'$ ならば　$Q \rightarrow\rightarrow \stackrel{\uparrow a}{\rightarrow} \rightarrow\rightarrow Q'$ となるQ'が存在し，$T \vdash P'=Q'$。

証明は，概要のみを示す。健全な理論中で，上記の非両立的な組を文脈に組み込み簡約について閉じている理論に対して，矛盾を引きおこすために，そのような非両立的な組を文脈として仮定できないということになる。

簡約可能定理での重要な帰結は，分離集合（isolation set）をつくることができる。分離できることは，それらを無矛盾で，かつ，最大の健全な理論を作れることにつながる。このことにより，簡約関係を基に真に規範な等価性を得ることができた。

また，簡約意味論による理論と観測による弱相互模倣性と一致させるためには，さらにマッチ演算子[u=v]Pと呼ばれる名前の同一性によるプロセスの動作を行なう演算子の導入が必要である。これは，本論文での操作的意味をあたえる時に名前（チャネル）を渡す仕組みがあるために，上記の観測可能定理で述べられる値としての名前が観測できないからである。つまりマッチ演算子の存在が，その差異をうめることができることがわかっている。

5．議論

5．1　通信型と簡約意味論の関連性

本論文であげた操作的意味論や観測等価性は，並行計算であるπ計算やν計算の操作的意味論や観測による等価性を基にしている。元のπ計算やν計算は，自由度が高いために様々な等価性が提案されている。これに対し通信プリミティブは，言語として通信に対する制約が多い。そのため，自由度が高い簡約に制約が強くつく。一方で，観測されるラベルの種類が一定のパターンでしめされたものになる。そのラベルのパターンが，通信型と同一になる。通信型は，型システムから導出され，ラベル列は，複数の簡約から別々に導出されるが，それらは一致する。ただし，型は，通信可能な項の対を決めることになるので，それらの型は双対となる。対して，おなじラベル列が観測される観測等価性（相互模倣性）は，対が同種になる。本論文では，その後，簡約意味論を導入したことによってこの差異を解消することができることになる。簡約は，通信可能な対の存在から導出されることになるので，当然ながら項の中に通信可能な項の組，いいかえると通信型の推論規則内にでてくる前提条件として出現する項の組が項中の双対の部分項となる。

6．結論

本論文では，連続な2者間の双方向通信を基本にした構造化プリミティブの操作的意味論の詳細を示し，簡約に基づく等価性を提案した。さらに，等価性の一部が，型システムによる型付けの整合性と関連性があることを示した。

謝辞

本研究は，本田耕平（Queen Mary & Westfield College, London），竹内格（NTT ソフトウェア研究所），Vasco. T. Vaconcelos（University of Lisbon, Portugal）氏との共同研究や示唆のもとに発展してきたものであり，さらに吉田展子氏の研究の成果からも多大なる影響を受けており，4氏には非常に感謝いたします。

参考文献

[1] Agha, G.: "Actors: A Model of Concurrent Computation in Distributed Systems", MIT Press, 1986.

[2] America, P.:"Operational Semantics of a Parallel Object-Oriented Language.", In Proc. 13th Annual ACM Symposium on Principle of Programming Languages, pp. 194－208, ACM press, 1986.

[3] 本田耕平，所真理雄：並行オブジェクト計算のための形式系,コンピュータソフトウェア，Vol. 9，No. 2， pp.75－89， 1992.

[4] 久保　誠,本田耕平,竹内格: 通信を基礎としたプログラミングのための構造化プリミティブ（I） 記述と意味論，コンピュータソフトウェア，Vol. 14，No. 5，pp32－54，1997.

[5] Honda, K., Vaconcelos, V., Makoto, K., Language Primitives and Type Discipline for Structured Communication-based Programming, In Proc. ESOP 98, LNCS, 1998.

[6] 久保　誠，構造化通信を基礎としたプログラミング言語の型システム，千葉商科大学論叢第38巻2号，2000年9月

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

[8] Milner, R.: Function as Processes, Journal of Mathmetical Structure in Computer Science, Vol. 2, No. 2: pp.119－141, 1992.

[9] Milner, R.: Calculus of Mobile Processes (Parts I and II), Information and Computation, Vol. 100., pp. 1 －77, 1992.

[10] Milner, R., Tofte, M., and Harper, R.: The definition of Standard ML, MIT Press, 1990.

[11] 猿渡 隆介，久保 誠，所 真理雄：連続な通信を基礎とした型つき並行言語の実装，日本ソフトウェア科学会第12回大会論文集，pp. 85－88，1995.

[12] Takeuchi, K., Honda, K., and Kubo, M.: Interaction Based Programming Language and its Typing System, In proc. PARLE'94, LNCS , 1994.

[13] Turner, D.: The π-calculus: Types, polymorphism and implementation, PhD Thesis, University of Edinburgh, 1996.

[14] Vasconcelos, V., A Process-Calculus Approach to Typed Concurrent Objects, Ph. D. thesis, Keio University, 1994.

[15] 本田耕平，吉田展子：並行計算のための簡約意味論，コンピュータソフトウェア，Vol. 11, No. 5, pp. 2 －20, September, 1994.