# CCB：ブロードキャスト機能をもつプロセス代数の提案

磯部 祥尚, 佐藤 豊, 大蒔 和仁

電子技術総合研究所 情報アーキテクチャ部 情報ベース研究室

システムの再利用性や拡張容易性を高めるために、システムを並行動作する複数の部品－エージェント－の集合体として構成するマルチエージェントモデルが知られている。このときエージェントの結合方式が重要となるが、システムの柔軟性を保つためには、エージェント間の動的な結合や、1対nの柔軟な結合が要求されてくる。このような要求に対して、送信エージェントはメッセージの受信エージェントを指定せず、そのメッセージの取捨選択は各エージェントに任せられるブロードキャスト的な通信方式が提案され実現されてきた。

マルチエージェントモデルの問題点は、その設計が困難であることにある。複数のエージェントの相互作用による予想外の結果の出力やデッドロックの発生を予防するため、何らかの設計支援が必要とされている。

並行システムを記述し解析する数学的道具としてプロセス代数が知られている。しかし、従来のプロセス代数で上記のブロードキャスト的な通信方式を扱うためにはいくつかの問題点を指摘することができる。例えば、エージェントの生成や消滅により、あるメッセージの受信エージェント数が不定になることが記述困難な1つの原因となる。

我々は送信先を指定しないブロードキャストの記述に適したプロセス代数CCB (a Calculus of Communication with Broadcasting) を提案する。本論文では従来のプロセス代数の問題点と、CCBの定義と特長について述べる。

# CCB : A Process Algebra for Broadcasting

Yoshinao ISOBE, Yutaka SATO, Kazuhito OHMAKI

Information Base Section, Computer Science Division, Electrotechnical Laboratory
1-1-4 Umezono, Tsukuba, Ibaraki 305, Japan

*Multiagent models* are useful for promoting reusability and extensibility of systems. In the models, systems consist of some parallel processes, called *agents*. Architecture of connection among agents is important, and it is required that agents are adaptively connected according to applications. For this requirement, connections with broadcasting was implemented. Thus, an agent sending messages does not specify their destination but broadcast them to all the other agents, then each agent decides whether to read the messages or discard.

A disadvantage of the multiagent models is that designs of systems are hard because there may be deadlocks, unexpected outputs, and so on, by communication and concurrency among agents. Therefore we want an assistant system for the designs.

In order to describe and analyze concurrent and communicating systems, *process algebras* are well known. Until now, many process algebras are proposed, but we can point out some problems of existing process algebras, to use them for broadcasting. For example, a cause is that a number of agents which accept certain messages is not fixed by generation and elimination of them.

In this paper, we propose a process algebra named CCB (a Calculus of Communication with Broadcasting), which can appropriately analyze communication with broadcasting. First, we explain the problems of existing process algebras against broadcasting. Second, we show definitions and properties of CCB.

---

isobe@etl.go.jp, ysato@etl.go.jp, ohmaki@etl.go.jp

## 1 はじめに

システムの変更や拡張を容易にするために、システムを複数の部品に分割し結合する方法が有効である。例えば、利用者インターフェース管理システム (UIMS : User Interface Management System) では、アプリケーションに依存する部分 (応用部) と、インターフェース (対話部) をできる限り分離して、対話部と応用部との独立な開発を支援し、結合環境を提供することが望まれている。

UIMS では、対話部と応用部をさらに機能単位の部品に分割し、各々の部品を並行動作する複数のエージェントとして実行し、エージェント間をメッセージあるいはイベントの交換により結合するマルチエージェントモデルが共通の基盤となりつつある [1]。

エージェントは必要に応じて、追加、修正、削除されることができ、さらに動的に生成、消滅することもある。このように動的に変化するエージェント接続の基盤として **VIABUS**[13] が実現されている (図1参照)。VIABUS では送信エージェントは全てのエージェントにメッセージをブロードキャストし、そのメッセージを受信する判断は各エージェントに任せられている。また、送信エージェントはメッセージの送信後にそのメッセージの受信エージェント数を知ることも可能である。この柔軟な1対 n の通信方式では、送信エージェントが受信エージェントの数を意識する必要がないので、既存のシステムを変更することなく、エージェントの追加や局所的な変更などが可能である。

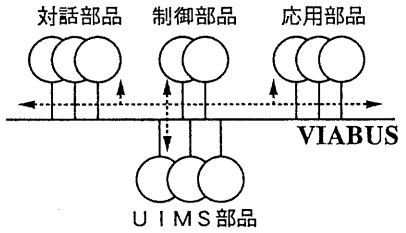


図 1: VIABUS のアーキテクチャ

この論文ではメッセージの送信方式としての2つの用語、マルチキャストとブロードキャストを次の意味で用いる：

- マルチキャスト: 受信者数を指定して送信する。
- ブロードキャスト: 受信希望者全てに送信する。

つまり、マルチキャストでは指定した受信者数より受信希望者が少ない場合はその送信が延期されたり、また多い場合は受信できない受信希望者がでてくることがある。これに対して、ブロードキャストでは送信時に受信を希望しているエージェント全てに送信されることになる。

再利用性などで利点をもつマルチエージェントモデルの問題点は、その設計が困難であることにある。個々に設計されたエージェントはメッセージの送受信によって協調しながら独立に各々の処理を行なうが、その全体の動作はエージェント数の増加と共に複雑になり、予想外の出力やデッドロックの発生をもたらす。そこで、この予想外の動作を実行前に発見し、設計者を支援する何らかの道具が必要となる。

並行システムを記述し解析する数学的道具としてプロセス代数が知られている。プロセス代数としては、1対1通信の CCS[2]、$\pi$-計算 [3]、CHOCS[4] をはじめ、1対 n 通信を記述可能な SCCS[5]、Meije[6]、CSP[7]、LOTOS など数多く提案されている。

しかし、ブロードキャストを記述するにはいくつかの問題点を指摘できる。例えば SCCS では受信者数を明示して送信 (マルチキャスト) を行なわなければならない。そこで、受信希望者全員に送信をするには、送信者が受信者数を動的に (実行中に) 数えなければならないが、SCCS ではこのような「受信希望者を数える動作」を記述することはできない。また、CSP では各受信者が「自分はこのイベント (メッセージ) の受信を希望している」ということを明示することによって、受信者全員が希望するイベントを受信することができる。しかし、送信後に送信者はその受信者数を知ることはできない。

我々は上記の問題点を解決し、ブロードキャストの記述に適したプロセス代数 **CCB** ( a Calculus of Communication with Broadcasting ) を提案する。CCB は次の特長をもつ：

1. ブロードキャストが可能。
2. 受信者数の動的カウントが可能。
3. 受信希望イベントの動的変更が可能。

以下、2節では従来のプロセス代数を用いてブロードキャストを記述するときの問題点について述べる。3節では CCB の特長を例を用いて説明する。4節では CCB の基礎となるプロセス代数 **Core-CCB** の形式的な定義を与え、5節で Core-CCB で使われる等価関係を定義し、成り立つ基本的な性質を紹介する。最後に、6節で Core-CCB を基に CCB を定義する。

## 2 プロセス代数

並行プロセスを解析する数学的道具としてプロセス代数が知られている。プロセス代数では、プロセスの動作を式の形で記述することによって、動作の等価性を代数的に調べることができる。現在までに数多くのプロセス代数が提案されており、その中には他対他通信を扱えるプロセス代数も含まれている。以下、プロセス代数を通信方式によって3種類に大別し、ブロードキャストを記述するうえでの問題点を指摘する。

## 2.1 CCS 系 ($\pi$- 計算,CHOCS,$\cdots$)

1 対 1 通信であるため、受信者の数だけイベントを繰り返し送信する必要がある。しかし、そのイベントを受信可能であるエージェントの数を動的に知ることは困難である。次の例は CCS[2] で、イベント $a$ の受信者数を数えようとして失敗する例である：

$$P_1 \stackrel{\text{def}}{=} a.0, \quad P_2 \stackrel{\text{def}}{=} b.0, \quad P_3 \stackrel{\text{def}}{=} a.0$$
$$C(i) \stackrel{\text{def}}{=} \overline{a}.C(i+1) + \overline{out}(i).0$$
$$SYS \stackrel{\text{def}}{=} (C(0)|P_1|P_2|P_3)\backslash\{a,b\}$$

エージェント $C$ はイベント $a$ による通信毎に内部変数 $i$ をインクリメントし、イベント $out$ を用いてその数を出力する。エージェント $P_1$ と $P_3$ がイベント $a$ を受信可能なので、$out$ からの出力として 2 を期待するが、実際には、次の等式が示すように 0,1,2 のどれが出力されるか全く予想できない[1]：

$$SYS \sim \overline{out}(0).0 + \tau.(\overline{out}(1).0 + \tau.\overline{out}(2).0)$$

これは、イベント $a$ を全て起こす前にイベント $out$ を起こしてしまう可能性があるためである。そこで、プライオリティ演算子 [10] を導入して、イベント $a$ をイベント $out$ より優先することが考えられる。しかし、エージェント $P_1$ が $P_1 \stackrel{\text{def}}{=} a.P_1$ の様に定義されていたならば、イベント $a$ の優先によって無限ループにおちいる。我々はこの場合でも $out$ からの出力として 2 を期待している。

この種の問題は高階計算 ($\pi$- 計算 [3],CHOCS[4]) にしても残される。

## 2.2 SCCS 系 (Meije,$\cdots$)

受信するエージェントの数を明示することによってマルチキャストを実現している。次の例は SCCS[5] でエージェント $P_0$ がエージェント $P_1, P_2$ にマルチキャストする例である[2]：

$$P_0 \stackrel{\text{def}}{=} a^{-2}:P_0', \quad P_1 \stackrel{\text{def}}{=} a^{1}:P_1', \quad P_2 \stackrel{\text{def}}{=} a^{1}:P_2'$$
$$P_0 \times P_1 \times P_2 \xrightarrow{1} P_0' \times P_1' \times P_2'$$

ここでは、$P_0$ は 2 つのエージェントにイベント $a$ をマルチキャストすることを明示している[3]。このため、イベント $a$ を要求するエージェントを追加した場合は、$P_0$ を修正する必要がある。そこで、イベント $a$ を受信可能なエージェントを動的に数える必要があるが、これは CCS 系のときと同じ理由で不可能である。

[1] $\sim$ は CCS[2] の強等価である。

[2] :,$\times$ は各々 SCCS のプレフィクスと並行合成演算子である。また、$\xrightarrow{1}$ の 1 は内部イベントに相当する。

[3] $P_0$ のイベント $a^{-2}$ の $-2$ に注目。負は送信を表す。

## 2.3 CSP 系 (LOTOS,$\cdots$)

受信を希望するイベントを明示することによってブロードキャストを実現している。例えば、CSP[7] では各エージェントに対して同期 (受信) するイベントの集合 (アルファベットと呼ばれている) を事前に定義することによって、また、LOTOS では合成演算子に同期するイベントを明示することによって実現している。次の例は CSP でエージェント $P_0$ がエージェント $P_1, P_2$ にブロードキャストする例である (簡単のため入出力を区別していない)：

$$a \in \alpha(P_0), \ a \in \alpha(P_1), \ a \in \alpha(P_2), \ a \notin \alpha(P_3)$$
$$P_0||P_1||P_2||P_3 \xrightarrow{\langle a \rangle} P_0'||P_1'||P_2'||P_3$$

ここで、$\alpha(P)$ は $P$ のアルファベット、$P_i \xrightarrow{\langle a \rangle} P_i'$ $(i \in 1,2,3)$ である。この場合、$P_0$ を修正することなくエージェントの追加に対応できる。しかし、$P_0$ は自分が送信したイベントの受信者数を知ることはできない。また、アルファベットは固定されており、動的に変えることはできない。LOTOS も同様の問題をもつ。

他にブロードキャスト用につくられた合成演算子がある [8]。CSP の演算子に似ているが、アルファベットの代わりに「状態遷移できない関係」を用いている。ただし、受信者数に対する考慮はされていない。

# 3 CCB の紹介

2節で述べてきた問題を解決するために、我々は CCS を拡張したプロセス代数 CCB ( a Calculus of Communication with Broadcasting ) を提案する。CCB には次の 2 種類の通信方式が用意されている：

- マルチキャスト: 受信者数を指定して送信する。
- ブロードキャスト: 受信希望者全てに送信する。

マルチキャストは SCCS の通信方式に似ており、受信エージェント数を 1 に限定すれば CCS の通信方式と同じになる。受信者数を受信希望者数より小さく設定すると、受信希望者のなかで受信できないエージェントが存在することになる。ブロードキャストは CCB の重要な特長であり不定数の受信希望者全てにメッセージを送信できる。また、送信後にそのメッセージを受信したエージェントの数を知ることもできる。CCB のイベントは

$$a\theta\langle x\rangle(y)$$

の形をもち、ここで $a$ はイベント名、$\theta$ はイベントの属性、$x$ は同期したイベントの数、$y$ はこのイベントによって渡される内容 (メッセージ) である。同期したイベントの数が 1 である場合は $\langle 1 \rangle$ を省略することもある。また、メッセージが空で単に同期をとるために使われた場合は、

$(y)$ を省略することもある。イベントの属性 $\theta$ は、次に示す4種類である：

| | 送信用 | 受信用 |
|---|---|---|
| マルチキャスト用 | ! | ? |
| ブロードキャスト用 | ‼ | ⁇ |

図 2: イベントの属性

特に、属性が ! で同期数が 0 であるイベント $a!\langle 0\rangle$ を内部イベントと呼ぶ (CCS の $\tau$ に相当する)。

次に上記の 2 つの通信方式の例を示す。

### マルチキャストの例

マルチキャストでは、送信者が送信可能になっても指定の受信者数より実際の受信者数が少ない場合は、実際の受信者数が指定の受信者数になるまで送信が延期される。また、実際の受信者数の方が多い場合は受信できない受信者が存在することになる。次の記述はエージェント $P_0$ がイベント $a$ を 2 つの受信エージェント $P_1, P_2$ に向けて送信する例である。しかし、$P_1$ は今現在 $a$ を受信できる状態にはなく、イベント $b$ を送信した後受信可能になるとする。

$$P_0 \stackrel{\text{def}}{=} a!\langle 2\rangle.end!.\mathbf{0} \qquad P_2 \stackrel{\text{def}}{=} a?.\mathbf{0}$$
$$P_1 \stackrel{\text{def}}{=} b!.a?.\mathbf{0} \qquad SYS \stackrel{\text{def}}{=} (P_0|P_1|P_2)\backslash a$$
$$SYS \sim b!.a!\langle 0\rangle.end!.\mathbf{0}$$

上の例では $P_0$ は $P_1$ のイベント $b$ の送信を待ってから、$P_1, P_2$ に $a$ を送信して終了している。

### ブロードキャストの例

ブロードキャストでは送信者が送信可能ならばいつでも送信可能であり、送信時の受信希望者全てがそのイベントを受信できる。ここでは、上の例をブロードキャストで記述してみる。ただし、イベント $a$ を受信したエージェントの数をイベント $end$ と共に出力するとする。

$$P_0 \stackrel{\text{def}}{=} a‼\langle x\rangle.end!(x).\mathbf{0} \qquad P_2 \stackrel{\text{def}}{=} a⁇.\mathbf{0}$$
$$P_1 \stackrel{\text{def}}{=} b!.a⁇.\mathbf{0} \qquad SYS \stackrel{\text{def}}{=} (P_0|P_1|P_2)\backslash a$$
$$SYS \sim b!.a!\langle 0\rangle.end!(2).\mathbf{0} + a!\langle 0\rangle.(\, b!.end!(1).\mathbf{0} + end!(1).b!.\mathbf{0})$$

$P_0$ のプレフィクス $a‼\langle x\rangle$ はイベント $a$ のブロードキャストを表しており、送信後に変数 $x$ にイベント $a$ の受信者数が束縛される。$P_1$ がイベント $b$ を送信してから $P_0$ がイベント $a$ をブロードキャストした場合は $a$ の受信者数は 2 となるが、ブロードキャストが先に起こった場合は $P_1$ が $a$ を受信することができないため受信者数は 1 になっている。

## 4 Core-CCB の定義

この節では CCB の基礎となる Core-CCB のシンタックスとセマンティクスを形式的に定義する。Core-CCB はデータ変数[4]をもたないプロセス代数であり、CCB の定義は Core-CCB を基にして 6節で与えられる。

### 4.1 シンタックス

この小節では Core-CCB のシンタックスを定義する。まず、我々は名前の集合 $\mathcal{N} = \{a, b, c, \cdots\}$ が与えられていると仮定する。このとき、イベントの集合 $Event$ は次の集合として与えられる：

$$Event = \{a\theta^n : a \in \mathcal{N}, \theta \in \mathcal{T}, n \in \mathcal{I}\} - \{a?^0, a⁇^0 : a \in \mathcal{N}\}$$

ここで、$\mathcal{I}$ は非負の整数の集合 $\{0, 1, 2, \cdots\}$、$\mathcal{T}$ はイベント属性の集合 $\{!, ?, ‼, ⁇\}$ である。

Core-CCB のシンタックスを次のように定義する。

**定義 4.1** *CCB* のエージェント式の集合 $\mathcal{E}$ は次の式を含む最小の集合である：

| | |
|---|---|
| $X$ | : 変数 $(X \in \mathcal{X})$ |
| $A$ | : 定数 $(A \in \mathcal{K})$ |
| $\mathbf{0}$ | : 無動作エージェント |
| $a\theta^n.E$ | : プレフィクス $(a\theta^n \in Event)$ |
| $E + F$ | : 選択 |
| $E\|F$ | : 並行合成 |
| $E[S]$ | : リネイミング $(S \in \mathcal{F})$ |
| $E\backslash L$ | : 制限 $(L \subseteq \mathcal{N})$ |

ここで、$E, F$ はすでに $\mathcal{E}$ の要素であるとする。$\mathcal{X}$ と $\mathcal{K}$ は各々エージェント変数とエージェント定数の集合、$\mathcal{F}$ はリネイミング関数 $(S : \mathcal{N} \to \mathcal{N})$ の集合である。 ∎

エージェント変数を含まないエージェント式をエージェントと呼び、エージェントの集合を $\mathcal{P}$ で表す。エージェント定数は定義式によって意味を与えられるエージェントである。実際に全てのエージェント定数 $A$ について、

$$A \stackrel{\text{def}}{=} P \qquad P \in \mathcal{P}$$

の形の定義式があると仮定する。さらに、$P$ の中で $A$ は弱ガードされていると仮定する[5]。弱ガードされていないエージェントの動作は予想できず実用的ではない [11]。CCB が複雑になることを避けるため我々は弱ガードされたエージェントのみを扱う。

[4] CCB のイベント $a\theta\langle x\rangle(y)$ の $\langle x\rangle(y)$ の部分に使われる変数

[5] 全ての $X$ が $E$ の部分式 $a\theta^n.F$ のなかに含まれているならば、$X$ は $E$ のなかでガードされているという

この論文を通して集合の要素上の変数として図3の記法を用いる。

| 集合 | 変数 | 集合 | 変数 |
|---|---|---|---|
| $\mathcal{N}$：名前 | $a, b$ | $\mathcal{P}$：エージェント | $P, Q$ |
| $\mathcal{T}$：イベント属性 | $\theta, \phi$ | $\mathcal{K}$：エージェント定数 | $A, B$ |
| $\mathcal{I}$：非負整数 | $n, m$ | $\mathcal{X}$：エージェント変数 | $X, Y$ |
| $\mathcal{E}$：エージェント式 | $E, F$ | $\mathcal{F}$：リネイミング関数 | $S$ |

図 3: 集合と集合要素上の変数の関係

## 4.2 セマンティクス

CCS と同様に、Core-CCB のセマンティクスも次のラベル付遷移システムによって与えられる：

$$(\mathcal{E}, Event, \{\xrightarrow{a\theta^n} : a\theta^n \in Event\})$$

例えば、$E \xrightarrow{a\theta^n} E'$ はエージェント式 $E$ がイベント $a\theta^n$ を実行でき、その実行後はエージェント式 $E'$ になることを意味している。

Core-CCB のセマンティクスを定義する前に、エージェント式 $E$ が現在どのイベントを受信可能であるのかを求める関数 $ac$ を定義する。

**定義 4.2** 各エージェント式 $E$ について、受信可能関数 $ac : \mathcal{E} \to 2^{\mathcal{N}}$ を次のように帰納的に定義する：

$$\begin{aligned}
ac(\mathbf{0}) &= \emptyset \\
ac(a\theta^n.E) &= \begin{cases} \{a\} & (\theta = ?^!) \\ \emptyset & (otherwise) \end{cases} \\
ac(E+F) &= ac(E) \cup ac(F) \\
ac(E|F) &= ac(E) \cup ac(F) \\
ac(E[S]) &= \{S(a) : a \in ac(E)\} \\
ac(E\backslash L) &= ac(E) - L \\
ac(A) &= ac(P) \qquad (A \stackrel{\text{def}}{=} P) \\
ac(X) &= \emptyset
\end{aligned}$$

∎

エージェント定数は弱ガードされているので、この関数 $ac$ は計算可能である。直観的にはブロードキャストされたイベント $a$ をエージェント式 $E$ が受信可能な状態にあるならば、$a \in ac(E)$ である。

次に Core-CCB のセマンティクスを与える。

**定義 4.3** エージェント式間の遷移関係 $\xrightarrow{a\theta^n}$ は次の推論規則を満たす最小の関係である。ここで、横棒の上が $0$ 個以上の仮定、右横が条件、下が結果を表している：

$$\mathbf{Event}\ \frac{}{a\theta^n.E \xrightarrow{a\theta^n} E}$$

$$\mathbf{Choice}_1\ \frac{E \xrightarrow{a\theta^n} E'}{E+F \xrightarrow{a\theta^n} E'}$$

$$\mathbf{Choice}_2\ \frac{F \xrightarrow{a\theta^n} F'}{E+F \xrightarrow{a\theta^n} F'}$$

$$\mathbf{Ren}\ \frac{E \xrightarrow{a\theta^n} E'}{E[S] \xrightarrow{S(a)\theta^n} E'[S]}$$

$$\mathbf{Res}_1\ \frac{E \xrightarrow{a\theta^n} E'}{E\backslash L \xrightarrow{a\theta^n} E'\backslash L}\ (a \notin L)$$

$$\mathbf{Res}_2\ \frac{E \xrightarrow{a\theta^0} E'}{E\backslash L \xrightarrow{a!^0} E'\backslash L}\ \left(\begin{array}{l}\theta \in \{!, ?^!\},\\ a \in L\end{array}\right)$$

$$\mathbf{Con}\ \frac{P \xrightarrow{a\theta^n} P'}{A \xrightarrow{a\theta^n} P'}\ (A \stackrel{\text{def}}{=} P)$$

$$\mathbf{Para}_1\ \frac{E \xrightarrow{a\theta^n} E'}{E|F \xrightarrow{a\theta^n} E'|F}\ \left(\begin{array}{l}\theta \in \{!, ?\}\ \text{or}\\ a \notin ac(F)\end{array}\right)$$

$$\mathbf{Para}_2\ \frac{F \xrightarrow{a\theta^n} F'}{E|F \xrightarrow{a\theta^n} E|F'}\ \left(\begin{array}{l}\theta \in \{!, ?\}\ \text{or}\\ a \notin ac(E)\end{array}\right)$$

$$\mathbf{Para}_3\ \frac{E \xrightarrow{a\theta^m} E' \qquad F \xrightarrow{a\phi^n} F'}{E|F \xrightarrow{a\theta^{(m-n)}} E'|F'}\ \left(\begin{array}{l}\theta \in \{!, ?^!\},\\ \phi = @(\theta),\\ m \geq n\end{array}\right)$$

$$\mathbf{Para}_4\ \frac{E \xrightarrow{a\phi^n} E' \qquad F \xrightarrow{a\theta^m} F'}{E|F \xrightarrow{a\theta^{(m-n)}} E'|F'}\ \left(\begin{array}{l}\theta \in \{!, ?^!\},\\ \phi = @(\theta),\\ m \geq n\end{array}\right)$$

$$\mathbf{Para}_5\ \frac{E \xrightarrow{a\theta^m} E' \qquad F \xrightarrow{a\theta^n} F'}{E|F \xrightarrow{a\theta^{(m+n)}} E'|F'}\ (\theta \in \{?, ?^!\})$$

ここで、$@$ は次のように定義される $\mathcal{T}$ から $\mathcal{T}$ への関数である：

$$@(!) = ?,\quad @(?) = !,\quad @(?^!) = ?^!,\quad @(!^?) = !^?$$

∎

このセマンティクスの定義のもとで次の命題が成り立つ。

**命題 4.1** $\exists E' \exists n \bullet E \xrightarrow{a?^n} E'$ iff $a \in ac(E)$ ∎

つまり、$\mathbf{Para}_1$ と $\mathbf{Para}_2$ の条件 $a \notin ac(E)$ は、エージェント式 $E$ が $a?^n$ のような受信可能イベントによる状態遷移を起こせないことを意味している。

## 5 同値関係

Core-CCB にも CCS と同様に同値関係を定義する。しかし、CCS では全ての内部イベントを $\tau$ で表しているのに対して、Core-CCB では $a!^0$ の形の全てのイベントを内部イベントとして用いているため、定義が多少異る[6]。

まず、**過剰強等価** $\sim_{ex}$ を定義する。CCS の強等価 $\sim$ との違いは、内部イベント $a!^0$ と $b!^0$ まで区別する点にあり、強等価より強い等価関係である[7]。しかし、定義は簡単であり (CCS の強等価の定義に同じ)、過剰強等価ならば強等価なので、等式を証明するうえで有効である。

**定義 5.1** 過剰強双模倣

エージェント上の関係 $\mathcal{S}$ が過剰強双模倣であるとは、$(P,Q) \in \mathcal{S}$ ならば、任意の $a\theta^n \in Event$ について、次の 2 つの条件が成り立つことである：

(*i*) $P \xrightarrow{a\theta^n} P'$ ならば、ある $Q'$ が存在して、$Q \xrightarrow{a\theta^n} Q'$ かつ $(P',Q') \in \mathcal{S}$ を満たす。

(*ii*) $Q \xrightarrow{a\theta^n} Q'$ ならば、ある $P'$ が存在して、$P \xrightarrow{a\theta^n} P'$ かつ $(P',Q') \in \mathcal{S}$ を満たす。 ∎

**定義 5.2** 過剰強等価

もし、ある過剰強双模倣 $\mathcal{S}$ において $(P,Q) \in \mathcal{S}$ ならば、エージェント $P$ と $Q$ は過剰強等価であるといい、$P \sim_{ex} Q$ と書く。 ∎

このとき過剰強等価は同値関係となる。さらに全ての演算子は過剰強等価を保存する。

**命題 5.1** もし $P_1 \sim_{ex} P_2$ ならば、

$a\theta^n.P_1 \sim_{ex} a\theta^n.P_2$, $P_1 + Q \sim_{ex} P_2 + Q$,
$P_1|Q \sim_{ex} P_2|Q$, $P_1\backslash L \sim_{ex} P_2\backslash L$, $P_1[S] \sim_{ex} P_2[S]$ ∎

また、合成演算子 $|$ について次の基本的な等式も得られる。

**命題 5.2** 任意のエージェント $P_1, P_2, P_3 \in \mathcal{P}$ について、次の等式が成り立つ：

$$P_1|P_2 \sim_{ex} P_2|P_1$$
$$P_1|(P_2|P_3) \sim_{ex} (P_1|P_2)|P_3$$
$$P_1|\mathbf{0} \sim_{ex} P_1$$ ∎

Core-CCB の等式に関する調査は未完成であり、多くの課題を残している。特に、CCS の**観測等価** $\approx$ が、次の例に示すように Core-CCB の合成演算子 $|$ に対して保存されないことが重要な問題である。

$$P_1 \stackrel{\text{def}}{=} a?^1.b!^0.P_1, \quad P_2 \stackrel{\text{def}}{=} a?^1.P_2$$
$$Q \stackrel{\text{def}}{=} a!!^0.out0!^1.Q + a!!^1.out1!^1.Q$$

$$P_1 \approx P_2 \text{ しかし } P_1|Q \not\approx P_2|Q$$

$b!^0$ は内部イベントであるので、$P_1$ と $P_2$ は観測等価であることを証明することはできる。しかし、その各々にエージェント $Q$ を合成するとその結果は観測等価にならない。それは、$P_2$ は常に $a$ を受信可能であるのに対し $P_1$ は受信できないときがあるためである。現在、Core-CCB の観測的な合同関係を得るために、2 つのエージェント $P$ と $Q$ が観測等価であることの条件に、$ac(P) = ac(Q)$ の条件を加えることを検討中である。

## 6 CCB の定義

4節と 5節ではデータ変数を扱っていなかった。この節ではデータ変数を扱うための方法を述べる。基本的には CCS で用いられた方法と同様である。

CCB の重要な特長の 1 つに、ブロードキャスト (不定数エージェントへのマルチキャスト) があった。しかし、4節で定義した Core-CCB ではその記述はやや複雑になる。次の記述は Core-CCB でエージェント $P$ がイベント $a$ をブロードキャストして、その受信エージェント数を知る例である[8]：

$$Q_1 \stackrel{\text{def}}{=} a?^1.Q_1'$$
$$Q_2 \stackrel{\text{def}}{=} b?^1.Q_2' \qquad P \stackrel{\text{def}}{=} \textstyle\sum_{n\in\mathcal{I}} a!!^n.P_n'$$
$$Q_3 \stackrel{\text{def}}{=} a?^1.Q_3' \qquad SYS \stackrel{\text{def}}{=} (P|Q_1|Q_2|Q_3)\backslash a$$

$$SYS \xrightarrow{a!^0} (P_2'|Q_1'|Q_2|Q_3')\backslash a$$

状態遷移後の $P_2'$ の添字の 2 が受信者数を表している。このように変数の変域をカバーするように**選択** $+$ を用いることによって、変数を扱うことができる。そこで、ブロードキャストや値の受渡しの記述を容易にするための CCB を与える。上の例を CCB を用いて書き直すと次のようになる：

$$Q_1 \stackrel{\text{def}}{=} a?\langle 1\rangle.Q_1'$$
$$Q_2 \stackrel{\text{def}}{=} b?\langle 1\rangle.Q_2' \qquad P \stackrel{\text{def}}{=} a!!\langle x\rangle.P'(x)$$
$$Q_3 \stackrel{\text{def}}{=} a?\langle 1\rangle.Q_3' \qquad SYS \stackrel{\text{def}}{=} (P|Q_1|Q_2|Q_3)\backslash a$$

$$SYS \xrightarrow{a!^0} (P'(2)|Q_1'|Q_2|Q_3')\backslash a$$

$x$ はエージェント $P$ の束縛変数であり、状態遷移後は受信者数の 2 が代入されている。以下、CCB を定義する。

[6] Core-CCB でも CCS と同様に内部イベントとして $\tau$ だけを用いるように簡単に変更できる。しかし、これにより冗長な記述が増え、セマンティクスを定義する推論規則も増加する。

[7] CCB のための強等価は $a!^0$ と $b!^0$ を同一視して得られる。

[8] $\sum_{i\in I} P_i$ は $P_0 + P_1 + P_2 + \cdots$ の速記法である。ただし、$I = \emptyset$ のときは $\mathbf{0}$ とする。

## 6.1 CCB のシンタックス

CCB のシンタックスは次のように定義される。

**定義 6.1** *CCB* のエージェント式の集合 $\mathcal{E}^+$ は次の式を含む最小の集合である：

$$
\begin{aligned}
X &: \text{変数 } (X \in \mathcal{X}) \\
A(e_1,\cdots,e_n) &: \text{定数 } (A \in \mathcal{K}) \\
\mathbf{0} &: \text{無動作エージェント} \\
a[e_1]!\langle c\rangle(e_2).E &: \text{送信 } M\ (a \in \mathcal{N}) \\
a[e_1]!!\langle x\rangle(e_2).E &: \text{送信 } B\ (a \in \mathcal{N}) \\
a[e]?\langle c\rangle(x).E &: \text{受信 } M\ (a \in \mathcal{N}) \\
a[e]?!\langle c\rangle(x).E &: \text{受信 } B\ (a \in \mathcal{N}) \\
a[e_1]!!\langle c\rangle(e_2):E &: \text{定数送信 } (a \in \mathcal{N}) \\
E+F &: \text{選択} \\
E|F &: \text{並行合成} \\
E[S] &: \text{リネイミング } (S \in \mathcal{F}) \\
E\backslash L &: \text{制限 } (L \subseteq \mathcal{N}) \\
[\![b]\!]E &: \text{条件}
\end{aligned}
$$

ここで、$E, F$ はすでに $\mathcal{E}^+$ の要素であるとする。$x$ は変数、$b$ は真偽式、$e, c$ は数式である。ただし、$c$ の計算結果は 0 以上の整数でなければならない。 ∎

イベント名 $a[e]$ は、基本イベント名 $a$ と添数 $[e]$ から構成され、基本イベント名と添数の両方が等しいときイベント名は等しいとみなされる。添数 0 の場合は $[0]$ の記述を省略することができる。

$n$ 引数をもつ各エージェント定数は、

$$A(x_1,\cdots,x_n) \stackrel{\text{def}}{=} E$$

の形の定義式をもつと仮定する。ここで、$E$ は変数 $x_1, \cdots, x_n$ 以外の自由変数を含まず、エージェント変数も含まないとする。

## 6.2 CCB のセマンティクス

CCB のエージェント式 $E \in \mathcal{E}^+$ の意味は、core-CCB のエージェント式 $\mathcal{B}(E)$ によって与えられる。ここで、$\mathcal{B}$ は次のように定義される CCB から Core-CCB への変換関数である。

**定義 6.2** 関数 $\mathcal{B}$ は $\mathcal{E}^+$ から $\mathcal{E}$ への関数であり、次のように定義される：

$$
\begin{aligned}
\mathcal{B}(X) &= X \\
\mathcal{B}(A(e_1,\cdots,e_n)) &= A_{e_1,\cdots,e_n} \\
\mathcal{B}(\mathbf{0}) &= \mathbf{0} \\
\mathcal{B}(a[e_1]!\langle c\rangle(e_2).E) &= a_{e_1,e_2}!^c.\mathcal{B}(E) \\
\mathcal{B}(a[e_1]!!\langle x\rangle(e_2).E) &= \textstyle\sum_{n\in I} a_{e_1,e_2}!!^n.\mathcal{B}(E\{n/x\}) \\
\mathcal{B}(a[e]?\langle c\rangle(x).E) &= \textstyle\sum_{v\in V} a_{e,v}?^c.\mathcal{B}(E\{v/x\}) \\
\mathcal{B}(a[e]?!\langle c\rangle(x).E) &= \textstyle\sum_{v\in V} a_{e,v}?!^c.\mathcal{B}(E\{v/x\}) \\
\mathcal{B}(a[e_1]!!\langle c\rangle(e_2):E) &= a_{e_1,e_2}!!^c.\mathcal{B}(E) \\
\mathcal{B}(E+F) &= \mathcal{B}(E)+\mathcal{B}(F) \\
\mathcal{B}(E|F) &= \mathcal{B}(E)|\mathcal{B}(F) \\
\mathcal{B}(E[S]) &= \mathcal{B}(E)[S'] \\
\mathcal{B}(E\backslash L) &= \mathcal{B}(E)\backslash L' \\
\mathcal{B}([\![b]\!]E) &= \begin{cases} \mathcal{B}(E) & \textit{if } b = \textit{true} \\ \mathbf{0} & \textit{otherwise} \end{cases}
\end{aligned}
$$

ここで、全ての値は集合 $V$ に含まれていると仮定している。また、

$$
\begin{aligned}
S'(a_{v_1,v_2}) &= S(a)_{v_1,v_2} \\
L' &= \{a_{v_1,v_2} : a \in L, (v_1,v_2) \in V^2\}
\end{aligned}
$$

であり、$\{v/x\}$ は変数 $x$ への値 $v$ の代入を表している。

∎

さらに、エージェント定数の定義式 $A(x_1,\cdots,x_n) \stackrel{\text{def}}{=} E$ は、定義式の集合

$$
\begin{aligned}
\{A_{v_1,\cdots,v_n} \stackrel{\text{def}}{=} \mathcal{B}(E\{v_1/x_1,\cdots,v_n/x_n\}) :& \\
(v_1,\cdots,v_n) \in V^n\}&
\end{aligned}
$$

に変換されて意味を与えられる。

## 6.3 CCB による記述例

下は、「イベントをブロードキャストしたエージェントに、そのイベントを受信した全てのエージェントが自分のプロセス ID を返信する動作」を CCB を用いて記述した例である[9]：

$$
\begin{aligned}
&P_1(id) \stackrel{\text{def}}{=} a!!\langle x\rangle(id).P_1'(x,id,\mathtt{nil}) \\
&P_1'(x,id,idlist) \\
&\stackrel{\text{def}}{=} [\![x \neq 0]\!]ack[id]?(cid).P_1'(x-1,id,cid::idlist) \\
&\quad +[\![x = 0]\!]P_1''(id,idlist) \\
&P_1''(id,idlist) \stackrel{\text{def}}{=} \cdots\cdots \\
&P_2(id) \stackrel{\text{def}}{=} b!!\langle x\rangle(id).\cdots
\end{aligned}
$$

$$
\begin{aligned}
&Q_1(id) \stackrel{\text{def}}{=} a?!(pid).(ack[pid]!(id).\cdots|Q_1(id)) \\
&Q_2(id) \stackrel{\text{def}}{=} b?!(pid).(ack[pid]!(id).\cdots|Q_2(id)) \\
&Q_3(id) \stackrel{\text{def}}{=} a?!(pid).(ack[pid]!(id).\cdots|Q_3(id)) \\
&\qquad\vdots
\end{aligned}
$$

[9] イベント $a\theta\langle c\rangle(e)$ の記述において、$c = 1$ の場合は $\langle 1\rangle$ の記述を省略している。また、$::$はリストの結合演算子である

$$SYS \stackrel{\text{def}}{=} (P_1(1)|P_1(2)|P_2(3)|\cdots \\ |Q_1(11)|Q_2(12)|Q_3(13)|\cdots)\backslash\{a,b,\cdots\}$$

エージェント $P_1$ はイベント $a$ を用いて自分のプロセス ID $id$ をブロードキャストした後、エージェント $P_1'$ に状態遷移する。$P_1'$ は変数として、イベント $a$ を受けとったエージェントの数 $x$、自分のプロセス ID $id$、$a$ を受けとったエージェントのプロセス ID を保存するためのリスト型の変数 $idlist$ をもっている。$x$ が 0 でなければ、イベント $ack$ を通して $a$ の受信者のプロセス ID を $cid$ に束縛し、そのプロセス ID を $idlist$ に加え、$x$ を 1 減じて $P_1'$ にもどる。$x$ が 0 であれば、$a$ の全ての受信者からプロセス ID を受けとったことになるので、次の処理をする $P_1''$ になる。

$Q_1$ はイベント $a$ を受信すると、変数 $pid$ に送信者のプロセス ID を束縛して、送信者に自分のプロセス ID をイベント $ack$ により送信する。それと同時に、次のイベント $a$ を受信するためエージェント $Q_1$ が並行に走り出す。

この記述の特長は、イベント $a$ に反応するエージェント $Q_n$ を追加した場合でも、既存の記述を変更する必要がないことにある。また、プロセス ID を渡すことにより、送信者に確実に返信することができる。

## 7 おわりに

従来のプロセス代数によるブロードキャストの記述の問題点を述べ、ブロードキャストの記述に適したプロセス代数 CCB を提案してきた。CCB は次の特長をもっている：

1. ブロードキャストが可能。
2. 受信者数の動的カウントが可能。
3. 受信希望イベントの動的変更が可能。

従来のプロセス代数でも工夫次第で上記の特長を記述できると思われるかも知れない。しかし、我々の知る限りではそれは極めて困難であった。特に「受信希望者全てに送信する」という条件を満たすには何らかの手法が必要である。CCB ではそのために受信可能関数 $ac$ を採用した。この方法が最良であるとは限らないが、「状態遷移できない条件」を「状態遷移」とは別に扱えるため、推論規則に「遷移できない関係 $\not\rightarrow$[8]」を加えるよりも、証明が容易であった。

CCB で採用した VIABUS のブロードキャスト方式の他に、「送信者は受信者を指定せず、メッセージの内容によって送信先が決定される通信方式」としては Linda[12] が知られている。Linda では送信されたメッセージをある空間 (タプル空間) に保存し、受信者はそのタプル空間からメッセージをマッチングによって取得する。この方法によってブロードキャストが可能であり、この方法ならば CCS によっても記述可能であると思われるかも知れない。しかし、不定数の全ての受信希望者がそのメッセージを取得したことを確実にするには別の次元である時間 [9] を考慮する必要がある。

CCB の研究はまだ初期の段階にある。今後は CCB のより多くの性質、特に等価性について調べていく予定である。

## 参考文献


[1] J.Coutaz, "Architecture Model for Interactive Software: Failures and Trends", Proc. of the IFIP TC 2/WG 2.7 Working Conference on Engineering for Human-Computer Interaction, pp.137 - 153, 1989.

[2] R.Milner, "Communication and Concurrency", Prentice-Hall, 1989.

[3] R.Milner, J.Parrow and D.Walker, "A Calculus of Mobile Processes, I and II", Information and Computation, 100, pp.1 - 40 and pp.41 - 77, 1992.

[4] B.Thomsen. "A Calculus of Higher Order Communicating Systems", In Proc. 16th ACM Annual Symposium on Principles of Programming Languages, pp.143 - 154, 1989.

[5] R.Milner, "Calculi for Synchrony and Asynchrony", Journal of Theoretical Computer Science, Vol.25, pp.267 - 310, 1983.

[6] R.de Simone, "Higher-level Synchronizing Devices in Meije-SCCS", Journal of Theoretical Computer Science, Vol.37, pp.245 - 267, 1985.

[7] C.A.R.Hoare,"Communicating Sequential Processes", Prentice-Hall, 1985.

[8] A.Pnueli,"Linear and Branching Structures in the Semantics and Logics of Reactive Systems", LNCS 194, Springer-Verlag, pp.15 - 32, 1985.

[9] Faron Moller and Chris Tofts,"An overview of TCCS", Proceedings of EUROMICRO'92, Athens, June 1992.

[10] L.Aceto, B.Bloom, and F.Vaandrager "Turning SOS Rules into Equations", Proc. Seventh Annual IEEE Symposium on Logic in Computer Science, pp.113 - 124, 1988.

[11] D.J.Walker, "Bisimulations and Divergence", Proc. of third annual IEEE symposium on Logic in Computer Science, pp.186 - 192, 1988.

[12] N.Carriero and D.Gelernter,"How to Write Parallel Programs : A Guide to the Perplexed", ACM Computing Surveys, Vol.21, No.3, pp.323-357, 1989.

[13] 佐藤豊, 真野芳久, "UIMS の試作とそのニュースリーダへの応用", 情報処理学会ソフトウェア工学研究会 SE77-14, pp.81-88, 1991.