

# 受信者数を考慮したブロードキャストシステムのためのプロセス代数

磯部祥尚 佐藤豊 大蒔和仁

システムの再利用性や拡張性を高めるために、システムを並行動作する複数の部品 (エージェント) に分割し、通信によって結合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、送り先を指定しないブロードキャスト的な通信方式が有効である。

マルチエージェントモデルの問題点は、全体の動作が複数のエージェントの協調動作によって決定されるため、その予測が困難であることにある。並行動作するエージェントの動作解析のための数学的枠組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ブロードキャスト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。

本論文で、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 Countable Broadcasting Systems) を提案する。他のプロセス代数と比較して、CCB ではブロードキャストされたメッセージの受信者数を適切に記述することが可能である。また CCB のための観測的な合同関係を定義し、有限エージェントに対する健全で完全な公理系を与える。

## 1 はじめに

ソフトウェアを効率的に開発していくためには、ソフトウェアを並行動作する複数の部品 (エージェント) に機能別に分割し、エージェント間の通信によって結合す

る方法が有効である。このような設計手法はマルチエージェントモデル [2] と呼ばれている。マルチエージェントモデルの利点は次のようにまとめられる。

1. 各エージェント毎に違う言語で記述できる。
2. ソフトウェアの再利用性が高い。
3. マシン依存性が低い。

マルチエージェントモデルではエージェントを柔軟に結合することが要求される。送り先を指定しないブロードキャスト通信方式は、メッセージの受信者数の変動に柔軟に対応できるため、拡張性の高いシステムを構築するために有効である。

ブロードキャスト通信をもつマルチエージェントモデルをもとにして、著者の一人は VIABUS [14] と呼ばれるエージェントの接続メカニズムを開発した。VIABUS はソフトウェアバス構造をもち、システム全体を停止させることなくエージェントの追加や修正を行なうことが可能である。VIABUS では次に示すブロードキャスト通信方式を採用している。

- メッセージは宛先の代わりに名前をもっている。
- 各エージェントは受信を希望するメッセージの名前を宣言する。

つまり、メッセージの受信は各エージェントに委ねられている。また、この受信を希望するメッセージの名前は固定ではなく動的に (実行時に) 変更可能である。図 1 に VIABUS における通信の例を示す。P1 から P4 まではエージェントを表し、各々 VIABUS に接続されている。各エージェントの右下の集合は受信を希望するメッセージの名前を表している。つまり、この例では P2 が名前 a をもつメッセージをブロードキャストし、P1 と P4 が

A Calculus of Countable Broadcasting Systems.  
Yoshinao Isobe, Yutaka Sato, Kazuhito Ohmaki, 電子  
技術総合研究所, Electrotechnical Laboratory.  
1995年1月30日受付。

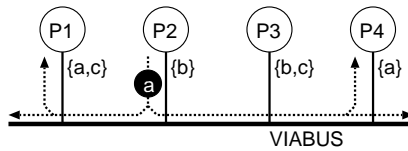


図1 VIABUSの通信例

それを受信している。ここで、もしメッセージ  $a$  の受信を希望するエージェント  $P_5$  が新しく追加された場合でも、 $P_2$  を修正する必要なく  $P_5$  もそれを受信することができる。

マルチエージェントモデルではソフトウェアの効率的な開発が可能であるが、エージェント間の相互作用による予想外の動作を防ぐため、エージェントの設計は慎重に行なわれなければならない。そこでエージェントの動作を設計時に解析するための道具が必要である。並行システムを記述し解析するための数学的道具としてはプロセス代数が知られており、現在までに数多くのプロセス代数が提案されている。特に CBS [12] [13] [4] はブロードキャストのために開発されたプロセス代数である。しかし、CBS ではブロードキャストされたメッセージの受信者数を扱うことが困難である。

利用者が自由にエージェントを追加修正できる場合、あるイベントの受信者数を固定することはできない。また、1つのエージェントにおいても、状況に応じて受信を希望するイベントを変える可能性もある。つまり、動的に受信者数を知ることは重要である。例えば、あるエージェントがイベントをブロードキャストして複数のエージェントを起動し、起動された全てのエージェントの終了を待ってから次の処理に進むような場合、そのイベントの受信者数(起動されたエージェントの数)を知ることは不可欠である。

図2の上の状態遷移図に示すように、一般に受信者数はフィードバックとタイムアウトを用いて数えられる。この例では、エージェント  $P$  はブロードキャストの後、その返信を数えてからエージェント  $P'(n)$  のように振舞うことを表している。しかし、CBS は有限時間を記述することができないため、返信を数えるためには無限の時間を必要とすることになり、実際にはこの振舞いを記述することはできない。

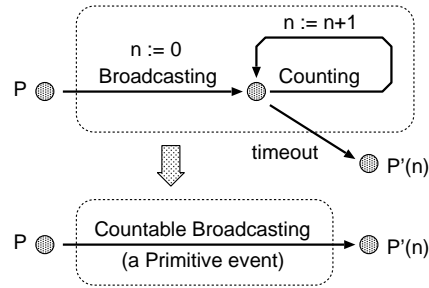


図2 可算ブロードキャストのモデル

我々はこの問題を解決するために、図2の下の状態遷移図に示すように、ブロードキャストとカウントの振舞いを1つの原子イベントに凝縮する。この原子イベントを可算ブロードキャスト (countable broadcast) と呼ぶ。そして、この可算ブロードキャストをもつプロセス代数として CCB (a Calculus of Countable Broadcasting Systems) を提案する。

一方、上記の CBS の問題は CBS に時間の概念 [11] を導入することによっても解決される。そこで時間の概念をもつ CBS と比較した CCB の特徴を次に記す。

1. CCB は解析コストが低い。
2. CCB は VIABUS の解析に適している。

受信者数のカウント動作はルーチンワークであり一度確立すれば何度も解析する対象ではない。CCB ではこのカウント動作は原子イベントとして確立しており、解析時のコストを低くすることができる。

また、VIABUS は可算ブロードキャストのための原子イベントをもっているため、VIABUS の通信は直接 CCB によって記述できる。実際には、VIABUS は図3のような星型の構造をもち、外部からは観測されない中心のエージェント  $C$  がブロードキャストを模倣するようにメッセージの集配を行なっている。このエージェント  $C$  は各メッセージの受信者数の情報を所有している。

プロセス代数の特徴はシステムの振舞いの等価性を代数的に証明できることであり、CCB に対しても観測的な合同関係が定義されていることが要求される。観測的とは外部から観測して違いが見られないことであり、合同関係とは全ての演算子で保存される等しい関係である。観測的な合同関係としては CCS で定義されている観測合同 [8] が知られているが、残念ながらこの観測合

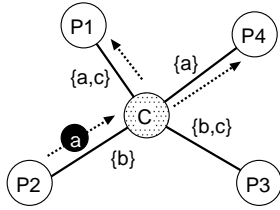


図3 VIABUSの実装 (cf:図1)

同はCCBでは合同関係にならない。そこで、本論文ではCCBのために観測的な合同関係を定義し、その特性について述べる。さらに有限エージェントのその合同関係に対する健全で完全な公理系を与える。

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のための観測的な合同関係を定義してその特性を示す。さらに有限エージェントのその合同関係に対する完全で健全な公理系を与える。6節でCore-CCBを基にCCBを定義する。

## 2 プロセス代数における通信方式

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、その中にはブロードキャスト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ための問題点を指摘する。

### 2.1 1対1通信

1対1通信をもつプロセス代数としてCCS[8]、 $\pi$ -計算[9]、CHOCS[16]などがある。1対1通信でも、受信希望者の数だけメッセージを繰り返し送信することによって、ブロードキャストを模倣できると考えられる。しかし、文献[4]で述べられているように、そのメッセージの受信希望者数を動的に知ることは困難である。次の例はCCSで、メッセージ $a$ の受信者数を数えようとし

てうまくいかない例である。

$$P_1 \stackrel{\text{def}}{=} a.0, \quad P_2 \stackrel{\text{def}}{=} b.0, \quad P_3 \stackrel{\text{def}}{=} a.0$$

$$C(i) \stackrel{\text{def}}{=} \bar{a}.C(i+1) + \overline{\text{out}}(i).0$$

$$SYS \stackrel{\text{def}}{=} (C(0)|P_1|P_2|P_3) \setminus \{a, b\}$$

エージェント $C(i)$ はメッセージ $a$ による通信毎に値変数 $i$ に1を加算して、メッセージ $\text{out}$ を用いてその値を出力することができる。エージェント $P_1$ と $P_3$ がメッセージ $a$ を受信可能なので、 $\text{out}$ 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<sup>†1</sup>。

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

これは、 $C(i)$ がメッセージ $a$ を $P_1$ と $P_3$ の両方に送信する前にメッセージ $\text{out}$ を送信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[1]を導入して、メッセージ $a$ をメッセージ $\text{out}$ 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、メッセージ $a$ 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 $\text{out}$ 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, CHOCS)を用いたとしても残される。

### 2.2 マルチキャスト

マルチキャストを記述可能なプロセス代数としてSCCS[10]、Meije[15]などがある。ここで、マルチキャストは「送信者が受信者数を指定してメッセージを送信する」通信方式を意味している。次の例はSCCS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<sup>†2</sup>。

$$P_0 \times P_1 \times P_2 \xrightarrow{1} P'_0 \times P'_1 \times P'_2$$

ここで、各成分は次のように定義されている。

$$P_0 \stackrel{\text{def}}{=} a^{-2}.P'_0, \quad P_1 \stackrel{\text{def}}{=} a^1.P'_1, \quad P_2 \stackrel{\text{def}}{=} a^1.P'_2$$

$P_0$ は2つのエージェントにマルチキャストすることを明示している<sup>†3</sup>。このためメッセージ $a$ を要求するエージェントを新しく1つ追加した場合は、 $P_0$ を

$$P_0 \stackrel{\text{def}}{=} a^{-3}.P'_0,$$

のように修正する必要がある。ブロードキャストを模倣するためには、CCSのときと同様にメッセージ $a$ の受

†1  $\sim$ はCCSの強等価[8]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。

信希望者を数える必要があるが、これは CCS のときと同じ理由で困難である<sup>†4</sup>。

### 2.3 ブロードキャスト

ブロードキャストを記述可能なプロセス代数として CSP [3]、LOTOS [18]、CBS [12]、CCS+b [5] などがあり、特に CBS と CCS+b はブロードキャストのために開発されたプロセス代数である。ブロードキャストでは送信者は受信者数を指定せずにメッセージを送信し、そのメッセージの全ての受信希望者がそれを受信することができる<sup>†5</sup>。次の例は、CBS で  $P_0$  がメッセージ  $a$  をブロードキャストし、 $P_1$  と  $P_3$  がそれを受信する例である<sup>†6</sup>。

$$(P_0|P_1|P_2|P_3) \xrightarrow{a!} (P'_0|P'_1|P_2|P'_3)$$

ここで、各成分は次のように定義されている。

$$P_0 \stackrel{\text{def}}{=} a!.P'_0, \quad P_1 \stackrel{\text{def}}{=} a?.P'_1 + b?.P''_1 \\ P_2 \stackrel{\text{def}}{=} b?.P'_2, \quad P_3 \stackrel{\text{def}}{=} a?.P'_3 + c?.P'_3''$$

$P_2$  は  $\xrightarrow{a?}$  のような状態遷移をもっていないため、メッセージ  $a$  を受信していない。ブロードキャストをプロセス代数で扱うためには、このように「状態遷移できない条件  $\xrightarrow{a?}$ 」を用いる方法が適切である。

残念ながら、1節で述べたように CBS を含むこれらのプロセス代数では、ブロードキャスト後にその受信者数を知ることは困難である。CBS については7節で再び考察する。

## 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 を提案する。CCB は次の特長をもっている。

1. 可算ブロードキャストを記述できる。
2. マルチキャストを記述できる。

マルチキャストは SCCS でも記述できるので、CCB 固有の特徴は可算ブロードキャストにある。

CCB のイベントは  $a[x]\theta\langle y\rangle(z)$  の形をもつ。ここで  $a$  は基本メッセージ名、 $[x]$  はポストフィクス、 $\theta$  はイベントの属性、 $\langle y\rangle$  は受信者数、 $(z)$  はこのイベントによって渡されるメッセージである<sup>†7</sup>。メッセージ名  $a[x]$  は基本メッセージ名  $a$  とポストフィクス  $[x]$  から構成され、2つのメッセージ名の基本メッセージ名とポストフィクスが各々等しいとき、それらのメッセージ名も等しいとみなされる。メッセージの名前が  $a[x]$  であるメッセージを、メッセージ  $a[x]$  と書く。 $\pi$  計算のように通信範囲の実行時の拡張などは記述できないが、ポストフィクスには変数が使えるのでエージェント間の結合を実行時に変更可能である。

イベントの属性  $\theta$  は  $!, !!, ?, ??$  の4種類であり、以下に各々の役割を説明する。

1.  $a[i]!\langle n\rangle(m)$  はメッセージ  $a[i]$  をマルチキャストするために使われるイベントである。 $m$  はこの通信で渡されるメッセージの内容であり、 $n$  はこのメッセージを受信するエージェントの数である。 $n$  は非負の整数であるが、特に  $n$  が 0 であるとき、 $a[i]!\langle 0\rangle(m)$  を内部イベントと呼ぶ (CCS の  $\tau$  に相当する)。便宜上、任意の内部イベントを表すための変数として  $\tau$  を用いる<sup>†8</sup>。内部イベントは観測されないイベントであり、観測的な解析を行なう時は可能な限り無視されるべきイベントである。
2.  $a[i]!!\langle x\rangle(m)$  はメッセージ  $a[i]$  をブロードキャストするために使われるイベントである。 $m$  はこの通信で渡されるメッセージの内容であり、 $x$  はこのメッセージの受信者数に束縛される変数である。
3.  $a[i]?\langle n\rangle(y)$  はマルチキャストされたメッセージ  $a[i]$  を受信するために使われるイベントである。 $n$  は正の整数で 2 以上を指定することにより 1 つのイベントで複数のエージェントのためにメッセージを受信することができる。 $y$  はこの通信で受けとるメッセージの内容に束縛される変数である。
4.  $a[i]??\langle n\rangle(y)$  はブロードキャストされたメッセージ  $a[i]$  を受信するために使われるイベントである。 $n$

<sup>†4</sup> SCCS でブロードキャスト用のプロセス代数 CBS を解釈する方法が文献 [4] に示されている。これについては7節で述べる。

<sup>†5</sup> CCS+b でのブロードキャストの定義は我々の定義と異なり、全ての受信希望者が受信する必要はない。

<sup>†6</sup> イベントの属性  $!?$  と  $??$  は各々メッセージの送信と受信を表している。

<sup>†7</sup> 0 のポストフィクス  $[0]$ 、1 の受信者数  $\langle 1\rangle$ 、空のメッセージ  $()$  は各々省略されることがある。

<sup>†8</sup> CCS と違い  $\tau$  は変数として使われる。違う 2 つの内部イベントを表すためには、 $\tau_1$  と  $\tau_2$  などを用いる。

と  $y$  の役割は  $a[i]? \langle n \rangle (y)$  と同じである。

受信を表す  $?$  と  $!$  の違いは、「 $?$  はメッセージ  $a[i]$  を受信できる」ことに対し「 $!$  はメッセージ  $a[i]$  を受信しなければならない」ことである。 $?$  は受信希望者は受信しなければならないというブロードキャストの本質を表している。CCB では受信者数を 1 に指定したマルチキャストは CCS の通信方式と同じになるように定義している。つまり、1 つのメッセージ送信に対して複数の受信者がいる場合は非決定的に 1 つのエージェントが受信できる通信方式である。全てのエージェントが受信しなければならないことを要求すると CCS のような非決定的な通信は成功しない。そこで、CCB では  $?$  と  $!$  の 2 種類の受信を用意している。

送信についても、マルチキャストを  $a! \langle \text{定数} \rangle$ 、ブロードキャストを  $a! \langle \text{変数} \rangle$  とせず、2 種類の属性  $!$  と  $!!$  を用意したのは、マルチキャストでも送信が起こるまでに束縛されていれば、受信者数に変数を使うことができるからである。例えば、

$$a?(1)(x).bl(x)(message).0$$

のように、メッセージ  $a$  で受けとった数  $x$  だけ、メッセージ  $b$  をマルチキャストすることが可能である。

以下の小節で CCB の記述例を示し、その特徴を説明する。

### 3.1 マルチキャストの例

下の記述はエージェント  $P_0$  がメッセージ  $a$  を受信者数を 2 に指定してマルチキャストする例である。

$$\begin{aligned} P_0 &\stackrel{\text{def}}{=} a! \langle 2 \rangle . P'_0 \\ P_i &\stackrel{\text{def}}{=} a?. P'_i \quad (i \in \{1, 2, 3\}) \\ SYS_1 &\stackrel{\text{def}}{=} (P_0 | P_1) \setminus \{a\} \\ SYS_2 &\stackrel{\text{def}}{=} (P_0 | P_1 | P_2) \setminus \{a\} \\ SYS_3 &\stackrel{\text{def}}{=} (P_0 | P_1 | P_2 | P_3) \setminus \{a\} \end{aligned}$$

次の等式が示すように、メッセージ  $a$  の受信希望者数が指定数より小さい場合は通信は起こらず、大きい場合は受信できないエージェントが存在する。どのエージェン

トが受信できないかは非決定的である。

$$\begin{aligned} SYS_1 &\sim 0 \\ SYS_2 &\sim \tau.(P'_0 | P'_1 | P'_2) \setminus \{a\} \\ SYS_3 &\sim \tau.(P'_0 | P'_1 | P'_2 | P'_3) \setminus \{a\} \\ &\quad + \tau.(P'_0 | P'_1 | P'_2 | P'_3) \setminus \{a\} \\ &\quad + \tau.(P'_0 | P_1 | P'_2 | P'_3) \setminus \{a\} \end{aligned}$$

### 3.2 可算ブロードキャストの例

次に上記の例をブロードキャストで記述する

$$\begin{aligned} P_0 &\stackrel{\text{def}}{=} a! \langle x \rangle . P'_0(x) \\ P_i &\stackrel{\text{def}}{=} a?. P'_i \quad (i \in \{1, 2, 3\}) \end{aligned}$$

$$\begin{aligned} SYS_1 &\stackrel{\text{def}}{=} (P_0 | P_1) \setminus \{a\} \\ SYS_2 &\stackrel{\text{def}}{=} (P_0 | P_1 | P_2) \setminus \{a\} \\ SYS_3 &\stackrel{\text{def}}{=} (P_0 | P_1 | P_2 | P_3) \setminus \{a\} \end{aligned}$$

次の等式が示すように、受信者数に依存せず全ての受信希望者がメッセージを受信することができている。また、通信後に変数  $x$  は受信者数に束縛されている。

$$\begin{aligned} SYS_1 &\sim \tau.(P'_0(1) | P'_1) \setminus \{a\} \\ SYS_2 &\sim \tau.(P'_0(2) | P'_1 | P'_2) \setminus \{a\} \\ SYS_3 &\sim \tau.(P'_0(3) | P'_1 | P'_2 | P'_3) \setminus \{a\} \end{aligned}$$

### 3.3 ポストフィクスの例

エージェント間の結合を実行時に変更するためにポストフィクスを使うことができる。下の記述は、 $P_1$  からの情報 'No1' を  $Q_1$  へ、 $P_2$  からの情報 'No2' を  $Q_2$  へ中継するエージェント  $M$  の例である。

$$\begin{aligned} P_1 &\stackrel{\text{def}}{=} id!(1).in[1]!(\text{'No1'}).0 \\ P_2 &\stackrel{\text{def}}{=} id!(2).in[2]!(\text{'No2'}).0 \\ M &\stackrel{\text{def}}{=} id?(i).in[i]?(x).out[i]!(x).M \\ Q_1 &\stackrel{\text{def}}{=} out[1]?(y).Q'_1(y) \\ Q_2 &\stackrel{\text{def}}{=} out[2]?(y).Q'_2(y) \\ SYS &\stackrel{\text{def}}{=} (P_1 | P_2 | M | Q_1 | Q_2) \setminus L \end{aligned}$$

$$\text{where } L = \{id, in, out\}$$

$M$  は転送先を間違えないために先ず  $P_i$  から ID を受けとり、変数  $i$  をその ID に束縛する。次に変数  $x$  をメッセージ  $in[i]$  によって受けとった内容に束縛し、メッセージ  $out[i]$  によって  $Q_i$  に転送する。ポストフィクス  $[i]$  によって、次の等式にみられるように各々の情報は確実に送り先に転送される。

$$\begin{aligned} SYS &\sim \tau.\tau.\tau.(0 | P_2 | M | Q'_1(\text{'No1'}) | Q_2) \setminus L \\ &\quad + \tau.\tau.\tau.(P_1 | 0 | M | Q_1 | Q'_2(\text{'No2'})) \setminus L \end{aligned}$$

### 3.4 記述例のまとめ

下は、「メッセージの送信者に、そのメッセージの受信者が自分のプロセス ID(P-ID) を返信する動作」を CCB を用いて記述した例の一部である<sup>†9</sup>。

$$\begin{aligned}
 P_1(id) &\stackrel{\text{def}}{=} a!!\langle x \rangle(id).P'_1(x, id, \text{nil}) \\
 P'_1(x, id, idlist) &\stackrel{\text{def}}{=} \llbracket x \neq 0 \rrbracket \text{ack}[id]?(cid).P'_1(x-1, id, cid::idlist) \\
 &\quad + \llbracket x = 0 \rrbracket P''_1(id, idlist) \\
 &\quad \vdots \\
 Q_1 &\stackrel{\text{def}}{=} a?(pid).(getid?(id).ack[pid]!(id).\dots|Q_1) \\
 Q_2 &\stackrel{\text{def}}{=} b?(pid).(getid?(id).ack[pid]!(id).\dots|Q_2) \\
 &\quad \vdots \\
 ID(i) &\stackrel{\text{def}}{=} getid!(i).ID(i+1) \\
 SYS &\stackrel{\text{def}}{=} (P_1(1)|P_1(2)|P_2(3)|\dots \\
 &\quad |Q_1|Q_2|Q_3|\dots|ID(\$QID)) \setminus \{a, b, \dots\}
 \end{aligned}$$

$P_1$  はメッセージ  $a$  を用いて自分の P-ID  $id$  をブロードキャストした後、 $P'_1$  のように振舞う。 $P'_1$  は変数として、 $a$  の受信者数  $x$ 、自分の P-ID  $id$ 、 $a$  の受信者の P-ID を保存するためのリスト変数  $idlist$  をもっている。 $x$  が 0 でなければ、メッセージ  $ack$  を通して  $a$  の受信者の P-ID を  $cid$  に束縛し、その P-ID を  $idlist$  に加え<sup>†10</sup>、 $x$  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 P-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  $a$  を受信すると、変数  $pid$  に  $a$  の送信者の P-ID を束縛して、自分の P-ID を  $ID(i)$  から受けとり、 $a$  の送信者に自分の P-ID を  $ack$  により返信する。それと同時に、次の  $a$  を受信するために複製された  $Q_1$  が並行に走り出す。

この記述の特長は、メッセージ  $a$  に反応する  $Q_n$  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、P-ID を渡してポストフィクスを用いることにより、送信者に確実に返信することができる。

## 4 Core-CCB の定義

CCB ではプロセス数は有限であり、扱うデータも離散的で有限であると仮定している。これは現実の計算機

環境には矛盾していない。実際の計算機ではプロセス数も有限であり、扱える実数も有限である。

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 は値変数<sup>†11</sup>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、我々は名前の有限集合  $\mathcal{N} = \{a, b, c, \dots\}$  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。

$$\begin{aligned}
 Event = &\{a\theta^n : a \in \mathcal{N}, \theta \in \mathcal{T}, n \in \mathcal{I}\} \\
 &\cup \{a!^0, a!!^0 : a \in \mathcal{N}\}
 \end{aligned}$$

ここで、 $\mathcal{I}$  は正の整数の無限集合  $\{1, 2, \dots\}$ 、 $\mathcal{T}$  はイベント属性の集合  $\{!, ?, !!, ??\}$  である。 $\mathcal{T}$  の要素は  $\theta, \phi, \dots$  によって表される。イベントを表すためには変数  $\alpha, \beta, \dots$  を用いるが、イベントの名前や属性を考慮するときは  $a\theta^n, b\phi^m, \dots$  の形を用いる。次に  $Event$  の部分集合として集合  $Tau$  を次のように与える。

$$Tau = \{a!^0 : a \in \mathcal{N}\}$$

この  $Tau$  の要素を内部イベントと呼び<sup>†12</sup>、内部イベントを表すために変数  $\tau, \tau', \tau_1, \dots$  を用いる。

名前の集合  $\mathcal{N}$  から  $\mathcal{N}$  への関数をリネイミング関数という。任意の  $i \in \{1, \dots, n\}$  について  $S(a_i) = a'_i$ 、それ以外では  $S(a) = a$  であるようなリネイミング関数  $S$  を、

$$(a'_1/a_1, a'_2/a_2, \dots, a'_n/a_n)$$

と記述する。また、通信の領域を制限するための  $L$  は

$$\{a_1, \dots, a_n\}$$

の形をもち、名前の有限集合  $\mathcal{N}$  の部分集合である。

このとき、Core-CCB のシンタックスを次のように定義する。

定義 4.1 Core-CCB のエージェント式の集合  $\mathcal{E}$  (要素を  $E, F, \dots$  で表す) は次の式を含む最小の集合である

†9  $\llbracket b \rrbracket P$  は条件分岐を表しており、真偽式  $b$  が真ならばエージェント  $P$  のように振舞う。

†10  $::$  はリストの結合演算子である。

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

†12  $a!^0$  は全ての  $a$  の受信希望者に  $a$  を受信させる役割を持っており、内部イベントではない。

†13.

 $X$  : エージェント変数 ( $X \in \mathcal{X}$ ) $A$  : エージェント定数 ( $A \in \mathcal{K}$ ) $0$  : 無動作エージェント $\alpha.E$  : プレフィクス ( $\alpha \in Event$ ) $E + F$  : 選択 $E|F$  : 並行合成 $E[S]$  : リネイミング ( $S$ : リネイミング関数) $E \setminus L$  : 制限 ( $L: \mathcal{N}$  の部分集合)

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  $\mathcal{P}$  で表し、その要素を  $P, Q, \dots$  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、

$$A \stackrel{\text{def}}{=} P \quad (P \in \mathcal{P})$$

の形の定義式があると仮定する。さらに、 $\mathcal{P}$  の中で  $A$  は弱ガードされていると仮定する<sup>†14</sup>。この仮定は各エージェントが受信を希望しているメッセージの名前を計算するために必要であり、CBS[12]でも同様の仮定が用いられている。

#### 4.2 セマンティクス

まず、エージェント式から名前の部分集合への関数である監視関数 (monitor function) を定義する。

定義 4.2 各エージェント式について、監視関数  $mon: \mathcal{E} \rightarrow 2^{\mathcal{N}}$  を次のように帰納的に定義する。

$$mon(0) = \emptyset$$

$$mon(a\theta^n.E) = \begin{cases} \{a\} & (\theta = ?) \\ \emptyset & (\text{otherwise}) \end{cases}$$

$$mon(E + F) = mon(E) \cup mon(F)$$

$$mon(E|F) = mon(E) \cup mon(F)$$

$$mon(E[S]) = \{S(a) : a \in mon(E)\}$$

$$mon(E \setminus L) = mon(E) - L$$

†13 Core-CCB で用いられる構文  $\sum_{i \in I} P_i$  は、 $I \neq \emptyset$  ならば  $P_1 + P_2 + \dots + P_n$ 、 $I = \emptyset$  ならば  $0$  によって書き換えられる有限選択に相当する。

†14 全ての  $X$  が  $E$  の部分式  $\alpha.F$  の中に含まれているならば、 $X$  は  $E$  の中で弱ガードされているという

$$mon(A) = mon(P) \quad (A \stackrel{\text{def}}{=} P)$$

$$mon(X) = \emptyset$$

エージェント定数は弱ガードされているので、この関数  $mon$  は効果的に計算可能である。

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。

$$(\mathcal{E}, Event, \{\overset{\alpha}{\rightarrow} : \alpha \in Event\})$$

例えば、 $E \overset{\alpha}{\rightarrow} E'$  はエージェント式  $E$  がイベント  $\alpha$  を実行でき、その実行後はエージェント式  $E'$  のように振舞うことを意味している。

定義 4.3 エージェント式間の遷移関係  $\overset{\alpha}{\rightarrow}$  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。

$$\text{Event} \frac{}{\alpha.E \overset{\alpha}{\rightarrow} E}$$

$$\text{Choice}_1 \frac{E \overset{\alpha}{\rightarrow} E'}{E + F \overset{\alpha}{\rightarrow} E'}$$

$$\text{Choice}_2 \frac{F \overset{\alpha}{\rightarrow} F'}{E + F \overset{\alpha}{\rightarrow} F'}$$

$$\text{Ren} \frac{E \overset{a\theta^n}{\rightarrow} E'}{E[S] \overset{S(a)\theta^n}{\rightarrow} E'[S]}$$

$$\text{Res}_1 \frac{E \overset{a\theta^n}{\rightarrow} E'}{E \setminus L \overset{a\theta^n}{\rightarrow} E' \setminus L} \quad (a \notin L)$$

$$\text{Res}_2 \frac{E \overset{a\theta^0}{\rightarrow} E'}{E \setminus L \overset{a!^0}{\rightarrow} E' \setminus L} \left( \begin{array}{l} \theta \in \{!, \#\}, \\ a \in L \end{array} \right)$$

$$\text{Con} \frac{P \overset{\alpha}{\rightarrow} P'}{A \overset{\alpha}{\rightarrow} P'} \quad (A \stackrel{\text{def}}{=} P)$$

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

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

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

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

$$\text{Para}_5 \frac{E \xrightarrow{a\theta^m} E' \quad F \xrightarrow{a\theta^n} F'}{E|F \xrightarrow{a\theta^{(m+n)}} E'|F'} \left( \theta \in \{?, ?\} \right)$$

ここで、@ は

$$@(!) = ?, @(?) = !, @(!!) = ?, @(?) = !!$$

の様に定義される  $T$  から  $T$  への関数である。

この定義のもとで次の命題が成り立つ。

命題 4.1  $E \xrightarrow{a\theta^n}$  for any  $n$  iff  $a \notin \text{mon}(E)$

証明 付録参照

つまり、推論規則  $\text{Para}_{1,2}$  の条件  $a \notin \text{mon}(E)$  は、ブロードキャストされたイベント  $a$  の受信を  $E$  が希望していないことを意味している。

## 5 観測的な合同関係

Core-CCB にも CCS と同様に観測的な合同関係を要求する。しかし、CCS で定義された観測合同  $\equiv$  が、次の例に示されるように Core-CCB では合同関係にならないことが重要な問題である。

$$P_1 = P_2, \quad P_1|Q \neq P_2|Q$$

ここで、各成分は次のように定義されている。

$$P_1 \stackrel{\text{def}}{=} a?!.\tau.P_1, \quad P_2 \stackrel{\text{def}}{=} a?!.P_2$$

$$Q \stackrel{\text{def}}{=} a!!^0.out0!^1.Q + a!!^1.out1!^1.Q$$

$\tau$  は内部イベントであるので、 $P_1$  と  $P_2$  は観測合同となる。しかし、その各々に  $Q$  を合成するとその結果は観測合同にならない。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。そこで、観測合同に可能な限り弱い条件を付加して観測的な Core-CCB の合同関係を定義する必要がある。まず、その定義のために必要な記法を準備する。

$Event^*$  を空列  $\varepsilon$  を含む有限イベント列の集合とする。 $t = \alpha_1 \cdots \alpha_k \in Event^*$  について、もし  $E \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_k} E'$  ならば、 $E \xrightarrow{t} E'$  とする。特に、 $E \xrightarrow{\varepsilon} E'$  ならば  $E' \equiv E$ <sup>†15</sup> である。また、イベント列

†15  $\equiv$  は構文的に等しいことを意味する。

$t$  に含まれるイベントが全て内部イベント ( $t = \tau_1 \cdots \tau_k$ ) のとき、 $E \xrightarrow{t} E'$  ならば、 $E \Longrightarrow E'$  とする。このとき、次の状態遷移関係を定義する。

定義 5.1  $t = \alpha_1 \cdots \alpha_k \in Event^*$  とする。もし、 $E \Longrightarrow \xrightarrow{\alpha_1} \Longrightarrow \cdots \Longrightarrow \xrightarrow{\alpha_k} \Longrightarrow E'$  ならば、 $E \xrightarrow{t} E'$  である。

次に、CCS の弱双模倣 (weak bisimulation) に相当する Core-CCB の双模倣関係を定義する。

定義 5.2 エージェント上の二項関係  $S$  が弱監視双模倣 (weak monitor bisimulation) であるとは、 $(P, Q) \in S$  ならば、任意の  $\alpha \in Event$  と  $a \in \mathcal{N}$  について、次の 4 つの条件が成り立つことである<sup>†16</sup>。

(i)  $P \xrightarrow{\alpha} P'$  ならば、ある  $Q'$  が存在して、

$Q \xrightarrow{\hat{\alpha}} Q'$  かつ  $(P', Q') \in S$  を満たす。

(ii)  $Q \xrightarrow{\alpha} Q'$  ならば、ある  $P'$  が存在して、

$P \xrightarrow{\hat{\alpha}} P'$  かつ  $(P', Q') \in S$  を満たす。

(iii)  $a \notin \text{mon}(P)$  ならば、ある  $Q', Q''$  が存在して、

$Q \Rightarrow Q' \Rightarrow Q'', a \notin \text{mon}(Q'), (P, Q'') \in S$  を満たす。

(iv)  $a \notin \text{mon}(Q)$  ならば、ある  $P', P''$  が存在して、

$P \Rightarrow P' \Rightarrow P'', a \notin \text{mon}(P'), (P'', Q) \in S$  を満たす。

弱監視双模倣は弱双模倣に条件 (iii)(iv) を追加して得られている。この弱監視双模倣を用いて弱監視等価を定義する。

定義 5.3 もし、ある弱監視双模倣  $S$  において  $(P, Q) \in S$  ならば、エージェント  $P$  と  $Q$  は弱監視等価 (weak monitor equivalence) であるといい、 $P \approx_m Q$  と書く。

次の命題 5.1 に示されるように、弱監視等価は CCS の観測等価  $\approx$  に含まれ、かつ Core-CCB の並行合成演算子  $|$  によって保存される最大の関係である。

命題 5.1 次の関係が成り立つ。

1.  $P_1 \approx_m P_2$  ならば、任意の  $Q$  について  $P_1|Q \approx_m P_2|Q$  である。

2.  $\mathcal{R}$  を、「 $\mathcal{R} \subseteq \approx$ 」かつ「 $(P_1, P_2) \in \mathcal{R}$  ならば、任意の  $Q$  について  $(P_1|Q, P_2|Q) \in \mathcal{R}$ 」を満たす関係とする<sup>†17</sup>。このとき、 $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$  のよう

†16  $\hat{t}$  はイベント列  $t$  の内部イベントを全て除いたイベント列である。

†17 実際には任意の  $Q$  である必要はなく、この命題の証明で定義するエージェント  $T$  についてのみで十分である。

な<sup>†18</sup>任意の  $(P_1, P_2) \in \mathcal{R}$  について、 $P_1 \simeq_m P_2$  である。

証明 付録参照

ここで、弱監視双模倣の条件 (iii), (iv) において、 $P \equiv P' \equiv P'', Q \equiv Q' \equiv Q''$  の特殊な場合を考える。これは、(iii), (iv) の内部イベントによる状態遷移を許さない場合であり、弱監視双模倣に対して強監視双模倣として定義できる。

定義 5.4 エージェント上の二項関係  $S$  が強監視双模倣 (strong monitor bisimulation) であるとは、 $(P, Q) \in S$  ならば、任意の  $\alpha \in Event$  について、次の3つの条件が成り立つことである。

- (i)  $P \xrightarrow{\alpha} P'$  ならば、ある  $Q'$  が存在して、  
 $Q \xrightarrow{\hat{\alpha}} Q'$  かつ  $(P', Q') \in S$  を満たす。
- (ii)  $Q \xrightarrow{\alpha} Q'$  ならば、ある  $P'$  が存在して、  
 $P \xrightarrow{\hat{\alpha}} P'$  かつ  $(P', Q') \in S$  を満たす。
- (iii)  $mon(P) = mon(Q)$

弱監視等価と同様に、強監視等価を定義する。

定義 5.5 もし、ある強監視双模倣  $S$  において  $(P, Q) \in S$  ならば、エージェント  $P$  と  $Q$  は強監視等価 (strong monitor equivalence) であるといい、 $P \simeq_m Q$  と書く。

以下、本論文ではこの特殊な場合である強監視等価についてのみ述べる。強監視等価ならば弱監視等価であるので、定義の簡単な強監視等価の性質から調べることは弱監視等価の性質を理解するためにも有効である。強監視等価は弱監視等価よりも強い条件を要求しているが、強等価の条件よりはまだ十分に弱く、観測的な等価性の性質を残している。

強監視等価は同値関係であることを示すことができるが、選択演算子  $+$  によって保存されないため合同関係ではない。そこで、強監視等価の条件に最も弱い条件を付加して合同関係を定義する。そのために、まず無数に存在する内部イベント間の違いを無視するため、イベント上の同値関係  $\equiv$  を導入する。

†18  $\mathcal{L}(P)$  は  $P$  に現れる全てのイベントの名前の集合を表す。 $\mathcal{N}$  は有限集合であるので、有限選択しかもたない Core-CCB でも名前を使い尽くすことはできる。残念ながら我々はこの条件なしに証明することができなかった。

定義 5.6 イベント等価  $\equiv$  は

$$\equiv = \{(\tau, \tau') : \tau, \tau' \in Tau\} \cup \{(\alpha, \alpha) : \alpha \in Event\}$$

のように定義されるイベント上の関係である。

このとき、強監視合同を次のように定義する。

定義 5.7 もし、任意の  $\alpha \in Event$  について、次の3つの条件を満たすならば、 $P$  と  $Q$  は強監視合同 (strong monitor congruence) であるといい、 $P \cong_m Q$  と書く。

- (i)  $P \xrightarrow{\alpha} P'$  ならば、ある  $Q', \alpha'$  が存在して、  
 $Q \xrightarrow{\alpha'} Q', P' \simeq_m Q', \alpha \equiv \alpha'$  を満たす。
- (ii)  $Q \xrightarrow{\alpha} Q'$  ならば、ある  $P', \alpha'$  が存在して、  
 $P \xrightarrow{\alpha'} P', P' \simeq_m Q', \alpha \equiv \alpha'$  を満たす。
- (iii)  $mon(P) = mon(Q)$

今までエージェント上に強監視合同を定義してきたが、ここでエージェント変数を含むエージェント式に拡張しておく。

定義 5.8 エージェント式  $E$  または  $F$  に含まれるエージェント変数を  $X_1, \dots, X_n$  (以後  $\tilde{X}$  と書く) とする。このとき、もし全ての  $\tilde{P}$  について、 $E\{\tilde{P}/\tilde{X}\} \cong_m F\{\tilde{P}/\tilde{X}\}$  ならば、 $E \cong_m F$  とする。

強監視合同について CCS の観測合同と同様な特性を証明することができる。以下にそのいくつかを示す。

命題 5.2 強監視合同は合同関係である。

証明 付録参照

命題 5.3  $P \simeq_m Q$  iff  $((P \cong_m Q) \text{ or } (P \cong_m \tau.Q + Q) \text{ or } (\tau.P + P \cong_m Q))$

証明 付録参照

命題 5.4 エージェント式  $\tilde{E}$  は高々エージェント変数  $\tilde{X}$  を含むとし、 $\tilde{X}$  は  $\tilde{E}$  のなかでガード<sup>†19</sup>され、かつ逐次的<sup>†20</sup>であるとする。このとき、もし、 $\tilde{P} \cong_m \tilde{E}\{\tilde{P}/\tilde{X}\}$  かつ  $\tilde{Q} \cong_m \tilde{E}\{\tilde{Q}/\tilde{X}\}$  ならば、 $\tilde{P} \cong_m \tilde{Q}$  である。

証明 付録参照

命題 5.3は定理 5.1の証明で使われるように、 $\cong_m$  に対する帰納的な証明に有効である。また、命題 5.4は強監視合同に関する唯一解の存在を示している。

†19  $X$  の全ての出現が  $\alpha.F$  の形をもつ  $E$  の部分式のなかにあるならば、 $X$  は  $E$  のなかでガードされているという、ただし、 $\alpha$  は内部イベントであってはならない。

†20 もし、 $X$  を含む  $E$  の全ての部分式 ( $X$  自身は除く) が  $\alpha.F$  が  $\sum \tilde{F}$  の形をもつならば、 $X$  は  $E$  のなかで、逐次的であるという。

この節の残りで、我々は有限エージェントのための健全で完全な公理系  $\mathcal{A}$  を与える。有限エージェントとはエージェント定数を含まない(再帰をもたない)エージェントのことである。

定義 5.9 二つの有限エージェント  $P$  と  $Q$  の等価性が公理系  $\mathcal{A}$  から推論されるならば、 $\mathcal{A} \vdash P = Q$  と書く。ここで、公理系  $\mathcal{A}$  は次の等式から構成される。

- **M1**  $P_1 + P_2 = P_2 + P_1$
- **M2**  $(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$
- **M3**  $P = P + P$
- **M4**  $P = P + 0$
- **T1**  $\alpha.(\tau.P + P) = \alpha.P$
- **T2**  $P + R + \tau.(P + Q) = R + \tau.(P + Q)$   
if  $\text{mon}(P) \subseteq \text{mon}(R)$
- **T3**  $\alpha.(P + \tau.Q) + \alpha.Q = \alpha.(P + \tau.Q)$
- **T4**  $\tau.P = \tau'.P$
- **E1**  $(\sum_{(i \in I_1)} a_i(\theta_i)^{m_i}.P'_i) | (\sum_{(i \in I_2)} b_i(\phi_i)^{n_i}.Q'_i)$   
 $= \sum_{(i \in I_1)} \{a_i(\theta_i)^{m_i}.(P'_i | Q)\}$   
 $\theta_i \in \{!, ?\}$  or  $a_i \notin \text{mon}(Q)$   
 $+ \sum_{(i \in I_2)} \{b_i(\phi_i)^{n_i}.(P | Q'_i)\}$   
 $\phi_i \in \{!, ?\}$  or  $b_i \notin \text{mon}(P)$   
 $+ \sum_{(i \in I_1)} \sum_{(j \in I_2)} \{a_i(\theta_i)^{(m_i - n_j)}.(P'_i | Q'_j)\}$   
 $a_i = b_j, \theta_i \in \{!, !!\}, \phi_j = @(\theta_i), m_i \geq n_j$   
 $+ \sum_{(i \in I_1)} \sum_{(j \in I_2)} \{b_j(\phi_j)^{(n_j - m_i)}.(P'_i | Q'_j)\}$   
 $a_i = b_j, \phi_j \in \{!, !!\}, \theta_i = @(\phi_j), n_j \geq m_i$   
 $+ \sum_{(i \in I_1)} \sum_{(j \in I_2)} \{a_i(\theta_i)^{(m_i + n_j)}.(P'_i | Q'_j)\}$   
 $a_i = b_j, \theta_i = \phi_j \in \{?, ??\}$
- **E2**  $(\sum_{(i \in I)} a_i(\theta_i)^{n_i}.P'_i) \setminus L$   
 $= \sum_{(i \in I)} \{a_i(\theta_i)^{n_i}.(P'_i \setminus L) : a_i \notin L\}$   
 $+ \sum_{(i \in I)} \{a_i!^0.(P'_i \setminus L) : a_i \in L, \theta_i \in \{!, !!\}, n_i = 0\}$
- **E3**  $(\sum_{(i \in I)} a_i(\theta_i)^{n_i}.P'_i)[S]$   
 $= \sum_{(i \in I)} S(a_i)(\theta_i)^{n_i}.(P'_i[S])$

**M1-4**、**T1-4**、**E1-3** は各々 CCS のモノイド規則、 $\tau$  規則、展開規則に相当する。次の定理 5.1 に示されるように、この公理系  $\mathcal{A}$  は有限エージェントの強監視合同について健全で完全である。

定理 5.1  $P$  と  $Q$  を有限エージェントとする。このとき、 $P \cong_m Q$  iff  $\mathcal{A} \vdash P = Q$ 。

証明 付録参照

## 6 CCB の定義

Core-CCB は値変数をもたないが、値変数の変域をカバーするように選択演算子  $+$  を用いることによって、値変数を扱うことができる。しかし、その記述はやや複雑になるため、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に CCB を与える。ただし、Core-CCB では有限選択  $+$  であるため、値として無限数は扱わない。特に同時に実行されるプロセス数の最大値を  $\max$  とし、0 以上  $\max$  以下の整数の集合を  $I_{\max}$  と記述する。

### 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 BNF 記法に従って定義される。

$$E ::= X \mid 0 \mid a[e_1]!(c)(e_2).E \mid a[e_1]!!(x)(e_2).E$$

$$\mid a[e]?(c)(x).E \mid a[e]?(c)(x).E \mid E + E$$

$$\mid E|E \mid E[S] \mid E \setminus L \mid [[b]E \mid A(\tilde{e})$$

$x$  は値変数、 $b$  は真偽式、 $e, c$  は式である。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<sup>†21</sup>。さらに、式  $b, e, c$  の中の全ての値変数は、それらの出現の左側で、すでに束縛されていなければならない。

$n$  引数をもつ各エージェント定数は、

$$A(\tilde{x}) \stackrel{\text{def}}{=} E$$

の形の定義式をもつと仮定する。ここで、 $E$  は変数  $\tilde{x}(= x_1, \dots, x_n)$  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  $\mathcal{B}(E) \in \mathcal{E}$  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。

<sup>†21</sup>  $\max$  以上の記述も可能であるが、受信者数が  $\max$  を越えた場合はブロードキャストは起らない。

定義 6.2 関数  $B: \mathcal{E}^+ \rightarrow \mathcal{E}$  は次の様に定義される。

$$\begin{aligned} B(X) &= X \\ B(A(e_1, \dots, e_n)) &= A_{e_1, \dots, e_n} \\ B(\mathbf{0}) &= \mathbf{0} \\ B(a[e_1]!(c)(e_2).E) &= a_{e_1, e_2}!^c.B(E) \\ B(a[e_1]!!(x)(e_2).E) &= \sum_{n \in I_{\max}} a_{e_1, e_2}!!^n.B(E\{n/x\}) \\ B(a[e]?(c)(x).E) &= \sum_{v \in V} a_{e, v}?.^c.B(E\{v/x\}) \\ B(a[e]?(c)(x).E) &= \sum_{v \in V} a_{e, v}?.^c.B(E\{v/x\}) \\ B(E + F) &= B(E) + B(F) \\ B(E|F) &= B(E)|B(F) \\ B(E[S]) &= B(E)[S'] \\ B(E \setminus L) &= B(E) \setminus L' \\ B([b]E) &= \begin{cases} B(E) & \text{if } b = \text{true} \\ \mathbf{0} & \text{otherwise} \end{cases} \end{aligned}$$

ここで、全ての値は有限集合  $V$  に含まれていると仮定している。また、

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

である。さらに、エージェント定数の定義式  $A(\tilde{x}) \stackrel{\text{def}}{=} E$  は、定義式の集合

$$\{A_{\tilde{v}} \stackrel{\text{def}}{=} B(E\{\tilde{v}/\tilde{x}\}) : \tilde{v} \in V^n\}$$

に変換される。 ■

## 7 関連研究

既に2節で述べたように、ブロードキャストのためのプロセス代数として CBS [12] が提案されている。ブロードキャストを導入するためには、「状態遷移できない関係 (負の状態遷移)  $\not\rightarrow$ 」の扱いが重要である。CBS では、ディスカードと呼ばれる特殊なイベントによる (正の) 状態遷移が負の状態遷移の代わりに使われている。このディスカードによる状態遷移を使う利点は同期代数 (synchronization algebra) [17] の枠組を適用できることにある。

我々は負の状態遷移の代わりに監視関数  $mon$  を採用している。この監視関数の利点は、等式の条件として監視関数を使えることにある。例として、公理系  $\mathcal{A}$  の等式 T2 の条件に監視関数が使われており、これによって公理系をより簡単に構築することができた。

ディスカードを用いても監視関数を用いてもその結果得られる効果に差はない。CCB と CBS の重要な違い

は受信者数にある。CCB ではイベントをブロードキャストしたエージェントはそのイベントの受信者数を知ることができるが、CBS ではそれは困難である。

他のプロセス代数によるブロードキャストの記述に関する研究として [4] があげられる。この研究では、CBS のエージェントを SCCS のエージェントに変換する方法を示し、CBS と SCCS の関係を明らかにしている。同様にして SCCS による CCB の解釈も考え得る。可能ならば CCB の監視合同が、SCCS でどのように解釈されるかに興味もたれる。

## 8 おわりに

エージェントを柔軟に結合するために、著者の一人は VIABUS を開発してきた。さらに、多様な言語で記述されたエージェントを制御するための共通の通信言語として  $\pi$  計算が VIABUS に実装され、その有効性が認められていた [14]。しかし、 $\pi$  計算は VIABUS のブロードキャストの記述には適していなかった。

この論文では、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、可算ブロードキャストを原子イベントとしてもつプロセス代数 CCB を提案した。さらに、強監視合同という観測的な合同関係を定義し、有限エージェントの強監視合同に対する健全で完全な公理系  $\mathcal{A}$  を与えた。

従来のプロセス代数で CCB の特長を記述することは困難である。その最も重要な原因は時間の概念がプロセス代数で記述できないことにある。そのためにエージェントを数えることができない典型的な例を 2.1 小節に示してきた。我々は、この問題をブロードキャストとカウントを 1 つの原子イベント (可算ブロードキャスト) に凝縮して解決した。

TCCS [11] のように、時間の概念をもつプロセス代数も提案されている。TCCS ではタイムアウトをもつループによってブロードキャストを模倣できると考えられる。すなわち、ブロードキャスト機能をもつ並列言語の開発に TCCS のようなプロセス代数は大変有効であると考えられる。他方、我々はそのような並列言語によって記述されたソフトウェアシステムを可能な限り簡単に解析することを望んでいる。このような目的に対して CCB は適している。

5節で定義した弱監視等価に関しては今後の課題である。また、CCBに時間や空間の概念を導入して拡張することも考えられる。

### 謝辞

本研究の機会を与えて頂いている、太田公廣情報アーキテクチャ部長に感謝いたします。また、命題の証明に関して助言を頂いた言語システム研究室の木下佳樹博士、日頃熱心に御討論頂く言語システム研究室の高橋孝一氏、小方一郎氏、情報ベース研究室の皆様には感謝いたします。

### 参考文献

- [1] Aceto, L., Bloom, B. and Vaandrager, F.: Turning SOS Rules into Equations, Proc. Seventh Annual IEEE Symposium on Logic in Computer Science, pp.113 - 124, 1988.
- [2] Coutaz, J.: 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.
- [3] Hoare, C.A.R.: Communicating Sequential Processes, Prentice-Hall, 1985.
- [4] Holmer, U.: Interpreting Broadcast Communication in SCCS, CONCUR'93, LNCS 715, Springer-Verlag, pp.188 - 201, 1993
- [5] 今井祐二, 結縁祥治, 坂部俊樹, 稲垣康善,: CCS+b: ブロードキャスト型通信機構を追加した CCS, 電子情報通信学会技術研究報告, Vol.91, No.224, COMP91-49, pp.51 - 57, 1991.
- [6] Isobe, Y., Sato, Y., and Ohmaki, K.: A Calculus of Countable Broadcasting Systems, AMAST'95, LNCS 936, Springer-Verlag, pp.498 - 504, 1995.
- [7] 磯部祥尚, 佐藤豊, 大時和仁,: ブロードキャストに適したプロセス代数, 第1回ソフトウェア工学の基礎ワークショップ (FOSE '94) 論文集, pp.33 - 40, 1994.
- [8] Milner, R.: Communication and Concurrency, Prentice-Hall, 1989.
- [9] Milner, R., Parrow, J. and Walker, D.: A Calculus of Mobile Processes, I and II, Information and Computation, 100, pp.1 - 40 and pp.41 - 77, 1992.
- [10] Milner, R.: Calculi for Synchrony and Asynchrony, Journal of Theoretical Computer Science, Vol.25, pp.267 - 310, 1983.
- [11] Moller, F. and Tofts, C.: An overview of TCCS, Proceedings of EUROMICRO'92, Athens, June 1992.
- [12] Prasad, K.V.S.: A Calculus of Broadcasting Systems, TAPSOFT'91, Vol.1:CAAP, LNCS 493, Springer-Verlag, pp.338 - 358, 1991
- [13] Prasad, K.V.S.: A Calculus of Value Broadcasts, PARLE'93, LNCS 694, Springer-Verlag, pp.391 -

402, 1993

- [14] 佐藤豊, 大時和仁,: 分散オブジェクト指向 UIMS の実行時アーキテクチャの設計と実現, 情報処理学会研究報告, PRG15-9, pp.65 - 72, 1994.
- [15] Simone, R.de: Higher-level Synchronizing Devices in Meije-SCCS, Journal of Theoretical Computer Science, Vol.37, pp.245 - 267, 1985.
- [16] Thomsen, B.: A Calculus of Higher Order Communicating Systems, In Proc. 16th ACM Annual Symposium on Principles of Programming Languages, pp.143 - 154, 1989.
- [17] Winskel, G.: Synchronization trees, Journal of Theoretical Computer Science, Vol.34, pp.33 - 82, 1984.
- [18] ISO 8807: Information Processing Systems-Open System Interconnection-LOTOS-A formal description technique based on the temporal ordering of observational behavior, Feb. 1989.

## A 付録

A.1小節では命題 4.1、5.1、5.2、5.3、5.4、定理 5.1 の証明を示す。A.2小節では命題 5.1の証明で使われる定義 A.1と補題 A.1、定理 5.1の証明で使われる定義 A.2と補題 A.2を与える。

### A.1 命題と定理の証明

命題 4.1の証明:

- ( $\Leftarrow$ ) 対偶: 「 $E \xrightarrow{a^n} E'$  ならば、 $a \in \text{mon}(E)$ 」を、 $E$ の構造上に帰納法を用いて証明する。紙面の都合により  $E \equiv E_1|E_2$  の場合のみ示す。  $E_1|E_2 \xrightarrow{a^n} E'$  を導いた最後の推論規則について場合分けをする。
  1. **Para<sub>1</sub>** による場合。このとき、 $E_1 \xrightarrow{a^n} E'_1$  かつ  $a \notin \text{mon}(E_2)$  かつ  $E \equiv E'_1|E_2$  が導かれる。帰納法の仮定より、 $a \in \text{mon}(E_1)$ 。よって、 $a \in \text{mon}(E_1) \cup \text{mon}(E_2) = \text{mon}(E_1|E_2)$  を得る。
  2. **Para<sub>2</sub>** による場合。**Para<sub>1</sub>** の場合に同様である。
  3. **Para<sub>3,4</sub>** による場合。これらの推論規則によって得られる状態遷移上のイベントの属性は !か!!のみであり、 $a^n$  を導くことはない。
  4. **Para<sub>5</sub>** による場合。このとき、 $E_1 \xrightarrow{a^{n_1}} E'_1$  かつ  $E_2 \xrightarrow{a^{n_2}} E'_2$  かつ  $n_1 + n_2 = n$  かつ  $E \equiv E'_1|E'_2$  が導かれる。帰納法の仮定より、 $a \in \text{mon}(E_1)$  かつ  $a \in \text{mon}(E_2)$ 。よって、

$a \in \text{mon}(E_1) \cup \text{mon}(E_2) = \text{mon}(E_1|E_2)$  を得る。

- (⇒) 対偶: 「 $a \in \text{mon}(E)$  ならば、ある  $n$  と  $E'$  で  $E \xrightarrow{a^n} E'$ 」を、 $E$  の構造上に帰納法を用いて証明する。再び、紙面の都合により  $E \equiv E_1|E_2$  の場合のみ示す。 $a \in \text{mon}(E_1|E_2)$  を仮定すると  $\text{mon}$  の定義より  $a \in \text{mon}(E_1) \cup \text{mon}(E_2)$ 。よって、次の3つの場合が考えられる。

1.  $a \in \text{mon}(E_1)$  かつ  $a \notin \text{mon}(E_2)$  の場合。  
帰納法の仮定より、ある  $n$  と  $E'_1$  で  $E_1 \xrightarrow{a^n} E'_1$ 。 $a \notin \text{mon}(E_2)$  より、**Para<sub>1</sub>** を用いて  $E_1|E_2 \xrightarrow{a^n} E'_1|E_2$  を得る。
2.  $a \notin \text{mon}(E_1)$  かつ  $a \in \text{mon}(E_2)$  の場合。上の場合と同様にして **Para<sub>2</sub>** を用いる。
3.  $a \in \text{mon}(E_1)$  かつ  $a \in \text{mon}(E_2)$  の場合。  
帰納法の仮定より、ある  $n_1, n_2$  と  $E'_1, E'_2$  で  $E_1 \xrightarrow{a^{n_1}} E'_1$  かつ  $E_2 \xrightarrow{a^{n_2}} E'_2$ 。**Para<sub>5</sub>** を用いて  $E_1|E_2 \xrightarrow{a^{n_1+n_2}} E'_1|E'_2$  を得る。

命題 5.1 の証明 :

1. 弱監視双模倣をみつける。(証明略)
2. エージェント定数  $\mathbf{T}$  を

$$\mathbf{T} \stackrel{\text{def}}{=} \sum_{a \in \mathcal{N}} a^{?^1} . a^{?^1} . \mathbf{T}$$

と定義し<sup>†22</sup>、 $\mathcal{R}$  を 「 $\mathcal{R} \subseteq \approx$ 」 かつ 「 $(P_1, P_2) \in \mathcal{R}$  ならば、 $(P_1|\mathbf{T}, P_2|\mathbf{T}) \in \mathcal{R}$ 」 を満たす関係と仮定する。

まず、「 $P_1|\mathbf{T} \approx P_2|\mathbf{T}$  かつ  $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$  ならば、任意の  $n \geq 0$  について  $P_1 \approx_{(n)} P_2$  である (\*)」ことを  $n$  の帰納法によって示す。ここで、 $\approx_{(n)}$  は A.2 小節の定義 A.1 に与えられている。

$P_1|\mathbf{T} \approx P_2|\mathbf{T}$  かつ  $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$  とする。

- $n = 0$  の場合。 $\approx_{(0)} \approx$  であるので、  
 $\{(P_1, P_2) : P_1|\mathbf{T} \approx P_2|\mathbf{T}, \mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}\}$  が弱双模倣であることを示す。この証明は A.2 小節の補題 A.1 の証明より簡単である。(証明略)
- $n = k + 1$  の場合。  
 $n = k$  のとき (\*) が成り立つと仮定する。

$P_1 \approx_{(k+1)} P_2$  を証明するために定義 A.1 の4つの条件 (i),  $\dots$ , (iv) を満たすことを示す。

(i) について  $P_1 \xrightarrow{a^{\theta^n}} P'_1$  を仮定する。補題 A.1 より、  
 $P_2 \xrightarrow{a^{\theta^n}} P'_2, P'_1|\mathbf{T} \approx P'_2|\mathbf{T}$

を得る。 $\mathcal{L}(P'_i) \subseteq \mathcal{L}(P_i), (i \in \{1, 2\})$  であるから帰納法の仮定より  $P'_1 \approx_{(k)} P'_2$  である。

(ii), (iii), (iv) についても (i) と同様に示せる。

よって、 $P_1 \approx_{(k+1)} P_2$  を得る。

- 帰納法により (\*) が証明された。

今、 $(P_1, P_2) \in \mathcal{R}$  かつ  $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$  を仮定する。 $\mathcal{R}$  の仮定より  $(P_1|\mathbf{T}, P_2|\mathbf{T}) \in \mathcal{R} \subseteq \approx$  である。従って、上で示した (\*) より任意の  $n \geq 0$  について  $P_1 \approx_{(n)} P_2$  である。ゆえに、不動点理論を用いて

$$(P_1, P_2) \in \approx_m (= \bigcap_{n \geq 0} \approx_{(n)})$$

を得る。 ■

命題 5.2 の証明 :

各々の演算子に対する強監視双模倣を見つめる。例えば並行合成であれば、

$$S = \{(P_1|Q, P_2|Q) : P_1 \simeq_m P_2\}$$

が強監視双模倣、つまり  $P_1|Q \simeq_m P_2|Q$  であることを示した後、定義 5.7 の3つの条件を満たすことを示す。

やや複雑なのは再帰 (エージェント定数) の場合で、

$S = \{(G\{\tilde{A}/\tilde{X}\}, G\{\tilde{B}/\tilde{X}\}) : G \text{ は } \tilde{X} \text{ 以外含まない}\}$  が、次の性質をもつことを証明する。

「もし  $G\{\tilde{A}/\tilde{X}\} \xrightarrow{\alpha} P'$  ならば、ある  $Q'$  で

$G\{\tilde{B}/\tilde{X}\} \xrightarrow{\alpha} Q'$  かつ  $P'S \simeq_m Q'$  である。」

ここで、 $\tilde{A}$  と  $\tilde{B}$  は  $\tilde{A} \stackrel{\text{def}}{=} \tilde{E}\{\tilde{A}/\tilde{X}\}, \tilde{B} \stackrel{\text{def}}{=} \tilde{F}\{\tilde{B}/\tilde{X}\}$  で、 $\tilde{E}$  と  $\tilde{F}$  は  $\tilde{X}$  以外の変数は含まず、 $\tilde{E} \simeq_m \tilde{F}$  である。この証明は文献 [8] の観測合同の合同関係の証明に同様である。ただし、強監視合同の場合はさらに  $\text{mon}(G\{\tilde{A}/\tilde{X}\}) = \text{mon}(G\{\tilde{A}/\tilde{X}\})$  を示さなければならない。これも  $G$  の構造に関して帰納法により証明される。例として並行合成であれば、

$$\begin{aligned} & \text{mon}((G_1|G_2)\{\tilde{A}/\tilde{X}\}) \\ &= \text{mon}(G_1\{\tilde{A}/\tilde{X}\}) \cup \text{mon}(G_2\{\tilde{A}/\tilde{X}\}) \\ &= \text{mon}(G_1\{\tilde{B}/\tilde{X}\}) \cup \text{mon}(G_2\{\tilde{B}/\tilde{X}\}) \text{ (帰納法)} \\ &= \text{mon}((G_1|G_2)\{\tilde{B}/\tilde{X}\}) \end{aligned}$$

となる。 ■

†22  $\mathcal{N}$  は有限集合であるので定義可能。

命題 5.3 の証明 :

( $\Leftarrow$ )  $\cong_m \subset \simeq_m$  より示せる。(証明略)

( $\Rightarrow$ )  $P \simeq_m Q$  を仮定する。

1. ある  $P'$  で  $P \xrightarrow{\tau'} P' \simeq_m Q$  の場合 (\*1)。

$P \cong_m \tau.Q + Q$  を示すために、定義 5.7 の (i),(ii),(iii) の条件を満たすことを証明する。

• (i)  $P \xrightarrow{\alpha} P'$  を仮定する。 $P \simeq_m Q$  より、ある  $Q'$  で  $Q \xrightarrow{\hat{\alpha}} Q'$  かつ  $P' \simeq_m Q'$  である。 $\alpha$  が内部イベントならば、 $Q \Rightarrow Q'$  となるが、 $\tau.Q + Q \xrightarrow{\tau} Q$  より  $\tau.Q + Q \xrightarrow{\tau} Q'$  を得る。 $\alpha$  が内部イベントでないならば、すぐに  $\tau.Q + Q \xrightarrow{\alpha} Q'$  を得る。

• (ii)  $\tau.Q + Q \xrightarrow{\alpha} Q'$  を仮定する。この状態遷移を導いた規則に関して場合分けをする。

(1)  $\tau.Q \xrightarrow{\alpha} Q'$ 、すなわち  $\alpha = \tau$  かつ  $Q' \equiv Q$  の場合。仮定 (\*1) より  $P \xrightarrow{\tau'} P' \simeq_m Q \equiv Q'$  である。

(2)  $Q \xrightarrow{\alpha} Q'$  による場合。 $\alpha$  が内部イベントならば、仮定 (\*1) より  $P \xrightarrow{\tau'} P' \simeq_m Q$  であるから、 $P' \Rightarrow P''$  かつ  $P'' \simeq_m Q'$  を得る。よって、 $P \xrightarrow{\tau'} P''$  である。 $\alpha$  が内部イベントでないならば、 $P \simeq_m Q$  より、すぐに  $P \xrightarrow{\alpha} P''$  かつ  $P'' \simeq_m Q'$  を得る。

• (iii)  $P \simeq_m Q$  ならば  $\text{mon}(P) = \text{mon}(Q)$  であり、 $\text{mon}(\tau.Q) = \emptyset$  であるので明らか。

2. ある  $Q'$  で  $Q \xrightarrow{\tau'} Q' \simeq_m P$  の場合 (\*2)。

$\tau.P + P \cong_m Q$  を示す。(\*1) の場合と同様。

3. 上記以外の場合 (\*3)。  $P \cong_m Q$  を示す。

• (i)  $P \xrightarrow{\alpha} P'$  を仮定する。 $P \simeq_m Q$  より、ある  $Q'$  で  $Q \xrightarrow{\hat{\alpha}} Q'$  かつ  $P' \simeq_m Q'$  を得る。 $\alpha$  が内部イベントならば、 $Q \Rightarrow Q'$  かつ  $P \xrightarrow{\tau'} P'$  であるが、(\*3) の仮定より  $Q \not\equiv Q'$  でなければならないので  $Q \xrightarrow{\tau''} Q'$  を得る。 $\alpha$  が内部イベントでないならば、すぐに  $Q \xrightarrow{\alpha} Q'$  を得る。

• (ii) 上の場合と同様。(iii) 明らか。

命題 5.4 の証明 :

この証明も文献 [8] の観測合同の唯一解の命題の証明と同様である。ただし、命題 5.2 のときと同様に  $\text{mon}(G\{\tilde{P}/\tilde{X}\}) = \text{mon}(G\{\tilde{Q}/\tilde{X}\})$  を示す必要がある。 ■

定理 5.1 の証明 :

( $\Leftarrow$ )  $\mathcal{A}$  の各々の等式について、定義 5.7 の条件を満たすことを証明する。(証明略)

( $\Rightarrow$ )  $P \cong_m Q$  を仮定する。まず、展開規則 E1-3 と補題 A.2 より、 $\mathcal{A} \vdash P = P_s$  かつ  $\mathcal{A} \vdash Q = Q_s$  である完全基本形  $P_s$  と  $Q_s$  が得られる<sup>†23</sup>。ここで、 $\mathcal{A}$  は健全であるので  $P_s \cong_m Q_s$  である。この  $P_s$  と  $Q_s$  の深さ<sup>†24</sup> の和に帰納法を用いる。共に深さが 0 であれば、

$$\mathcal{A} \vdash \mathbf{0} = \mathbf{0}$$

である。それ以外の場合、すなわち  $a\theta^n.P'_s$  が  $P_s$  の選択の成分の 1 つであると仮定する。このとき、

$$P_s \xrightarrow{a\theta^n} P'_s$$

である。イベントの属性に関して次のように場合分けを行なう。

1.  $a\theta^n = \tau$  (すなわち  $\theta = !$  かつ  $n = 0$ ) の場合。

$P_s \cong_m Q_s$  より、ある  $Q'_s$  で

$$Q_s \xrightarrow{\tau'} Q'_s, P'_s \simeq_m Q'_s$$

を得る。さらに  $Q'_s$  も完全基本形であるので、 $\tau'.Q'_s$  は  $Q_s$  の選択の成分である。 $P'_s \simeq_m Q'_s$  と命題 5.3 より、 $P'_s \cong_m Q'_s$  または  $P'_s \cong_m \tau''.Q'_s + Q'_s$  または  $\tau''.P'_s + P'_s \cong_m Q'_s$  である。

(a)  $P'_s \cong_m Q'_s$  の場合。次の場合より簡単。

(b)  $P'_s \cong_m \tau''.Q'_s + Q'_s$  の場合。 $\tau''.Q'_s + Q'_s$  は完全基本形で、その深さは  $Q_s$  以下である。 $P'_s$  の深さは  $P_s$  よりも 1 以上減少している。よって帰納法の仮定より、

$$\mathcal{A} \vdash P'_s = \tau''.Q'_s + Q'_s \quad (*)$$

を得る。ゆえに、T1 と T4 より次を得る。

$$\mathcal{A} \vdash \tau.P'_s = \tau.(\tau''.Q'_s + Q'_s) = \tau.Q'_s = \tau'.Q'_s$$

(c)  $\tau''.P'_s + P'_s \cong_m Q'_s$  の場合。(b) に同様。

†23 CCB の完全基本形は CCS のものと少し異り、A.2 小節の定義 A.2 のように定義される。

†24 エージェント  $P$  の深さ  $\delta(P)$  は次のように定義される。

(i)  $\delta(\mathbf{0}) = 0$

(ii)  $\delta(\sum_{i=1}^m a_i(\theta_i)^{n_i}.P_i) = \max_{(1 \leq i \leq m)} \delta(P_i) + 1$

2.  $\theta = ?$  の場合。

$P_s \cong_m Q_s$  より、ある  $Q'_s$  で

$$Q_s \xrightarrow{a^{\theta^n}} Q'_s, P'_s \simeq_m Q'_s$$

を得る。 $\theta = ?$  であるが、 $a \in \text{mon}(P_s) = \text{mon}(Q_s)$  であるので完全基本形の性質を使うことができる。後は上の場合に同様である。

3. それ以外の場合。上の場合より簡単である。

従って、 $P_s$  の選択の任意の成分は  $A$  によって  $Q_s$  のある選択の成分に等しくなることが示された。逆も同様に示される。ゆえに、

$$A \vdash P = P_s = Q_s = Q$$

が得られる。 ■

## A.2 定義と補題

定義 A.1 関係  $\approx_{(n)}$  ( $n \geq 0$ ) を次のように定義する。

•  $\approx_{(0)} = \approx$

•  $P \approx_{(n+1)} Q$  iff 任意の  $\alpha \in \text{Event}$  と  $a \in \mathcal{N}$  につ

いて、次の 4 つの条件が成り立つ。

(i)  $P \xrightarrow{\alpha} P'$  ならば、ある  $Q'$  が存在して、

$Q \xrightarrow{\hat{\alpha}} Q'$  かつ  $P' \approx_{(n)} Q'$  を満たす。

(ii)  $Q \xrightarrow{\alpha} Q'$  ならば、ある  $P'$  が存在して、

$P \xrightarrow{\hat{\alpha}} P'$  かつ  $P' \approx_{(n)} Q'$  を満たす。

(iii)  $a \notin \text{mon}(P)$  ならば、ある  $Q', Q''$  が存在して、 $Q \Rightarrow Q' \Rightarrow Q''$ ,  $a \notin \text{mon}(Q')$ ,  $P \approx_{(n)} Q''$  を満たす。

(iv)  $a \notin \text{mon}(Q)$  ならば、ある  $P', P''$  が存在して、 $P \Rightarrow P' \Rightarrow P''$ ,  $a \notin \text{mon}(P')$ ,  $P'' \approx_{(n)} Q$  を満たす。 ■

定義 A.2 次の  $P$  は完全基本形である。

(i)  $P \equiv \sum_{i=1}^m a_i (\theta_i)^{n_i} . P_i$  (各  $P_i$  も完全基本形),

(ii)  $P \xrightarrow{a^{\theta^n}} P'$  かつ ( $\theta \neq ?$  or  $a \in \text{mon}(P)$ ) ならば  $P \xrightarrow{a^{\theta^n}} P'$ . ■

補題 A.1 エージェント  $\mathbf{T}$  を

$$\mathbf{T} \stackrel{\text{def}}{=} \sum_{a \in \mathcal{N}} a^{?^1} . a^{?^1} . \mathbf{T}$$

と定義する。このとき、 $P_1 | \mathbf{T} \approx P_2 | \mathbf{T}$  かつ  $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$  ならば、任意の  $\alpha \in \text{Event}$  と  $a \in \mathcal{N}$  につ

いて、次の関係が成り立つ。

(i)  $P_1 \xrightarrow{\alpha} P'_1$  ならば、ある  $P'_2$  が存在して、

$$P_2 \xrightarrow{\hat{\alpha}} P'_2, P'_1 | \mathbf{T} \approx P'_2 | \mathbf{T}.$$

(ii)  $P_2 \xrightarrow{\alpha} P'_2$  ならば、ある  $P'_1$  が存在して、

$$P_1 \xrightarrow{\hat{\alpha}} P'_1, P'_1 | \mathbf{T} \approx P'_2 | \mathbf{T}.$$

(iii)  $a \notin \text{mon}(P_1)$  ならば、ある  $P'_2, P''_2$  が存在して、

$$P_2 \Rightarrow P'_2 \Rightarrow P''_2, a \notin \text{mon}(P'_2), P_1 | \mathbf{T} \approx P''_2 | \mathbf{T}.$$

(iv)  $a \notin \text{mon}(P_2)$  ならば、ある  $P'_1, P''_1$  が存在して、

$$P_1 \Rightarrow P'_1 \Rightarrow P''_1, a \notin \text{mon}(P'_1), P'_1 | \mathbf{T} \approx P''_1 | \mathbf{T}.$$

証明  $P_1 | \mathbf{T} \approx P_2 | \mathbf{T}$  かつ  $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$  とする。

• (i)  $P_1 \xrightarrow{a^{\theta^n}} P'_1$  を仮定する。

まず、 $b \notin \mathcal{L}(P_1) \cup \mathcal{L}(P_2)$  となる  $b$  を選ぶ。

$\mathbf{T}$  の定義より、

$$\mathbf{T} \xrightarrow{b^{?^1}} b^{?^1} . \mathbf{T}$$

であり、 $b \notin \text{mon}(P_1) \subseteq \mathcal{L}(P_1)$  であるから、**Para2** を用いて

$$P_1 | \mathbf{T} \xrightarrow{b^{?^1}} P_1 | (b^{?^1} . \mathbf{T})$$

を導く。ここで、 $P_1 | \mathbf{T} \approx P_2 | \mathbf{T}$  より、ある  $R_1$  で

$$P_2 | \mathbf{T} \xrightarrow{b^{?^1}} R_1, P_1 | (b^{?^1} . \mathbf{T}) \approx R_1$$

を得る。 $P_2$  はイベント  $b^{?^1}$  を起こすことができず、 $\mathbf{T}$  は内部イベントを起こすことができないので、

$$P_2 \Rightarrow P_{21}, R_1 \equiv P_{21} | (b^{?^1} . \mathbf{T}) \quad (*)$$

でなければならない。次に、 $P_1 \xrightarrow{a^{\theta^n}} P'_1$  と  $a \notin \text{mon}(b^{?^1} . \mathbf{T})$  より、**Para1** を用いて

$$P_1 | (b^{?^1} . \mathbf{T}) \xrightarrow{a^{\theta^n}} P'_1 | (b^{?^1} . \mathbf{T})$$

を導く。このとき、 $P_1 | (b^{?^1} . \mathbf{T}) \approx R_1 \equiv P_{21} | (b^{?^1} . \mathbf{T})$  より、ある  $R_2$  で

$$P_{21} | (b^{?^1} . \mathbf{T}) \xrightarrow{a^{\theta^n}} R_2, P'_1 | (b^{?^1} . \mathbf{T}) \approx R_2$$

を得る。 $(b^{?^1} . \mathbf{T})$  はイベント  $a^{\theta^n}$  も内部イベントも起こすことができないので

$$P_{21} \xrightarrow{a^{\theta^n}} P_{22}, R_2 \equiv P_{22} | (b^{?^1} . \mathbf{T})$$

でなければならない。最後に、 $b^{?^1} . \mathbf{T} \xrightarrow{b^{?^1}} \mathbf{T}$  と  $b \notin \text{mon}(P'_1) \subseteq \mathcal{L}(P'_1) \subseteq \mathcal{L}(P_1)$  より、**Para2** を用いて

$$P'_1 | (b^{?^1} . \mathbf{T}) \xrightarrow{b^{?^1}} P'_1 | \mathbf{T}$$

を導く。このとき、 $P'_1 | (b^{?^1} . \mathbf{T}) \approx R_2 \equiv P_{22} | (b^{?^1} . \mathbf{T})$  より、ある  $R_3$  で

$$P_{22} | (b^{?^1} . \mathbf{T}) \xrightarrow{b^{?^1}} R_3, P'_1 | \mathbf{T} \approx R_3$$

を得る。(\*) のときと同様に

$$P_{22} \Rightarrow P'_2, R_3 \equiv P'_2 | \mathbf{T}$$

でなければならない。以上の結果をまとめて次の関係を得る。

$$P_2 \Rightarrow P_{21} \xrightarrow{\widehat{a\theta^n}} P_{22} \Rightarrow P'_2, P'_1 | \mathbf{T} \approx P'_2 | \mathbf{T}$$

- (ii) 上の (i) の場合に同様である。
- (iii)  $a \notin \text{mon}(P_1)$  を仮定する。

$\mathbf{T}$  の定義より、

$$\mathbf{T} \xrightarrow{a^{?1}} \xrightarrow{a^{?1}} \mathbf{T}$$

であり、 $a \notin \text{mon}(P_1)$  であるから、**Para<sub>2</sub>** を用いて

$$P_1 | \mathbf{T} \xrightarrow{a^{?1}} \xrightarrow{a^{?1}} P_1 | \mathbf{T}$$

を導く。ここで、 $P_1 | \mathbf{T} \approx P_2 | \mathbf{T}$  の仮定より、ある  $R'$  で

$$P_2 | \mathbf{T} \xrightarrow{a^{?1}} \xrightarrow{a^{?1}} R', P_1 | \mathbf{T} \approx R'$$

を得る。この中の  $a^{?1}$  による状態遷移は

- (1)  $a \in \text{mon}(\mathbf{T})$  である、(**Para<sub>1</sub>** は使えない)
  - (2) 属性が  $?$  である、(**Para<sub>3,4</sub>** は使えない)
  - (3) 受信者数が 1 である、(**Para<sub>5</sub>** は使えない)
- ことを考慮すると、**Para<sub>2</sub>** によって導かれなければならない。さらに、 $\mathbf{T}$  は内部イベントを起こすことができないことを考慮すると、ある  $P'_2, P''_2$  で次の関係が得られる。

$$P_2 \Rightarrow P'_2 \Rightarrow P''_2, a \notin \text{mon}(P'_2), P_1 | \mathbf{T} \approx P''_2 | \mathbf{T}$$

- (iv) 上の (iii) の場合に同様である。

**補題 A.2** もし  $P \xrightarrow{a\theta^n} P'$  であり、 $\theta \neq ?$  または  $a \in \text{mon}(P)$  ならば、 $A \vdash P = P + a\theta^n.P'$ 。

証明  $P \xrightarrow{a\theta^n} P'$  を仮定して、 $P$  の構造に関する帰納法

を用いる。

1.  $a\theta^n.P'$  が  $P$  の選択の成分の 1 つである場合。

**M3** より明らか。

2.  $a\theta^n.Q$  が  $P$  の選択の成分の 1 つであり、 $Q \xrightarrow{\tau} P'$  の場合。帰納法の仮定により

$$A \vdash Q = Q + \tau.P' \quad (*1)$$

である。ゆえに、次の等式が得られる。

$$\begin{aligned} A \vdash P & \\ &= P + a\theta^n.Q && \text{by M3} \\ &= P + a\theta^n.(Q + \tau.P') && \text{by (*1)} \\ &= P + a\theta^n.(Q + \tau.P') + a\theta^n.P' && \text{by T3} \\ &= P + a\theta^n.P' && \text{by (*1), M3} \end{aligned}$$

3.  $\tau.Q$  が  $P$  の選択の成分の 1 つであり、 $Q \xrightarrow{a\theta^n} P'$  の場合。もし  $\theta \neq ?$  ならば、 $\text{mon}(a\theta^n.P') = \emptyset \subseteq \text{mon}(P)$  である。一方、もし  $\theta = ?$  ならば、仮定より  $a \in \text{mon}(P)$  であるので  $\text{mon}(a\theta^n.P') = \{a\} \subseteq \text{mon}(P)$  である。よって、

$$\text{mon}(a\theta^n.P') \subseteq \text{mon}(P) \quad (*2)$$

が成り立つ。また、帰納法の仮定により

$$A \vdash Q = Q + a\theta^n.P' \quad (*3)$$

である。ゆえに、次の等式が得られる。

$$\begin{aligned} A \vdash P & \\ &= P + \tau.Q && \text{by M3} \\ &= P + \tau.(Q + a\theta^n.P') && \text{by (*3)} \\ &= P + \tau.(Q + a\theta^n.P') + a\theta^n.P' && \text{by (*2), T2} \\ &= P + a\theta^n.P' && \text{by (*3), M3} \end{aligned}$$

すなわち全ての場合で成り立つ。 ■