SURE 静岡大学学術リポジトリ
Shizuoka University REpository

プロセス代数による並行システムの検証と合成に関する研究

メタデータ	言語: ja 出版者: 静岡大学 公開日: 2012-03-29 キーワード (Ja): キーワード (En): 作成者: 磯部, 祥尚 メールアドレス: 所属:
URL	https://doi.org/10.14945/00006532

静岡大学博士論文

プロセス代数による並行システムの検証と合成に関する研究

平成 13 年 6 月

産業技術総合研究所 情報処理研究部門

磯部 祥尚

要旨

情報の分散と処理の高速化に対し，情報を一局で集中的に処理するのではなく，局所的に並行に処理しながら，必要に応じて他局と通信して情報を交換する並行システム (分散システム) が利用されている．非常に多くの情報を操作でき，かつ局所的に高速に処理できる利点がある．しかし，並行システム全体の動作は複数の並行に動作するプロセスの相互作用によって決まるため，その設計は困難である．本論文では，形式的な枠組みであるプロセス代数やプロセス論理を用いて，並行システムの検証や合成に関する理論的側面を明らかにする．

プロセス代数では並行システムや逐次システムの振舞いを式の形で表現し，振舞いの等価性を形式的に判定することができる．また，プロセス論理では振舞いに対する仕様 (要求) を式の形で表現し，プロセス代数で記述された振舞いが仕様を満たしているかを形式的に判定することができる．本論文では，まず 2章で基本的なプロセス代数について説明した後，3章でプロセス代数によるプロダクションルールの振舞いの解析例を示す．次に，4章と 5章で，各々ブロードキャスト通信や減衰を伴う通信をもつ並行プロセスの振舞いを検証する方法を与える．そして，6章と 7章で，柔軟な仕様からより詳細な仕様や分散システムを合成する方法を与える．以下，各章の概要を述べる．

3章 プロダクションルールには，反応すべき状況変化と，そのときに実行すべき動作が記述されており，プロダクションルールの集合をもつアクティブデータベースは状況に応じてルールを呼び出して実行し，データを自動的に更新することができる．このとき，複数のルールが同時に実行されることもあり，その実行順序は必ずしも呼び出された順番とは限らない．そこで，ルールの実行順序が期待通りであるかを事前に検証することが重要となる．3章ではプロセス代数 CCSPR を提案し，CCSPR によってプロダクションルールが容易に記述でき，その振舞いが解析できることを示す．

4章 ブロードキャストの送信者がその受信者数を知るまでの過程を一つの命令で表現する通信方式がある．これを可算ブロードキャストと呼ぶ．受信者を指定しないブロードキャスト通信は拡張性の高いシステムを構築するために有効であり，受信者数を知ることはその後の処理を続けるために重要である．4章では，可算ブロードキャスト通信を一つのアクションで表現できるプロセス代数 CCB を提案し，CCB に適した振舞いの等価性として監視合同を定義する．監視合同は，振舞いが観測的に区別できない二つのプロセスを等しいとみなす等価性である．

5章 メッセージを送信するときにその重要性を指定し，その重要性に応じてその受信範囲が決まる通信方式がある．5章では，メッセージの重要性と距離の概念を導入し，距離と共にその重要性が減衰する通信を表現できるプロセス代数 CCSG を提案する．さらに，観測レベルの概念を導入し，観測レベルを低くすることによって，遠くの重要でない情報を無視した近似的な解析が可能な等価性を定義する．この近似解析によって，情報量の爆発をさけることができ，大規模システムの振舞いの概要を得ることができる．

6章 大規模システムの完全な仕様を一度に記述することは容易ではない．そこで，完全な仕様の代わりに複数の柔軟な (不完全な) 仕様記述を許すことによって設計者の負担を軽減することが考えられる．このとき，複数の仕様の無矛盾性を判定し，それらの共通部分に相当する仕様を合成することが重要である．6章では，柔軟な仕様を記述できるように論理演算子 (特に最小不動点) をプロセス代数に導入して μLOTOS を提案し，複数の柔軟な仕様から詳細な仕様を合成する方法と，その無矛盾性を判定する方法を与える．

7章 分散システムでは複数のプロセスの相互作用を考慮する必要があり，その設計は容易ではない．7章では分散システムを記述するために並行性を明記できる真の並行プロセス代数 *DS*@，その柔軟な仕様を記述するためにプロセス論理 *SP*@ を定義し，停止性は保証されていないが，仕様の充足可能性 (仕様を満たす分散システムが存在するか) を判定し，その仕様を満たす分散システムを合成する方法を与える．さらに *SP*@ で記述された仕様の充足可能性は決定不能であることも示す．すなわち，停止性が保証された *SP*@ の充足可能性を判定するアルゴリズムはない．*DS*@ は非常に簡単な真の並行プロセス代数であり，この結果は他の多くの真の並行プロセス代数にも適用できる。

目 次

第 1 章

序論

情報の分散と処理の高速化に対し，情報を一局で集中的に処理するのではなく，局所的に並行に処理しながら，必要に応じて他局と通信して情報を交換する並行システム (分散システム) が利用されている．非常に多くの情報を操作でき，かつ局所的に高速に処理できる利点がある．しかし，並行システム全体の動作は複数の局所的な処理の相互作用によって決定されるため，その設計は困難である．本研究では，形式的手法を用いて，並行システムの検証や合成に関する理論的側面を明らかにする．

システムの検証や合成には，図 1.1に示す**充足性判定**と**充足可能性判定**が重要である．充足性判定とはシステムと仕様が与えられたとき，システムが仕様を満たしているかを判定することである．開発されたシステムが仕様を満たしていることを確認できれば安全にそのシステムを利用できる．しかし，仕様が複雑になってくると，それを満たすようにシステムを開発すること自体が困難になってくる．また，仕様が矛盾しており，その仕様を満たすシステムが存在しない可能性もある．そこで，その仕様を満たすことができるか，すなわち充足可能性を判定することは，その仕様を満たすシステムを合成するためにも重要である．

充足性や充足可能性を判定するためには振舞いを明確に記述する形式的手法が有効である．本研究ではその形式的手法として**プロセス代数**を用いる．プロセス代数に関する研究は並行システムに数学的意味を与えるために 1970 年半ばから始められ，代表的なプロセス代数として CSP[18]，CCS[38]，ACP[4] 等が知られている．また，1989 年には，OSI(Open System Intetrcommunication) 仕様記述言語の国際標準としてプロセス代数 LOTOS[56] が採用されている．プロセス代数では，プロセスの振舞いを式の

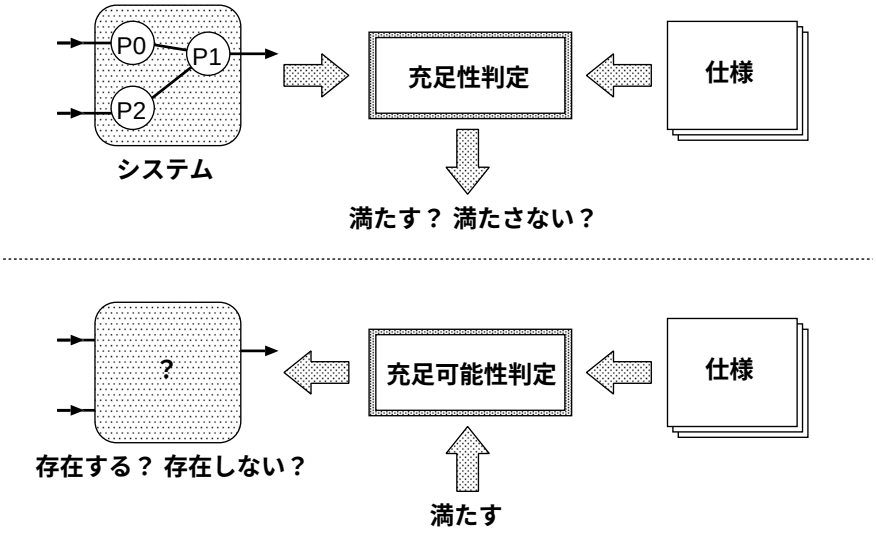

図 1.1: 充足性判定と充足可能性判定

形で記述し，二つのプロセスの振舞いが同じであるかを形式的に検証できる．すなわち，並行システム (並行的記述) と仕様 (逐次的記述) を記述して，並行システムと仕様の振舞いが等価であることを検証できる．

また，プロセス代数で記述されたシステムの振舞いに対する要求を記述するために**プロセス論理** [38, 52] を利用できる．プロセス論理は通常の論理演算子に振舞いを表す様相演算子が加えられた論理であり，Hennessy-Milner 論理とも呼ばれている．仕様をプロセス論理で記述することによって，プロセス代数で記述されたシステムが仕様を充足しているかを検証できる．

上記のように，プロセス代数で記述された並行システムを検証するとき，仕様をプロセス代数で記述する場合とプロセス論理で記述する場合が考えられる．プロセス代数で仕様を記述する場合，システムと仕様の関係に等価性が使われるため，システム全体の振舞いを把握できる．ただし，並行システムの振舞いは複雑であり，その振舞いを完全に正確に記述することは容易ではない．一方，プロセス論理で仕様を記述する場合，システムとの関係に充足性が使われているため，ある部分的な特性を記述し，それを満たしているかを判定することができる．

本論文では，プロセス代数による解析例，検証方法，合成方法について述べる．図 1.2に本論文の構成を示す．まず，呼び出されたプロセスが階層構造 (木構造) をもつアクティブデータベースシステムのプロセス代数による解析例を示す (3章)．次に，ブ

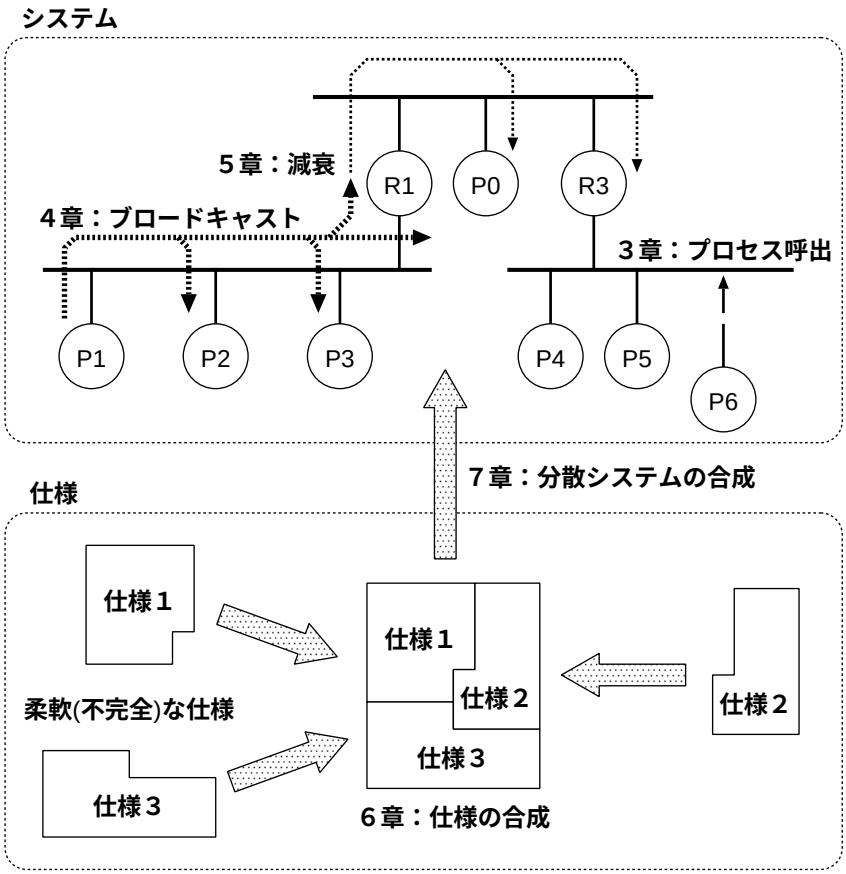

図 1.2: 本論文で対象とするシステムと仕様のイメージ

ロードキャスト通信 (4章) や減衰通信 (5章) をもつシステムを検証可能なプロセス代数を提案し，各々のシステムに最も適した振舞いの等価性や，それを判定するために有効な健全で完全な公理系を与える．そして，複数の柔軟な仕様からより詳細な仕様を合成する方法 (6章) や，柔軟な仕様から分散システムを合成する方法 (7章) を与える．以下，各章の概要を述べる．

2章では，CCS をもとに基本的なプロセス代数について説明する．3章以降で提案されるプロセス代数は CCS をもとにしており，本章で定義される多くの記法は本論文を通して利用される．

3章では，**カップリングモード**をもつアクティブデータベース [16, 37] の**プロダクションルール**の動作を検証するためのプロセス代数として CCSPR を提案する．プロダクションルールとはある状況である動作をするように明記された規則の集合である．つまり，アクティブデータベースでは状況に応じて自動的にデータを更新することが可

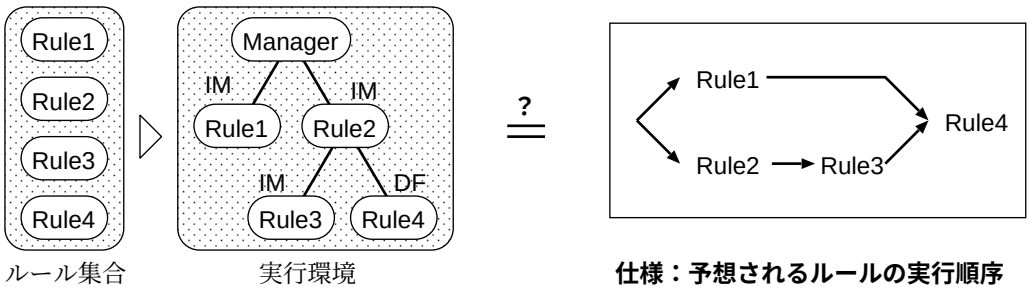

図 1.3: CCSPR によるアクティブデータベースシステムの振舞い検証のイメージ

能である．カップリングモードとはルールの実行順序を制御するためのルール間の結合方法の一つであり，これによってルールは呼び出された順序で実行されるとは限らない．CCSPR は，親プロセスがルール集合から複数のルールを子プロセスとして呼び出してプロセス木を構築するとき，その親子関係を明確に表現できる特徴をもつ．この特徴を利用して，CCSPR によるプロダクションルールの振舞いの検証例を示す．図 1.3にこの振舞い検証のイメージを示す．左図は 4 つのルールをもつルール集合と，それを呼び出して実行する環境から構成されるシステムである．図中，IM や DF は各々「即実行」と「延期実行」を表すカップリングモードであり，これによってルールの実行順序は図 1.3右の仕様のようになると予想される．実際に，CCSPR によって図 1.3 左のシステムは右の仕様のように振舞うことを証明できる．

4章では，VIABUS[50] 上に構築されたシステムの検証に適したプロセス代数として CCB を提案する．VIABUS はマルチエージェントモデルをもとに開発されたプロセスの接続メカニズムであり，システムの拡張性を高めるためにプロセス間の通信方式としてブロードキャストを採用している．ブロードキャストではその受信者数が重要な場合が多く，VIABUS では送信者がメッセージをブロードキャストしてからその受信者数を受け取るまでが一つの命令として与えられている．例えば図 1.4に示すような，P がメッセージ a をブロードキャストし，それを受信したプロセス (Q_1, Q_3, Q_4) の数 3 が返信されるまでの一連の動作を一つの命令で記述できる．このようなブロードキャストを**可算ブロードキャスト**と呼ぶ．CCB は可算ブロードキャストを一つのアクションで表わせるように CCS を拡張したプロセス代数である．

5章では，送信されたメッセージの重要性に応じてその受信範囲が決まる通信方式をもつシステムの解析に適したプロセス代数として CCSG を提案する．このような

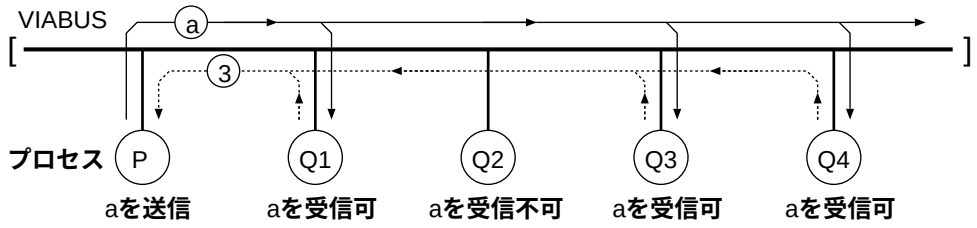

図 1.4: CCB における可算ブロードキャストのイメージ

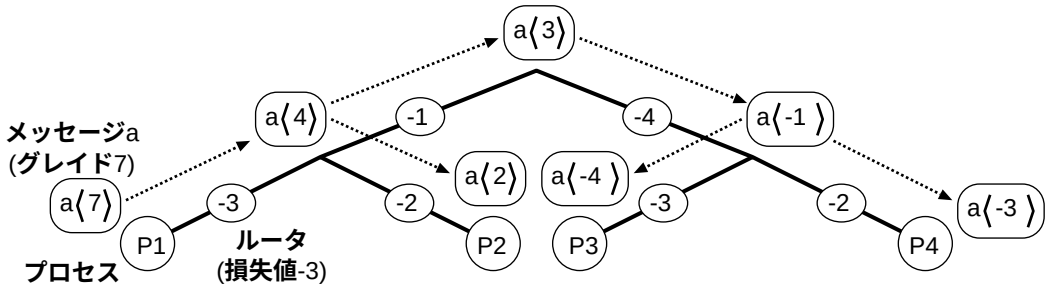

図 1.5: CCSG におけるグレイドと損失のイメージ

通信方式として MBone[57] がある．MBone では，メッセージの受信範囲 (重要度) を TTL(Time To Live) で表し，TTL が 31 以下で組織内，TTL が 127 以下で国内全体，TTL が 128 以上で海外と決められている．CCSG ではメッセージにその重要度を表す実数を付加して送信する．この実数を**送信グレイド** (大きい程遠くまで届く) と呼ぶ．メッセージはルータによって転送されるが，各ルータにはグレイドの損失値が与えられており，ルータを通るごとにメッセージのグレイドは減少する．その例を図 1.5に示す．プロセス P_1 によって送信されたメッセージ a の送信グレイド 7 は，ルータを通過するごとに損失値だけ減少する．プロセス P_4 に届いたこのメッセージのグレイドは -3 になっているが，これを受信できるかは，P_4 のこのメッセージに対する**受信グレイド** (大きい程多くの情報を受け取る) に依存する．この例では，受信グレイド 3 以上で受信できる．CCSG では，観測者の受信グレイド (**観測レベル**と呼ぶ) r における振舞いの等価性として**レベル** $\langle r \rangle$ **等価**が定義されている．この観測レベル r を下げることによって，遠くの重要でない振舞いを無視した近似的な解析が可能である．

6章では，柔軟な仕様の段階的詳細化に適したプロセス代数 μLOTOS を提案する．ここで，柔軟な仕様とは異なる複数のプロセスによって満たされる仕様である [46, 32].

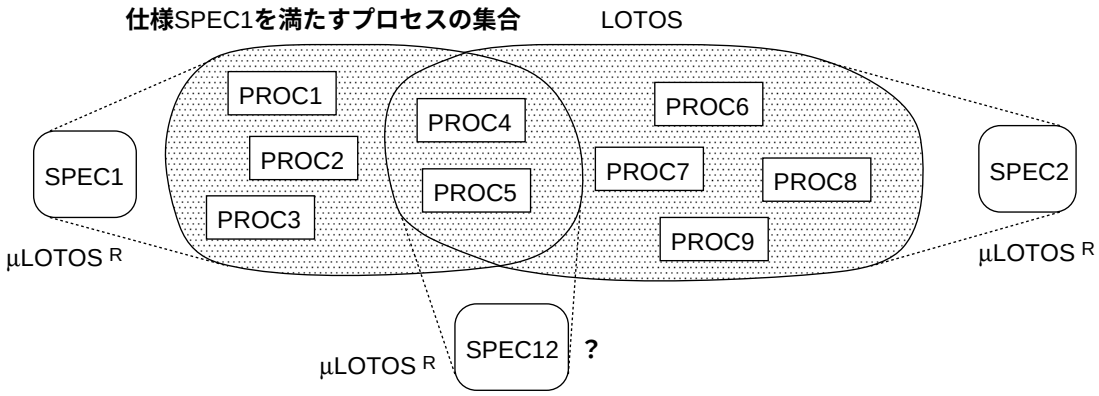

図 1.6: μLOTOS による仕様合成のイメージ

一方，完全な仕様とは，それを満たすプロセスの振舞いが一意に定まる仕様である．大規模システムの設計では，設計者の負担を軽減するために，完全な仕様の代わりに複数の柔軟な仕様が与えられることがある．このとき，与えられた複数の柔軟な仕様の共通部分に相当する仕様を合成し，その仕様を満たすプロセスが存在するか (充足可能性) を判定することが重要である．例えば，図 1.6のように，$SPEC_1$ と $SPEC_2$ を各々プロセス $PROC_{1,\cdots,5}$ と $PROC_{4,\cdots,9}$ によって満たされる柔軟な仕様とすると，その共通のプロセス $PROC_{4,5}$ によって満たされる仕様 $SPEC_{12}$ を合成することが重要である．μLOTOS では，**離接演算子** $\vee$ や**最小不動点**を用いて柔軟に仕様を記述し，複数の柔軟な仕様の共通部分に相当する仕様を**合接演算子** $\wedge$ を用いて記述できる．さらに，仕様の充足可能性を判定する方法と，仕様からそれを満たす逐次プロセスを合成する方法を与える．柔軟な仕様から実行可能なプロセスまで同じ枠組 (μLOTOS) で記述できるため，仕様からプロセスへの変換が容易である特徴をもつ．

7章では，分散システムを記述するために**真の並行プロセス代数** $DS@$，その仕様を柔軟に記述するためにプロセス論理 $SP@$ を定義し，停止性は保証されていないが，積極的に仕様の充足可能性を判定し，それを満たす分散システムを合成するアルゴリズムを与える．プロセス論理では可能性や必然性を表現でき，前章の μLOTOS より柔軟に仕様を記述できる．特に，$SP@$ は単にアクションの実行順序を要求するだけでなく，各アクションをどのプロセスが実行するかを明示的に指定できる特徴をもつ．これによって，逐次動作と並行動作を明確に区別できるようになり (この概念は**真の並行性** [6, 13, 30] と呼ばれる)，分散システムに対する要求を適切に記述できる．例えば，図

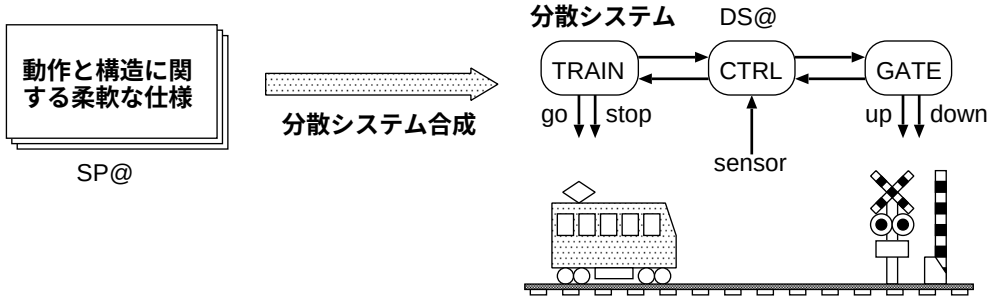

図 1.7: *SP*@ と *DS*@ よる分散システム合成のイメージ

1.7は列車と踏切を制御する分散システムの例である．go, stop は列車を制御するアクションであり，up, down は遮断機を制御するアクションである．また，これらの実行時期は，sensor の情報をもとに制御される．このシステムが安全に動作するためには，アクションの実行順序について様々な制約が課せられる．例えば，遮断機が下りる前に列車は通過してはならない等である．このような振舞いに対する要求は一般のプロセス論理でも記述できるが，振舞いに対する要求からだけでは適切に分散システムを合成することはできない．*SP*@ は，go, stop は列車用のプロセス TRAIN，up, down は踏切用のプロセス GATE で実行される等の構造に関する要求も記述でき，分散システムに対する仕様を適切に記述できる特徴をもつ．

8章では，本研究で得られた成果をまとめ，この成果から発展可能な今後の研究について述べる．本研究の重要な成果として，7章のプロセス論理 *SP*@ で記述された仕様の充足可能性は決定不能であることを証明したことがあげられる．すなわち，充足可能性を判定する停止性が保証されたアルゴリズムは存在しない．この結果は仕様が有限状態をもち，離接演算子をもたず，通信要求をもたない場合でも成り立つ．*DS*@ は非常に簡単な真の並行プロセス代数であり，この成果は他の多くの真の並行プロセス代数に対しても適用できる．

第 2 章

プロセス代数の基礎

2.1 はじめに

並行システムを検証する数学的道具として**プロセス代数**が知られている．基本的なプロセス代数である CCS[38]，CSP[18]，ACP[4] をはじめ，通信チャンネルを動的に変えられるプロセス代数 [39]，時間の概念をもつプロセス代数 [41]，空間の概念をもつプロセス代数 [6]，確率的な振舞いを記述できるプロセス代数 [11] など数多くのプロセス代数が提案されている．また，近年ではセキュリティの検証にプロセス代数を利用する研究も行われている [1, 15, 49]．本章では Milner によって提案された代表的なプロセス代数である CCS[38] (a Calculus of Communicating Systems) を紹介する．本章で与えられる命題は全て文献 [38] に与えられており，ここではその証明を省略する．3 章以降で提案されるプロセス代数は CCS をもとにしており，本章で定義される多くの記法は本論文を通して利用される．

本章の構成は次の通りである．まず，2.2節で CCS によるシステムの検証をバッファの例をもとに説明する．2.3節では CCS の構文と意味の定義を示す．2.4節では CCS の代表的な等価性を紹介する．2.5節では CCS のためのプロセス論理について述べる．

2.2 検証例

プロセス代数 CCS では，振舞いの最小単位は**アクション**と呼ばれ，プロセスへの入出力やプロセス間の通信がアクションに相当する．**プロセス**はアクションの実行順序

を記述した式で表される．例えば，アクション in の後にアクション $\overline{out}$ を実行して停止するプロセスは次のように記述される．

$$in.\overline{out}.\mathbf{0}$$

ここで，ピリオド '.' は**逐次演算子**であり，アクションを順番に逐次的に実行することを表している[1]．また，アクションのオーバーラインは一般に出力を表し，アクション in はチャンネル in からの入力動作，アクション $\overline{out}$ はチャンネル out への出力動作を表す．$\mathbf{0}$ は**無動作プロセス**である．尚，簡単のため，受け渡すメッセージの記述は省略する．

無限に in と $\overline{out}$ を交互に繰り返すプロセス $(in.\overline{out}.in.\overline{out}.in.\overline{out}.\cdots)$ を有限に記述するために，次のように**定数**を利用できる．

$$BUFF_1 \stackrel{\text{def}}{=} in.\overline{out}.BUFF_1$$

ここで，$BUFF_1$ は定数であり，その振舞いは定義式 (定数 $\stackrel{\text{def}}{=}$ プロセス) によって与えられる．右辺は定数を含むことができ，定数 $BUFF_1$ は in と $\overline{out}$ を実行した後，再び $BUFF_1$ のように振舞うように定義されている．この $BUFF_1$ は入力 (in) された情報を一つ蓄えてから出力 $(\overline{out})$ するバッファの振舞いを表している．

プロセスの振舞いはラベル付遷移システムによって与えられ，$BUFF_1$ は次のように状態遷移する．

$$BUFF_1 \xrightarrow{in} \overline{out}.BUFF_1 \xrightarrow{\overline{out}} BUFF_1$$

ここで，$\overline{out}.BUFF_1$ は $BUFF_1$ が in を実行した後の一つの状態であるが，$\overline{out}$ を実行後に $BUFF_1$ として振舞うプロセスでもある．プロセス代数では「状態」と「プロセス」を特に区別しない．

上記の $BUFF_1$ を並行に二つ合成して $BUFF_2$ を次のように定義する．

$$BUFF_2 \equiv (BUFF_1[com/out] \mid BUFF_1[com/in])\backslash\{com\}$$

ここで，$\mid$ は**並行合成演算子**，$[com/out]$ はアクションの名前を out から com に変更する**リラベリング演算子**，$\backslash\{com\}$ は外部からアクション $com, \overline{com}$ を実行することを禁止する**制限演算子**である．また，$\equiv$ は左辺と右辺が構文的に同じであることを表し，上記の例では単に右辺を $BUFF_2$ と略記するために使われている．$BUFF_2$ の構造を図 2.1に

[1]一般に '.' はプレフィックス (prefix) と呼ばれるが，ここでは逐次演算子とよぶ．

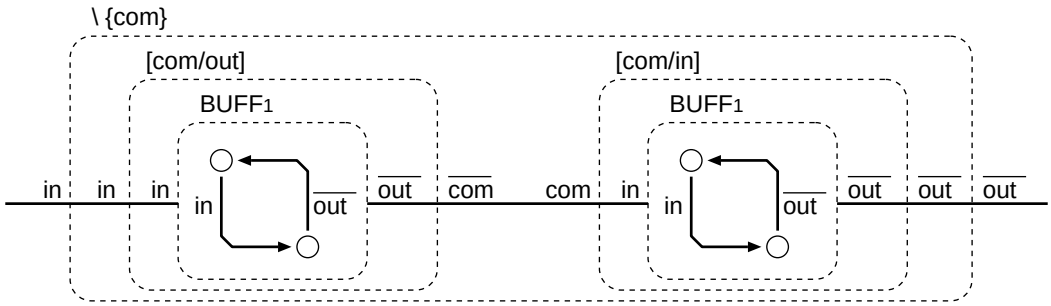

図 2.1: $BUFF_2 \equiv (BUFF_1[com/out] \mid BUFF_1[com/in])\backslash\{com\}$ の構造

示す．CCS では同期式の通信を採用しており，$(com, \overline{com})$ のように相補的なアクションは同期して通信することができる．$BUFF_2$ の例については，

$$(\overline{out}.BUFF_1)[com/out] \xrightarrow{\overline{com}} BUFF_1[com/out]$$
$$BUFF_1[com/in] \xrightarrow{com} (\overline{out}.BUFF_1)[com/in]$$

であるので，次のような同期通信が可能である．

$$((\overline{out}.BUFF_1)[com/out] \mid BUFF_1[com/in])\backslash\{com\}$$
$$\xrightarrow{\tau} (BUFF_1[com/out] \mid (\overline{out}.BUFF_1)[com/in])\backslash\{com\}$$

ここで，τ は内部で自動的に起こる制御できないアクションであり，**内部アクション**と呼ばれる．

次に容量 2 のバッファの仕様を考える．容量が 2 であるので，入力を 1 回実行した後，それをすぐに出力することもできるが，さらに続けてもう 1 回入力することもできる．2 回続けて入力した後は出力無しにはさらなる入力はできない．この振舞いは次のように記述できる．

$$SPEC_{20} \stackrel{\text{def}}{=} in.SPEC_{21}$$
$$SPEC_{21} \stackrel{\text{def}}{=} in.SPEC_{22} + \overline{out}.SPEC_{20}$$
$$SPEC_{22} \stackrel{\text{def}}{=} \overline{out}.SPEC_{21}$$

ここで，$+$ は**選択演算子**であり，アクションによってその後の振舞いが選択される．すなわち，$SPEC_{21}$ では，in の後は $SPEC_{22}$ のように振舞い，$\overline{out}$ の後は $SPEC_{20}$ のように振舞うことを表している．$SPEC_{2i}$ の i は今バッファが蓄えている情報量に相当する．

このとき，$BUFF_2$ と $SPEC_{20}$ の振舞いは観測的に等価であることを証明できる．

$$BUFF_2 \approx SPEC_{20}$$

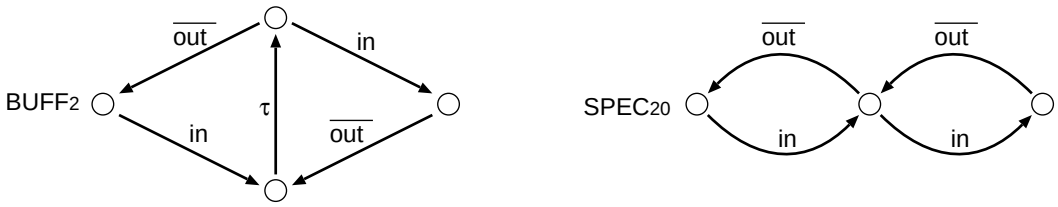

図 2.2: $BUFF_2$ と $SPEC_{20}$ の状態遷移図

ここで，≈は**観測等価**と呼ばれる同値関係である．図 2.2に $BUFF_2$ と $SPEC_{20}$ の状態遷移図を示す．ここで注目すべきは，$BUFF_2$ と $SPEC_{20}$ の状態遷移図は同じではないことである．観測等価では内部の振舞い τ を可能な限り (全てではない) 無視して等価性を判定するように定義されている．この観測等価はツール (Concurrency Workbench)[12] を用いて自動的に判定することもできる．

この例では，$BUFF_2$ を実際に実装される並行システム，$SPEC_{20}$ をその並行システムに対する仕様であるとみなすことができる．すなわち，上記の等式が成り立つことによって，$BUFF_2$ は容量 2 のバッファとして振舞うことが証明できたことになる．

CCS で記述されたシステムの仕様を記述するために**プロセス論理** [38, 52] を利用することもできる．プロセス論理は様相演算子をもつ論理であり，様相演算子には**可能性様相演算子** $\langle\langle\,\rangle\rangle$ や**必然性様相演算子** $[[\,]]$ がある．ここで，仕様 $\langle\langle a\rangle\rangle S$ は，(0 回以上の内部アクションを実行後に) アクション a を実行でき，その後は (0 回以上の内部アクションを実行後に) 仕様 S を満たすことができることを要求する．一方，仕様 $[[a]]S$ は，もし (0 回以上の内部アクションを実行後に) アクション a を実行できるならば，その後は (0 回以上の内部アクションを実行後の全ての状態が) 必ず仕様 S を満たすことを要求する．例えば，次の仕様 $LSPEC_1$ は，アクション in を実行した後は，必ずアクション $\overline{out}$ を実行できることを要求している．

$$LSPEC_1 \equiv [[in]]\langle\langle\overline{out}\rangle\rangle\mathtt{tt}$$

ここで，tt は全てのプロセスによって満たされる仕様である．また，次の仕様 $LSPEC_2$ は，アクション $\overline{out}$ を実行した後は，必ず ff を満たすことを要求している．

$$LSPEC_2 \equiv [[\overline{out}]]\mathtt{ff}$$

ここで，ff は満たすことができない仕様である．すなわち，$LSPEC_2$ は $\overline{out}$ の実行禁止を要求している．

表 2.1: 集合と要素上の変数

集合の記法	集合の名前	要素上変数
$\mathcal{N}$	名前の集合	$a, b, \cdots$
$\overline{\mathcal{N}} = \{\overline{a} : a \in \mathcal{N}\}$	余名の集合	$\overline{a}, \overline{b}, \cdots$
$\mathcal{L} = \mathcal{N} \cup \overline{\mathcal{N}}$	ラベルの集合	$l, l', \cdots$
$Act = \mathcal{L} \cup \{\tau\}$	アクションの集合	$\alpha, \beta, \cdots$

このとき，上記のシステム $BUFF_1$ と $BUFF_2$ は仕様 $LSPEC_1$ と $LSPEC_2$ の両方を満たすことを証明できる．

$$BUFF_1 \models LSPEC_1 \wedge LSPEC_2$$
$$BUFF_2 \models LSPEC_1 \wedge LSPEC_2$$

ここで，$\models$ は左辺のプロセス代数式が右辺のプロセス論理式を満たすことを表す．$LSPEC_2$ は $\overline{out}$ の実行禁止を要求しているが，それはプロセスが何か (観測可能な) 別のアクションを実行するまでであり，$BUFF_i$ のように in を実行後に $\overline{out}$ を実行することは可能である．

このように，プロセス論理で仕様を記述し，システムがその仕様を満たしているかを検証できる．プロセス代数で記述した上記の仕様の例 $SPEC_{20}$ と比較すると，$SPEC_{20}$ の方が $BUFF_2$ の全ての振舞いを知るためには有効であるが，大規模システムでは全ての振舞いの仕様を正確に記述することは難しい．プロセス論理では要求を部分的に記述して検証できる利点がある．

2.3 構文と意味

まず，**名前**の無限集合 $\mathcal{N}$ が与えられていると仮定する．このとき，表 2.1のように各集合を与える[2]．ここで，$\bar{\cdot}$ は名前と**余名**の間の全単射であり，$\overline{\overline{a}} = a$ である．τ は**内部アクション**と呼ばれる観測されない特殊なアクションを表しており，$\mathcal{N}$ には含まれていないとする．

さらに，**プロセス定数** ($A, \cdots$ で表す) の集合 $\mathcal{K}_{ccs}$ とプロセス変数 ($X, \cdots$ で表す)

[2]注：$L \subseteq \mathcal{L}$ とするとき，集合 $\overline{L}$ は L の補集合ではなく，$\overline{L} = \{\overline{l} : l \in L\}$ である

の集合 $\mathcal{X}_{ccs}$ が与えられていると仮定する．また，$\mathcal{L}$ から $\mathcal{L}$ への関数 $f : \mathcal{L} \to \mathcal{L}$ で，$f(\bar{l}) = \overline{f(l)}$ を満たす関数を**リラベリング関数**と呼ぶ．便宜上，$f(\tau) = \tau$ とし，f をアクションの集合上に拡張する．

このとき，CCS の構文は次のように与えられる．

定義 2.3.1 プロセス式 ($E, F, \cdots$ で表す) の集合 $\mathcal{E}_{ccs}$ は次の式を含む最小の集合である

$$
\begin{aligned}
X &: \text{プロセス変数 } (X \in \mathcal{X}_{ccs}) \\
A &: \text{プロセス定数 } (A \in \mathcal{K}_{ccs}) \\
\alpha.E &: \text{逐次 } (\alpha \in Act) \\
\textstyle\sum_{i\in I} E_i &: \text{選択 } (I \text{ はインデックス集合 }) \\
E|F &: \text{並行合成} \\
E[f] &: \text{リラベリング } (f \text{ はリラベリング関数 }) \\
E\backslash L &: \text{制限 } (L \subseteq \mathcal{L})
\end{aligned}
$$

ここで，E, E_i, F はすでに $\mathcal{E}_{ccs}$ の要素であるとする． ∎

プロセス変数を含まないプロセス式を**プロセス** ($P, Q, R, \cdots$ で表す) と呼び，プロセスの集合を $\mathcal{P}_{ccs}$ で表す．プロセス定数は**定義式**によって意味を与えられるプロセスである．実際に全てのプロセス定数 $A \in \mathcal{K}_{ccs}$ について，$A \stackrel{\text{def}}{=} P$ の形の定義式があると仮定する．ここで，$P \in \mathcal{P}_{ccs}$ であり，P は定数 A を含むことができるので，繰り返し動作を表現できる．また，$E \equiv F$ は E と F が構文的に同じであることを表す．

選択演算子において $I = \{1, 2\}$ の場合は，$\sum_{i\in\{1,2\}} E_i$ の代わりに $E_1 + E_2$ の記述を用いる．また，無動作プロセス $\mathbf{0}$ は，$\mathbf{0} \stackrel{\text{def}}{=} \sum_{i\in\emptyset} E_i$ によって定義されるプロセス定数として与えられる．演算子の結合の優先順位は { 制限 , リラベリング } > 逐次 > 並行合成 > 選択，である．

次に CCS のプロセス式に意味を与える．

定義 2.3.2 プロセス式の意味は次のラベル付遷移システムにより与えられる．

$$(\mathcal{E}_{ccs}, Act, \longrightarrow)$$

ここで，ラベル付遷移の集合 $\longrightarrow \subseteq \mathcal{E}_{ccs} \times Act \times \mathcal{E}_{ccs}$ は図 2.3の推論規則を満たす最小の集合である．ここで，$(E, \alpha, E') \in \longrightarrow$ を $E \xrightarrow{\alpha} E'$ と書く．また，ある $(E_1, \cdots, E_{n-1})$ について $E_0 \xrightarrow{\alpha_1} E_1 \xrightarrow{\alpha_2} \cdots \xrightarrow{\alpha_{n-1}} E_{n-1} \xrightarrow{\alpha_n} E_n$ であるとき，$E_0 \xrightarrow{\alpha_1}\xrightarrow{\alpha_2} \cdots \xrightarrow{\alpha_{n-1}}\xrightarrow{\alpha_n} E_n$ と書く． ∎

名前	仮定	⊢	結果
Act		⊢	$\alpha.E \xrightarrow{\alpha} E$
$\mathbf{Choice}_j$	$E_j \xrightarrow{\alpha} E',\ j \in I$	⊢	$\sum_{i\in I} E_i \xrightarrow{\alpha} E'$
$\mathbf{Com}_1$	$E \xrightarrow{\alpha} E'$	⊢	$E\|F \xrightarrow{\alpha} E'\|F$
$\mathbf{Com}_2$	$F \xrightarrow{\alpha} F'$	⊢	$E\|F \xrightarrow{\alpha} E\|F'$
$\mathbf{Com}_3$	$E \xrightarrow{l} E',\ F \xrightarrow{\overline{l}} F'$	⊢	$E\|F \xrightarrow{\tau} E'\|F'$
Rel	$E \xrightarrow{\alpha} E'$	⊢	$E[f] \xrightarrow{f(\alpha)} E'[f]$
Res	$E \xrightarrow{\alpha} E',\ \alpha \notin L \cup \overline{L}$	⊢	$E\backslash L \xrightarrow{\alpha} E'\backslash L$
Rec	$P \xrightarrow{\alpha} P',\ A \stackrel{\text{def}}{=} P$	⊢	$A \xrightarrow{\alpha} P'$

図 2.3: プロセス式上のラベル付遷移の推論規則

規則 $\mathbf{Choice}_j$ は複数のプロセスの中から，アクションによって 1 つのプロセスを選択することを表している．次の状態遷移は規則 $\mathbf{Choice}_3$ による選択の例である．

$$a.P_1 + b.P_2 + c.P_3 + d.P_4 \xrightarrow{c} P_3$$

規則 $\mathbf{Com}_{1,2}$ は 2 つのプロセス E と F が並行に独立に動作できることを表し，規則 $\mathbf{Com}_3$ は相補的なアクション $(l, \overline{l})$ による E と F の間の通信を表している．通信後、$(l, \overline{l})$ は 1 つの内部アクション τ に置き換えられるため，さらに他のプロセスと通信することはできない．つまり，CCS では送信と受信が同期する 1 対 1 通信を採用しており，次のように $\overline{a}.P_1$ が $a.P_2$ と $a.P_3$ の両方と通信可能な場合は，非決定的にどちらか一方と通信する．

$$(\overline{a}.P_1|a.P_2|a.P_3)\backslash\{a\} \xrightarrow{\tau} (P_1|P_2|a.P_3)\backslash\{a\}$$
$$(\overline{a}.P_1|a.P_2|a.P_3)\backslash\{a\} \xrightarrow{\tau} (P_1|a.P_2|P_3)\backslash\{a\}$$

一般に CCS では送信側のアクションにオーバーラインを付ける．また，上の例では a は制限されているため，規則 **Res** により a を用いた外部との通信は起こらない．

規則 **Rel** は，アクションの名前を変えるときに使われる．a を b に変えるリラベリング関数は (b/a) のように記述され，次のように使われる．

$$(a.P)[b/a] \xrightarrow{b} P[b/a]$$

繰り返し動作はプロセス定数によって記述される．例えば，無限にアクション a を実行でき，アクション b を実行すると停止するプロセス A は次のように定義される．

$$A \stackrel{\text{def}}{=} a.A + b.\mathbf{0}$$

規則 **Rec** によって，プロセス定数 A はプロセス $a.A + b.\mathbf{0}$ のように振舞う．

2.4 等価性

プロセス代数ではプロセスの等価性が与えられており，システムの振舞いの検証に利用できる．本節では**双模倣**の概念に基づいた等価性として，**強等価**，**観測等価**，**観測合同**を紹介する．強等価は定義が簡単な合同関係であり，双模倣の基本的な性質を調べることができる．観測等価は内部の振舞い (内部アクション) を可能な限り無視することができる同値関係であり，強等価よりも弱い関係である．ただし，観測等価は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．観測合同は観測等価に含まれる最大の合同関係である．

2.4.1 強等価

2 つのプロセスが双模倣とは，一方のプロセスがあるアクションを実行できるならば，他方のプロセスもそのアクションを実行でき，アクションを実行後の各々のプロセスも双模倣になっていることである．まず，内部アクションの個数まで同じであることが要求される強い双模倣の定義を与える．

定義 2.4.1 プロセス上の二項関係 $\mathcal{S}$ が**強双模倣** (strong bisimulation) であるとは，$(P, Q) \in \mathcal{S}$ ならば，任意の $\alpha \in Act$ について，次の 2 つの条件が成り立つことである．

(i) $P \stackrel{\alpha}{\longrightarrow} P'$ ならば，ある Q' について，$Q \stackrel{\alpha}{\longrightarrow} Q'$ かつ $(P', Q') \in \mathcal{S}$ を満たす．

(ii) $Q \stackrel{\alpha}{\longrightarrow} Q'$ ならば，ある P' について，$P \stackrel{\alpha}{\longrightarrow} P'$ かつ $(P', Q') \in \mathcal{S}$ を満たす．

∎

例えば，プロセス $SSPEC_{20}$ を次のように定義する．

$$SSPEC_{20} \stackrel{\text{def}}{=} in.\tau.SSPEC_{21}$$
$$SSPEC_{21} \stackrel{\text{def}}{=} in.SSPEC_{22} + \overline{out}.SSPEC_{20}$$
$$SSPEC_{22} \stackrel{\text{def}}{=} \overline{out}.\tau.SSPEC_{21}$$

$SSPEC_{20}$ はプロセスであるが，(通信も考慮した) 容量 2 のバッファの仕様とみなすこともできる．2.2節で紹介した $SPEC_{20}$ との違いは，内部アクション τ が挿入されていることである．このとき，次の $\mathcal{S}^{strong}_{BUFF}$ は強双模倣である．

$$\begin{aligned}\mathcal{S}^{strong}_{BUFF} = \{\, &(BUFF_2,\ SSPEC_{20}), (BUFF'_2,\ \tau.SSPEC_{21}),\\ &(BUFF_{21},\ SSPEC_{21}), (BUFF_{22},\ SSPEC_{22})\,\}\end{aligned}$$

ここで，$BUFF_1$ と $BUFF_2$ は 2.2節で紹介したバッファの記述であり，$BUFF'_2$，$BUFF_{21}$，$BUFF_{22}$ は次のように与えられる．

$$\begin{aligned}BUFF'_2 &\equiv ((\overline{out}.BUFF_1)[com/out] \mid BUFF_1[com/in])\backslash\{com\}\\ BUFF_{21} &\equiv (BUFF_1[com/out] \mid (\overline{out}.BUFF_1)[com/in])\backslash\{com\}\\ BUFF_{22} &\equiv ((\overline{out}.BUFF_1)[com/out] \mid (\overline{out}.BUFF_1)[com/in])\backslash\{com\}\end{aligned}$$

$BUFF'_2$ は内部アクションを実行して $BUFF_{21}$ になるプロセスである．強等価では内部アクションの個数も一致する必要があるため，$\mathcal{S}^{strong}_{BUFF}$ に含まれる $(BUFF'_2, \tau.SSPEC_{21})$ の τ を無視することはできない．

この強双模倣をもとに，強等価は最大の強双模倣として与えられる．

定義 2.4.2 もしある強双模倣 $\mathcal{S}$ において $(P,Q) \in \mathcal{S}$ ならば，プロセス P と Q は**強等価** (strong equivalence) であるといい，$P \sim Q$ と書く． ∎

このとき，期待通り次の命題が成り立つ．

命題 2.4.1 $P \sim Q$ ならば，そのときに限り任意の $\alpha \in Act$ について，次の 2 つの条件が成り立つ．

(*i*) $P \xrightarrow{\alpha} P'$ ならば，ある Q' について，$Q \xrightarrow{\alpha} Q'$ かつ $P' \sim Q'$ を満たす．
(*ii*) $Q \xrightarrow{\alpha} Q'$ ならば，ある P' について，$P \xrightarrow{\alpha} P'$ かつ $P' \sim Q'$ を満たす．

∎

命題 2.4.2 強等価 $\sim$ は同値関係である．すなわち，反射律 $(P \sim P)$，対称律 $(P \sim Q$ ならば $Q \sim P)$，推移律 $(P \sim Q$ かつ $Q \sim R$ ならば $P \sim R)$ が成り立つ． ∎

上記の例 $SSPEC_{20}$ では，$(BUFF_2, SSPEC_{20}) \in \mathcal{S}^{strong}_{BUFF}$ であるので，次の等式を得る．

$$BUFF_2 \sim SSPEC_{20}$$

このように，$P \sim Q$ を証明するには，(P,Q) を含む強双模倣を見つければよい．しかし，強双模倣では**強等価な組**を複数含む場合があり，予想以上に多くの組を含むことがある．ここで，$P \sim P'$ かつ $Q \sim Q'$ であるとき，(P,Q) と (P',Q') は強等価な組という．そこで定義されるのが次の “強等価な組を除く強双模倣” である．

定義 2.4.3 プロセス上の二項関係 $\mathcal{S}$ が**強等価な組を除く強双模倣** (strong bisimulation up to $\sim$) であるとは，$P\mathcal{S}Q$ ならば，任意の $\alpha \in Act$ について，次の 2 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' について，$Q \xrightarrow{\alpha} Q'$ かつ $P' \sim \mathcal{S} \sim Q'$ を満たす．
(ii) $Q \xrightarrow{\alpha} Q'$ ならば，ある P' について，$P \xrightarrow{\alpha} P'$ かつ $P' \sim \mathcal{S} \sim Q'$ を満たす．

ここで，$P\mathcal{S}Q$ は $(P,Q) \in \mathcal{S}$ を表し，$P' \sim \mathcal{S} \sim Q'$ は，$P' \sim P''$ かつ $P''\mathcal{S}Q''$ かつ $Q'' \sim Q'$ となる P'' と Q'' が存在することを表している． ∎

このとき，次の命題が成り立つ．

命題 2.4.3 もし $\mathcal{S}$ が強等価な組を除く強双模倣であるならば，$\mathcal{S} \subseteq \sim$ である． ∎

すなわち，$P \sim Q$ を証明するためには，$(P,Q) \in \mathcal{S}$ のような強等価な組を除く強双模倣 $\mathcal{S}$ をみつければ十分である．

強等価は内部アクションの個数も一致することを要求する強い関係であるが，観測等価より定義が簡単であり，強等価は観測等価に含まれるので，いくつかの等式は強等価をもとに与えられる．次の命題は逐次的な振舞いと並行的な振舞いを関係付けるために重要であり，展開規則と呼ばれる．

命題 2.4.4 $P \equiv (P_1[f_1]|\cdots|P_n[f_n])\backslash L$ とする．このとき，次の等式が成り立つ．

$$\begin{aligned} P \sim \quad & \sum\{f_i(\alpha).(P_1[f_1]|\cdots|P_i'[f_i]|\cdots|P_n[f_n])\backslash L : P_i \xrightarrow{\alpha} P_i',\ f_i(\alpha) \notin L \cup \overline{L}\} \\ & + \sum\{\tau.(P_1[f_1]|\cdots|P_i'[f_i]|\cdots|P_j'[f_j]|\cdots|P_n[f_n])\backslash L : \\ & \qquad P_i \xrightarrow{l} P_i',\ P_j \xrightarrow{l'} P_j',\ f_i(l) = \overline{f_j(l')},\ i < j\} \end{aligned}$$

∎

例えば，この展開規則を適用して次の等式が得られる．

$$(a.P|\overline{a}.Q) \sim a.(P|\overline{a}.Q) + \overline{a}.(a.P|Q) + \tau.(P|Q)$$

右辺第 1 項と第 2 項は各々PとQが独立にアクションを起こす場合に相当し，第 3 項は P と Q が同期通信する場合に相当する．

ここまでは，(変数を含まない) プロセスに対して強等価 $\sim$ を与えてきた．ここで，強等価を次のようにプロセス式上に拡張する．

定義 2.4.4 E と F は高々変数 X_i $(i \in I)$ を含むとする．このとき，任意のプロセス P_i $(i \in I)$ について，$E\{P_i/X_i\}_{i\in I} \sim F\{P_i/X_i\}_{i\in I}$ ならば，$E \sim F$ である．ここで，$E\{P_i/X_i\}_{i\in I}$ は，各 $i \in I$ について，E に含まれる全ての変数 X_i へプロセス P_i を代入して得られるプロセスである． ∎

例えば，プロセス式 E_{par} と E_{seq} を次のように定義する．

$$E_{par} \equiv (a.X|\overline{a}.\mathbf{0})$$
$$E_{seq} \equiv a.(X|\overline{a}.\mathbf{0}) + \overline{a}.(a.X|\mathbf{0}) + \tau.(X|\mathbf{0})$$

このとき，次の $\mathcal{S}$ は強双模倣であることを証明できる．

$$\mathcal{S} = \{(E_{par}\{P/X\}, E_{seq}\{P/X\}) : P \in \mathcal{P}_{ccs}\} \cup \{(P, P) : P \in \mathcal{P}_{ccs}\}$$

すなわち，任意のプロセス P について，$E_{par}\{P/X\} \sim E_{seq}\{P/X\}$ であるので，定義 2.4.4より，$E_{par} \sim E_{seq}$ である．

次に，強等価は全ての演算子と再帰定義によって保存されることを示す．すなわち，強等価は合同関係である．

命題 2.4.5 $E_1 \sim E_2$ とする．このとき次の等式が成り立つ．

(1) $\alpha.E_1 \sim \alpha.E_2$
(2) $E_1 + F \sim E_2 + F$
(3) $E_1|F \sim E_2|F$
(4) $E_1\backslash L \sim E_2\backslash L$
(5) $E_1[f] \sim E_2[f]$

∎

命題 2.4.6 プロセス式 E_i, F_i $(i \in I)$ は高々変数 X_j $(j \in I)$ を含むとする．このとき，各 $i \in I$ について，$A_i \stackrel{\text{def}}{=} E_i\{A_j/X_j\}_{j\in I}$ かつ $B_i \stackrel{\text{def}}{=} F_i\{B_j/X_j\}_{j\in I}$ かつ $E_i \sim F_i$ ならば，各 $i \in I$ について，$A_i \sim B_i$ である． ∎

上記の例 E_{par} と E_{seq} について，

$$A_{par} \stackrel{\text{def}}{=} E_{par}\{A_{par}/X\}$$
$$A_{seq} \stackrel{\text{def}}{=} E_{seq}\{A_{seq}/X\}$$

とすると，$E_{par} \sim E_{seq}$ であるので，命題 2.4.6より，$A_{par} \sim A_{seq}$ である．

本小節の残りで強等価の唯一解の存在を示す．ここで，強等価の解とは両辺を等しするプロセスのことであり，例えばプロセス式 E が高々変数 X を含むとき，等式

$$X \sim E$$

の解は $P \sim E\{P/X\}$ のようなプロセス P である．このような解は (強等価なプロセスを除いて) 唯一であることが保証される．例えば，$BUFF_1 \stackrel{\text{def}}{=} in.\overline{out}.BUFF_1$ によって定義される定数 $BUFF_1$ は等式

$$X \sim in.\overline{out}.X$$

の解であり，この等式の解は全て $BUFF_1$ に強等価となる．

まず，唯一解の条件である**弱いガード**を定義する．

定義 2.4.5 もしプロセス式 E に含まれる全ての変数 X が E のある部分式 $\alpha.F$ に含まれるならば，X は E において**弱くガード**されているという． ∎

次の 2 つの命題は定数 $A \stackrel{\text{def}}{=} E\{A/X\}$ は $X \sim E$ の唯一解であることを示している．

命題 2.4.7 もし $A \stackrel{\text{def}}{=} P$ ならば $A \sim P$ である． ∎

命題 2.4.8 各 $i \in I$ について，プロセス式 E_i は高々変数 X_j $(j \in I)$ を含み，各変数 X_j $(j \in I)$ は E_i において弱くガードされているとする．このとき，各 $i \in I$ について，$P_i \sim E_i\{P_j/X_j\}_{j\in I}$ かつ $Q_i \sim E_i\{Q_j/X_j\}_{j\in I}$ ならば，各 $i \in I$ について，$P_i \sim Q_i$ である． ∎

すでに，$BUFF_2 \sim SSPEC_{20}$ が成り立つことは，$(BUFF_2, SSPEC_{20})$ を含む強双模倣 $\mathcal{S}^{strong}_{BUFF}$ が存在することによって示したが，展開規則と唯一解の命題を用いても証明できる．まず，展開規則によって次の等式が得られる．

$$BUFF_2 \sim in.BUFF'_2$$
$$BUFF'_2 \sim \tau.BUFF_{21}$$
$$BUFF_{21} \sim in.BUFF_{22} + \overline{out}.BUFF_2$$
$$BUFF_{22} \sim \overline{out}.BUFF'_2$$

さらに，命題 2.4.5(1) より，

$$BUFF_2 \sim in.\tau.BUFF_{21}$$
$$BUFF_{21} \sim in.BUFF_{22} + \overline{out}.BUFF_2$$
$$BUFF_{22} \sim \overline{out}.\tau.BUFF_{21}$$

を得る．また，命題 2.4.7より，

$$SSPEC_{20} \sim in.\tau.SSPEC_{21}$$
$$SSPEC_{21} \sim in.SSPEC_{22} + \overline{out}.SSPEC_{20}$$
$$SSPEC_{22} \sim \overline{out}.\tau.SSPEC_{21}$$

である．すなわち，

$$E_0 \equiv in.\tau.X_1$$
$$E_1 \equiv in.X_2 + \overline{out}.X_0$$
$$E_2 \equiv \overline{out}.\tau.X_1$$

とすると，各 $i \in \{0,1,2\}$ について，

$$BUFF_{2i} \sim E_i\{BUFF_{2j}/X_j\}_{j\in\{0,1,2\}}$$
$$SSPEC_{2i} \sim E_i\{SSPEC_{2j}/X_j\}_{j\in\{0,1,2\}}$$

である．ただし，$BUFF_{20} \equiv BUFF_2$ である．E_i において変数は弱くガード されているので，命題 2.4.8より，

$$BUFF_2 \sim SSPEC_{20}$$

を得る．この例のように再帰定義をもつプロセスの等価性は，式を規則にしたがって変形した後，唯一解の命題を用いて判定できる．

2.4.2 観測等価と観測合同

強等価では内部アクションの個数も一致する必要があるため，$BUFF_2 \sim SSPEC_{20}$ ではあるが，$BUFF_2 \sim SPEC_{20}$ は成り立たない ($SPEC_{20}$ は 2.2節で示した仕様の例)．そこで，強等価より弱い等価性として，0 個以上の内部アクションを無視できる観測等価が与えられている．まず，連続した遷移を扱えるようにいくつかの記法を定義する．

定義 2.4.6 $\mathcal{A} \subseteq Act$ とする．このとき，$\mathcal{A}^*$ (要素を $s,t,\cdots$ で表す) は $\mathcal{A}$ に含まれる 0 個以上のアクションを逐次結合してできる有限アクション列の集合である．とくに ε は長さ 0 のアクション列であり，$\varepsilon s = s\varepsilon = s$ である． ∎

定義 2.4.7 集合 $\Longrightarrow \subseteq \mathcal{E}_{ccs} \times Act^* \times \mathcal{E}_{ccs}$ は次の条件を満たす弱い遷移の集合である．

$$E \stackrel{\varepsilon}{\Longrightarrow} E' \iff E \stackrel{\tau}{\longrightarrow} \cdots \stackrel{\tau}{\longrightarrow} E' \quad (0 \text{ 個以上の } \tau \text{ 遷移 })$$
$$E \stackrel{s}{\Longrightarrow} E' \iff E \stackrel{\varepsilon}{\Longrightarrow} \stackrel{\alpha_1}{\longrightarrow} \stackrel{\varepsilon}{\Longrightarrow} \stackrel{\alpha_2}{\longrightarrow} \stackrel{\varepsilon}{\Longrightarrow} \cdots \stackrel{\varepsilon}{\Longrightarrow} \stackrel{\alpha_{n-1}}{\longrightarrow} \stackrel{\varepsilon}{\Longrightarrow} \stackrel{\alpha_n}{\longrightarrow} \stackrel{\varepsilon}{\Longrightarrow} E' \quad (n \geq 1)$$

ここで，$s = \alpha_1 \cdots \alpha_n$ である．∎

定義 2.4.8 $s \in Act^*$ とすると，$\widehat{s} \in \mathcal{L}^*$ は s から全ての τ を取り除いて得られるアクション列である．∎

例えば，$s = ab\tau c\tau\tau de$ ならば，$\widehat{s} = abcde$ である．

これらの記法を用いて，弱双模倣は次のように定義される．

定義 2.4.9 プロセス上の二項関係 $\mathcal{S}$ が**弱双模倣** (weak bisimulation) であるとは，$(P,Q) \in \mathcal{S}$ ならば，任意の $\alpha \in Act$ について，次の 2 つの条件が成り立つことである．

(i) $P \stackrel{\alpha}{\longrightarrow} P'$ ならば，ある Q' について，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．
(ii) $Q \stackrel{\alpha}{\longrightarrow} Q'$ ならば，ある P' について，$P \stackrel{\widehat{\alpha}}{\Longrightarrow} P'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．

∎

例えば，2.2節で与えられたバッファ $BUFF_2$ と仕様 $SPEC_{20}$ について，次の $\mathcal{S}^{weak}_{BUFF}$ は弱双模倣である．

$$\mathcal{S}^{weak}_{BUFF} = \{\, (BUFF_2,\ SPEC_{20}), (BUFF'_2,\ SPEC_{21}),$$
$$(BUFF_{21},\ SPEC_{21}), (BUFF_{22},\ SPEC_{22}) \,\}$$

ここで，$BUFF'_2$，$BUFF_{21}$，$BUFF_{22}$ は 2.4.1小節で与えられている．この $\mathcal{S}^{weak}_{BUFF}$ で注目すべきは，組 $(BUFF'_2,\ SPEC_{21})$ である．遷移 $BUFF'_2 \stackrel{\tau}{\longrightarrow} BUFF_{21}$ に対し，$SPEC_{21} \stackrel{\widehat{\tau}}{\Longrightarrow} SPEC_{21}$ が対応できる．

強等価の定義と同様に観測等価は次のように定義される．

定義 2.4.10 もしある弱双模倣 $\mathcal{S}$ において $(P,Q) \in \mathcal{S}$ ならば，プロセス P と Q は**観測等価** (observation-equivalence) であるといい，$P \approx Q$ と書く．∎

命題 2.4.9 観測等価 $\approx$ は同値関係である．∎

上記の例 $SPEC_{20}$ では，$(BUFF_2, SPEC_{20}) \in \mathcal{S}^{weak}_{BUFF}$ であるので，次の等式を得る．

$$BUFF_2 \approx SPEC_{20}$$

強等価と観測等価の違いを顕著に表している等式を次の命題に示す．

命題 2.4.10 $P \approx \tau.P$ ∎

観測等価で注意すべきことは，観測等価が選択演算子によって保存されないことである．例えば，$b.\mathbf{0} \approx \tau.b.\mathbf{0}$ であるが，

$$a.\mathbf{0} + b.\mathbf{0} \not\approx a.\mathbf{0} + \tau.b.\mathbf{0}$$

である．右辺は τ によって b だけが実行できる状態に遷移することができるが，左辺ではそのような状態をもたないため観測等価とはならない．

そこで，選択演算子によって保存されるように観測等価に条件を付加して観測合同が定義される．

定義 2.4.11 もし次の 2 つの条件が成り立つならば，プロセス P と Q は**観測合同** (observation-congruence) であるといい，$P =_{ccs} Q$ と書く．

(i) $P \xrightarrow{\alpha} P'$ ならば，ある Q' について，$Q \stackrel{\alpha}{\Longrightarrow} Q'$ かつ $P' \approx Q'$ を満たす．
(ii) $Q \xrightarrow{\alpha} Q'$ ならば，ある P' について，$P \stackrel{\alpha}{\Longrightarrow} P'$ かつ $P' \approx Q'$ を満たす．

∎

観測等価との違いは，$\stackrel{\widehat{\alpha}}{\Longrightarrow}$ ではなく $\stackrel{\alpha}{\Longrightarrow}$ が要求されていることである．ただし，この要求は最初の遷移に対してのみであり，それ以降は観測等価 $(P' \approx Q')$ が要求される．すなわち，最初の遷移については一方が τ を実行できるならば，他方も τ を実行できなければならない．観測合同は次の 2 つの命題に示されるとおり，観測等価に含まれ，かつ選択演算子によって保存される最大の関係である．

命題 2.4.11 $P_1 =_{ccs} P_2$ ならば，任意の Q について $P_1 + Q =_{ccs} P_2 + Q$ である． ∎

命題 2.4.12 $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$ とする．任意の Q について $P_1 + Q \approx P_2 + Q$ ならば，$P_1 =_{ccs} P_2$ である．ここで，$\mathcal{L}(P)$ は P に現れる全てのラベルの集合である． ∎

観測合同では最初の τ を無視できないので，$P \neq_{ccs} \tau.P$ である．観測合同のための τ に対する等式を次に示す．

命題 2.4.13

(1) $\alpha.\tau.P = \alpha.P$

(2) $P + \tau.P = \tau.P$

(3) $\alpha.(P + \tau.Q) + \alpha.Q = \alpha.(P + \tau.Q)$

∎

(1) 式が示すように 2 番目以降の内部アクション τ は無視することできる．(2) 式は今すぐ起こせるアクションでも，内部アクションの後に同じように起こせるならば，そのアクションを内部アクションの後に延期しても振舞いは変わらないことを意味している．(3) 式も (2) 式と同様にプロセス Q の延期を意味している．この 3 つの等式は τ 規則と呼ばれる CCS の重要な規則である．

次に強等価と同様に観測合同をプロセス式上に拡張する．

定義 2.4.12 E と F は高々変数 X_i $(i \in I)$ を含むとする．このとき，任意のプロセス P_i $(i \in I)$ について，$E\{P_i/X_i\}_{i\in I} =_{ccs} F\{P_i/X_i\}_{i\in I}$ ならば，$E =_{ccs} F$ である． ∎

観測合同は全ての演算子と再帰定義によって保存される，すなわち合同関係である．

命題 2.4.14 $E_1 =_{ccs} E_2$ とする．このとき次の等式が成り立つ．

(1) $\alpha.E_1 =_{ccs} \alpha.E_2$

(2) $E_1 + F =_{ccs} E_2 + F$

(3) $E_1|F =_{ccs} E_2|F$

(4) $E_1\backslash L =_{ccs} E_2\backslash L$

(5) $E_1[f] =_{ccs} E_2[f]$

∎

命題 2.4.15 プロセス式 E_i, F_i $(i \in I)$ は高々変数 X_j $(j \in I)$ を含むとする．このとき，各 $i \in I$ について，$A_i \stackrel{\text{def}}{=} E_i\{A_j/X_j\}_{j\in I}$ かつ $B_i \stackrel{\text{def}}{=} F_i\{B_j/X_j\}_{j\in I}$ かつ $E_i =_{ccs} F_i$ ならば，各 $i \in I$ について，$A_i =_{ccs} B_i$ である． ∎

観測合同に対する唯一解の存在を示すためには，強等価で使われた条件 (弱いガード) を強める必要がある．まず，次のように**逐次性**と**ガード**を定義する．

定義 2.4.13 もしプロセス式 E の変数 X を含む全ての部分式が $\alpha.F$ か $\sum_{i\in I} F_i$ か X の形であるならば，X は E において**逐次的**であるという．∎

定義 2.4.14 もしプロセス式 E に含まれる全ての変数 X が E のある部分式 $\alpha.F$ に含まれ，かつ $\alpha \neq \tau$ ならば，X は E において**ガード**されているという．∎

例えば，$(\tau.X+a.\mathbf{0})$ において X は逐次的であるがガードされていない．また，$(a.X|b.\mathbf{0})$ において X はガードされているが逐次的ではない．逐次的かつガードされていれば，次の命題が示すように観測合同は唯一解をもつ．

命題 2.4.16 各 $i \in I$ について，プロセス式 E_i は高々変数 X_j $(j \in I)$ を含み，各変数 X_j $(j \in I)$ は E_i において逐次的かつガードされているとする．このとき，各 $i \in I$ について，$P_i =_{ccs} E_i\{P_j/X_j\}_{j\in I}$ かつ $Q_i =_{ccs} E_i\{Q_j/X_j\}_{j\in I}$ ならば，各 $i \in I$ について，$P_i =_{ccs} Q_i$ である．∎

以上，強等価 $\sim$，観測等価 $\approx$，観測合同 $=_{ccs}$ の定義とその特性を紹介してきた．最後に，これら等価性の包含関係を示す．

命題 2.4.17 もし $P \sim Q$ ならば $P =_{ccs} Q$ である．もし $P =_{ccs} Q$ ならば $P \approx Q$ である．∎

2.4.3 公理系

各等価性の公理系を示すことは，式変形によって等価性を証明するためだけでなく，等価性の特性を明らかにするためにも重要である．本小節では有限プロセスに対する強等価と観測合同の健全で完全な公理系を示す．有限プロセスとは定数を含まない (繰り返し動作をもたない) プロセスである．

全ての有限プロセスは，命題 2.4.4の展開規則を用いて，それに強等価な有限逐次プロセスに展開できる．ここで，有限逐次プロセスとは，並行合成，制限，リラベリング，定数を含まないプロセスである．

まず，有限逐次プロセスに対する 2 つの公理系 $\mathcal{A}^{\sim}_{ccs}$ と $\mathcal{A}_{ccs}$ を次のように与える．

定義 2.4.15 2 つの有限逐次プロセス P と Q の等価性が公理系 $\mathcal{A}^{\sim}_{ccs}$ から推論されるな

らば，$\mathcal{A}^{\sim}_{ccs} \vdash P = Q$ と書く．ここで，公理系 $\mathcal{A}^{\sim}_{ccs}$ は次の等式から構成される．

A1 $P + Q = Q + P$
A2 $P + (Q + R) = (P + Q) + R$
A3 $P + P = P$
A4 $P + \mathbf{0} = P$

∎

定義 2.4.16 2つの有限逐次プロセス P と Q の等価性が公理系 $\mathcal{A}_{ccs}$ から推論されるならば，$\mathcal{A}_{ccs} \vdash P = Q$ と書く．ここで，公理系 $\mathcal{A}_{ccs}$ は $\mathcal{A}^{\sim}_{ccs}$ の等式と次の等式から構成される．

T1 $\alpha.\tau.P = \alpha.P$
T2 $P + \tau.P = \tau.P$
T3 $\alpha.(P + \tau.Q) + \alpha.Q = \alpha.(P + \tau.Q)$

∎

公理系 $\mathcal{A}_{ccs}$ で追加された等式は命題 2.4.13に示した τ 規則である．次の命題に示すとおり，τ 規則は観測合同を証明するための必要にして十分な規則である．

命題 2.4.18 P と Q を有限逐次プロセスとする．

(1) $P \sim Q \iff \mathcal{A}^{\sim}_{ccs} \vdash P = Q$
(2) $P =_{ccs} Q \iff \mathcal{A}_{ccs} \vdash P = Q$

∎

例えば，観測等価な等式 $P + \tau.(P + Q) =_{ccs} \tau.(P + Q)$ は公理系 $\mathcal{A}_{ccs}$ により次のように証明できる．

$$
\begin{aligned}
\mathcal{A}_{ccs} \vdash P + \tau.(P + Q) &= P + ((P + Q) + \tau.(P + Q)) && \text{by } \mathbf{T2} \\
&= (P + (P + Q)) + \tau.(P + Q) && \text{by } \mathbf{A2} \\
&= ((P + P) + Q) + \tau.(P + Q) && \text{by } \mathbf{A2} \\
&= (P + Q) + \tau.(P + Q) && \text{by } \mathbf{A3} \\
&= \tau.(P + Q) && \text{by } \mathbf{T2}
\end{aligned}
$$

2.5 プロセス論理

2.4節では，システムと仕様をプロセス代数で記述し，等価性を用いてシステムの振舞いを検証する方法について述べた．本節では，仕様をプロセス論理で記述し，充足性を用いてシステムの振舞いを検証する方法について述べる．文献 [38] では，強い充足性をもつプロセス論理と弱い充足性をもつプロセス論理が与えられている．先ず定義の簡単な強い充足性について説明する．

定義 2.5.1 仕様 ($S, T, \cdots$ で表す) の集合 $\mathcal{PL}^{\sim}$ は次の式を含む最小の集合である．

$$\begin{array}{rcl} \langle\alpha\rangle S & : & \text{可能 } (\alpha \in Act) \\ \neg S & : & \text{否定} \\ \bigwedge_{i\in I} S_i & : & \text{合接 } (I \text{ はインデックス集合 }) \end{array}$$

ここで，S, S_i はすでに $\mathcal{PL}^{\sim}$ の要素であるとする． ∎

直観的には，仕様 $\langle\alpha\rangle S$ は，α を実行でき，その後に仕様 S を満たすことができることを要求している．また，次の記法も用いる．ここで，$\mathtt{tt}$ は全てのプロセスによって満たされる仕様である．

$$\begin{array}{rcl} S_1 \wedge S_2 \equiv \bigwedge_{i\in\{1,2\}} S_i & : & (\text{二項}) \text{ 合接} \\ \mathtt{tt} \equiv \bigwedge_{i\in\emptyset} S_i & : & \text{真} \end{array}$$

例えば，次の仕様は a を実行でき，b を実行できないことを要求する．

$$\langle a\rangle\mathtt{tt} \wedge \neg\langle b\rangle\mathtt{tt}$$

例えば，この仕様はプロセス $(a.\mathbf{0})$ によって満たされるが，プロセス $(a.\mathbf{0} + b.\mathbf{0})$ によっては満たされない．ここで注目すべきは，この仕様は a と b 以外のアクションについては何も要求していないことである．さらに，a を実行した後は全く何も要求していない．例えば，この仕様はプロセス $(a.b.\mathbf{0} + c.\mathbf{0})$ によっても満たされる．このように，プロセス論理では部分的な要求を記述することができる．

プロセスの仕様に対する充足性は形式的に次のように定義される．

定義 2.5.2 充足関係 $\models \subseteq \mathcal{P}_{ccs} \times \mathcal{PL}^{\sim}$ は仕様の構造について帰納的に次のように与えられる．

(1) $P \models \langle\alpha\rangle S \quad \Leftrightarrow \quad \exists P'.\ (P \xrightarrow{\alpha} P',\ P' \models S)$

(2) $P \models \neg S \quad \Leftrightarrow \quad P \not\models S$

(3) $P \models \bigwedge_{i\in I} S_i \quad \Leftrightarrow \quad \forall i \in I.\ P \models S_i$

∎

例えば，次の関係が成り立つ．

$$a.(b.\mathbf{0} + c.\mathbf{0}) \models \langle a\rangle(\langle b\rangle\mathtt{tt} \wedge \langle c\rangle\mathtt{tt})$$
$$a.b.\mathbf{0} + a.c.\mathbf{0} \not\models \langle a\rangle(\langle b\rangle\mathtt{tt} \wedge \langle c\rangle\mathtt{tt})$$

仕様 $\langle a\rangle(\langle b\rangle\mathtt{tt} \wedge \langle c\rangle\mathtt{tt})$ は a を実行後に b と c の両方が実行できる状態があることを要求しており，$a.b.\mathbf{0} + a.c.\mathbf{0}$ はこの仕様を満たさない．

この充足関係と強等価の間には次の関係が成り立つ．すなわち，P と Q が強等価でないならば，そのときに限りそれらを区別できるプロセス論理式 S が存在する．

命題 2.5.1 $P \not\sim Q \iff \exists S \in \mathcal{PL}^{\sim}.\ (P \models S,\ Q \not\models S)$ ∎

この命題が示すように，可能，否定，合接の演算子で十分な表現力をもつが，否定 $\neg$ を多く含む仕様の意味は分かりにくくなる．否定を減らすために，略記法として次のような演算子が用意されている．

$$[\alpha]S \equiv \neg\langle\alpha\rangle\neg S \quad : \quad \text{必然}$$
$$\bigvee_{i\in I} S_i \equiv \neg\bigwedge_{i\in I}\neg S_i \quad : \quad \text{離接}$$
$$S_1 \vee S_2 \equiv \bigvee_{i\in\{1,2\}} S_i \quad : \quad \text{(二項) 離接}$$
$$\mathtt{ff} \equiv \neg\mathtt{tt} \quad : \quad \text{偽}$$

ここで，仕様 $[\alpha]S$ は，もしアクション α を実行するならば，その後は必ず S を満たすことを要求する．また，$\mathtt{ff}$ は満たすことができない仕様である．例えば，次の仕様は，アクション a を実行した後は，必ず b を実行でき，かつ c を実行できないことを要求している．

$$[a](\langle b\rangle\mathtt{tt} \wedge [c]\mathtt{ff})$$

これらの演算子を利用すれば，全ての仕様を否定を含まない仕様に変換できる．

次に，弱い充足性をもつプロセス論理について述べる．

定義 2.5.3 仕様 ($S, T, \cdots$ で表す) の集合 $\mathcal{PL}^{\approx}$ は次の式を含む最小の集合である．

$$\begin{aligned} \langle\langle\varepsilon\rangle\rangle S &: \text{弱可能} \\ \langle\langle l\rangle\rangle S &: \text{弱可能 } (l \in \mathcal{L}) \\ \neg S &: \text{否定} \\ \textstyle\bigwedge_{i\in I} S_i &: \text{合接 } (I \text{ はインデックス集合 }) \end{aligned}$$

ここで，S, S_i はすでに $\mathcal{PL}^{\approx}$ の要素であるとする． ∎

$\mathcal{PL}^{\sim}$ の場合と同様に，$\mathcal{PL}^{\approx}$ に対しても弱必然や離接を定義できる．充足性については，$\xrightarrow{\alpha}$ の代わりに $\stackrel{\alpha}{\Longrightarrow}$ が使われる．

定義 2.5.4 充足関係 $\models \subseteq \mathcal{P}_{ccs} \times \mathcal{PL}^{\approx}$ は仕様の構造について帰納的に次のように与えられる．

(1) $P \models \langle\langle\varepsilon\rangle\rangle S \quad \Leftrightarrow \quad \exists P'.\ (P \stackrel{\varepsilon}{\Longrightarrow} P',\ P' \models S)$
(2) $P \models \langle\langle l\rangle\rangle S \quad \Leftrightarrow \quad \exists P'.\ (P \stackrel{l}{\Longrightarrow} P',\ P' \models S)$
(3) $P \models \neg S \quad \Leftrightarrow \quad P \not\models S$
(4) $P \models \bigwedge_{i\in I} S_i \quad \Leftrightarrow \quad \forall i \in I.\ P \models S_i$

∎

仕様 $\langle\langle l\rangle\rangle S$ は 0 個以上の内部アクションを実行後に観測可能なアクション l を実行でき，その後 0 個以上の内部アクションを実行後に S を満たすことができることを要求する．また，仕様 $\langle\langle\varepsilon\rangle\rangle S$ は 0 個以上の内部アクションを実行後に S を満たすことができることを要求する．期待通り，観測等価との間に次の関係が成り立つ．

命題 2.5.2 $P \not\approx Q \iff \exists S \in \mathcal{PL}^{\approx}.\ (P \models S,\ Q \not\models S)$ ∎

2.6 おわりに

本章では，基本的なプロセス代数 CCS を文献 [38] をもとに紹介してきた．まず，プロセス代数でシステムと仕様を記述し，システムと仕様の振舞いが等価であるかを検証する方法について説明した．等価性は観測合同のように内部の振舞いを無視できるほど弱く，かつ合同関係であることが望まれる．また，健全で完全な公理系を示すこ

とは，式変形によって等価性を証明するだけでなく，等価性の特性を明らかにするためにも重要である．

プロセス代数を拡張した場合は，観測合同のように弱い合同関係を適切に定義し，その公理系を示すことが重要である．4章と 5章では，ブロードキャスト通信や減衰通信をもつシステムの解析ができるようにプロセス代数を拡張した場合の合同関係や公理系について，本章と同様に議論する．

さらに，プロセス論理を用いて仕様を部分的に記述し，システムが仕様を充足していることを検証できることを紹介した．6章では，論理演算子をプロセス代数に導入して部分的 (柔軟な) 仕様からシステムまで，同じ枠組で記述する方法について述べる．また，7章では，プロセス論理で記述された仕様からプロセス代数で記述された分散システムを合成する方法について述べる．

第 3 章

プロダクションルール解析用プロセス代数

3.1 はじめに

データベースが状況に応じて自動的にデータを更新できれば，その管理者の負担を大幅に軽減できる．**アクティブデータベース** [16, 37] は，そのような自動更新のために**プロダクションルール**を利用したデータベースである．ここで，プロダクションルールとは，ある変化 (event) が起こったとき，ある条件 (condition) が満たされていたならば，ある動作 (action) をすることを規定するルールである．これは ECA ルールと呼ばれている．アクティブデータベースの管理者は必要に応じてプロダクションルールを追加・修正することによって，より状況に適応できるデータベースを構築できる．

アクティブデータベースは状況の変化に応じたルールを呼び出して，そのルールを実行するためのプロセスを起動する．さらにこのルールの動作によって状況が変化すると，その変化に応じた別のルールが呼び出される．このように，一つの状況変化から次々にルールが呼び出されて実行されることが可能であるが，そのルールの実行順序は呼び出された順番が適切とは限らない．例えば，情報の表示などは他のルールとは独立に並行に実行され，無矛盾性の判定などは他のルールの処理が終了してから実行されることが期待される．このような要求に対し，ルール間の実行順序を制御する方法として**カップリングモード** [21] を利用することが有効であり，カップリングモードをもつアクティブデータベースとして，SAMOS[16]，HiPAC[37] などがある．

カップリングモードはルールを効率良く適切に実行するために有効であるが，ルールの数が増えたときに実際にどのような順番でルールが実行されるかを把握し設計することは困難になる．そこで本章では，カップリングモードをもつアクティブデータベースのプロダクションルールの動作を解析するためのプロセス代数として CCSPR (a Calculus of Communicating Systems with Production Rules) を提案する．CCSPR は，親プロセスがルール集合から一つのルールを子プロセスとして呼び出してプロセス木を構築するとき，その親子関係を明確に表現できる特徴をもつ．そして，CCSPR によってプロダクションルールを記述し，ルールの振舞いが仕様を満たしているかを検証する方法を示す．

本章の構成は次のとおりである．まず，3.2節でアクティブデータベースを紹介する．次に 3.3節で，成長するプロセス木や親子間の局所通信が CCSPR によってどのように記述されるかを，従来のプロセス代数と比較して説明する．3.4節で CCSPR の構文と意味を定義し，3.5節で CCSPR の強等価に対するいくつかの特性を示す．3.6節では，CCSPR によってプロダクションルールを解析する方法について例をもとに説明する．

3.2 アクティブデータベース

アクティブデータベース [16, 37] は，プロダクションルールによって状況に応じた動作を自動的に実行できる．各プロダクションルールには，状況変化 (event)，条件 (condition)，動作 (action) が書かれており，その状況変化が生じたとき，その条件が満たされていたならば，その動作が実行されることを要求する．アクティブデータベースは状況を監視し，状況が変化したとき，その変化に反応するプロダクションルールを呼び出し，そのルールを処理するプロセスを起動する．もし，その変化に反応する複数のルールがあるときは，それら全てを呼び出して並行に処理する．さらに，あるルール R_1 の処理によって状況が変化し，その変化に反応するルール R_2 があるときは，ルール R_1 を処理しているプロセスに対して子プロセスを起動し，その子プロセスによってルール R_2 を処理する [37]．このルールの順次呼び出しによってプロセス木が構築される．

図 3.1の例をもとに，プロダクションルールの振舞いを説明する．各ルールの E,C,A は各々，状況変化 (event)，条件 (condition)，動作 (action) であり，CM はカップリングモードを表している．状況変化 E に updatex とあるのは，このルールが変数 x の更

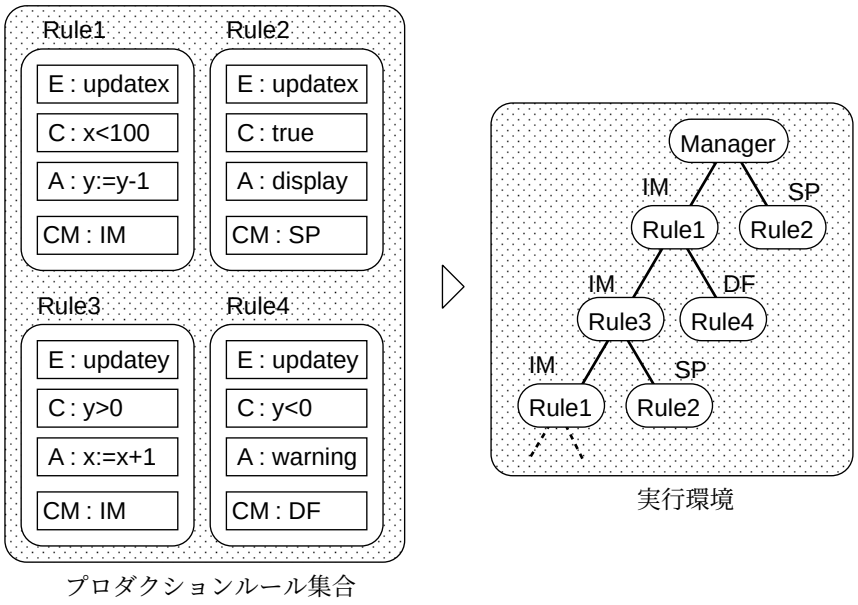

図 3.1: プロダクションルール集合とその実行例

新によって呼び出されることを表している．updatey についても同様である．今，変数 x が更新されたとする．このとき状況の監視プロセス (Manager) は Rule1 と Rule2 を呼び出す．呼び出された Rule1 は，もし条件 $(x < 100)$ が成り立つならば，y を 1 減じ，Rule2 は変数 x を表示する．このとき y が更新されるため，Rule3 と Rule4 が呼び出される．ここで，Rule3 と Rule4 を処理するプロセスは (それらを呼び出した)Rule1 を処理するプロセスの子プロセスとなる．これにより，図 3.1に示すようにプロセス木が構築される．そして，更に Rule3 が x を更新することによって，Rule1 と Rule2 が再び呼び出される可能性もある．

呼び出されたルールの実行順序を制御するためにカップリングモード [21] が有効であり，カップリングモードをもつアクティブデータベースとして，SAMOS[16]，HiPAC[37] などが知られている．SAMOS，HiPAC では，次の 3 つのカップリングモードが用意されている．

1. **即実行モード** (immediate mode)：子プロセスの処理はすぐに実行され，その処理の完了を待ってその親プロセスも処理を完了する．

2. **分離実行モード** (separate mode)：子プロセスの処理はすぐに実行され，その処理の完了を待たずにその親プロセスは処理を完了する．

3. **延期実行モード** (deferred mode) : 子プロセスの処理は，その子プロセスを含むプロセス木の一番上の親プロセスの完了直前まで延期されて実行される．

図 3.1では，IM, SP, DF が各々，即実行モード，分離実行モード，延期実行モードを表している．変数 x の表示は独立に処理できるので，Rule2 は分離実行モードをもち，全体の処理完了後に変数 y が負になっていないかをチェックする Rule4 は延期実行モードをもつ．このようにカップリングモードは親子プロセス間の関係だけでなく，延期実行モードのように全体のプロセスに関係する制御も表現できる．

3.3 プロセス代数による検証

アクティブデータベースの利用者はプロダクションルールを自由に追加 (修正) できることを望むが，そのルールの追加が全体の振舞いにどのように影響するかを予測することはそれほど容易ではない．ルールが別のルールを呼び出し，それらのルールの実行順序はカップリングモードによって制御されているためである．そこで，設計時に振舞いを検証するツールが必要になる．

アクティブデータベースに対して，いくつかの静的解析方法が提案されている．例えば，有向起動グラフ (directed triggering graphs) がルールの振舞いについての情報を与えるために利用されている [3, 9]．また，ペトリネットが合成イベントの探索に使われている [17]．しかし，従来の方法ではカップリングモードは考慮されていなかった．本研究では，カップリングモードをもつプロダクションルールの実行順序を，プロセス代数によって検証する．まず 3.3.1小節で，従来のプロセス代数でプロダクションルールの振舞いを記述しようとするとき，どのような点が記述困難であるかを指摘する．そして 3.3.2小節で，プロセス代数 CCSPR がプロダクションルールの振舞いの記述に適していることを示す．

3.3.1 従来のプロセス代数による記述

プロダクションルールを記述するために重要なことは，成長可能なプロセス木を表現できることである．従来のプロセス代数でも不変なプロセス木は簡単に表現できる．例えば，プロセス代数 CSP[18] には，従属演算子 $/\!\!/$ があり，

$$Q /\!\!/ P$$

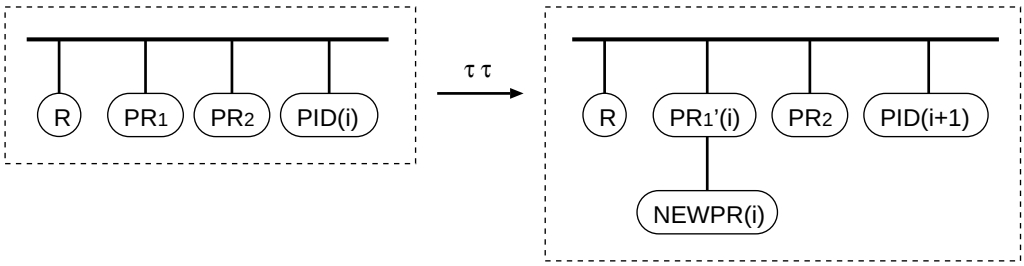

図 3.2: プロセス生成の例

はプロセス Q はプロセス P の従属プロセス (子プロセス) であることを明確に (構文的に) 表している．しかし，成長するプロセス木を表現するためには適していない．プロセス木を記述する他の方法として，固有の値 ID によって親子プロセスを関係付ける方法が考えられる．例えば，成長するプロセス木は ID を用いてプロセス代数 CCS によって次のように記述できる．

$$(R \mid PR_1 \mid PR_2 \mid PID(i)) \xrightarrow{\tau}\xrightarrow{\tau} (R \mid NEWPR(i) \mid PR_1'(i) \mid PR_2 \mid PID(i+1))$$

ここで，各プロセスは次のように定義されている．

$$\begin{aligned} R &\stackrel{\text{def}}{=} a(x).(R \mid NEWPR(x)) \\ PR_1 &\stackrel{\text{def}}{=} getid(x).\overline{a}(x).PR_1'(x) \\ PR_2 &\stackrel{\text{def}}{=} getid(x).\overline{a}(x).PR_2'(x) \\ PID(i) &\stackrel{\text{def}}{=} \overline{getid}(i).PID(i+1) \end{aligned}$$

ここで，値 i と変数 x をもつ相補的なアクション $\overline{getid}(i)$ と $getid(x)$ が同期すると，値 i が渡されて変数 x に代入される．

プロセス PR_1 は ID を管理しているプロセス PID から ID として値 i をチャンネル $getid$ を通して受け取り，変数 x に代入する (上記の遷移の最初の τ)．プロセス PID は保持する値 i を 1 増やし，次の要求 $getid$ に備える．ID として i を受け取った PR_1 はチャンネル a を通して値 i をプロセス R に渡し，新しいプロセス $NEWPR(i)$ を生成することを要求する (上記の遷移の 2 番目の τ)．$PR_1'(i)$ と $NEWPR(i)$ の親子関係は値 i によって維持される．もし ID がなければ親子関係を特定するとはできない．図 3.2にこの子プロセスの生成過程を示す．

3.3.2 プロセス代数 CCSPR による記述

前小節で述べたように，成長するプロセス木を ID を用いて表現することができる．しかし，ID は記述を複雑にし，その検証を困難にする．また，プロセス間の親子関係も明確ではない．

そこで，成長するプロセス木の記述に適したプロセス代数として CCSPR を提案する．CCSPR では，成長するプロセス木を**従属演算子** $\rangle$ によって構文的に記述できる特徴をもつ．例えば，3.3.1小節で使われた例は，CCSPR では次のように記述される．

$$R \triangleright (PR_1 \mid PR_2) \xrightarrow{\tau} R \triangleright ((PR_1' \rangle NEWPR) \mid PR_2)$$

ここで，各プロセスは次のように定義されている．

$$\begin{aligned} R &\equiv \{\!\{a.NEWPR\}\!\} \\ PR_1 &\stackrel{\text{def}}{=} \overline{a}.PR_1' \\ PR_2 &\stackrel{\text{def}}{=} \overline{a}.PR_2' \end{aligned}$$

ここで，$\{\!\{\ \}\!\}$ は**リソース演算子**であり，R はアクション a によって呼び出されるプロセス $NEWPR$ を保持する**リソース**を表している．また，$\triangleright$ はリソースからプロセスに新しいプロセスを供給する演算子である．上記の τ 遷移で注目すべきは，生成されたプロセス $NEWPR$ の位置である．$(PR_1' \rangle NEWPR)$ は生成された $NEWPR$ が PR_1' の子プロセスであることを明確に表している．

一方，上記の記述については，次の遷移も可能である．

$$R \triangleright (PR_1 \mid PR_2) \xrightarrow{\tau} R \triangleright ((PR_1' \mid PR_2) \rangle NEWPR)$$

このとき，PR_1 と PR_2 の両方が $NEWPR$ の親プロセスとなる．このように，CCSPR では，複数のプロセスが親プロセスになることもできる．上記の例では，$NEWPR$ の親が PR_1' だけか，または PR_1' と PR_2 の両方であるかは非決定的に決まるが，親プロセスが決定的に決まるように記述することもできる．そのためには，制限演算子 $\backslash$ と**パック演算子** $[\![\ ]\!]$ を用いる．例えば，次の記述では必ず P_1 と P_2 が Q_1 の親となる．

$$\{\!\{a.Q_1\}\!\} \triangleright ([\![\overline{a}.P_1 | P_2]\!] \backslash \overline{a} | P_3) \xrightarrow{\tau} \{\!\{a.Q_1\}\!\} \triangleright (([\![P_1 | P_2]\!] \rangle Q_1) \backslash \overline{a} | P_3)$$

供給 $\triangleright$ は，制限 $\backslash$ や並行合成 $|$ の内側に新しいプロセスを供給することができるが，$[\![\ ]\!]$ の内側には供給できない．また，制限 $\backslash\overline{a}$ は，制限の外側に新しいプロセスが供給され

ることを禁止している．もし制限$\backslash\overline{a}$やパック$[\![\ ]\!]$がなければ次の遷移も可能となる．

$$\{\!\{a.Q_1\}\!\} \triangleright ([\![\overline{a}.P_1|P_2]\!]|P_3) \xrightarrow{\tau} \{\!\{a.Q_1\}\!\} \triangleright (([\![P_1|P_2]\!]|P_3)\rangle Q_1)$$
$$\{\!\{a.Q_1\}\!\} \triangleright ((\overline{a}.P_1|P_2)\backslash\overline{a}|P_3) \xrightarrow{\tau} \{\!\{a.Q_1\}\!\} \triangleright (((P_1\rangle Q_1)|P_2)\backslash\overline{a}|P_3)$$

制限がなければ，P_1, P_2, P_3 の全てが Q_1 の親プロセスになることもでき，パックがなければ，P_1 だけが Q_1 の親プロセスになることもできる．このように，呼び出されたプロセスは制限の内側，かつパックの外側に生成されることが重要である．

次に孫プロセスを生成する例を示す．

$$\begin{aligned}
&(\{\!\{a.(\overline{b}.Q_1)\backslash\overline{b}\}\!\} :: \{\!\{b.Q_2\}\!\}) \triangleright (\overline{a}.P)\backslash\overline{a} \\
&\quad \xrightarrow{\tau} (\{\!\{a.(\overline{b}.Q_1)\backslash\overline{b}\}\!\} :: \{\!\{b.Q_2\}\!\}) \triangleright (P\rangle(\overline{b}.Q_1)\backslash\overline{b})\backslash\overline{a} \\
&\quad \xrightarrow{\tau} (\{\!\{a.(\overline{b}.Q_1)\backslash\overline{b}\}\!\} :: \{\!\{b.Q_2\}\!\}) \triangleright (P\rangle(Q_1\rangle Q_2)\backslash\overline{b})\backslash\overline{a} \quad (*)
\end{aligned}$$

ここで，$::$ はリソースの**結合演算子**である．このとき，Q_1 は P の子プロセスであり，Q_2 は Q_1 の子プロセスである．すなわち，Q_2 は P の孫プロセスである．

一方，同じプロセスが続けて複数のプロセスを呼び出すことによって，次の例のように複数の子プロセスをもつこともできる．

$$\begin{aligned}
&(\{\!\{a.Q_1\}\!\} :: \{\!\{b.Q_2\}\!\}) \triangleright (\overline{a}.\overline{b}.P)\setminus\{\overline{a},\overline{b}\} \\
&\quad \xrightarrow{\tau} (\{\!\{a.Q_1\}\!\} :: \{\!\{b.Q_2\}\!\}) \triangleright (\overline{b}.P\rangle Q_1)\setminus\{\overline{a},\overline{b}\} \\
&\quad \xrightarrow{\tau} (\{\!\{a.Q_1\}\!\} :: \{\!\{b.Q_2\}\!\}) \triangleright ((P\rangle Q_1)\rangle Q_2)\setminus\{\overline{a},\overline{b}\} \quad (**)
\end{aligned}$$

この場合，Q_1 と Q_2 は P の子プロセスである．$(*)$ と $(**)$ の括弧の位置が重要である．

プロセスの親子関係が構築されることによって，親子間の局所通信が可能となる．次の例に示すように，この通信は親プロセスの親アクション $\lceil a\rceil$ と全ての子プロセスの子アクション $\lfloor a\rfloor$ による 1 対他通信である．

$$((\lceil a\rceil.P\rangle\lfloor a\rfloor.Q_1)\rangle\lfloor a\rfloor.Q_2)/\lceil a\rceil \xrightarrow{\tau} ((P\rangle Q_1)\rangle Q_2)/\lceil a\rceil$$

ここで，Q_1 と Q_2 は P の子プロセスである．**隠蔽演算子** $/\lceil a\rceil$ は，親プロセスが $\lceil a\rceil$ によってこれ以上他の子プロセスと同期しないように，$\lceil a\rceil$ を τ に変換する．

リソースと親子アクション $\lfloor a\rfloor, \lceil a\rceil$ によって，プロセスを呼び出すと自動的に親子間の局所通信が可能になる振舞いを記述できる．次の例は，$\overline{a}$ でプロセスを呼び出すと $\lfloor b\rfloor, \lceil b\rceil$ によって親子間局所通信が可能になる例である．

$$\begin{aligned}
\{\!\{a.\lfloor b\rfloor.Q\}\!\} \triangleright ((\overline{a}.\lceil b\rceil.P)\backslash a)/\lceil b\rceil &\xrightarrow{\tau} \{\!\{a.\lfloor b\rfloor.Q\}\!\} \triangleright ((\lceil b\rceil.P\rangle\lfloor b\rfloor.Q)\backslash a)/\lceil b\rceil \\
&\xrightarrow{\tau} \{\!\{a.\lfloor b\rfloor.Q\}\!\} \triangleright ((P\rangle Q)\backslash a)/\lceil b\rceil
\end{aligned}$$

表 3.1: 集合と要素上の変数

集合の記法	集合の名前	要素上変数
$\mathcal{N}$	名前の集合	$a, b, \cdots$
$\overline{\mathcal{N}} = \{\overline{a} : a \in \mathcal{N}\}$	余名の集合	$\overline{a}, \overline{b}, \cdots$
$\mathcal{L}_G = \mathcal{N} \cup \overline{\mathcal{N}}$	大域ラベルの集合	$\omega, \omega', \cdots$
$\mathcal{L}_P = \{\lceil a \rceil : a \in \mathcal{N}\}$	親ラベルの集合	$\lceil a \rceil, \lceil b \rceil, \cdots$
$\mathcal{L}_C = \{\lfloor a \rfloor : a \in \mathcal{N}\}$	子ラベルの集合	$\lfloor a \rfloor, \lfloor b \rfloor, \cdots$
$\mathcal{L} = \mathcal{L}_G \cup \mathcal{L}_P \cup \mathcal{L}_C$	ラベルの集合	$l, l', \cdots$
$Act = \mathcal{L} \cup \{\tau\}$	アクションの集合	$\alpha, \alpha', \cdots$

また，次の例に示すように，リソースに同じアクションで呼び出される複数のプロセスがあるときは，それらは全て同時に呼び出され並行に処理される．

$$(\{\!\{a.Q_1\}\!\} :: \{\!\{a.Q_2\}\!\}) \rhd (\overline{a}.P) \backslash a \xrightarrow{\tau} (\{\!\{a.Q_1\}\!\} :: \{\!\{a.Q_2\}\!\}) \rhd (P \rangle (Q_1 \parallel Q_2)) \backslash a$$

ここで，$\parallel$ は子アクション $\lfloor a \rfloor$ は必ず同期することを要求する**同期並行合成演算子**であり，子アクション $\lfloor a \rfloor$ の独立実行を禁止する以外並行合成 $|$ と同じである．これにより，全ての子プロセスの情報を確実に親プロセスに伝えることができる．

3.4 CCSPR の構文と意味

まず，**名前** ($a, b, \cdots$ で表す) の無限集合 $\mathcal{N}$ が与えられていると仮定する．このとき，表 3.1のように各集合を与える．τ は**内部アクション**と呼ばれる観測されない特殊なアクションを表しており，$\mathcal{N}$ には含まれていないとする．

CCS と比較して親アクション $\lceil a \rceil$ (親ラベルのついたアクション) と子アクション $\lfloor a \rfloor$ が親子間の局所的な通信のために追加されている．これは，一つの親プロセスとその全ての子プロセスが同期する 1 対他通信であり，全ての子プロセスの処理が完了したことを伝えるために有効である．

また，定数 ($A, \cdots$ で表す) の集合 $\mathcal{K}_{ccspr}$ が与えられているとする．このとき CCSPR の構文を次のように与える．

定義 3.4.1 プロセス ($P, Q, \cdots$ で表す) の集合 $\mathcal{P}_{ccspr}$ は次の式を含む最小の集合である

$$
\begin{aligned}
A &: \text{プロセス定数 } (A \in \mathcal{K}_{ccspr}) \\
R &: \text{リソース } (R \in \mathcal{R}_{ccspr}) \\
\alpha.P &: \text{逐次 } (\alpha \in Act) \\
\textstyle\sum_{i\in I} P_i &: \text{選択 } (I \text{ はインデックス集合 }) \\
P|Q &: \text{並行合成} \\
P \| Q &: \text{同期並行合成} \\
P \rangle Q &: \text{従属} \\
P[f] &: \text{リラベリング } (f \text{ はリラベリング関数 }) \\
P \backslash L &: \text{制限 } (L \subseteq \mathcal{L}_G) \\
P/L &: \text{隠蔽 } (L \subseteq \mathcal{L}_P) \\
[\![P]\!] &: \text{パック} \\
R \triangleright P &: \text{供給 } (R \in \mathcal{R}_{ccspr})
\end{aligned}
$$

ここで，P, P_i, Q はすでに $\mathcal{P}_{ccspr}$ の要素であるとする．また，**リソース** ($R, \cdots$ で表す) の集合 $\mathcal{R}_{ccspr}$ は次の式を含む最小の集合である．

$$
\begin{aligned}
\{\!\{\omega.P\}\!\} &: \text{リソース } (\omega \in \mathcal{L}_G, P \in \mathcal{P}_{ccspr}) \\
R_1 :: R_2 &: \text{結合}
\end{aligned}
$$

ここで，R_1, R_2 はすでに $\mathcal{R}_{ccspr}$ の要素であるとする． ∎

リラベリング関数 $f : Act \to Act$ は次の条件を満たす関数である．すなわち，リラベリグ関数によってラベルの種類が変わることはない．

- $f(\alpha) \in \mathcal{L}_G \iff \alpha \in \mathcal{L}_G$
- $f(\alpha) \in \mathcal{L}_P \iff \alpha \in \mathcal{L}_P$
- $f(\alpha) \in \mathcal{L}_C \iff \alpha \in \mathcal{L}_C$
- $\overline{f(\omega)} = f(\overline{\omega})$
- $\lceil f(a) \rceil = f(\lceil a \rceil)$
- $\lfloor f(a) \rfloor = f(\lfloor a \rfloor)$
- $f(\tau) = \tau$

リソース $\{\!\{\omega_1.P_1\}\!\} :: \cdots :: \{\!\{\omega_n.P_n\}\!\}$ は，アクション ω_i によって呼び出されるプロセス P_i の集合である．しばしば，$\{\!\{\omega_1.P_1\}\!\} :: \cdots :: \{\!\{\omega_n.P_n\}\!\}$ を $\{\!\{\omega_i.P_i : i \in \{1, \cdots, n\}\}\!\}$ と書く．また，CCS と同様に，$\sum_{i\in\{1,2\}} P_i$ を $P_1 + P_2$ と書く．

全ての定数 $A \in \mathcal{K}_{ccspr}$ について，$A \stackrel{\text{def}}{=} P$ の形の定義式があるとする．特殊な定数として，**0** と **I** を次のように定義する．

$$
\begin{aligned}
\mathbf{0} &\stackrel{\text{def}}{=} \textstyle\sum_{i\in\emptyset} P_i \quad &: \text{無動作プロセス} \\
\mathbf{I} &\stackrel{\text{def}}{=} \textstyle\sum_{a\in\mathcal{N}} \lfloor a \rfloor.\mathbf{I} \quad &: \text{子アクション同期用プロセス}
\end{aligned}
$$

定数 $\mathbf{I}$ は同期並行合成 $\|$ に対する単位プロセスに相当する．例えば，P と $P\|\mathbf{I}$ の振舞いは等しいが，P と $P\|\mathbf{0}$ の振舞いは異なる．

次に，CCSPR のプロセスの振舞いを定義する．

定義 3.4.2 プロセスの意味は次のラベル付遷移システムにより与えられる．

$$(\mathcal{P}_{ccspr}, Act, \longrightarrow)$$

ここで，ラベル付遷移の集合 $\longrightarrow \subseteq \mathcal{P}_{ccspr} \times Act \times \mathcal{P}_{ccspr}$ は図 3.3と図 3.4 の推論規則を満たす最小の集合である．図 3.4中の関数 $call : \mathcal{R}_{ccspr} \to 2^{\mathcal{L}_G}$ はプロセスの呼び出しに使われるアクションをリソースから取り出す関数であり，次のように定義される．

$$call(\{\!\{\omega.P\}\!\}) = \{\omega\}$$
$$call(R_1 :: R_2) = call(R_1) \cup call(R_2)$$

∎

CCSPR の遷移の推論規則は図 3.3と図 3.4に示されるように 2 つに大別できる．図 3.3はプロセスの基本的な振舞いのための規則であり，図 3.4はプロセスを呼び出すための規則である．

規則 $\mathbf{Subo}_3$ にみられるように，親子アクション $\lceil a \rceil, \lfloor a \rfloor$ によって通信した後のアクションは内部アクション τ ではなく $\lceil a \rceil$ である．これは，親プロセスがさらに他の子プロセスと通信できることを意味している．親プロセスが他の子プロセスと通信しないようにするには，制限 $\backslash$ ではなく隠蔽 $/$ を用いる．

同期並行合成では規則 $\mathbf{Sync}_{1,2}$ にみられるように，子アクションは必ず同期しなければならない．これは，全ての子プロセスからの情報を確実に親プロセスに伝えるために有効である．そこで，リソースから一度に複数のプロセスが呼び出されるときは，$\mathbf{Uni}_3$ にみられるように，それらのプロセスを同期並行合成する．

名前に $\mathbf{S}.$ のついた規則 ($\mathbf{S.Choice}_j$, $\mathbf{S.Com}_{1,2}$, $\mathbf{S.Sync}_{1,2}$, $\mathbf{S.Subo}_{1,2}$, $\mathbf{S.Rel}$, $\mathbf{S.Res}$, $\mathbf{S.Hide}$, $\mathbf{S.Rec}$) は，リソースから呼び出されたプロセスが各演算子 ($+, |, \|, \rangle, [f], \backslash, /$) の内側に入り込み，呼び出したプロセスの下にくるために必要である．例えば，遷移

$$\{\!\{a.Q\}\!\} \triangleright (\overline{a}.P_1 \mid P_2) \xrightarrow{\tau} \{\!\{a.Q\}\!\} \triangleright ((P_1 \rangle Q) \mid P_2)$$

名前	仮定	$\vdash$	結果
Act		$\vdash$	$\alpha.P \xrightarrow{\alpha} P$
Choice$_j$	$P_j \xrightarrow{\alpha} P',\ j \in I$	$\vdash$	$\sum_{i\in I} P_i \xrightarrow{\alpha} P'$
Com$_1$	$P \xrightarrow{\alpha} P'$	$\vdash$	$P\|Q \xrightarrow{\alpha} P'\|Q$
Com$_2$	$Q \xrightarrow{\alpha} Q'$	$\vdash$	$P\|Q \xrightarrow{\alpha} P\|Q'$
Com$_3$	$P \xrightarrow{\omega} P',\ Q \xrightarrow{\overline{\omega}} Q'$	$\vdash$	$P\|Q \xrightarrow{\tau} P'\|Q'$
Sync$_1$	$P\|Q \xrightarrow{\alpha} P'\|Q',\ \alpha \notin \mathcal{L}_C$	$\vdash$	$P \parallel Q \xrightarrow{\alpha} P' \parallel Q'$
Sync$_2$	$P \xrightarrow{\lfloor a \rfloor} P',\ Q \xrightarrow{\lfloor a \rfloor} Q'$	$\vdash$	$P \parallel Q \xrightarrow{\lfloor a \rfloor} P' \parallel Q'$
Subo$_1$	$P\|Q \xrightarrow{\alpha} P'\|Q',\ \alpha \notin \mathcal{L}_P \cup \mathcal{L}_C$	$\vdash$	$P \rangle Q \xrightarrow{\alpha} P' \rangle Q'$
Subo$_2$	$P \xrightarrow{\lfloor a \rfloor} P'$	$\vdash$	$P \rangle Q \xrightarrow{\lfloor a \rfloor} P' \rangle Q$
Subo$_3$	$P \xrightarrow{\lceil a \rceil} P',\ Q \xrightarrow{\lfloor a \rfloor} Q'$	$\vdash$	$P \rangle Q \xrightarrow{\lceil a \rceil} P' \rangle Q'$
Rel	$P \xrightarrow{\alpha} P'$	$\vdash$	$P[f] \xrightarrow{f(\alpha)} P'[f]$
Res	$P \xrightarrow{\alpha} P',\ \alpha \notin L \cup \overline{L}$	$\vdash$	$P \setminus L \xrightarrow{\alpha} P' \setminus L$
Hide$_1$	$P \xrightarrow{\lceil a \rceil} P',\ \lceil a \rceil \in L$	$\vdash$	$P/L \xrightarrow{\tau} P'/L$
Hide$_2$	$P \xrightarrow{\alpha} P',\ \alpha \notin L$	$\vdash$	$P/L \xrightarrow{\alpha} P'/L$
Pack	$P \xrightarrow{\alpha} P'$	$\vdash$	$[\![P]\!] \xrightarrow{\alpha} [\![P']\!]$
Rec	$P \xrightarrow{\alpha} P',\ A \stackrel{\text{def}}{=} P$	$\vdash$	$A \xrightarrow{\alpha} P'$

図 3.3: プロセス上のラベル付遷移の推論規則 1

は次のように導かれる.

$$
\begin{aligned}
&\vdash \{\!\{a.Q\}\!\} \xrightarrow{a} \{\!\{a.Q\}\!\} \triangleright Q \cdots\cdots (*1) && \text{by } \mathbf{Reso}\\
&\vdash \overline{a}.P_1 \xrightarrow{\overline{a}} P_1 \cdots\cdots (*2) && \text{by } \mathbf{Act}\\
(*1), (*2) &\vdash \{\!\{a.Q\}\!\} \triangleright \overline{a}.P_1 \xrightarrow{\tau} \{\!\{a.Q\}\!\} \triangleright (P_1 \rangle Q) \cdots\cdots (*3) && \text{by } \mathbf{Supp}_1\\
(*3) &\vdash \{\!\{a.Q\}\!\} \triangleright (\overline{a}.P_1 | P_2) \xrightarrow{\tau} \{\!\{a.Q\}\!\} \triangleright ((P_1 \rangle Q) | P_2) && \text{by } \mathbf{S.Com}_1
\end{aligned}
$$

もし規則 $\mathbf{S.Com}_1$ がなければ，呼び出された Q は P_2 の内側に入り込むことができず，P_1 との間に親子関係を構築できない.

図3.3のパックのための規則**Pack**は意味が無いようにみえるが，図3.4に規則**S.Pack**が無いことが重要である．これは，呼び出されたプロセスはパックの内側に入れない

名前	仮定	$\vdash$	結果
Reso		$\vdash$	$\{\!\{\omega.P\}\!\} \xrightarrow{\omega} \{\!\{\omega.P\}\!\} \triangleright P$
$\mathbf{Uni}_1$	$R_1 \xrightarrow{\omega} R_1 \triangleright P,\ \omega \notin call(R_2)$	$\vdash$	$R_1 :: R_2 \xrightarrow{\omega} R_1 :: R_2 \triangleright P$
$\mathbf{Uni}_2$	$R_2 \xrightarrow{\omega} R_2 \triangleright P,\ \omega \notin call(R_1)$	$\vdash$	$R_1 :: R_2 \xrightarrow{\omega} R_1 :: R_2 \triangleright P$
$\mathbf{Uni}_3$	$R_1 \xrightarrow{\omega} R_1 \triangleright P,\ R_2 \xrightarrow{\omega} R_2 \triangleright Q,$	$\vdash$	$R_1 :: R_2 \xrightarrow{\omega} R_1 :: R_2 \triangleright (P \parallel Q)$
$\mathbf{Supp}_1$	$R \xrightarrow{\omega} R \triangleright Q,\ P \xrightarrow{\overline{\omega}} P'$	$\vdash$	$R \triangleright P \xrightarrow{\tau} R \triangleright (P' \rangle Q)$
$\mathbf{Supp}_2$	$P \xrightarrow{\alpha} P'$	$\vdash$	$R \triangleright P \xrightarrow{\alpha} R \triangleright P'$
$\mathbf{S.Choice}_j$	$R \triangleright P_j \xrightarrow{\tau} P',\ (j \in I)$	$\vdash$	$R \triangleright (\sum_{i \in I} P_i) \xrightarrow{\tau} P'$
$\mathbf{S.Com}_1$	$R \triangleright P \xrightarrow{\tau} R \triangleright P'$	$\vdash$	$R \triangleright (P \mid Q) \xrightarrow{\tau} R \triangleright (P' \mid Q)$
$\mathbf{S.Com}_2$	$R \triangleright Q \xrightarrow{\tau} R \triangleright Q'$	$\vdash$	$R \triangleright (P \mid Q) \xrightarrow{\tau} R \triangleright (P \mid Q')$
$\mathbf{S.Sync}_1$	$R \triangleright P \xrightarrow{\tau} R \triangleright P'$	$\vdash$	$R \triangleright (P \parallel Q) \xrightarrow{\tau} R \triangleright (P' \parallel Q)$
$\mathbf{S.Sync}_2$	$R \triangleright Q \xrightarrow{\tau} R \triangleright Q'$	$\vdash$	$R \triangleright (P \parallel Q) \xrightarrow{\tau} R \triangleright (P \parallel Q')$
$\mathbf{S.Subo}_1$	$R \triangleright P \xrightarrow{\tau} R \triangleright P'$	$\vdash$	$R \triangleright (P \rangle Q) \xrightarrow{\tau} R \triangleright (P' \rangle Q)$
$\mathbf{S.Subo}_2$	$R \triangleright Q \xrightarrow{\tau} R \triangleright Q'$	$\vdash$	$R \triangleright (P \rangle Q) \xrightarrow{\tau} R \triangleright (P \rangle Q')$
S.Rel	$R \triangleright P \xrightarrow{\tau} R \triangleright P'$	$\vdash$	$R \triangleright (P[f]) \xrightarrow{\tau} R \triangleright (P'[f])$
S.Res	$R \triangleright P \xrightarrow{\tau} R \triangleright P'$	$\vdash$	$R \triangleright (P \backslash L) \xrightarrow{\tau} R \triangleright (P' \backslash L)$
S.Hide	$R \triangleright P \xrightarrow{\tau} R \triangleright P'$	$\vdash$	$R \triangleright (P/L) \xrightarrow{\tau} R \triangleright (P'/L)$
S.Rec	$R \triangleright P \xrightarrow{\tau} R \triangleright P',\ A \stackrel{\text{def}}{=} P$	$\vdash$	$R \triangleright A \xrightarrow{\tau} R \triangleright P'$

図 3.4: プロセス上のラベル付遷移の推論規則 2

ことを意味しており，3.3.2小節で説明したように複数のプロセスを確実に親プロセスにすることができる．

3.5 CCSPR における等価性

CCSPR のプロセス記述を CCS のプロセス記述に展開することを主な目的として，CCS の強等価を用いる．CCS のプロセス記述に展開することによって，すでに確立されている CCS の観測等価やその判定ツールを利用できる．強等価の定義を示す．

定義 3.5.1 プロセス上の二項関係 $\mathcal{S}$ が**強双模倣** (strong bisimulation) であるとは，$(P,Q) \in \mathcal{S}$ ならば，任意の $\alpha \in Act$ について，次の 2 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' について，$Q \xrightarrow{\alpha} Q'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．

(ii) $Q \xrightarrow{\alpha} Q'$ ならば，ある P' について，$P \xrightarrow{\alpha} P'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．

∎

定義 3.5.2 もし，ある強双模倣 $\mathcal{S}$ において $(P,Q) \in \mathcal{S}$ ならば，プロセス P と Q は**強等価** (strong equivalence) であるといい，$P \sim Q$ と書く． ∎

この定義は 2.4.1小節で紹介した強等価の定義 2.4.2と同じである．CCSPR に対しても CCS と同様に，命題 2.4.1や命題 2.4.2等が成り立つ．

CCSPR では，基本的にプロセスは $R \triangleright P$ の形をしている．そこで，$R \triangleright P$ の形のプロセスの等価性が各演算子についてどのように保存されるかを明らかにしていく．まず，$R \triangleright P$ の R について強等価は保存されることを示す．すなわち，リソース R を安全に交換することができる．

命題 3.5.1 もし $R_1 \sim R_2$ ならば，$R_1 \triangleright P \sim R_2 \triangleright P$ である．

証明 次の集合 $\mathcal{S}$ が強双模倣であることを示す．

$$\mathcal{S} = \bigcup_{n \geq 0} \mathcal{S}^{(n)}$$

ここで，$\mathcal{S}^{(n)}$ は帰納的に次のように定義される $(n \geq 0)$.

$$\begin{aligned}
\mathcal{S}^{(0)} &= \mathcal{S}_1^{(0)} \cup \mathcal{S}_2^{(0)} \cup \mathcal{S}_3^{(0)} \\
\mathcal{S}_1^{(0)} &= \{(R_1 \triangleright P, R_2 \triangleright P) : R_1 \sim R_2\} \\
\mathcal{S}_2^{(0)} &= \{(R_1 \triangleright P_1, R_2 \triangleright P_2) : R_1 \sim R_2, R_1 \triangleright P_1 \sim R_2 \triangleright P_2\} \\
\mathcal{S}_3^{(0)} &= \{(R \triangleright [\![P_{11}]\!], R \triangleright [\![P_{21}]\!]) : P_{11} \sim P_{21}\}
\end{aligned}$$

$$\begin{aligned}
\mathcal{S}^{(n+1)} &= \mathcal{S}_1^{(n+1)} \cup \mathcal{S}_2^{(n+1)} \cup \mathcal{S}_3^{(n+1)} \cup \mathcal{S}_4^{(n+1)} \cup \mathcal{S}_5^{(n+1)} \cup \mathcal{S}_6^{(n+1)} \\
\mathcal{S}_1^{(n+1)} &= \{(R_1 \triangleright (P_{11}|P_{12}), R_2 \triangleright (P_{21}|P_{22})) : \forall i \in \{1,2\}.\ (R_1 \triangleright P_{1i}, R_2 \triangleright P_{2i}) \in \mathcal{S}^{(n)}\} \\
\mathcal{S}_2^{(n+1)} &= \{(R_1 \triangleright (P_{11} \| P_{12}), R_2 \triangleright (P_{21} \| P_{22})) : \forall i \in \{1,2\}.\ (R_1 \triangleright P_{1i}, R_2 \triangleright P_{2i}) \in \mathcal{S}^{(n)}\} \\
\mathcal{S}_3^{(n+1)} &= \{(R_1 \triangleright (P_{11} \rangle P_{12}), R_2 \triangleright (P_{21} \rangle P_{22})) : \forall i \in \{1,2\}.\ (R_1 \triangleright P_{1i}, R_2 \triangleright P_{2i}) \in \mathcal{S}^{(n)}\} \\
\mathcal{S}_4^{(n+1)} &= \{(R_1 \triangleright (P_{11}[f]), R_2 \triangleright (P_{21}[f])) : (R_1 \triangleright P_{11}, R_2 \triangleright P_{21}) \in \mathcal{S}^{(n)}\} \\
\mathcal{S}_5^{(n+1)} &= \{(R_1 \triangleright (P_{11} \backslash L), R_2 \triangleright (P_{21} \backslash L)) : (R_1 \triangleright P_{11}, R_2 \triangleright P_{21}) \in \mathcal{S}^{(n)}\} \\
\mathcal{S}_6^{(n+1)} &= \{(R_1 \triangleright (P_{11}/L), R_2 \triangleright (P_{21}/L)) : (R_1 \triangleright P_{11}, R_2 \triangleright P_{21}) \in \mathcal{S}^{(n)}\}
\end{aligned}$$

本命題を証明するためには $\mathcal{S}_1^{(0)}$ が重要であるが，$\mathcal{S}_1^{(0)}$ だけでは強双模倣にならない．リソースからのプロセス呼出を考慮すると，上記のような集合 $\mathcal{S}$ が必要である．ただし，$\mathcal{S}_3^{(0)}$ は本命題を証明するためではなく，後の命題 3.5.3を証明するために加えられている．本命題と命題 3.5.3の証明では共通部分が多いため，本証明で両方を証明できる強双模倣を与えている．

以下，各 $(R_1 \rhd P_1, R_2 \rhd P_2) \in \mathcal{S}^{(n')}$ について，定義 3.5.1の (i) の条件が満たされることを，n' に関する帰納法を用いて証明する ((ii) は (i) に対称である)．

- $n' = 0$ の場合 $(R_1 \rhd P_1, R_2 \rhd P_2) \in \mathcal{S}^{(0)}$：すなわち，ある $k \in \{1,2,3\}$ について，$(R_1 \rhd P_1, R_2 \rhd P_2) \in \mathcal{S}_k^{(0)}$ である．各場合について示す．

  - $(R_1 \rhd P_1, R_2 \rhd P_2) \in \mathcal{S}_1^{(0)}$ の場合：すなわち，$R_1 \sim R_2$ かつ $P_1 \equiv P_2$ である．

    (i) $R_1 \rhd P_1 \xrightarrow{\alpha} R_1 \rhd P_1'$ とする ($R_1 \rhd P_1 \xrightarrow{\alpha} P_1''$ ならば，P_1'' は必ずある P_1' で $P_1'' \equiv R_1 \rhd P_1'$ の形をもつ)．この遷移を導いた推論規則の数に関する帰納法を用いて，ある P_2' について $R_2 \rhd P_2 \equiv R_2 \rhd P_1 \xrightarrow{\alpha} R_2 \rhd P_2'$ かつ $(R_1 \rhd P_1', R_2 \rhd P_2') \in \mathcal{S}$ となることを証明する．$R_1 \rhd P_1 \xrightarrow{\alpha} R_1 \rhd P_1'$ を導いた最後の規則について次のように場合分けする．

    1. $\mathbf{Supp}_1$ による場合：ある Q_1, P_1'', ω について，$R_1 \xrightarrow{\omega} R_1 \rhd Q_1$ かつ $P_1 \xrightarrow{\overline{\omega}} P_1''$ かつ $\alpha = \tau$ かつ $P_1' \equiv P_1'' \rangle Q_1$ である．仮定 $R_1 \sim R_2$ より，ある Q_2 について $R_2 \xrightarrow{\omega} R_2 \rhd Q_2$ かつ $R_1 \rhd Q_1 \sim R_2 \rhd Q_2$ を得る．すなわち，$\mathbf{Supp}_1$ により $R_2 \rhd P_1 \xrightarrow{\tau} R_2 \rhd (P_1'' \rangle Q_2)$ を導ける．また，$(R_1 \rhd P_1'', R_2 \rhd P_1'') \in \mathcal{S}_1^{(0)}$ かつ $(R_1 \rhd Q_1, R_2 \rhd Q_2) \in \mathcal{S}_2^{(0)}$ より，$(R_1 \rhd (P_1'' \rangle Q_1), R_2 \rhd (P_1'' \rangle Q_2)) \in \mathcal{S}_3^{(1)}$ である．
    2. $\mathbf{Supp}_2$ による場合：$P_1 \xrightarrow{\alpha} P_1'$ である．すなわち，$\mathbf{Supp}_2$ により，$R_2 \rhd P_1 \xrightarrow{\alpha} R_2 \rhd P_1'$ を導ける．また，$(R_1 \rhd P_1', R_2 \rhd P_1') \in \mathcal{S}_1^{(0)}$ である．
    3. $\mathbf{S.Choice}_j$ による場合：ある $j \in I$ と P_{1i} $(i \in I)$ について $R_1 \rhd P_{1j} \xrightarrow{\tau} R_1 \rhd P_1'$ かつ $P_1 \equiv \sum_{i \in I} P_{1i}$ かつ $\alpha = \tau$ である．ここで，推論規則の数に関する帰納法の仮定より，ある P_2' について $R_2 \rhd P_{1j} \xrightarrow{\alpha} R_2 \rhd P_2'$ かつ $(R_1 \rhd P_1', R_2 \rhd P_2') \in \mathcal{S}$ を得る．さらに，$j \in I$ であるので，$\mathbf{S.Choice}_j$ より $R_2 \rhd P_1 \equiv R_2 \rhd (\sum_{i \in I} P_{1i}) \xrightarrow{\tau} R_2 \rhd P_2'$ を導ける．
    4. $\mathbf{S.Com}_1$ による場合：ある P_{11}, P_{11}', P_{12} について，$R_1 \rhd P_{11} \xrightarrow{\tau} R_1 \rhd P_{11}'$ かつ $P_1 \equiv P_{11} | P_{12}$ かつ $P_1' \equiv P_{11}' | P_{12}$ かつ $\alpha = \tau$ である．推論規則の数

に関する帰納法の仮定より，ある P'_{21} について $R_2 \triangleright P_{11} \xrightarrow{\tau} R_2 \triangleright P'_{21}$ かつ $(R_1 \triangleright P'_{11}, R_2 \triangleright P'_{21}) \in \mathcal{S}$ を得る．また，$(R_1 \triangleright P_{12}, R_2 \triangleright P_{12}) \in \mathcal{S}_1^{(0)} \subseteq \mathcal{S}^{(0)}$ でもあるので，$(R_1 \triangleright (P'_{11}|P_{12}), R_2 \triangleright (P'_{21}|P_{12})) \in \mathcal{S}$ を得る．さらに，$R_2 \triangleright P_{11} \xrightarrow{\tau} R_2 \triangleright P'_{21}$ から，**S.Com**$_1$ より $R_2 \triangleright P_1 \equiv R_2 \triangleright (P_{11}|P_{12}) \xrightarrow{\tau} R_2 \triangleright (P'_{21}|P_{12})$ を導ける．

5. 他の規則 **S.Com**$_2$, **S.Sync**$_{1,2}$, **S.Subo**$_{1,2}$, **S.Res**, **S.Hide**, **S.Rec** による場合も **S.Com**$_1$ による場合と同様に証明できる．

– $(R_1 \triangleright P_1, R_2 \triangleright P_2) \in \mathcal{S}_2^{(0)}$ の場合：すなわち，$R_1 \sim R_2$ かつ $R_1 \triangleright P_1 \sim R_2 \triangleright P_2$ である．

(i) $R_1 \triangleright P_1 \xrightarrow{\alpha} R_1 \triangleright P'_1$ とする．仮定より，$R_1 \triangleright P_1 \sim R_2 \triangleright P_2$ であるので，ある P'_2 ついて，$R_2 \triangleright P_2 \xrightarrow{\alpha} R_2 \triangleright P'_2$ かつ $R_1 \triangleright P'_1 \sim R_2 \triangleright P'_2$ を得る．すなわち，$(R_1 \triangleright P'_1, R_2 \triangleright P'_2) \in \mathcal{S}_2^{(0)} \subseteq \mathcal{S}$ である．

– $(R_1 \triangleright P_1, R_2 \triangleright P_2) \in \mathcal{S}_3^{(0)}$ の場合：すなわち，$R \equiv R_1 \equiv R_2$ かつ $P_{11} \sim P_{21}$ かつ $P_1 \equiv [\![P_{11}]\!]$ かつ $P_2 \equiv [\![P_{21}]\!]$ である．

(i) $R \triangleright P_1 \equiv R \triangleright [\![P_{11}]\!] \xrightarrow{\alpha} R \triangleright P'_1$ とする．この遷移を導いた最後の規則は **Supp**$_1$ か **Supp**$_2$ である．

1. **Supp**$_1$ による場合：ある Q, P''_1, ω について，$R \xrightarrow{\omega} R \triangleright Q$ かつ $[\![P_{11}]\!] \xrightarrow{\overline{\omega}} P''_1$ かつ $\alpha = \tau$ かつ $P'_1 \equiv P''_1 \rangle Q$ である．ここで，$[\![P_{11}]\!] \xrightarrow{\overline{\omega}} P''_1$ を導く最後の規則は **Pack** のみであるので，ある P'_{11} について，$P_{11} \xrightarrow{\overline{\omega}} P'_{11}$ かつ $P''_1 \equiv [\![P'_{11}]\!]$ を得る．すなわち，$P'_1 \equiv [\![P'_{11}]\!] \rangle Q$ である．ここで，仮定 $P_{11} \sim P_{21}$ より，ある P'_{21} について，$P_{21} \xrightarrow{\overline{\omega}} P'_{21}$ かつ $P'_{11} \sim P'_{21}$ を得る．この遷移から，**Pack** により $[\![P_{21}]\!] \xrightarrow{\overline{\omega}} [\![P'_{21}]\!]$ を導ける．さらに，$R \xrightarrow{\omega} R \triangleright Q$ であるので，**Supp**$_1$ より $R \triangleright [\![P_{21}]\!] \xrightarrow{\tau} R \triangleright ([\![P'_{21}]\!] \rangle Q)$ を導ける．また，$P'_{11} \sim P'_{21}$ であるので，$(R \triangleright [\![P'_{11}]\!], R \triangleright [\![P'_{21}]\!]) \in \mathcal{S}_3^{(0)}$ である．すなわち，$(R \triangleright Q, R \triangleright Q) \in \mathcal{S}_1^{(0)}$ であるので，$(R \triangleright ([\![P'_{11}]\!] \rangle Q), R \triangleright ([\![P'_{21}]\!] \rangle Q)) \in \mathcal{S}_3^{(1)}$ を得る．

2. **Supp**$_2$ による場合：$[\![P_{11}]\!] \xrightarrow{\alpha} P'_1$ である．この遷移を導く最後の規則は **Pack** のみであるので，ある P'_{11} について，$P_{11} \xrightarrow{\alpha} P'_{11}$ かつ $P'_1 \equiv [\![P'_{11}]\!]$ を得る．ここで，仮定 $P_{11} \sim P_{21}$ より，ある P'_{21} について，$P_{21} \xrightarrow{\alpha} P'_{21}$ かつ $P'_{11} \sim P'_{21}$ を得る．この遷移から，**Pack** により $[\![P_{21}]\!] \xrightarrow{\alpha} [\![P'_{21}]\!]$

を導ける．さらに，$\mathbf{Supp}_2$ より $R \rhd [\![P_{21}]\!] \xrightarrow{\alpha} R \rhd [\![P'_{21}]\!]$ を導く．また，$P'_{11} \sim P'_{21}$ より $(R \rhd [\![P'_{11}]\!], R \rhd [\![P'_{21}]\!]) \in \mathcal{S}_3^{(0)} \subseteq \mathcal{S}$ である．

- $n' = n+1$ の場合 $(R_1 \rhd P_1, R_2 \rhd P_2) \in \mathcal{S}^{(n+1)}$：すなわち，ある $k \in \{1, \cdots, 6\}$ について，$(R_1 \rhd P_1, R_2 \rhd P_2) \in \mathcal{S}_k^{(n+1)}$ である．ここでは，$k = 1$ の場合のみ示す．他の場合も同様に示せる．すなわち，各 $i \in \{1, 2\}$ について，$P_i \equiv P_{i1}|P_{i2}$ かつ $(R_1 \rhd P_{1i}, R_2 \rhd P_{2i}) \in \mathcal{S}^{(n)}$ となる P_{i1} と P_{i2} が存在する．このとき，必ず $R_1 \sim R_2$ である．

  (i) $R_1 \rhd P_1 \equiv R_1 \rhd (P_{11}|P_{12}) \xrightarrow{\alpha} R_1 \rhd P'_1$ とする．この遷移を導く最後の規則は $\mathbf{Supp}_1$ か $\mathbf{Supp}_2$ か $\mathbf{S.Com}_1$ か $\mathbf{S.Com}_2$ である．

  1. $\mathbf{Supp}_1$ による場合:ある Q_1, P''_1, ω について，$R_1 \xrightarrow{\omega} R_1 \rhd Q_1$ かつ $P_{11}|P_{12} \xrightarrow{\overline{\omega}} P''_1$ かつ $\alpha = \tau$ かつ $P'_1 \equiv P''_1 \rangle Q_1$ である．仮定 $R_1 \sim R_2$ より，ある Q_2 について，$R_2 \xrightarrow{\overline{\omega}} R_2 \rhd Q_2$ かつ $R_1 \rhd Q_1 \sim R_2 \rhd Q_2$ を得る．すなわち，$(R_1 \rhd Q_1, R_2 \rhd Q_2) \in \mathcal{S}^{(0)}$ である．一方，$P_{11}|P_{12} \xrightarrow{\overline{\omega}} P''_1$ は $\mathbf{Com}_1$ か $\mathbf{Com}_2$ によって導かれるが，これらは対称であるので $\mathbf{Com}_1$ による場合を示す．すなわち，ある P'_{11} について，$P_{11} \xrightarrow{\overline{\omega}} P'_{11}$ かつ $P''_1 \equiv P'_{11}|P_{12}$ を得る．さらに，$\mathbf{Supp}_2$ より $R_1 \rhd P_{11} \xrightarrow{\overline{\omega}} R_1 \rhd P'_{11}$ を導ける．ここで，$(R_1 \rhd P_{11}, R_2 \rhd P_{21}) \in \mathcal{S}^{(n)}$ であるので，n' に関する帰納法の仮定より，ある P'_{21} について，$R_2 \rhd P_{21} \xrightarrow{\overline{\omega}} R_2 \rhd P'_{21}$ かつ $(R_1 \rhd P'_{11}, R_2 \rhd P'_{21}) \in \mathcal{S}$ を得る．このとき，$R_2 \rhd P_{21} \xrightarrow{\overline{\omega}} R_2 \rhd P'_{21}$ を導く規則は，$\overline{\omega} \neq \tau$ であるので $\mathbf{Supp}_2$ のみである．すなわち，$P_{21} \xrightarrow{\overline{\omega}} P'_{21}$ でなければならない．この遷移から，$\mathbf{Com}_1$ より $P_{21}|P_{22} \xrightarrow{\overline{\omega}} P'_{21}|P_{22}$ を導ける．さらに，$P'_2 \equiv (P'_{21}|P_{22}) \rangle Q_2$ とおくと，$R_2 \xrightarrow{\omega} R_2 \rhd Q_2$ であるので，$\mathbf{Supp}_1$ より $R_2 \rhd P_2 \equiv R_2 \rhd (P_{21}|P_{22}) \xrightarrow{\tau} R_2 \rhd P'_2$ を導ける．また，$(R_1 \rhd P'_{11}, R_2 \rhd P'_{21}) \in \mathcal{S}$ かつ $(R_1 \rhd P_{12}, R_2 \rhd P_{22}) \in \mathcal{S}$ であるので $(R_1 \rhd (P'_{11}|P_{12}), R_2 \rhd (P'_{21}|P_{22})) \in \mathcal{S}$ を得る．さらに $(R_1 \rhd Q_1, R_2 \rhd Q_2) \in \mathcal{S}^{(0)}$ であるので，$(R_1 \rhd P'_1, R_2 \rhd P'_2) \equiv (R_1 \rhd ((P'_{11}|P_{12}) \rangle Q_1), R_2 \rhd ((P'_{21}|P_{22}) \rangle Q_2)) \in \mathcal{S}$ を得る．
  2. $\mathbf{Supp}_2$ による場合：$P_{11}|P_{12} \xrightarrow{\alpha} P'_1$ である．この遷移を導く最後の規則 $\mathbf{Com}_{1,2,3}$ について場合分けする．

     (1) $\mathbf{Com}_1$ による場合：ある P'_{11} について $P_{11} \xrightarrow{\alpha} P'_{11}$ かつ $P'_1 \equiv P'_{11}|P_{12}$ である．さらに，$\mathbf{Supp}_2$ より，$R_1 \rhd P_{11} \xrightarrow{\alpha} R_1 \rhd P'_{11}$ を導ける．ここで，

$(R_1 \triangleright P_{11}, R_2 \triangleright P_{21}) \in \mathcal{S}^{(n)}$ であるので，n' に関する帰納法の仮定より，ある P'_{21} について，$R_2 \triangleright P_{21} \xrightarrow{\alpha} R_2 \triangleright P'_{21}$ かつ $(R_1 \triangleright P'_{11}, R_2 \triangleright P'_{21}) \in \mathcal{S}$ を得る．

- $\alpha \neq \tau$ の場合：$R_2 \triangleright P_{21} \xrightarrow{\alpha} R_2 \triangleright P'_{21}$ を導く最後の規則は $\mathbf{Supp}_2$ のみである．すなわち，$P_{21} \xrightarrow{\alpha} P'_{21}$ を得る．これは，$\mathbf{Com}_1$ より，$P_{21}|P_{22} \xrightarrow{\alpha} P'_{21}|P_{22}$ を導く．さらに，$P'_2 \equiv P'_{21}|P_{22}$ とおくと，$\mathbf{Supp}_2$ より $R_2 \triangleright P_2 \equiv R_2 \triangleright (P_{21}|P_{22}) \xrightarrow{\alpha} R_2 \triangleright P'_2$ を導ける．
- $\alpha = \tau$ の場合：$R_2 \triangleright P_{21} \xrightarrow{\tau} R_2 \triangleright P'_{21}$ であるので，$P'_2 \equiv P'_{21}|P_{22}$ とおくと，$\mathbf{S}.\mathbf{Com}_1$ より $R_2 \triangleright P_2 \equiv R_2 \triangleright (P_{21}|P_{22}) \xrightarrow{\tau} R_2 \triangleright P'_2$ を導ける．

すなわち，どちらの場合でも，$R_2 \triangleright P_2 \xrightarrow{\alpha} R_2 \triangleright P'_2 \equiv R_2 \triangleright (P'_{21}|P_{22})$ が得られる．また，$(R_1 \triangleright P'_{11}, R_2 \triangleright P'_{21}) \in \mathcal{S}$ かつ $(R_1 \triangleright P_{12}, R_2 \triangleright P_{22}) \in \mathcal{S}$ より，$(R_1 \triangleright P'_1, R_2 \triangleright P'_2) \equiv (R_1 \triangleright (P'_{11}|P_{12}), R_2 \triangleright (P'_{21}|P_{22})) \in \mathcal{S}$ を得る．

(2) $\mathbf{Com}_2$ による場合：$\mathbf{Com}_1$ の場合に対称である．

(3) $\mathbf{Com}_3$ による場合：ある P'_{11}, P'_{12}, ω について，$P_{11} \xrightarrow{\omega} P'_{11}$ かつ $P_{12} \xrightarrow{\overline{\omega}} P'_{12}$ かつ $\alpha = \tau$ かつ $P'_1 \equiv P'_{11}|P'_{12}$ である．さらに，これらの遷移から，$\mathbf{Supp}_2$ より，$R_1 \triangleright P_{11} \xrightarrow{\omega} R_1 \triangleright P'_{11}$ かつ $R_1 \triangleright P_{12} \xrightarrow{\overline{\omega}} R_1 \triangleright P'_{12}$ を導ける．ここで，$(R_1 \triangleright P_{11}, R_2 \triangleright P_{21}) \in \mathcal{S}^{(n)}$ であるので，n' に関する帰納法の仮定より，ある P'_{21} について，$R_2 \triangleright P_{21} \xrightarrow{\omega} R_2 \triangleright P'_{21}$ かつ $(R_1 \triangleright P'_{11}, R_2 \triangleright P'_{21}) \in \mathcal{S}$ を得る．同様に，$(R_1 \triangleright P_{12}, R_2 \triangleright P_{22}) \in \mathcal{S}^{(n)}$ より，ある P'_{22} について，$R_2 \triangleright P_{22} \xrightarrow{\overline{\omega}} R_2 \triangleright P'_{22}$ かつ $(R_1 \triangleright P'_{12}, R_2 \triangleright P'_{22}) \in \mathcal{S}$ を得る．$\omega \neq \tau$ であるので，$R_2 \triangleright P_{21} \xrightarrow{\omega} R_2 \triangleright P'_{21}$ と $R_2 \triangleright P_{22} \xrightarrow{\overline{\omega}} R_2 \triangleright P'_{22}$ を導く最後の規則は $\mathbf{Supp}_2$ のみである．すなわち，$P_{21} \xrightarrow{\omega} P'_{21}$ かつ $P_{22} \xrightarrow{\overline{\omega}} P'_{22}$ でなければならない．これらの遷移から，$\mathbf{Com}_2$ より，$P_{21}|P_{22} \xrightarrow{\tau} P'_{21}|P'_{22}$ を導ける．さらに，$P'_2 \equiv P'_{21}|P'_{22}$ とおくと，$\mathbf{Supp}_2$ より $R_2 \triangleright P_2 \equiv R_2 \triangleright (P_{21}|P_{22}) \xrightarrow{\tau} R_2 \triangleright P'_2$ を導ける．また，$(R_1 \triangleright P'_{11}, R_2 \triangleright P'_{21}) \in \mathcal{S}$ かつ $(R_1 \triangleright P'_{12}, R_2 \triangleright P'_{22}) \in \mathcal{S}$ であるので，$(R_1 \triangleright P'_1, R_2 \triangleright P'_2) \equiv (R_1 \triangleright (P'_{11}|P'_{12}), R_2 \triangleright (P'_{21}|P'_{22})) \in \mathcal{S}$ を得る．

3. $\mathbf{S}.\mathbf{Com}_1$ による場合：ある P'_{11} について，$R_1 \triangleright P_{11} \xrightarrow{\tau} R \triangleright P'_{11}$ かつ $P'_1 \equiv P'_{11}|P_{12}$ かつ $\alpha = \tau$ である．すなわち，$(R_1 \triangleright P_{11}, R_2 \triangleright P_{21}) \in \mathcal{S}^{(n)}$ であるので，n' に関する帰納法の仮定より，ある P'_{21} について，$R_2 \triangleright P_{21} \xrightarrow{\tau} R_2 \triangleright P'_{21}$ かつ $(R_1 \triangleright P'_{11}, R_2 \triangleright P'_{21}) \in \mathcal{S}$ を得る．このとき，$P'_2 \equiv P'_{21}|P_{22}$ と

おくと，$\mathbf{S}.\mathbf{Com}_1$ より，$R_2 \rhd P_2 \equiv R_2 \rhd (P_{21}|P_{22}) \xrightarrow{\tau} R_2 \rhd P_2'$ を導ける．また，$(R_1 \rhd P_{11}', R_2 \rhd P_{21}') \in \mathcal{S}$ かつ $(R_1 \rhd P_{12}, R_2 \rhd P_{22}) \in \mathcal{S}$ であるので，$(R_1 \rhd P_1', R_2 \rhd P_2') \equiv (R_1 \rhd (P_{11}'|P_{12}), R_2 \rhd (P_{21}'|P_{22})) \in \mathcal{S}$ を得る．

4. $\mathbf{S}.\mathbf{Com}_2$ による場合：$\mathbf{S}.\mathbf{Com}_1$ の場合に対称である．

∎

一方，$R \rhd P$ のプロセス P については強等価は保存されるない．例えば，$(\overline{a}.\mathbf{0})\backslash\overline{a} \sim \mathbf{0}$ であるが，$\{\!\{a.b.\mathbf{0}\}\!\} \rhd (\overline{a}.\mathbf{0})\backslash\overline{a} \not\sim \{\!\{a.\mathbf{0}\}\!\} \rhd \mathbf{0}$ である．これは，$\{\!\{a.b.\mathbf{0}\}\!\} \rhd (\overline{a}.\mathbf{0})\backslash\overline{a}$ が τb 遷移をもつためである．すなわち，強等価 $\sim$ は CCSPR において合同関係ではない．そこで，プロセス P の構造に着目して次の命題を与える．

命題 3.5.2 各 $i \in \{1,2\} \cup I$ について，$R \rhd P_{1i} \sim R \rhd P_{2i}$ とする．このとき，次の等式が成り立つ．

(1) $R \rhd (\alpha.P_{11}) \sim R \rhd (\alpha.P_{21})$
(2) $R \rhd (\sum_{i\in I} P_{1i}) \sim R \rhd (\sum_{i\in I} P_{2i})$
(3) $R \rhd (P_{11}|P_{12}) \sim R \rhd (P_{21}|P_{22})$
(4) $R \rhd (P_{11} \| P_{12}) \sim R \rhd (P_{21} \| P_{22})$
(5) $R \rhd (P_{11}\rangle P_{12}) \sim R \rhd (P_{21}\rangle P_{22})$
(6) $R \rhd (P_{11}[f]) \sim R \rhd (P_{21}[f])$
(7) $R \rhd (P_{11}\backslash L) \sim R \rhd (P_{21}\backslash L)$
(8) $R \rhd (P_{11}/L) \sim R \rhd (P_{21}/L)$

証明 (3) から (8) については，命題 3.5.1 の証明に用いた集合 $\mathcal{S}$ を次のように利用できる：各 $i \in \{1,2\} \cup I$ について，$R \rhd P_{1i} \sim R \rhd P_{2i}$ であるので，$(R \rhd P_{1i}, R \rhd P_{2i}) \in \mathcal{S}^{(0)}$ である．すなわち，例えば並行合成の (3) については，$(R \rhd (P_{11}|P_{12}), R \rhd (P_{21}|P_{22})) \in \mathcal{S}_1^{(1)}$ である．$\mathcal{S}$ は強双模倣であるので，$R \rhd (P_{11}|P_{12}) \sim R \rhd (P_{21}|P_{22})$ を得る．他の場合 $(4), \cdots, (8)$ についても同様である．以下，(1) と (2) について，命題 2.4.1 の条件 (i) が満たされることを示す ((ii) については対称である)．

(1) $R \rhd (\alpha.P_{11}) \xrightarrow{\alpha'} R \rhd P_1'$ とする．この遷移を導く最後の規則は $\mathbf{Supp}_1$ か $\mathbf{Supp}_2$ である．各場合について証明する．

- $\mathbf{Supp}_1$ による場合：ある Q, P_1'', ω について，$R \xrightarrow{\omega} R \rhd Q$ かつ $\alpha.P_{11} \xrightarrow{\overline{\omega}} P_1''$ かつ $\alpha' = \tau$ かつ $P_1' \equiv P_1''\rangle Q$ である．$\alpha.P_{11} \xrightarrow{\overline{\omega}} P_1''$ であるので，規則 **Act** により $\alpha = \overline{\omega}$ かつ $P_{11} \equiv P_1''$ である．すなわち，**Act** により $\alpha.P_{12} \xrightarrow{\overline{\omega}} P_{12}$ を導ける．ここで，$R \xrightarrow{\omega} R \rhd Q$ であるので，$\mathbf{Supp}_1$ より $R \rhd (\alpha.P_{12}) \xrightarrow{\tau} R \rhd (P_{12}\rangle Q)$ を導ける．一方，$R \rhd P_{11} \sim R \rhd P_{21}$ であるので，本命題 3.5.2(5) より，$R \rhd P_1' \equiv R \rhd (P_{11}\rangle Q) \sim R \rhd (P_{21}\rangle Q)$ を得る．

- **Supp**$_2$ による場合：$\alpha.P_{11} \xrightarrow{\alpha'} P_1'$ である．すなわち，規則 **Act** により $\alpha = \alpha'$ かつ $P_{11} \equiv P_1'$ である．ここで，規則 **Act** により，$\alpha.P_{21} \xrightarrow{\alpha'} P_{21}$ を導ける．さらに，**Supp**$_2$ より $R \rhd (\alpha.P_{21}) \xrightarrow{\alpha'} R \rhd P_{21}$ を導く．一方，仮定より $R \rhd P_1' \equiv R \rhd P_{11} \sim R \rhd P_{21}$ である．

(2) $R \rhd (\sum_{i \in I} P_{1i}) \xrightarrow{\alpha} R \rhd P_1'$ とする．この遷移を導く最後の規則は **Supp**$_1$ か **Supp**$_2$ か **S.Choice** である．各場合について証明する．

- **Supp**$_1$ による場合：ある Q, P_1'', ω について，$R \xrightarrow{\omega} R \rhd Q$ かつ $\sum_{i \in I} P_{1i} \xrightarrow{\overline{\omega}} P_1''$ かつ $\alpha = \tau$ かつ $P_1' \equiv P_1'' \rangle Q$ である．$\sum_{i \in I} P_{1i} \xrightarrow{\overline{\omega}} P_1''$ であるので，規則 **Choice** により，ある $j \in I$ について，$P_{1j} \xrightarrow{\overline{\omega}} P_1''$ を得る．この遷移から **Supp**$_2$ により $R \rhd P_{1j} \xrightarrow{\overline{\omega}} R \rhd P_1''$ を導ける．ここで，仮定 $R \rhd P_{1j} \sim R \rhd P_{2j}$ より，ある P_2'' について，$R \rhd P_{2j} \xrightarrow{\overline{\omega}} R \rhd P_2''$ かつ $R \rhd P_1'' \sim R \rhd P_2''$ を得る．このとき，$\overline{\omega} \neq \tau$ であるので，$R \rhd P_{2j} \xrightarrow{\overline{\omega}} R \rhd P_2''$ を導く最後の規則は **Supp**$_2$ のみである．すなわち，$P_{2j} \xrightarrow{\overline{\omega}} P_2''$ を得る．よって，$j \in I$ であるので，**Choice**$_j$ により $\sum_{i \in I} P_{2i} \xrightarrow{\overline{\omega}} P_2''$ を導ける．さらに，$R \xrightarrow{\omega} R \rhd Q$ であるので，**Supp**$_1$ より $R \rhd (\sum_{i \in I} P_{2i}) \xrightarrow{\tau} R \rhd (P_2'' \rangle Q)$ を導く．一方，$R \rhd P_1'' \sim R \rhd P_2''$ であるので，本命題 3.5.2(5) より，$R \rhd P_1' \equiv R \rhd (P_1'' \rangle Q) \sim R \rhd (P_2'' \rangle Q)$ を得る．
- **Supp**$_2$ による場合：$\sum_{i \in I} P_{1i} \xrightarrow{\alpha} P_1'$ である．この遷移を導く最後の規則は **Choice** であるので，ある $j \in I$ について，$P_{1j} \xrightarrow{\alpha} P_1'$ を得る．この遷移は **Supp**$_2$ により $R \rhd P_{1j} \xrightarrow{\alpha} R \rhd P_1'$ を導く．ここで，仮定 $R \rhd P_{1j} \sim R \rhd P_{2j}$ より，ある P_2' で $R \rhd P_{2j} \xrightarrow{\alpha} R \rhd P_2'$ かつ $R \rhd P_1' \sim R \rhd P_2'$ を得る．
  - $\alpha \neq \tau$ の場合：このとき，$R \rhd P_{2j} \xrightarrow{\alpha} R \rhd P_2'$ を導く最後の規則は **Supp**$_2$ のみである．すなわち，$P_{2j} \xrightarrow{\alpha} P_2'$ を得る．よって，$j \in I$ であるので，**Choice**$_j$ により $\sum_{i \in I} P_{2i} \xrightarrow{\alpha} P_2'$ を導ける．さらに，**Supp**$_2$ より $R \rhd (\sum_{i \in I} P_{2i}) \xrightarrow{\alpha} R \rhd P_2'$ を導く．
  - $\alpha = \tau$ の場合：このとき，$R \rhd P_{2j} \xrightarrow{\tau} R \rhd P_2'$ かつ $j \in I$ であるので，**S.Choice**$_j$ より $R \rhd (\sum_{i \in I} P_{2i}) \xrightarrow{\tau} R \rhd P_2'$ を導ける．
- **S.Choice**$_j$ による場合：ある $j \in I$ について，$R \rhd P_{1j} \xrightarrow{\tau} R \rhd P_1'$ かつ $\alpha = \tau$ である．すなわち，仮定 $R \rhd P_{1j} \sim R \rhd P_{2j}$ より，ある P_2' について

$R \triangleright P_{2j} \xrightarrow{\tau} R \triangleright P_2'$ かつ $R \triangleright P_1' \sim R \triangleright P_2'$ を得る．さらに，$j \in I$ であるので，$\mathbf{S.Choice}_j$ より $R \triangleright (\sum_{i \in I} P_{2i}) \xrightarrow{\tau} R \triangleright P_2'$ を導ける．

∎

一方，パック $[\![\ ]\!]$ については命題 3.5.2 の演算子と異なり次の等式が成り立つ．

命題 3.5.3 もし $P \sim Q$ ならば，$R \triangleright [\![P]\!] \sim R \triangleright [\![Q]\!]$ である．

証明 命題 3.5.1の証明に用いた集合 $\mathcal{S}$ について，$P \sim Q$ ならば $(R \triangleright [\![P]\!], R \triangleright [\![Q]\!]) \in \mathcal{S}_3^{(0)} \subseteq \mathcal{S}$ である．ここで，$\mathcal{S}$ は強双模倣であるので，$R \triangleright [\![P]\!] \sim R \triangleright [\![Q]\!]$ を得る． ∎

上記の命題によって，部分的なプロセスの等式から，より大きなプロセスの等式を得ることができる．例えば，$R \triangleright (P)\mathbf{I}) \sim R \triangleright P$ は簡単に証明できるので，命題 3.5.2(4) より，$R \triangleright ((P)\mathbf{I}) \,\|\, Q) \sim R \triangleright (P \,\|\, Q)$ を得ることができる．

次に，リソース R がどのようにプロセスを供給するかを明確にするために命題 3.5.4 を与える．そして，$R \triangleright P$ の形のプロセスの振舞いを明確にするために，その形のプロセスを逐次的なプロセスに展開する規則を命題 3.5.5と命題 3.5.6に与える．この展開規則は P の構造に対して帰納的に与えられる．

命題 3.5.4 $R \in \mathcal{R}_{ccspr}$ とする．このとき，次の関係が成り立つ．

$$R \xrightarrow{\omega} P' \quad \Longleftrightarrow \quad P' \equiv R \triangleright \mathtt{CP}(\omega, R) \text{ かつ } \omega \in call(R)$$

ここで，関数 $\mathtt{CP} : \mathcal{R}_{ccspr} \to \mathcal{P}_{ccspr}$ は，リソース R から ω によって呼び出されるプロセスを取り出すための関数であり，次のように帰納的に定義される．

$$\mathtt{CP}(\omega, \{\!\{\omega'.Q\}\!\}) \equiv \begin{cases} Q & (\omega = \omega') \\ \mathbf{I} & (\omega \neq \omega') \end{cases}$$

$$\mathtt{CP}(\omega, R_1 :: R_2) \equiv \begin{cases} \mathtt{CP}(\omega, R_1) \,\|\, \mathtt{CP}(\omega, R_2) & (\omega \in call(R_1), \omega \in call(R_2)) \\ \mathtt{CP}(\omega, R_1) & (\omega \in call(R_1), \omega \notin call(R_2)) \\ \mathtt{CP}(\omega, R_2) & (\omega \notin call(R_1), \omega \in call(R_2)) \\ \mathbf{I} & (\omega \notin call(R_1), \omega \notin call(R_2)) \end{cases}$$

証明 $(\Rightarrow)$ の場合：$R \xrightarrow{\omega} P'$ を導いた推論規則の数に関する帰納法を用いる．以下，$R \xrightarrow{\omega} P'$ を導いた最後の規則について場合分けする．

1. **Reso** による場合：ある Q について，$R \equiv \{\!\{\omega.Q\}\!\}$ かつ $P' \equiv \{\!\{\omega.Q\}\!\} \triangleright Q$ である．よって，$\omega \in call(\{\!\{\omega.Q\}\!\}) = call(R)$ かつ $P' \equiv \{\!\{\omega.Q\}\!\} \triangleright Q \equiv R \triangleright Q \equiv R \triangleright \mathtt{CP}(\omega, \{\!\{\omega.Q\}\!\}) \equiv R \triangleright \mathtt{CP}(\omega, R)$ を得る．

2. $\mathbf{Uni}_1$ による場合：ある R_1, R_2, Q について，$R \equiv R_1 :: R_2$ かつ $R_1 \xrightarrow{\omega} R_1 \triangleright Q$ かつ $\omega \notin call(R_2)$ かつ $P' \equiv R \triangleright Q$ である．帰納法の仮定より，$R_1 \triangleright Q \equiv R_1 \triangleright \mathtt{CP}(\omega, R_1)$ かつ $\omega \in call(R_1)$ を得る．すなわち，$Q \equiv \mathtt{CP}(\omega, R_1)$ である．ここで，$\omega \in call(R_1)$ かつ $\omega \notin call(R_2)$ より，$\mathtt{CP}(\omega, R_1 :: R_2) \equiv \mathtt{CP}(\omega, R_1) \equiv Q$ である．ゆえに，$\omega \in call(R_1) \subseteq call(R_1) \cup call(R_2) = call(R)$ かつ $P' \equiv R \triangleright Q \equiv R \triangleright \mathtt{CP}(\omega, R_1 :: R_2) \equiv R \triangleright \mathtt{CP}(\omega, R)$ を得る．

3. $\mathbf{Uni}_2$ による場合：$\mathbf{Uni}_1$ の場合に対称である．

4. $\mathbf{Uni}_3$ による場合：ある R_1, R_2, Q_1, Q_2 について，$R \equiv R_1 :: R_2$ かつ $R_1 \xrightarrow{\omega} R_1 \triangleright Q_1$ かつ $R_2 \xrightarrow{\omega} R_2 \triangleright Q_2$ かつ $P' \equiv R \triangleright (Q_1 \parallel Q_2)$ である．帰納法の仮定より，各 $i \in \{1, 2\}$ について，$R_i \triangleright Q_i \equiv R_i \triangleright \mathtt{CP}(\omega, R_i)$ かつ $\omega \in call(R_i)$ を得る．すなわち，$Q_i \equiv \mathtt{CP}(\omega, R_i)$ である．ここで，$\omega \in call(R_1)$ かつ $\omega \in call(R_2)$ より，$\mathtt{CP}(\omega, R_1 :: R_2) \equiv \mathtt{CP}(\omega, R_1) \parallel \mathtt{CP}(\omega, R_2) \equiv Q_1 \parallel Q_2$ である．ゆえに，$\omega \in call(R_1) \cup call(R_2) = call(R)$ かつ $P' \equiv R \triangleright (Q_1 \parallel Q_2) \equiv R \triangleright \mathtt{CP}(\omega, R_1 :: R_2) \equiv R \triangleright \mathtt{CP}(\omega, R)$ を得る．

($\Leftarrow$) の場合：$P' \equiv R \triangleright \mathtt{CP}(\omega, R)$ かつ $\omega \in call(R)$ とし，R の構造に関する帰納法を用いる．

1. $R \equiv \{\!\{\omega'.Q\}\!\}$ の場合：$\omega \in call(R) = call(\{\!\{\omega'.Q\}\!\}) = \{\omega'\}$ より，$\omega' = \omega$ である．すなわち，$P' \equiv R \triangleright \mathtt{CP}(\omega, R) \equiv R \triangleright \mathtt{CP}(\omega, \{\!\{\omega.Q\}\!\}) \equiv R \triangleright Q$ である．ゆえに，**Reso** より $R \equiv \{\!\{\omega.Q\}\!\} \xrightarrow{\omega} R \triangleright Q \equiv P'$ を導ける．

2. $R \equiv R_1 :: R_2$ の場合：$\omega \in call(R) = call(R_1 :: R_2) = call(R_1) \cup call(R_2)$ より，次の 3 つの場合に分けて証明する．

   - $\omega \in call(R_1)$ かつ $\omega \notin call(R_2)$ の場合：このとき，$P' \equiv R \triangleright \mathtt{CP}(\omega, R) \equiv R \triangleright \mathtt{CP}(\omega, R_1)$ である．すなわち，$P'_1 \equiv R_1 \triangleright \mathtt{CP}(\omega, R_1)$ とおくと，帰納法の仮定より $R_1 \xrightarrow{\omega} P'_1$ を得る．さらに，$\omega \notin call(R_2)$ であるので，$\mathbf{Uni}_1$ より $R \equiv R_1 :: R_2 \xrightarrow{\omega} R_1 :: R_2 \triangleright \mathtt{CP}(\omega, R_1) \equiv P'$ を導ける．

- $\omega \notin call(R_1)$ かつ $\omega \in call(R_2)$ の場合：上記の場合と対称である．
- $\omega \in call(R_1)$ かつ $\omega \in call(R_2)$ の場合：このとき，$P' \equiv R \rhd \mathtt{CP}(\omega, R) \equiv R \rhd (\mathtt{CP}(\omega, R_1) \,\|\, \mathtt{CP}(\omega, R_2))$ である．すなわち，各 $i \in \{1,2\}$ について，$P'_i \equiv R_i \rhd \mathtt{CP}(\omega, R_i)$ とおくと，帰納法の仮定より $R_i \xrightarrow{\omega} P'_i$ を得る．さらに，$\mathbf{Uni}_3$ より $R \equiv R_1 :: R_2 \xrightarrow{\omega} R_1 :: R_2 \rhd (\mathtt{CP}(\omega, R_1) \,\|\, \mathtt{CP}(\omega, R_2)) \equiv P'$ を導ける． ∎

命題 3.5.5 (R の展開規則) $R \in \mathcal{R}_{ccspr}$ とする．このとき，次の等式が成り立つ．

$$R \sim \sum\{\omega.(R \rhd \mathtt{CP}(\omega, R)) : \omega \in call(R)\}$$

証明　命題 2.4.1の条件 (i) と (ii) が満たされることを示す．

(i) リソースの遷移は必ず大域ラベルによるので $R \xrightarrow{\omega} P'$ とする．命題 3.5.4より，$P' \equiv R \rhd \mathtt{CP}(\omega, R)$ かつ $\omega \in call(R)$ である．すなわち，規則 **Act** と **Choice** により，$\sum\{\omega'.(R \rhd \mathtt{CP}(\omega', R)) : \omega' \in call(R)\} \xrightarrow{\omega} P'$ を導ける．ここで，$P' \sim P'$ である．

(ii) $\sum\{\omega'.(R \rhd \mathtt{CP}(\omega', R)) : \omega' \in call(R)\} \xrightarrow{\alpha} P'$ とする．規則 **Act** と **Choice** により，ある ω について，$\alpha = \omega \in call(R)$ かつ $P' \equiv R \rhd \mathtt{CP}(\omega, R)$ でなければならない．すなわち，命題 3.5.4より $R \xrightarrow{\omega} P'$ を得る． ∎

命題 3.5.6 ($R \rhd P$ の展開規則) P の構造について帰納的に次のように展開できる．

1. $R \rhd (\alpha.P')$ の場合：

$$R \rhd (\alpha.P') \sim \alpha.(R \rhd P') + \textstyle\sum_{\omega \in call(R)}\{\tau.(R \rhd (P' \rangle \mathtt{CP}(\omega, R))) : \alpha = \overline{\omega}\}$$

2. $R \rhd (\sum_{i \in I} P_i)$ の場合：

$$R \rhd (\textstyle\sum_{i \in I} P_i) \sim \sum_{i \in I} (R \rhd P_i)$$

3. $R \rhd (P_1 | P_2)$ の場合：各 $i \in \{1,2\}$ について $R \rhd P_i \sim \sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij})$ とする．

$$\begin{aligned} R \rhd (P_1|P_2) \sim \quad & \textstyle\sum_{j \in J_1} \alpha_{1j}.(R \rhd (P_{1j}|P_2)) \\ & + \textstyle\sum_{j \in J_2} \alpha_{2j}.(R \rhd (P_1|P_{2j})) \\ & + \textstyle\sum_{j_1 \in J_1} \sum_{j_2 \in J_2} \{\tau.(R \rhd (P_{1j_1}|P_{2j_2})) : \exists \omega.\ \alpha_{1j_1} = \omega, \alpha_{2j_2} = \overline{\omega}\} \\ & + \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_1} \{\tau.(R \rhd ((P_{1j}|P_2) \rangle \mathtt{CP}(\omega, R))) : \alpha_{1j} = \overline{\omega}\} \\ & + \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_2} \{\tau.(R \rhd ((P_1|P_{2j}) \rangle \mathtt{CP}(\omega, R))) : \alpha_{2j} = \overline{\omega}\} \end{aligned}$$

4. $R \rhd (P_1 \parallel P_2)$ の場合：各 $i \in \{1,2\}$ について $R \rhd P_i \sim \sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij})$ とする．

$$\begin{aligned} R \rhd (P_1 \parallel P_2) \sim \quad & \textstyle\sum_{j \in J_1} \{\alpha_{1j}.(R \rhd (P_{1j} \parallel P_2)) : \alpha_{1j} \notin \mathcal{L}_C\} \\ & + \textstyle\sum_{j \in J_2} \{\alpha_{2j}.(R \rhd (P_1 \parallel P_{2j})) : \alpha_{2j} \notin \mathcal{L}_C\} \\ & + \textstyle\sum_{j_1 \in J_1} \sum_{j_2 \in J_2} \{\tau.(R \rhd (P_{1j_1} \parallel P_{2j_2})) : \exists \omega.\ \alpha_{1j_1} = \omega, \alpha_{2j_2} = \overline{\omega}\} \\ & + \textstyle\sum_{j_1 \in J_1} \sum_{j_2 \in J_2} \{\lfloor a \rfloor.(R \rhd (P_{1j_1} \parallel P_{2j_2})) : \alpha_{1j_1} = \alpha_{2j_2} = \lfloor a \rfloor\} \\ & + \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_1} \{\tau.(R \rhd ((P_{1j} \parallel P_2) \rangle \mathtt{CP}(\omega, R))) : \alpha_{1j} = \overline{\omega}\} \\ & + \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_2} \{\tau.(R \rhd ((P_1 \parallel P_{2j}) \rangle \mathtt{CP}(\omega, R))) : \alpha_{2j} = \overline{\omega}\} \end{aligned}$$

5. $R \rhd (P_1 \rangle P_2)$ の場合：各 $i \in \{1,2\}$ について $R \rhd P_i \sim \sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij})$ とする．

$$\begin{aligned} R \rhd (P_1 \rangle P_2) \sim \quad & \textstyle\sum_{j \in J_1} \{\alpha_{1j}.(R \rhd (P_{1j} \rangle P_2)) : \alpha_{1j} \notin \mathcal{L}_P\} \\ & + \textstyle\sum_{j \in J_2} \{\alpha_{2j}.(R \rhd (P_1 \rangle P_{2j})) : \alpha_{2j} \notin \mathcal{L}_P \cup \mathcal{L}_C\} \\ & + \textstyle\sum_{j_1 \in J_1} \sum_{j_2 \in J_2} \{\tau.(R \rhd (P_{1j_1} \rangle P_{2j_2})) : \exists \omega.\ \alpha_{1j_1} = \omega, \alpha_{2j_2} = \overline{\omega}\} \\ & + \textstyle\sum_{j_1 \in J_1} \sum_{j_2 \in J_2} \{\lceil a \rceil.(R \rhd (P_{1j_1} \rangle P_{2j_2})) : \alpha_{1j_1} = \lceil a \rceil, \alpha_{2j_2} = \lfloor a \rfloor\} \\ & + \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_1} \{\tau.(R \rhd ((P_{1j} \rangle P_2) \rangle \mathtt{CP}(\omega, R))) : \alpha_{1j} = \overline{\omega}\} \\ & + \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_2} \{\tau.(R \rhd ((P_1 \rangle P_{2j}) \rangle \mathtt{CP}(\omega, R))) : \alpha_{2j} = \overline{\omega}\} \end{aligned}$$

6. $R \rhd (P_1[f])$ の場合：$R \rhd P_1 \sim \sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j})$ とする．

$$\begin{aligned} R \rhd (P_1[f]) \sim \quad & \textstyle\sum_{j \in J_1} f(\alpha_{1j}).(R \rhd (P_{1j}[f])) \\ & + \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_1} \{\tau.(R \rhd ((P_{1j}[f]) \rangle \mathtt{CP}(\omega, R))) : f(\alpha_{1j}) = \overline{\omega}\} \end{aligned}$$

7. $R \rhd (P_1 \backslash L)$ の場合：$R \rhd P_1 \sim \sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j})$ とする．

$$\begin{aligned} R \rhd (P_1 \backslash L) \sim \quad & \textstyle\sum_{j \in J_1} \{\alpha_{1j}.(R \rhd (P_{1j} \backslash L)) : \alpha_{1j} \notin L \cup \overline{L}\} \\ & + \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_1} \{\tau.(R \rhd ((P_{1j} \backslash L) \rangle \mathtt{CP}(\omega, R))) : \alpha_{1j} = \overline{\omega} \notin L \cup \overline{L}\} \end{aligned}$$

8. $R \rhd (P_1/L)$ の場合：$R \rhd P_1 \sim \sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j})$ とする．

$$\begin{aligned} R \rhd (P_1/L) \sim \quad & \textstyle\sum_{j \in J_1} \{\alpha_{1j}.(R \rhd (P_{1j}/L)) : \alpha_{1j} \notin L\} \\ & + \textstyle\sum_{j \in J_1} \{\tau.(R \rhd (P_{1j}/L)) : \alpha_{1j} \in L\} \\ & + \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_1} \{\tau.(R \rhd ((P_{1j}/L) \rangle \mathtt{CP}(\omega, R))) : \alpha_{1j} = \overline{\omega}\} \end{aligned}$$

9. $R \rhd [\![P_1]\!]$ の場合：$P_1 \sim \sum_{j \in J_1} \alpha_{1j}.P_{1j}$ とする．

$$\begin{aligned} R \rhd [\![P_1]\!] \sim \quad & \textstyle\sum_{j \in J_1} \alpha_{1j}.(R \rhd [\![P_{1j}]\!])) \\ & + \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_1} \{\tau.(R \rhd ([\![P_{1j}]\!] \rangle \mathtt{CP}(\omega, R))) : \alpha_{1j} = \overline{\omega}\} \end{aligned}$$

証明 各場合について，命題 2.4.1の条件 (i),(ii) が満たされることを示す.

1. $R \rhd (\alpha.P')$ の場合：$S_1 \equiv \alpha.(R \rhd P')$，$S_2 \equiv \sum_{\omega \in call(R)} \{\tau.(R \rhd (P' \rangle \mathtt{CP}(\omega, R))) : \alpha = \overline{\omega}\}$ とおく.

   (i) $R \rhd (\alpha.P') \xrightarrow{\alpha'} P''$ とする．この遷移は規則 $\mathbf{Supp}_1$ か $\mathbf{Supp}_2$ によって導かれる．各場合について証明する.

   - $\mathbf{Supp}_1$ による場合：ある Q, P_1, ω について，$R \xrightarrow{\omega} R \rhd Q$ かつ $\alpha.P' \xrightarrow{\overline{\omega}} P_1$ かつ $\alpha' = \tau$ かつ $P'' \equiv R \rhd (P_1 \rangle Q)$ である．$\alpha.P' \xrightarrow{\overline{\omega}} P_1$ であるので，規則 **Act** により $\alpha = \overline{\omega}$ かつ $P' \equiv P_1$ である．また，$R \xrightarrow{\omega} R \rhd Q$ であるので，命題 3.5.4より $R \rhd Q \equiv R \rhd \mathtt{CP}(\omega, R)$ かつ $\omega \in call(R)$ である．一方，$\alpha = \overline{\omega}$ かつ $\omega \in call(R)$ であるので，規則 **Act** と **Choice** によって $S_2 \xrightarrow{\tau} R \rhd (P' \rangle \mathtt{CP}(\omega, R)) \equiv P''$ を導ける．さらに，**Choice** によって $S_1 + S_2 \xrightarrow{\tau} P''$ を導く．ここで，$P'' \sim P''$ である.
   - $\mathbf{Supp}_2$ による場合：ある P_1 について，$\alpha.P' \xrightarrow{\alpha'} P_1$ かつ $P'' \equiv R \rhd P_1$ である．$\alpha.P' \xrightarrow{\alpha'} P_1$ であるので，規則 **Act** により $\alpha = \alpha'$ かつ $P' \equiv P_1$ である．一方，規則 **Act** によって $S_1 \xrightarrow{\alpha} R \rhd P' \equiv P''$ を導ける．さらに，**Choice** によって $S_1 + S_2 \xrightarrow{\alpha} P''$ を導く.

   (ii) $S_1 + S_2 \xrightarrow{\alpha'} P''$ とする．この遷移を導く最後の規則は $\mathbf{Choice}_1$ か $\mathbf{Choice}_2$ である．各場合について証明する.

   - $\mathbf{Choice}_1$ による場合：$S_1 \xrightarrow{\alpha'} P''$ である．さらに，規則 **Act** によって，$\alpha' = \alpha$ かつ $P'' \equiv R \rhd P'$ を得る．一方，**Act** により $\alpha.P' \xrightarrow{\alpha} P'$ を導ける．さらに，$\mathbf{Supp}_2$ によって $R \rhd (\alpha.P') \xrightarrow{\alpha} R \rhd P' \equiv P''$ を導く.
   - $\mathbf{Choice}_2$ による場合：$S_2 \xrightarrow{\alpha'} P''$ である．さらに，規則 **Act** と **Choice** によって，ある ω について，$P'' \equiv R \rhd (P' \rangle \mathtt{CP}(\omega, R))$ かつ $\alpha = \overline{\omega}$ かつ $\omega \in call(R)$ かつ $\alpha' = \tau$ が得られる．一方，$\omega \in call(R)$ であるので，命題 3.5.4より $R \xrightarrow{\omega} R \rhd \mathtt{CP}(\omega, R)$ を得る．また，**Act** により $\alpha.P' \xrightarrow{\alpha} P'$ を導ける．さらに，$\alpha = \overline{\omega}$ であるので，$\mathbf{Supp}_1$ によって $R \rhd (\alpha.P') \xrightarrow{\tau} R \rhd (P' \rangle \mathtt{CP}(\omega, R)) \equiv P''$ を導く.

2. $R \rhd (\sum_{i \in I} P_i)$ の場合：

   (i) $R \rhd (\sum_{i \in I} P_i) \xrightarrow{\alpha} P'$ とする．この遷移を導く最後の規則は $\mathbf{Supp}_1$ か $\mathbf{Supp}_2$ か $\mathbf{S.Choice}_j$ である．各場合について証明する．

   - $\mathbf{Supp}_1$ による場合：ある Q, P'', ω について，$R \xrightarrow{\omega} R \rhd Q$ かつ $\sum_{i \in I} P_i \xrightarrow{\overline{\omega}} P''$ かつ $\alpha = \tau$ かつ $P' \equiv R \rhd (P'' \rangle Q)$ である．$\sum_{i \in I} P_i \xrightarrow{\overline{\omega}} P''$ であるので，**Choice** により，ある $j \in I$ について，$P_j \xrightarrow{\overline{\omega}} P''$ でなければならない．すなわち，$R \xrightarrow{\omega} R \rhd Q$ かつ $P_j \xrightarrow{\overline{\omega}} P''$ であるので，$\mathbf{Supp}_1$ より $R \rhd P_j \xrightarrow{\tau} R \rhd (P'' \rangle Q) \equiv P'$ を導く．さらに，$j \in I$ であるので，$\mathbf{Choice}_j$ により $\sum_{i \in I} (R \rhd P_i) \xrightarrow{\tau} P'$ を導ける．
   - $\mathbf{Supp}_2$ による場合：ある P'' について，$\sum_{i \in I} P_i \xrightarrow{\alpha} P''$ かつ $P' \equiv R \rhd P''$ である．$\sum_{i \in I} P_i \xrightarrow{\alpha} P''$ であるので，**Choice** により，ある $j \in I$ について，$P_j \xrightarrow{\alpha} P''$ を得る．すなわち，$\mathbf{Supp}_2$ より，$R \rhd P_j \xrightarrow{\alpha} R \rhd P'' \equiv P'$ を導く．さらに，$j \in I$ であるので，$\mathbf{Choice}_j$ により $\sum_{i \in I} (R \rhd P_i) \xrightarrow{\alpha} P'$ を導ける．
   - $\mathbf{S.Choice}_j$ による場合：ある $j \in I$ について，$R \rhd P_j \xrightarrow{\tau} P'$ かつ $\alpha = \tau$ である．すなわち，$\mathbf{Choice}_j$ により $\sum_{i \in I} (R \rhd P_i) \xrightarrow{\tau} P'$ を導ける．

   (ii) $\sum_{i \in I} (R \rhd P_i) \xrightarrow{\alpha} P'$ とする．この遷移を導く最後の規則は **Choice** のみである．すなわち，ある $j \in I$ について，$R \rhd P_j \xrightarrow{\alpha} P'$ でなければならない．アクション α について，次の場合に分けて証明する．

   - $\alpha \neq \tau$ の場合：このとき，$R \rhd P_j \xrightarrow{\alpha} P'$ を導く最後の規則は $\mathbf{Supp}_2$ のみである．すなわち，ある P'' について，$P_j \xrightarrow{\alpha} P''$ かつ $P' \equiv R \rhd P''$ である．この遷移 $P_j \xrightarrow{\alpha} P''$ から，$j \in I$ であるので，規則 $\mathbf{Choice}_j$ により $\sum_{i \in I} P_i \xrightarrow{\alpha} P''$ を導く．さらにこの遷移から，$\mathbf{Supp}_2$ により $R \rhd (\sum_{i \in I} P_i) \xrightarrow{\alpha} R \rhd P'' \equiv P'$ を導ける．
   - $\alpha = \tau$ の場合：このとき，$R \rhd P_j \xrightarrow{\tau} P'$ であるので，$\mathbf{S.Choice}_j$ により $R \rhd (\sum_{i \in I} P_i) \xrightarrow{\tau} P'$ を得る．

3. $R \rhd (P_1 | P_2)$ の場合：次の $R \rhd (P_1 \parallel P_2)$ の場合より簡単であるので省略する．

4. $R \rhd (P_1 \,\|\, P_2)$ の場合：各 $i \in \{1,2\}$ について $R \rhd P_i \sim \sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij})$ とし，

$$
\begin{aligned}
S &\equiv \textstyle\sum_{i \in \{1,\cdots,6\}} S_i \\
S_1 &\equiv \textstyle\sum_{j \in J_1} \{\alpha_{1j}.(R \rhd (P_{1j} \,\|\, P_2)) : \alpha_{1j} \notin \mathcal{L}_C\} \\
S_2 &\equiv \textstyle\sum_{j \in J_2} \{\alpha_{2j}.(R \rhd (P_1 \,\|\, P_{2j})) : \alpha_{2j} \notin \mathcal{L}_C\} \\
S_3 &\equiv \textstyle\sum_{j_1 \in J_1} \sum_{j_2 \in J_2} \{\tau.(R \rhd (P_{1j_1} \,\|\, P_{2j_2})) : \exists \omega.\, \alpha_{1j_1} = \omega, \alpha_{2j_2} = \overline{\omega}\} \\
S_4 &\equiv \textstyle\sum_{j_1 \in J_1} \sum_{j_2 \in J_2} \{\lfloor a \rfloor.(R \rhd (P_{1j_1} \,\|\, P_{2j_2})) : \alpha_{1j_1} = \alpha_{2j_2} = \lfloor a \rfloor\} \\
S_5 &\equiv \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_1} \{\tau.(R \rhd ((P_{1j} \,\|\, P_2)\rangle \mathtt{CP}(\omega, R))) : \alpha_{1j} = \overline{\omega}\} \\
S_6 &\equiv \textstyle\sum_{\omega \in call(R)} \sum_{j \in J_2} \{\tau.(R \rhd ((P_1 \,\|\, P_{2j})\rangle \mathtt{CP}(\omega, R))) : \alpha_{2j} = \overline{\omega}\}
\end{aligned}
$$

とおく．

(i) $R \rhd (P_1 \,\|\, P_2) \xrightarrow{\alpha} P'$ とする．この遷移を導く最後の規則は $\mathbf{Supp}_1$ か $\mathbf{Supp}_2$ か $\mathbf{S.Sync}_1$ か $\mathbf{S.Sync}_2$ である．各場合について証明する．

- $\mathbf{Supp}_1$ による場合：ある Q, P'_{12}, ω について，$R \xrightarrow{\omega} R \rhd Q$ かつ $P_1 \,\|\, P_2 \xrightarrow{\overline{\omega}} P'_{12}$ かつ $\alpha = \tau$ かつ $P' \equiv R \rhd (P'_{12}\rangle Q)$ である．$R \xrightarrow{\omega} R \rhd Q$ であるので，命題 3.5.4より，$Q \equiv \mathtt{CP}(\omega, R)$ かつ $\omega \in call(R)$ を得る．一方，$\overline{\omega}$ は大域ラベルであるので，遷移 $P_1 \,\|\, P_2 \xrightarrow{\overline{\omega}} P'_{12}$ は $\mathbf{Sync}_1$ によってのみ導かれる．すなわち，ある P'_1 と P'_2 について，$P_1 | P_2 \xrightarrow{\overline{\omega}} P'_{12} \equiv P'_1 | P'_2$ である．さらに，この遷移は $\mathbf{Com}_1$ か $\mathbf{Com}_2$ によって導かれるが，これらは対称であるので $\mathbf{Com}_1$ による場合のみ示す．つまり，$P_1 \xrightarrow{\overline{\omega}} P'_1$ かつ $P_2 \equiv P'_2$ を得る．この遷移から，$\mathbf{Supp}_2$ より $R \rhd P_1 \xrightarrow{\overline{\omega}} R \rhd P'_1$ を導ける．ここで，$R \rhd P_1 \sim \sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j})$ であるので，ある S'' について，$\sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j}) \xrightarrow{\overline{\omega}} S''$ かつ $R \rhd P'_1 \sim S''$ を得る．さらに，この遷移は $\mathbf{Choice}$ と $\mathbf{Act}$ によって導かれるので，ある $j \in J_1$ について，$S'' \equiv R \rhd P_{1j}$ かつ $\alpha_{1j} = \overline{\omega}$ を得る．すなわち，$R \rhd P'_1 \sim S'' \equiv R \rhd P_{1j}$ であるので，命題 3.5.2(4) より，$R \rhd P'_{12} \equiv R \rhd (P'_1 \,\|\, P_2) \sim R \rhd (P_{1j} \,\|\, P_2)$ を得る．さらに，$R \rhd Q \equiv R \rhd \mathtt{CP}(\omega, R)$ であるので，命題 3.5.2(5) より，$P' \equiv R \rhd (P'_{12}\rangle Q) \equiv R \rhd ((P'_1 \,\|\, P_2)\rangle Q) \sim R \rhd ((P_{1j} \,\|\, P_2)\rangle \mathtt{CP}(\omega, R))$ を得る．すなわち，$S' \equiv R \rhd ((P_{1j} \,\|\, P_2)\rangle \mathtt{CP}(\omega, R))$ とおくと，$j \in J_1$ かつ $\alpha_{1j} = \overline{\omega}$ かつ $\omega \in call(R)$ であるので，$\mathbf{Choice}$ と $\mathbf{Act}$ により $S_5 \xrightarrow{\tau} S'$ を導ける．さらに，$\mathbf{Choice}_5$ により $S \xrightarrow{\tau} S'$ を導く．以上，$\alpha = \tau$ より $S \xrightarrow{\alpha} S'$ かつ $P' \sim S'$ である．

- $\mathbf{Supp}_2$ による場合：ある P'_{12} について，$P_1 \| P_2 \xrightarrow{\alpha} P'_{12}$ かつ $P' \equiv R \rhd P'_{12}$ である．この遷移 $P_1 \| P_2 \xrightarrow{\alpha} P'_{12}$ を導く最後の規則は $\mathbf{Sync}_1$ か $\mathbf{Sync}_2$ である．

  - $\mathbf{Sync}_1$ による場合：ある P'_1 と P'_2 について，$P_1 | P_2 \xrightarrow{\alpha} P'_{12} \equiv P'_1 | P'_2$ かつ $\alpha \notin \mathcal{L}_C$ である．さらに，この遷移を導く最後の規則は $\mathbf{Com}_1$ か $\mathbf{Com}_2$ か $\mathbf{Com}_3$ である．各場合に分けて証明する．

    (a) $\mathbf{Com}_1$ による場合:$P_1 \xrightarrow{\alpha} P'_1$ かつ $P_2 \equiv P'_2$ である．このとき，$\mathbf{Supp}_2$ より $R \rhd P_1 \xrightarrow{\alpha} R \rhd P'_1$ を導ける．$R \rhd P_1 \sim \sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j})$ であるので，ある S'' について，$\sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j}) \xrightarrow{\alpha} S''$ かつ $R \rhd P'_1 \sim S''$ を得る．さらに，この遷移は **Choice** と **Act** によって導かれるので，ある $j \in J_1$ で $S'' \equiv R \rhd P_{1j}$ かつ $\alpha_{1j} = \alpha$ を得る．ここで，$R \rhd P'_1 \sim S'' \equiv R \rhd P_{1j}$ であるので，命題 3.5.2(4) より $P' \equiv R \rhd P'_{12} \equiv R \rhd (P'_1 \| P_2) \sim R \rhd (P_{1j} \| P_2)$ を得る．すなわち，$S' \equiv R \rhd (P_{1j} \| P_2)$ とおくと，$j \in J_1$ かつ $\alpha \notin \mathcal{L}_C$ であるので，**Choice** と **Act** により $S_1 \xrightarrow{\alpha_{1j}} S'$ を導ける．さらに，$\mathbf{Choice}_1$ により $S \xrightarrow{\alpha_{1j}} S'$ を導く．以上，$\alpha = \alpha_{1j}$ より $S \xrightarrow{\alpha} S'$ かつ $P' \sim S'$ である．

    (b) $\mathbf{Com}_2$ による場合：$P_2 \xrightarrow{\alpha} P'_2$ かつ $P_1 \equiv P'_1$ である．$\mathbf{Com}_1$ による場合と対称である．

    (c) $\mathbf{Com}_3$ による場合：ある ω について，$P_1 \xrightarrow{\omega} P'_1$ かつ $P_2 \xrightarrow{\overline{\omega}} P'_2$ かつ $\alpha = \tau$ である．このとき，$\mathbf{Supp}_2$ より，$R \rhd P_1 \xrightarrow{\omega} R \rhd P'_1$ と $R \rhd P_2 \xrightarrow{\overline{\omega}} R \rhd P'_2$ を導ける．すなわち，各 $i \in \{1, 2\}$ について，仮定 $R \rhd P_i \sim \sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij})$ より，$\sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j}) \xrightarrow{\omega} S''_1$ かつ $\sum_{j \in J_2} \alpha_{2j}.(R \rhd P_{2j}) \xrightarrow{\overline{\omega}} S''_2$ かつ $R \rhd P'_i \sim S''_i$ となる S''_i が存在する．これらの遷移は **Choice** と **Act** によって導かれるので，ある $j_i \in J_i$ について，$S''_i \equiv R \rhd P_{ij_i}$ かつ $\alpha_{1j_1} = \omega$ かつ $\alpha_{2j_2} = \overline{\omega}$ を得る．ここで，各 $i \in \{1, 2\}$ について $R \rhd P'_i \sim S''_i \equiv R \rhd P_{ij_i}$ であるので，命題 3.5.2(4) より，$P' \equiv R \rhd P'_{12} \equiv R \rhd (P'_1 \| P'_2) \sim R \rhd (P_{1j_1} \| P_{2j_2})$ を得る．すなわち，$S' \equiv R \rhd (P_{1j_1} \| P_{2j_2})$ とおくと，$j_i \in J_i$ かつ $\alpha_{1j_1} = \omega$ かつ $\alpha_{2j_2} = \overline{\omega}$ であるので，**Choice** と **Act** により $S_3 \xrightarrow{\tau} S'$ を導ける．さらに，$\mathbf{Choice}_3$ により $S \xrightarrow{\tau} S'$ を導く．以上，$\alpha = \tau$ より $S \xrightarrow{\alpha} S'$ かつ $P' \sim S'$ である．

– **Sync**$_2$ による場合：ある a, P'_1, P'_2 について，$P_1 \xrightarrow{\lfloor a \rfloor} P'_1$ かつ $P_2 \xrightarrow{\lfloor a \rfloor} P'_2$ かつ $\alpha = \lfloor a \rfloor$ である．各 $i \in \{1,2\}$ について，**Supp**$_2$ より $R \rhd P_i \xrightarrow{\lfloor a \rfloor} R \rhd P'_i$ を導ける．仮定 $R \rhd P_i \sim \sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij})$ より，ある S''_i について，$\sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij}) \xrightarrow{\lfloor a \rfloor} S''_i$ かつ $R \rhd P'_i \sim S''_i$ を得る．この遷移は **Choice** と **Act** によって導かれるので，ある $j_i \in J_i$ について，$S''_i \equiv R \rhd P_{ij_i}$ かつ $\alpha_{ij_i} = \lfloor a \rfloor$ を得る．ここで，各 $i \in \{1,2\}$ について $R \rhd P'_i \sim S''_i \equiv R \rhd P_{ij_i}$ であるので，命題 3.5.2(4) より，$P' \equiv R \rhd P'_{12} \equiv R \rhd (P'_1 \| P'_2) \sim R \rhd (P_{1j_1} \| P_{2j_2})$ を得る．すなわち，$S' \equiv R \rhd (P_{1j_1} \| P_{2j_2})$ とおくと，$j_i \in J_i$ かつ $\alpha_{ij_i} = \lfloor a \rfloor$ であるので，**Choice** と **Act** により $S_4 \xrightarrow{\lfloor a \rfloor} S'$ を導く．さらに，**Choice**$_4$ により $S \xrightarrow{\lfloor a \rfloor} S'$ を導く．以上，$\alpha = \lfloor a \rfloor$ より $S \xrightarrow{\alpha} S'$ かつ $P' \sim S'$ である．

- **S.Sync**$_1$ による場合：ある P'_1 について，$R \rhd P_1 \xrightarrow{\tau} R \rhd P'_1$ かつ $P'_{12} \equiv P'_1 \| P_2$ かつ $\alpha = \tau$ である．ここで，仮定 $R \rhd P_1 \sim \sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j})$ より，ある S'' について，$\sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j}) \xrightarrow{\tau} S''$ かつ $R \rhd P'_1 \sim S''$ を得る．さらに，この遷移は **Choice** と **Act** によって導かれるので，ある $j \in J_1$ について，$S'' \equiv R \rhd P_{1j}$ かつ $\alpha_{1j} = \tau$ を得る．ここで，$R \rhd P'_1 \sim S'' \equiv R \rhd P_{1j}$ であるので，命題 3.5.2(4) より，$P' \equiv R \rhd P'_{12} \equiv R \rhd (P'_1 \| P_2) \sim R \rhd (P_{1j} \| P_2)$ を得る．すなわち，$S' \equiv R \rhd (P_{1j} \| P_2)$ とおくと，$j \in J_1$ かつ $\alpha_{1j} = \tau \notin \mathcal{L}_C$ であるので，**Choice** と **Act** により $S_1 \xrightarrow{\alpha_{1j}} S'$ を導く．さらに，**Choice**$_1$ により $S \xrightarrow{\alpha_{1j}} S'$ を導く．以上，$\alpha = \tau = \alpha_{1j}$ より $S \xrightarrow{\alpha} S'$ かつ $P' \sim S'$ である．
- **S.Sync**$_2$ による場合：ある P'_2 について，$R \rhd P_2 \xrightarrow{\tau} R \rhd P'_2$ かつ $P'_{12} \equiv P_1 \| P'_2$ かつ $\alpha = \tau$ である．**S.Sync**$_1$ による場合と対称である．

(ii) $S \xrightarrow{\alpha} S'$ とする．この遷移を導く最後の規則は **Choice**$_{1,\cdots,6}$ のどれかである．各場合について証明する．

- **Choice**$_1$ による場合：$S_1 \equiv \sum_{j' \in J_1} \{\alpha_{1j'}.(R \rhd (P_{1j'} \| P_2)) : \alpha_{1j'} \notin \mathcal{L}_C\} \xrightarrow{\alpha} S'$ である．この遷移は **Choice** と **Act** によって導かれるので，ある $j \in J_1$ について，$S' \equiv R \rhd (P_{1j} \| P_2)$ かつ $\alpha = \alpha_{1j} \notin \mathcal{L}_C$ を得る．一方，$j \in J_1$ であるので，**Choice** と **Act** により，$\sum_{j' \in J_1} \alpha_{1j'}.(R \rhd P_{1j'}) \xrightarrow{\alpha_{1j}} R \rhd P_{1j}$ を導ける．ここで，仮定 $R \rhd P_1 \sim \sum_{j' \in J_1} \alpha_{1j'}.(R \rhd P_{1j'})$ より，ある P'_1 について，$R \rhd P_1 \xrightarrow{\alpha_{1j}} R \rhd P'_1$ かつ $R \rhd P_{1j} \sim R \rhd P'_1$ を得る．すなわち，命題 3.5.2(4)

より，$S' \equiv R \rhd (P_{1j} \| P_2) \sim R \rhd (P_1' \| P_2)$ である．次の各場合について証明する．

- $\alpha_{1j} \neq \tau$ の場合：このとき，遷移 $R \rhd P_1 \xrightarrow{\alpha_{1j}} R \rhd P_1'$ を導く最後の規則は $\mathbf{Supp}_2$ のみである．すなわち，$P_1 \xrightarrow{\alpha_{1j}} P_1'$ である．この遷移から，$\alpha_{1j} \notin \mathcal{L}_C$ であるので，$\mathbf{Com}_1$ と $\mathbf{Sync}_1$ より，$P_1 \| P_2 \xrightarrow{\alpha_{1j}} P_1' \| P_2$ を導ける．さらに，$\mathbf{Supp}_2$ より，$R \rhd (P_1 \| P_2) \xrightarrow{\alpha_{1j}} R \rhd (P_1' \| P_2)$ を導く．
- $\alpha_{1j} = \tau$ の場合：遷移 $R \rhd P_1 \xrightarrow{\tau} R \rhd P_1'$ から，規則 $\mathbf{S.Sync}_1$ により $R \rhd (P_1 \| P_2) \xrightarrow{\tau} R \rhd (P_1' \| P_2)$ を導ける．

よって，$R \rhd (P_1 \| P_2) \xrightarrow{\alpha} R \rhd (P_1' \| P_2)$ かつ $S' \sim R \rhd (P_1' \| P_2)$ である．

- $\mathbf{Choice}_2$ による場合：$S_2 \equiv \sum_{j' \in J_2} \{\alpha_{2j'}.(R \rhd (P_1 \| P_{2j'})) : \alpha_{2j'} \notin \mathcal{L}_C\} \xrightarrow{\alpha} S'$ である．$\mathbf{Choice}_1$ による場合と対称である．
- $\mathbf{Choice}_3$ による場合：$S_3 \equiv \sum_{j_1' \in J_1} \sum_{j_2' \in J_2} \{\tau.(R \rhd (P_{1j_1'} \| P_{2j_2'})) : \alpha_{1j_1'} = \omega, \alpha_{2j_2'} = \overline{\omega}\} \xrightarrow{\alpha} S'$ である．この遷移は $\mathbf{Choice}$ と $\mathbf{Act}$ によって導かれるので，ある $j_1 \in J_1$ と $j_2 \in J_2$ について，$S' \equiv R \rhd (P_{1j_1} \| P_{2j_2})$ かつ $\alpha_{1j_1} = \omega$ かつ $\alpha_{2j_2} = \overline{\omega}$ かつ $\alpha = \tau$ を得る．一方，各 $i \in \{1,2\}$ について，$j_i \in J_i$ であるので，$\mathbf{Choice}$ と $\mathbf{Act}$ により $\sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij}) \xrightarrow{\alpha_{ij_i}} R \rhd P_{ij_i}$ を導ける．ここで，仮定 $R \rhd P_i \sim \sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij})$ より，ある P_i' について，$R \rhd P_i \xrightarrow{\alpha_{ij_i}} R \rhd P_i'$ かつ $R \rhd P_{ij_i} \sim R \rhd P_i'$ を得る．すなわち，命題 3.5.2(4) より，$S' \equiv R \rhd (P_{1j_1} \| P_{2j_2}) \sim R \rhd (P_1' \| P_2')$ である．また，各 $i \in \{1,2\}$ について，$\alpha_{ij_i} \neq \tau$ であるので，$R \rhd P_i \xrightarrow{\alpha_{ij_i}} R \rhd P_i'$ を導く最後の規則は $\mathbf{Supp}_2$ のみである．すなわち，$P_i \xrightarrow{\alpha_{ij_i}} P_i'$ である．ここで，$\alpha_{1j_1} = \omega$ かつ $\alpha_{2j_2} = \overline{\omega}$ であるので，$\mathbf{Com}_3$ と $\mathbf{Sync}_1$ より $P_1 \| P_2 \xrightarrow{\tau} P_1' \| P_2'$ を導ける．さらに，$\mathbf{Supp}_2$ より，$R \rhd (P_1 \| P_2) \xrightarrow{\tau} R \rhd (P_1' \| P_2')$ を導く．よって，$R \rhd (P_1 \| P_2) \xrightarrow{\alpha} R \rhd (P_1' \| P_2')$ かつ $S' \sim R \rhd (P_1' \| P_2')$ である．
- $\mathbf{Choice}_4$ による場合：$S_4 \equiv \sum_{j_1' \in J_1} \sum_{j_2' \in J_2} \{\lfloor a \rfloor.(R \rhd (P_{1j_1'} \| P_{2j_2'})) : \alpha_{1j_1'} = \alpha_{2j_2'} = \lfloor a \rfloor\} \xrightarrow{\alpha} S'$ である．この遷移は $\mathbf{Choice}$ と $\mathbf{Act}$ によって導かれるので，ある $j_1 \in J_1$ と $j_2 \in J_2$ について，$S' \equiv R \rhd (P_{1j_1} \| P_{2j_2})$ かつ $\alpha_{1j_1} = \alpha_{2j_2} = \alpha = \lfloor a \rfloor$ を得る．一方，各 $i \in \{1,2\}$ について，$j_i \in J_i$ であるので，$\mathbf{Choice}$ と $\mathbf{Act}$ により $\sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij}) \xrightarrow{\alpha_{ij_i}} R \rhd P_{ij_i}$ を導ける．ここで，仮定 $R \rhd P_i \sim \sum_{j \in J_i} \alpha_{ij}.(R \rhd P_{ij})$ より，ある P_i' について，$R \rhd P_i \xrightarrow{\alpha_{ij_i}} R \rhd P_i'$ かつ $R \rhd P_{ij_i} \sim R \rhd P_i'$ を得る．すなわち，命題 3.5.2(4)

より，$S' \equiv R \rhd (P_{1j_1} \| P_{2j_2}) \sim R \rhd (P'_1 \| P'_2)$ である．また，各 $i \in \{1,2\}$ について，$\alpha_{ij_i} \neq \tau$ であるので，$R \rhd P_i \xrightarrow{\alpha_{ij_i}} R \rhd P'_i$ を導く最後の規則は $\mathbf{Supp}_2$ のみである．すなわち，$P_i \xrightarrow{\alpha_{ij_i}} P'_i$ である．ここで，$\alpha_{1j_1} = \alpha_{2j_2} = \lfloor a \rfloor$ であるので，$\mathbf{Sync}_2$ より $P_1 \| P_2 \xrightarrow{\lfloor a \rfloor} P'_1 \| P'_2$ を導ける．さらに，$\mathbf{Supp}_2$ より，$R \rhd (P_1 \| P_2) \xrightarrow{\lfloor a \rfloor} R \rhd (P'_1 \| P'_2)$ を導く．よって，$\alpha = \lfloor a \rfloor$ であるので，$R \rhd (P_1 \| P_2) \xrightarrow{\alpha} R \rhd (P'_1 \| P'_2)$ かつ $S' \sim R \rhd (P'_1 \| P'_2)$ である．

- $\mathbf{Choice}_5$ による場合：$S_5 \equiv \sum_{\omega' \in call(R)} \sum_{j' \in J_1} \{\tau.(R \rhd ((P_{1j'} \| P_2) \rangle \mathtt{CP}(\omega', R))) : \alpha_{1j'} = \overline{\omega'}\} \xrightarrow{\alpha} S'$ である．この遷移は **Choice** と **Act** によって導かれるので，ある $\omega \in call(R)$ と $j \in J_1$ について，$S' \equiv R \rhd ((P_{1j} \| P_2) \rangle \mathtt{CP}(\omega, R))$ かつ $\alpha_{1j} = \overline{\omega}$ かつ $\alpha = \tau$ を得る．一方，$j \in J_1$ であるので，**Choice** と **Act** により $\sum_{j' \in J_1} \alpha_{1j'}.(R \rhd P_{1j'}) \xrightarrow{\alpha_{1j}} R \rhd P_{1j}$ を導ける．ここで，仮定 $R \rhd P_1 \sim \sum_{j' \in J_1} \alpha_{1j'}.(R \rhd P_{1j'})$ より，ある P'_1 について，$R \rhd P_1 \xrightarrow{\alpha_{1j}} R \rhd P'_1$ かつ $R \rhd P_{1j} \sim R \rhd P'_1$ を得る．すなわち，命題 3.5.2(4) より，$R \rhd (P_{1j} \| P_2) \sim R \rhd (P'_1 \| P_2)$ であり，命題 3.5.2(5) より，$S' \equiv R \rhd ((P_{1j} \| P_2) \rangle \mathtt{CP}(\omega, R)) \sim R \rhd ((P'_1 \| P_2) \rangle \mathtt{CP}(\omega, R))$ を得る．また，$\alpha_{1j} = \overline{\omega} \neq \tau$ であるので，$R \rhd P_1 \xrightarrow{\alpha_{1j}} R \rhd P'_1$ を導く最後の規則は $\mathbf{Supp}_2$ のみである．よって，$P_1 \xrightarrow{\alpha_{1j}} P'_1$ を得る．このとき，$\alpha_{1j} = \overline{\omega} \notin \mathcal{L}_C$ であるので，$\mathbf{Com}_1$ と $\mathbf{Sync}_1$ より，$P_1 \| P_2 \xrightarrow{\alpha_{1j}} P'_1 \| P_2$ を導ける．一方，$\omega \in call(R)$ であるので，命題 3.5.4より，$R \xrightarrow{\omega} R \rhd \mathtt{CP}(\omega, R)$ である．すなわち，$P_1 \| P_2 \xrightarrow{\alpha_{1j}} P'_1 \| P_2$ かつ $\alpha_{1j} = \overline{\omega}$ であるので，$\mathbf{Supp}_1$ より $R \rhd (P_1 \| P_2) \xrightarrow{\tau} R \rhd ((P'_1 \| P_2) \rangle \mathtt{CP}(\omega, R))$ を導ける．よって，$R \rhd (P_1 \| P_2) \xrightarrow{\alpha} R \rhd ((P'_1 \| P_2) \rangle \mathtt{CP}(\omega, R)) \sim S'$ である．

5. $R \rhd (P_1 \rangle P_2)$ の場合：$R \rhd (P_1 \| P_2)$ の場合と同様である．

6. $R \rhd (P_1[f])$ の場合：$R \rhd P_1 \sim \sum_{j \in J_1} \alpha_{1j}.(R \rhd P_{1j})$ とし，

$$S_1 \equiv \sum_{j \in J_1} f(\alpha_{1j}).(R \rhd (P_{1j}[f]))$$
$$S_2 \equiv \sum_{\omega \in call(R)} \sum_{j \in J_1} \{\tau.(R \rhd ((P_{1j}[f]) \rangle \mathtt{CP}(\omega, R))) : f(\alpha_{1j}) = \overline{\omega}\}$$

とおく．

(i) $R \rhd (P_1[f]) \xrightarrow{\alpha} P'$ とする．この遷移を導く最後の規則は $\mathbf{Supp}_1$ か $\mathbf{Supp}_2$ か **S.Rel** である．各場合について証明する．

- **Supp**$_1$ による場合：ある Q, P'_{11}, ω について，$R \xrightarrow{\omega} R \triangleright Q$ かつ $P_1[f] \xrightarrow{\overline{\omega}} P'_{11}$ かつ $\alpha = \tau$ かつ $P' \equiv R \triangleright (P'_{11} \rangle Q)$ である．$R \xrightarrow{\omega} R \triangleright Q$ であるので，命題 3.5.4より，$Q \equiv \mathtt{CP}(\omega, R)$ かつ $\omega \in call(R)$ を得る．一方，遷移 $P_1[f] \xrightarrow{\overline{\omega}} P'_{11}$ は **Rel** によって導かれるので，ある P'_1 と ω' について，$P_1 \xrightarrow{\overline{\omega'}} P'_1$ かつ $P'_{11} \equiv P'_1[f]$ かつ $f(\omega') = \omega$ である．このとき，規則 **Supp**$_2$ より，$R \triangleright P_1 \xrightarrow{\overline{\omega'}} R \triangleright P'_1$ を導ける．ここで，仮定 $R \triangleright P_1 \sim \sum_{j \in J_1} \alpha_{1j}.(R \triangleright P_{1j})$ より，ある S'' について，$\sum_{j \in J_1} \alpha_{1j}.(R \triangleright P_{1j}) \xrightarrow{\overline{\omega'}} S''$ かつ $R \triangleright P'_1 \sim S''$ を得る．さらに，この遷移は **Choice** と **Act** によって導かれるので，ある $j \in J_1$ について，$S'' \equiv R \triangleright P_{1j}$ かつ $\alpha_{1j} = \overline{\omega'}$ を得る．ここで，$R \triangleright P'_1 \sim S'' \equiv R \triangleright P_{1j}$ であるので，命題 3.5.2(6) より，$R \triangleright P'_{11} \equiv R \triangleright (P'_1[f]) \sim R \triangleright (P_{1j}[f])$ を得る．さらに，$R \triangleright Q \equiv R \triangleright \mathtt{CP}(\omega, R)$ であるので，命題 3.5.2(5) より $P' \equiv R \triangleright (P'_{11} \rangle Q) \equiv R \triangleright ((P'_1[f]) \rangle Q) \sim R \triangleright ((P_{1j}[f]) \rangle \mathtt{CP}(\omega, R))$ を得る．すなわち，$j \in J_1$ かつ $f(\alpha_{1j}) = f(\overline{\omega'}) = \overline{f(\omega')} = \overline{\omega}$ かつ $\omega \in call(R)$ であるので，$S' \equiv R \triangleright ((P_{1j}[f]) \rangle \mathtt{CP}(\omega, R))$ とおくと，**Choice** と **Act** により $S_2 \xrightarrow{\tau} S'$ を導ける．さらに，**Choice**$_2$ により $S_1 + S_2 \xrightarrow{\tau} S'$ を導く．以上，$\alpha = \tau$ より $S_1 + S_2 \xrightarrow{\alpha} S'$ かつ $P' \sim S'$ である．

- **Supp**$_2$ による場合：ある P'_{11} について，$P_1[f] \xrightarrow{\alpha} P'_{11}$ かつ $P' \equiv R \triangleright P'_{11}$ である．この遷移 $P_1[f] \xrightarrow{\alpha} P'_{11}$ は **Rel** によって導かれるので，ある P'_1 と α' について，$P_1 \xrightarrow{\alpha'} P'_1$ かつ $P'_{11} \equiv P'_1[f]$ かつ $f(\alpha') = \alpha$ である．このとき，**Supp**$_2$ より $R \triangleright P_1 \xrightarrow{\alpha'} R \triangleright P'_1$ を導ける．ここで，仮定 $R \triangleright P_1 \sim \sum_{j \in J_1} \alpha_{1j}.(R \triangleright P_{1j})$ より，ある S'' について，$\sum_{j \in J_1} \alpha_{1j}.(R \triangleright P_{1j}) \xrightarrow{\alpha'} S''$ かつ $R \triangleright P'_1 \sim S''$ を得る．さらに，この遷移は **Choice** と **Act** によって導かれるので，ある $j \in J_1$ で $S'' \equiv R \triangleright P_{1j}$ かつ $\alpha_{1j} = \alpha'$ を得る．すなわち，$R \triangleright P'_1 \sim S'' \equiv R \triangleright P_{1j}$ であるので，命題 3.5.2(6) より $P' \equiv R \triangleright P'_{11} \equiv R \triangleright (P'_1[f]) \sim R \triangleright (P_{1j}[f])$ を得る．すなわち，$S' \equiv R \triangleright (P_{1j}[f])$ とすると，**Choice** と **Act** により，$j \in J_1$ であるので，$S_1 \xrightarrow{f(\alpha_{1j})} S'$ を導ける．さらに，**Choice**$_1$ により $S_1 + S_2 \xrightarrow{f(\alpha_{1j})} S'$ を導く．以上，$f(\alpha_{1j}) = f(\alpha') = \alpha$ より $S_1 + S_2 \xrightarrow{\alpha} S'$ かつ $P' \sim S'$ である．

- **S.Rel** による場合：ある P'_1 について，$R \triangleright P_1 \xrightarrow{\tau} R \triangleright P'_1$ かつ $P' \equiv P'_1[f]$ かつ $\alpha = \tau$ である．仮定 $R \triangleright P_1 \sim \sum_{j \in J_1} \alpha_{1j}.(R \triangleright P_{1j})$ より，ある S'' について，$\sum_{j \in J_1} \alpha_{1j}.(R \triangleright P_{1j}) \xrightarrow{\tau} S''$ かつ $R \triangleright P'_1 \sim S''$ を得る．さらに，この遷移は **Choice** と **Act** によって導かれるので，ある $j \in J_1$ について，$S'' \equiv R \triangleright P_{1j}$

かつ $\alpha_{1j} = \tau$ を得る．ここで，$R \rhd P_1' \sim S'' \equiv R \rhd P_{1j}$ であるので，命題 3.5.2(6) より，$P' \equiv R \rhd P_1' \equiv R \rhd (P_1'[f]) \sim R \rhd (P_{1j}[f])$ を得る．すなわち，$S' \equiv R \rhd (P_{1j}[f])$ とすると，$j \in J_1$ であるので，**Choice** と **Act** により $S_1 \xrightarrow{f(\alpha_{1j})} S'$ を導ける．さらに，**Choice**$_1$ により $S_1 + S_2 \xrightarrow{f(\alpha_{1j})} S'$ を導く．以上，$f(\alpha_{1j}) = f(\tau) = \tau = \alpha$ より $S_1 + S_2 \xrightarrow{\alpha} S'$ かつ $P' \sim S'$ である．

(ii) $S_1 + S_2 \xrightarrow{\alpha} S'$ とする．この遷移を導く最後の規則は **Choice**$_{1,2}$ のどちらかである．各場合について証明する．

- **Choice**$_1$ による場合：$S_1 \equiv \sum_{j' \in J_1} f(\alpha_{1j'}).(R \rhd (P_{1j'}[f])) \xrightarrow{\alpha} S'$ である．この遷移は **Choice** と **Act** によって導かれるので，ある $j \in J_1$ について，$S' \equiv R \rhd (P_{1j}[f])$ かつ $\alpha = f(\alpha_{1j})$ を得る．一方，$j \in J_1$ であるので，**Choice** と **Act** により $\sum_{j' \in J_1} \alpha_{1j'}.(R \rhd P_{1j'}) \xrightarrow{\alpha_{1j}} R \rhd P_{1j}$ を導ける．ここで，仮定 $R \rhd P_1 \sim \sum_{j' \in J_1} \alpha_{1j'}.(R \rhd P_{1j'})$ より，ある P_1' について，$R \rhd P_1 \xrightarrow{\alpha_{1j}} R \rhd P_1'$ かつ $R \rhd P_{1j} \sim R \rhd P_1'$ である．すなわち，命題 3.5.2(6) より $S' \equiv R \rhd (P_{1j}[f]) \sim R \rhd (P_1'[f])$ である．次の各場合について証明する．
  - $\alpha = f(\alpha_{1j}) \neq \tau$ の場合：このとき，遷移 $R \rhd P_1 \xrightarrow{\alpha_{1j}} R \rhd P_1'$ を導く最後の規則は **Supp**$_2$ のみである．すなわち，$P_1 \xrightarrow{\alpha_{1j}} P_1'$ である．この遷移から，**Rel** により $P_1[f] \xrightarrow{f(\alpha_{1j})} P_1'[f]$ を導ける．これはさらに，**Supp**$_2$ より $R \rhd (P_1[f]) \xrightarrow{f(\alpha_{1j})} R \rhd (P_1'[f])$ を導く．
  - $\alpha = f(\alpha_{1j}) = \tau$ の場合：規則 **S.Rel** により遷移 $R \rhd P_1 \xrightarrow{\tau} R \rhd P_1'$ は $R \rhd (P_1[f]) \xrightarrow{\tau} R \rhd (P_1'[f])$ を導く．

  よって，$R \rhd (P_1[f]) \xrightarrow{\alpha} R \rhd (P_1'[f])$ かつ $S' \sim R \rhd (P_1'[f])$ である．
- **Choice**$_2$ による場合：$S_2 \equiv \sum_{\omega' \in call(R)} \sum_{j' \in J_1} \{\tau.(R \rhd ((P_{1j'}[f]) \rangle \mathtt{CP}(\omega', R))) : f(\alpha_{1j'}) = \overline{\omega'}\} \xrightarrow{\alpha} S'$ である．この遷移は **Choice** と **Act** によって導かれるので，ある $\omega \in call(R)$ と $j \in J_1$ について，$S' \equiv R \rhd ((P_{1j}[f]) \rangle \mathtt{CP}(\omega, R))$ かつ $f(\alpha_{1j}) = \overline{\omega}$ かつ $\alpha = \tau$ を得る．一方，$j \in J_1$ であるので，**Choice** と **Act** により $\sum_{j' \in J_1} \alpha_{1j'}.(R \rhd P_{1j'}) \xrightarrow{\alpha_{1j}} R \rhd P_{1j}$ を導ける．ここで，仮定 $R \rhd P_1 \sim \sum_{j' \in J_1} \alpha_{1j'}.(R \rhd P_{1j'})$ より，ある P_1' について，$R \rhd P_1 \xrightarrow{\alpha_{1j}} R \rhd P_1'$ かつ $R \rhd P_{1j} \sim R \rhd P_1'$ を得る．すなわち，命題 3.5.2(6) より $R \rhd (P_{1j}[f]) \sim R \rhd (P_1'[f])$ である．さらに，命題 3.5.2(5) より $S' \equiv R \rhd ((P_{1j}[f]) \rangle \mathtt{CP}(\omega, R)) \sim R \rhd ((P_1'[f]) \rangle \mathtt{CP}(\omega, R))$ である．また，$f(\alpha_{1j}) = \overline{\omega} \neq \tau$ であるので，$R \rhd P_1 \xrightarrow{\alpha_{1j}}$

$R \triangleright P_1'$ を導く最後の規則は $\mathbf{Supp}_2$ のみである．よって，$P_1 \xrightarrow{\alpha_{1j}} P_1'$ を得る．このとき，**Rel** より，$P_1[f] \xrightarrow{f(\alpha_{1j})} P_1'[f]$ を導ける．一方，$\omega \in call(R)$ であるので，命題 3.5.4より $R \xrightarrow{\omega} R \triangleright \mathtt{CP}(\omega, R)$ である．すなわち，$R \xrightarrow{\omega} R \triangleright \mathtt{CP}(\omega, R)$ かつ $P_1[f] \xrightarrow{f(\alpha_{1j})} P_1'[f]$ かつ $f(\alpha_{1j}) = \overline{\omega}$ であるので，$\mathbf{Supp}_1$ より $R \triangleright (P_1[f]) \xrightarrow{\tau} R \triangleright ((P_1'[f]) \rangle \mathtt{CP}(\omega, R))$ を得る．

7. $R \triangleright (P_1 \backslash L)$ の場合：$R \triangleright (P_1[f])$ の場合に同様である．

8. $R \triangleright (P_1/L)$ の場合：$R \triangleright (P_1[f])$ の場合に同様である．

9. $R \triangleright [\![P_1]\!]$ の場合：$P_1 \sim \sum_{j \in J_1} \alpha_{1j}.P_{1j}$ とし，

$$S_1 \equiv \sum_{j \in J_1} \alpha_{1j}.(R \triangleright [\![P_{1j}]\!]))$$
$$S_2 \equiv \sum_{\omega \in call(R)} \sum_{j \in J_1} \{\tau.(R \triangleright ([\![P_{1j}]\!] \rangle \mathtt{CP}(\omega, R))) : \alpha_{1j} = \overline{\omega}\}$$

とおく．

(i) $R \triangleright [\![P_1]\!] \xrightarrow{\alpha} P'$ とする．この遷移を導く最後の規則は $\mathbf{Supp}_1$ か $\mathbf{Supp}_2$ である．各場合について証明する．

- $\mathbf{Supp}_1$ による場合：ある Q, P_{11}', ω について，$R \xrightarrow{\omega} R \triangleright Q$ かつ $[\![P_1]\!] \xrightarrow{\overline{\omega}} P_{11}'$ かつ $\alpha = \tau$ かつ $P' \equiv R \triangleright (P_{11}' \rangle Q)$ である．$R \xrightarrow{\omega} R \triangleright Q$ であるので，命題 3.5.4より $Q \equiv \mathtt{CP}(\omega, R)$ かつ $\omega \in call(R)$ を得る．一方，遷移 $[\![P_1]\!] \xrightarrow{\overline{\omega}} P_{11}'$ は **Pack** によって導かれるので，ある P_1' について，$P_1 \xrightarrow{\overline{\omega}} P_1'$ かつ $P_{11}' \equiv [\![P_1']\!]$ である．ここで，$P_1 \xrightarrow{\overline{\omega}} P_1'$ であるので，仮定 $P_1 \sim \sum_{j \in J_1} \alpha_{1j}.P_{1j}$ より，ある Q' について，$\sum_{j \in J_1} \alpha_{1j}.P_{1j} \xrightarrow{\overline{\omega}} Q'$ かつ $P_1' \sim Q'$ を得る．さらに，この遷移は **Choice** と **Act** によって導かれるので，ある $j \in J_1$ について，$Q' \equiv P_{1j}$ かつ $\alpha_{1j} = \overline{\omega}$ を得る．すなわち，$P_1' \sim Q' \equiv P_{1j}$ であるので，命題 3.5.3より $R \triangleright [\![P_1']\!] \sim R \triangleright [\![P_{1j}]\!]$ を得る．さらに，$R \triangleright Q \equiv R \triangleright \mathtt{CP}(\omega, R)$ であるので，命題 3.5.2(5) より，$P' \equiv R \triangleright (P_{11}' \rangle Q) \equiv R \triangleright ([\![P_1']\!] \rangle Q) \sim R \triangleright ([\![P_{1j}]\!] \rangle \mathtt{CP}(\omega, R))$ を得る．すなわち，$S' \equiv R \triangleright ([\![P_{1j}]\!] \rangle \mathtt{CP}(\omega, R))$ とおくと，$j \in J_1$ かつ $\alpha_{1j} = \overline{\omega}$ かつ $\omega \in call(R)$ であるので，**Choice** と **Act** により $S_2 \xrightarrow{\tau} S'$ を導く．さらに，$\mathbf{Choice}_2$ により $S_1 + S_2 \xrightarrow{\tau} S'$ を導く．以上，$\alpha = \tau$ より $S_1 + S_2 \xrightarrow{\alpha} S'$ かつ $P' \sim S'$ である．

- **Supp**$_2$ による場合：ある P'_{11} について，$[\![P_1]\!] \xrightarrow{\alpha} P'_{11}$ かつ $P' \equiv R \triangleright P'_{11}$ である．この遷移 $[\![P_1]\!] \xrightarrow{\alpha} P'_{11}$ は **Pack** によって導かれるので，ある P'_1 について，$P_1 \xrightarrow{\alpha} P'_1$ かつ $P'_{11} \equiv [\![P'_1]\!]$ である．すなわち，仮定 $P_1 \sim \sum_{j\in J_1} \alpha_{1j}.P_{1j}$ より，ある Q' について，$\sum_{j\in J_1} \alpha_{1j}.P_{1j} \xrightarrow{\alpha} Q'$ かつ $P'_1 \sim Q'$ を得る．さらに，この遷移は **Choice** と **Act** によって導かれるので，ある $j \in J_1$ について，$Q' \equiv P_{1j}$ かつ $\alpha_{1j} = \alpha$ を得る．ここで，$P'_1 \sim Q' \equiv P_{1j}$ であるので，命題 3.5.3より $P' \equiv R \triangleright P'_{11} \equiv R \triangleright [\![P'_1]\!] \sim R \triangleright [\![P_{1j}]\!]$ を得る．すなわち，$S' \equiv R \triangleright [\![P_{1j}]\!]$ とおくと，$j \in J_1$ であるので，**Choice** と **Act** により $S_1 \xrightarrow{\alpha_{1j}} S'$ を導ける．さらに，**Choice**$_1$ により $S_1 + S_2 \xrightarrow{\alpha_{1j}} S'$ を導く．以上，$\alpha_{1j} = \alpha$ より $S_1 + S_2 \xrightarrow{\alpha} S'$ かつ $P' \sim S'$ である．

(ii) $S_1 + S_2 \xrightarrow{\alpha} S'$ とする．この遷移を導く最後の規則は **Choice**$_{1,2}$ のどちらかである．各場合について証明する．

- **Choice**$_1$ による場合：$S_1 \equiv \sum_{j'\in J_1} \alpha_{1j'}.(R \triangleright [\![P_{1j'}]\!]) \xrightarrow{\alpha} S'$ である．この遷移は **Choice** と **Act** によって導かれるので，ある $j \in J_1$ について，$S' \equiv R \triangleright [\![P_{1j}]\!]$ かつ $\alpha = \alpha_{1j}$ を得る．一方，$j \in J_1$ であるので，**Choice** と **Act** により $\sum_{j'\in J_1} \alpha_{1j'}.P_{1j'} \xrightarrow{\alpha_{1j}} P_{1j}$ を導ける．ここで，仮定 $P_1 \sim \sum_{j'\in J_1} \alpha_{1j'}.P_{1j'}$ より，ある P'_1 について，$P_1 \xrightarrow{\alpha_{1j}} P'_1$ かつ $P_{1j} \sim P'_1$ を得る．すなわち，命題 3.5.3より $S' \equiv R \triangleright [\![P_{1j}]\!] \sim R \triangleright [\![P'_1]\!]$ である．一方，遷移 $P_1 \xrightarrow{\alpha_{1j}} P'_1$ であるので，**Pack** により $[\![P_1]\!] \xrightarrow{\alpha_{1j}} [\![P'_1]\!]$ を導ける．さらに，**Supp**$_2$ より $R \triangleright [\![P_1]\!] \xrightarrow{\alpha_{1j}} R \triangleright [\![P'_1]\!]$ を導く．ゆえに，$R \triangleright [\![P_1]\!] \xrightarrow{\alpha} R \triangleright [\![P'_1]\!]$ かつ $S' \sim R \triangleright [\![P'_1]\!]$ である．
- **Choice**$_2$ による場合：$S_2 \equiv \sum_{\omega'\in call(R)} \sum_{j'\in J_1} \{\tau.(R \triangleright ([\![P_{1j'}]\!] \rangle \mathtt{CP}(\omega', R))) : \alpha_{1j'} = \overline{\omega'}\} \xrightarrow{\alpha} S'$ である．この遷移は **Choice** と **Act** によって導かれるので，ある $\omega \in call(R)$ と $j \in J_1$ について，$S' \equiv R \triangleright ([\![P_{1j}]\!] \rangle \mathtt{CP}(\omega, R))$ かつ $\alpha_{1j} = \overline{\omega}$ かつ $\alpha = \tau$ を得る．一方，$j \in J_1$ であるので，**Choice** と **Act** により $\sum_{j'\in J_1} \alpha_{1j'}.P_{1j'} \xrightarrow{\alpha_{1j}} P_{1j}$ である．ここで，仮定 $P_1 \sim \sum_{j'\in J_1} \alpha_{1j'}.P_{1j'}$ より，ある P'_1 について，$P_1 \xrightarrow{\alpha_{1j}} P'_1$ かつ $P_{1j} \sim P'_1$ を得る．すなわち，命題 3.5.3より $R \triangleright [\![P_{1j}]\!] \sim R \triangleright [\![P'_1]\!]$ である．ここでさらに，命題 3.5.2(5) より $S' \equiv R \triangleright ([\![P_{1j}]\!] \rangle \mathtt{CP}(\omega, R)) \sim R \triangleright ([\![P'_1]\!] \rangle \mathtt{CP}(\omega, R))$ である．また，$P_1 \xrightarrow{\alpha_{1j}} P'_1$ であるので，**Pack** より $[\![P_1]\!] \xrightarrow{\alpha_{1j}} [\![P'_1]\!]$ を導ける．一方，$\omega \in call(R)$ であるので，命題 3.5.4より $R \xrightarrow{\omega} R \triangleright \mathtt{CP}(\omega, R)$ である．すなわち，$R \xrightarrow{\omega} R \triangleright \mathtt{CP}(\omega, R)$

かつ $[\![P_1]\!] \xrightarrow{\alpha_{1j}} [\![P_1']\!]$ かつ $\alpha_{1j} = \overline{\omega}$ であるので，$\mathbf{Supp}_1$ より $R \triangleright [\![P_1]\!] \xrightarrow{\tau} R \triangleright ([\![P_1']\!] \rangle \mathtt{CP}(\omega, R))$ を導ける．よって，$R \triangleright [\![P_1]\!] \xrightarrow{\alpha} R \triangleright ([\![P_1']\!] \rangle \mathtt{CP}(\omega, R)) \sim S'$ である．

∎

3.6 CCSPR によるプロダクションルールの解析

まず，3.6.1小節でプロダクションルールによるスケジューラの例を紹介する．次に，3.6.2小節でそのスケジューラを CCSPR によって記述し，その振舞いを検証する．

3.6.1 プロダクションルールの例

本小節では，7つのプロダクションルールをもつスケジューラの例を示す．各ルールは ECA ルールの形をとるが，簡単のため条件 C は常に満たされると仮定して省略されている．すなわち，各ルールは，反応する状況変化 (event)，そのときの動作 (action)，カップリングモード (CM) から構成される．

スケジュールされるアクションは $\overline{a_1}, \cdots, \overline{a_7}$ であり，ルールが反応する状況変化 (これもアクションである) は $\overline{e_1}, \cdots, \overline{e_7}$ である．各ルール $Rule_i$ はアクション $\overline{a_i}$ を実行するとともに必要に応じて $\overline{e_j}$ によって他のルールを呼び出すこともできる．各ルールの振舞いを表 3.2に示す．例えば，$Rule_1$ は $\overline{e_1}$ の状況変化に反応し，アクション $\overline{a_1}$ と実行するとともに状況変化 $\overline{e_3}$ を引き起こす．

ルールを管理するプロセス M は，環境からアクション $start$ を受け取ると，状況変化 $\overline{e_1}$ を起こすとする．この状況変化 $\overline{e_1}$ は，$Rule_1$ と $Rule_2$ を呼び出す．このとき，$Rule_1$ は状況変化 $\overline{e_3}$ を起こして $Rule_3$ を呼び出し，$Rule_2$ は状況変化 $\overline{e_4}$ を起こして $Rule_4$ と $Rule_5$ を呼び出す．さらに，$Rule_3$ と $Rule_4$ は各々 $Rule_6$ と $Rule_7$ を呼び出すため，最終的に図 3.5に示すプロセス木が生成される．

このとき，各ルールのカップリングモードを考慮して，スケジュールされるアクション $\overline{a_1}, \cdots, \overline{a_7}$ の期待される実行順序を図 3.6に示す．例えば，$Rule_3$ は延期実行モードをもつため，その実行はプロセス木のトップの管理プロセス M が実行を完了する直前まで延期される．管理プロセス M は $Rule_1$ と $Rule_2$ が処理を完了するまで待ち，$Rule_2$

表 3.2: 各ルールの振舞い

名前	反応する状況変化	そのときの動作	カップリングモード
$Rule_1$	$\overline{e_1}$	$\overline{a_1}; \overline{e_3}$	即実行 (IM)
$Rule_2$	$\overline{e_1}$	$\overline{a_2}; \overline{e_4}$	即実行 (IM)
$Rule_3$	$\overline{e_3}$	$\overline{a_3}; \overline{e_6}$	延期実行 (DF)
$Rule_4$	$\overline{e_4}$	$\overline{a_4}; \overline{e_7}$	分離実行 (SP)
$Rule_5$	$\overline{e_4}$	$\overline{a_5}$	即実行 (IM)
$Rule_6$	$\overline{e_6}$	$\overline{a_6}$	即実行 (IM)
$Rule_7$	$\overline{e_7}$	$\overline{a_7}$	延期実行 (DF)

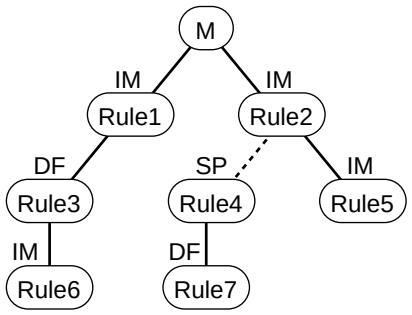

図 3.5: プロダクションルールによって生成されるプロセス木

は $Rule_5$ が処理を完了するまで待つ．すなわち，$Rule_3$ は $Rule_1$, $Rule_2$, $Rule_5$ の処理が完了した後に実行される．ここで，$Rule_3$ が $Rule_4$ の完了を待たないのは，$Rule_4$ が分離実行モードをもつためである．$Rule_7$ も延期実行モードをもつが，$Rule_4$ は他から分離しているため，$Rule_7$ の延期については $Rule_4$ 以外を考慮する必要はない．

3.6.2 CCSPR による解析

前小節で 7 つのプロダクションルールをもつスケジューラの例を紹介した．このスケジューラの振舞いは並行性と逐次性を合わせもっており，その振舞いを事前に正確に予測することは容易ではない．そこで，実際にこのスケジューラが図 3.6に示される

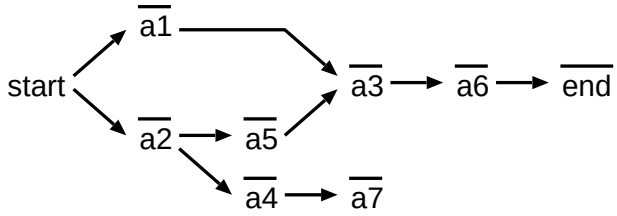

図 3.6: スケジュールされるアクションの期待される実行順序

順序で実行されるかを検証することは重要である．本小節では，スケジューラがどのように CCSPR によって記述され，解析されるかを説明する．

先ず各プロダクションルールを表 3.2に従って CCSPR によって次のように記述する．

$$
\begin{aligned}
Rule_1 &\stackrel{\text{def}}{=} e_1.Rule(A_1, IM), & Rule_5 &\stackrel{\text{def}}{=} e_4.Rule(A_5, IM),\\
Rule_2 &\stackrel{\text{def}}{=} e_1.Rule(A_2, IM), & Rule_6 &\stackrel{\text{def}}{=} e_6.Rule(A_6, IM),\\
Rule_3 &\stackrel{\text{def}}{=} e_3.Rule(A_3, DF), & Rule_7 &\stackrel{\text{def}}{=} e_7.Rule(A_7, DF),\\
Rule_4 &\stackrel{\text{def}}{=} e_4.Rule(A_4, SP),
\end{aligned}
$$

ここで，プロセス $Rule(P, Q)$ は次のように定義される略記法である．

$$
\begin{aligned}
Rule(P, Q) &\equiv (\llbracket s.P[d/done] | Q \rrbracket \setminus L)/H\\
L &= \{s, d, e_1, e_2, e_3, e_4, e_5, e_6, e_7\}\\
H &= \{\lceil c \rceil, \lceil sd \rceil, \lceil cd \rceil\}
\end{aligned}
$$

$Rule(P, Q)$ の P にルールが実行する動作を代入し，Q にその実行時期を制御するカップリングモードを代入する．アクション s が実行開始，d が実行完了に相当する．$done$ はプロセスの処理の終了を知らせるアクションである．P と Q はパック $\llbracket\ \rrbracket$ されているため，P によって呼び出されたルールは P と Q の両方の子プロセスとして処理される．

各ルールの動作 $A1, \cdots, A7$ は表 3.2 をもとに次のように定義される．

$$
\begin{aligned}
A_1 &\stackrel{\text{def}}{=} \overline{a_1}.\overline{e_3}.\overline{done}.\mathbf{0}, & A_5 &\stackrel{\text{def}}{=} \overline{a_5}.\overline{done}.\mathbf{0},\\
A_2 &\stackrel{\text{def}}{=} \overline{a_2}.\overline{e_4}.\overline{done}.\mathbf{0}, & A_6 &\stackrel{\text{def}}{=} \overline{a_6}.\overline{done}.\mathbf{0},\\
A_3 &\stackrel{\text{def}}{=} \overline{a_3}.\overline{e_6}.\overline{done}.\mathbf{0}, & A_7 &\stackrel{\text{def}}{=} \overline{a_7}.\overline{done}.\mathbf{0},\\
A_4 &\stackrel{\text{def}}{=} \overline{a_4}.\overline{e_7}.\overline{done}.\mathbf{0},
\end{aligned}
$$

また，IM, SP, DF はカップリングモードであり，各々即実行，分離実行，延期実行を表している．ここでは，親子アクションが子プロセスの処理の完了を親プロセスに

確実に伝えるために使われる．これは，延期実行モードのルールの実行開始時期を決めるために必要である．各カップリングモードは次のように定義される．

$$
\begin{aligned}
IM &\stackrel{\text{def}}{=} \overline{s}.d.\lceil c\rceil.\lfloor c\rfloor.\lfloor sd\rfloor.\lceil sd\rceil.\lceil cd\rceil.\lfloor cd\rfloor.\mathbf{I} \\
SP &\stackrel{\text{def}}{=} ((\overline{s}.d.\lceil c\rceil.\lceil sd\rceil.\lceil cd\rceil.\mathbf{0})|\mathbf{I}) \\
DF &\stackrel{\text{def}}{=} \lfloor c\rfloor.\lfloor sd\rfloor.\overline{s}.d.\lceil c\rceil.\lceil sd\rceil.\lceil cd\rceil.\lfloor cd\rfloor.\mathbf{I}
\end{aligned}
$$

親子アクション $\lceil c\rceil$ と $\lfloor c\rfloor$ は即実行モードのルールが実行完了したことを親に伝えるために使われ，$\lceil cd\rceil$ と $\lfloor cd\rfloor$ は延期実行モードのルールが実行完了したことを親に伝えるために使われる．親子アクション $\lceil sd\rceil$ と $\lfloor sd\rfloor$ は，延期実行モードのルールの実行開始を子に伝えるために使われる．

即実行モード IM では，すぐにルールの実行を開始 s し，処理完了 d を受け取ると，子プロセスの完了 $\lceil c\rceil$ を受け取ってから，親プロセスに処理完了 $\lfloor c\rfloor$ を伝える．延期実行モードのためのアクション $\lfloor sd\rfloor$, $\lceil sd\rceil$, $\lceil cd\rceil$, $\lfloor cd\rfloor$ は単に転送される．

分離実行モード SP では，$\mathbf{I}$ によって全ての子アクションが常に実行できるので，この親プロセスはこのルールの振舞いに全く影響を与えない．

延期実行モード DF では，まず即実行モードの完了 $\lfloor c\rfloor$ を親に伝え，延期実行モードの開始 $\lfloor sd\rfloor$ を待つ．実行後は，子プロセスの完了 $\lceil c\rceil$ を受け取ってから，子プロセスに延期実行モードの開始を伝える．これは，延期実行モードのルールの下にも延期実行モードのルールが呼び出されることがあるためである．

ここで重要なことは，上記のカップリングモードの記述がルールの振舞いに依存していないことである．すなわち，この記述はどのようなルールに対しても修正することなく利用することができる．このような記述は従来のプロセス代数では困難である．

最後に，各ルールを実行する環境を次のようにシステム SYS として与える．

$$
\begin{aligned}
SYS &\equiv R \triangleright ((M \backslash L)/H) \\
R &\equiv \{\!\{Rule_1\}\!\} :: \{\!\{Rule_2\}\!\} :: \{\!\{Rule_3\}\!\} :: \{\!\{Rule_4\}\!\} :: \{\!\{Rule_5\}\!\} :: \{\!\{Rule_6\}\!\} :: \{\!\{Rule_7\}\!\} \\
M &\stackrel{\text{def}}{=} start.\overline{e_1}.\lceil c\rceil.\lceil sd\rceil.\lceil cd\rceil.\overline{end}.\overline{done}.\mathbf{0}
\end{aligned}
$$

プロセス M はルールを管理するプロセスである．

次に，図 3.6に示されるアクションの期待される実行順序を記述する．

$$
ORDER \stackrel{\text{def}}{=} ev1.(EA_1 ||| (EA_2; (EA_5 ||| \langle\!\langle EA_4; EA_7 \rangle\!\rangle)); EA_3; EA_6; END)
$$

ここで，EA_i はスケジュールされるアクションを実行する次のプロセスであり，

$$\begin{aligned} EA_1 &\stackrel{\text{def}}{=} \overline{a_1}.\overline{done}.\mathbf{0}, & EA_5 &\stackrel{\text{def}}{=} \overline{a_5}.\overline{done}.\mathbf{0}, \\ EA_2 &\stackrel{\text{def}}{=} \overline{a_2}.\overline{done}.\mathbf{0}, & EA_6 &\stackrel{\text{def}}{=} \overline{a_6}.\overline{done}.\mathbf{0}, \\ EA_3 &\stackrel{\text{def}}{=} \overline{a_3}.\overline{done}.\mathbf{0}, & EA_7 &\stackrel{\text{def}}{=} \overline{a_7}.\overline{done}.\mathbf{0}, \\ EA_4 &\stackrel{\text{def}}{=} \overline{a_4}.\overline{done}.\mathbf{0}, & END &\stackrel{\text{def}}{=} \overline{end}.\overline{done}.\mathbf{0} \end{aligned}$$

プロセス $P;Q$，$P|||Q$，$\langle\!\langle P \rangle\!\rangle$ は次のように与えられるプロセスの略記法である．

$$\begin{aligned} P;Q &\equiv (P[d/done]|d.Q)\backslash\{d\} \\ P|||Q &\equiv (P[d_1/done]|Q[d_2/done]|d_1.d_2.\overline{done})\backslash\{d_1,d_2\} \\ \langle\!\langle P \rangle\!\rangle &\equiv (P[d/done]|d.\mathbf{0}|\overline{done}.\mathbf{0})\backslash\{d\} \end{aligned}$$

$P;Q$ はプロセス P と Q を逐次的に実行するプロセスである．$P|||Q$ はプロセス P と Q を並行に実行するプロセスであり，両方のプロセスが終了したとき $\overline{done}$ を実行する．$\langle\!\langle P \rangle\!\rangle$ はプロセス P の分離実行を表す．

CCSPR では，命題 3.5.5と命題 3.5.6の展開規則を用いて CCSPR の記述を CCS の逐次的な記述に展開してから，CCS の枠組で検証することができる．実際に，上記の SYS と振舞いが強等価な CCS の記述 SYS_{ccs} を得ることができる．一方，$ORDER$ は既に CCS の記述でもある．そして，

$$SYS_{ccs} =_{ccs} ORDER$$

を CCS の検証ツール (Concurrency workbench)[38] により確認した．ここで，$=_{ccs}$ は観測合同 (定義 2.4.11) である．プロセスの振舞いの状態数が有限であれば，CCSPR の記述を CCS の記述に自動的に変換することは可能であり，CCSPR によるプロダクションルールの記述し易さと，CCS の検証能力の両方を利用できる．

3.7 おわりに

本章では，カップリングモードをもつアクティブデータベースのプロダクションルールの振舞いを検証するためのプロセス代数として CCSPR を提案した．CCSPR では，成長するプロセス木を容易に表現でき，そのプロセス木において親子間の 1 対他局所通信が可能である．この特徴は，カップリングモードをもつアクティブデータベース SAMOS[16] や HiPAC[37] の振舞い記述に適している．

3.6.2節では，例をもとにプロダクションルールの解析方法を示した．CCSPR では，プロセスの呼び出しによる親子関係は自動的に構築されて維持されるため，ルールを記述する際には考慮する必要はない．CCSPR によって記述されたシステムを，それと強等価な CCS の記述に変換することは可能であり，システムの検証に CCS のツールを利用することができる．

従来のプロセス代数でもプロダクションルールの振舞いを検証可能であるが，CCSPR の利点はその記述し易さにある．検証に従来のプロセス代数を利用する場合でも，先ず CCSPR で記述してから従来の記述に自動変換することによって，設計者の記述の負担を軽減することができる．

第 4 章

可算ブロードキャスト用プロセス代数

4.1 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エージェント (プロセス) に機能別に分割し，プロセス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** [10] 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. 各プロセス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プロセス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，VIABUS[50] と呼ばれるプロセスの接続メカニズムが開発され，VIABUS 上にメールシステムやニュースシステムが構築されてきた．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プロセスの追加や修正が可能である．例えば，利用者の要求に応じてユーザーインターフェース等を随時変更することができる．VIABUS では次のようなブロードキャスト通信方式を採用している．

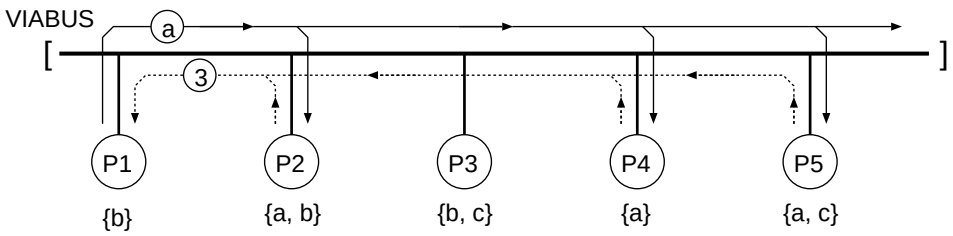

図 4.1: VIABUS の通信例

1. メッセージは宛先の代わりに名前をもっている．

2. 各プロセスは受信を希望するメッセージの名前を宣言する．

3. 送信してから，その受信者数が返信されるまでが一つの命令である．

つまり，メッセージの受信は各プロセスに委ねられている．また，この受信を希望するメッセージは動的に (実行時に) 変更可能である．図 4.1に VIABUS における通信の例を示す．P1 から P5 まではプロセスを表し，各々VIABUS に接続されている．各プロセスの下に書かれている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 P1 が名前 a をもつメッセージをブロードキャストすると，P2, P4, P5 がそれを受信して，その受信者数として 3 が送信者 P1 に返信されている．

利用者が自由にプロセスを追加や修正できる場合，あるメッセージの受信者数を固定することはできない．また，1 つのプロセスにおいても，状況に応じて受信を希望するメッセージ名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，あるプロセスがメッセージをブロードキャストして複数のプロセスを起動した後，起動された全てのプロセスの終了を待ってから次の処理に進むような場合，そのメッセージの受信者数 (起動されたプロセスの数) を知ることは重要である．

VIABUS は実際には図 4.2のような星型の構造で実装されており，外部からは観測されない中心のプロセス 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．このプロセス C は各メッセージの受信者数の情報を所有しており，VIABUS ではブロードキャストしてからその受信者数が返信されるまでを一つの命令で表現できる．このような受信者数を考慮したブロードキャストを**可算ブロードキャスト** (countable broadcast) と呼ぶ．

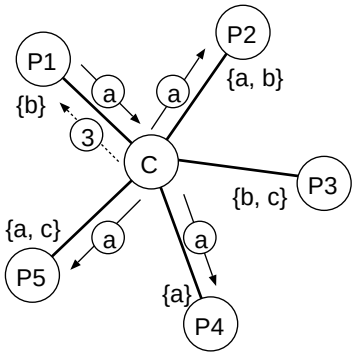

図 4.2: VIABUS の実装 (比較:図 4.1)

本章では，VIABUS 上に構築されたシステムの解析に適したプロセス代数として，可算ブロードキャストを一つのアクションで表現できる CCB (a Calculus of Countable Broadcasting Systems) を提案する．ここで，CCS の観測合同 (定義 2.4.11) $=_{ccs}$ が，CCB に対しては合同関係にならないことが重要である．そこで，CCB のための観測的な合同関係として**監視合同**を定義し，有限プロセスに対する監視合同の公理系を与える．

以下，4.2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．4.3節では CCB の特長を例を用いて説明する．4.4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，4.5節で Core-CCB のための観測的な合同関係として監視合同を定義してその特性を示す．さらに有限プロセスに対する監視合同の完全で健全な公理系を与える．4.6節で Core-CCB を基に CCB を定義する．

4.2 プロセス代数における通信方式

2章で紹介したように，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，動作の等価性を形式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，可算ブロードキャストを記述するための問題点を指摘する．

4.2.1 1 対 1 通信

1 対 1 通信をもつプロセス代数として CCS[38]，π-計算 [39]，CHOCS[53] 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 [19] 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 の受信者を数えようとしてうまくいかないシステムの例である．

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

ここで，各プロセスは次のように定義されている．

$$\begin{aligned} C(i) &\stackrel{\text{def}}{=} \overline{a}.C(i+1) + \overline{out}(i).\mathbf{0} \\ P_1 &\stackrel{\text{def}}{=} a.\mathbf{0} \\ P_2 &\stackrel{\text{def}}{=} b.\mathbf{0} \\ P_3 &\stackrel{\text{def}}{=} a.\mathbf{0} \end{aligned}$$

プロセス $C(i)$ はメッセージ a を送信するごとに値変数 i に 1 を加算して，メッセージ out を用いてその値を出力するプロセスである．2 つのプロセス P_1 と P_3 がメッセージ a を受信可能であるので，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．

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

ここで，$\sim$ は定義 2.4.2の強等価である．これは，$C(i)$ がメッセージ a を P_1 と P_3 に送信する前に，メッセージ 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 [2] を導入して，メッセージ a をメッセージ out より優先することが考えられる．しかし，この方法では，プロセス P_1 が $P_1 \stackrel{\text{def}}{=} a.P_1$ のように再帰的に定義されていた場合，メッセージ a の優先によって無限ループにおちいる．実際にはこの場合でも 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 (π-計算,CHOCS) を用いたとしても残される．

4.2.2 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[40]，Meije[51] などがある．ここで，マルチキャストとは，送信者が受信者数を指定してメッセージを送信する通

信方式である．次の例は SCCS でプロセス P_0 がプロセス P_1 と P_3 にマルチキャストする例である．

$$P_0|P_1|P_2|P_3 \xrightarrow{\tau} P_0'|P_1'|P_2|P_3'$$

ここで，各プロセスは次のように定義されている．

$$P_0 \stackrel{\text{def}}{=} a^{-2}.P_0', \qquad P_1 \stackrel{\text{def}}{=} a^{1}.P_1', \qquad P_2 \stackrel{\text{def}}{=} b^{1}.P_2', \qquad P_3 \stackrel{\text{def}}{=} a^{1}.P_3'$$

P_0 のアクション a^{-2} の -2 は，P_0 が 2 つのプロセスにマルチキャストすることを明示している (負は送信を表す)．すなわち，メッセージ a を要求するプロセスを新しく 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．

4.2.3 ブロードキャスト

ブロードキャストが記述可能なプロセス代数として CSP[18]，LOTOS[56]，CBS[43, 44, 19]，CCS+b[20] などがあり，特に CBS と CCS+b はブロードキャストのために提案されたプロセス代数である．ブロードキャストは，送信者が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通信方式である．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．

$$(P_0|P_1|P_2|P_3) \xrightarrow{a!} (P_0'|P_1'|P_2|P_3')$$

ここで，各プロセスは次のように定義されている．

$$P_0 \stackrel{\text{def}}{=} a!.P_0', \qquad P_1 \stackrel{\text{def}}{=} a?.P_1' + b?.P_1''$$
$$P_2 \stackrel{\text{def}}{=} b?.P_2', \qquad P_3 \stackrel{\text{def}}{=} a?.P_3' + c?.P3''$$

ここで，アクションの属性 ! と ? は各々メッセージの送信と受信を表している．すなわち，P_1, P_2, P_3 が受信を希望するメッセージの集合は，各々$\{a, b\}$, $\{b\}$, $\{a, c\}$ である．このように，CBS により適切にブロードキャスト通信を記述することができる．

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．タイムアウトを用いて受信者からの返信を数えることも考えられるが，CBS は時間の概念をもたないため，返信を数えるための有限時間を記述できない．CBS に時間の概念 [41] を導入することによって返信を数えることが可能になるが，可算ブロードキャストごとにこの返信動作を解析するため，必要以上に解析コストは高くなる問題がある．

4.3 CCB の紹介

4.2節で述べた問題を解決するためにプロセス代数 CCB を提案する．CCB は可算ブロードキャストを一つのアクションで表現できる特長をもち，VIABUS 上に構築されたシステムの解析に適している．また，受信者数を指定するマルチキャストは，受信者数が指定した数になるまで可算ブロードキャストを繰り返すことによって模倣できるが，解析コストを低くするために，CCB ではマルチキャストのためのアクションも用意している．本節では，CCB の概要を説明する．

CCB のアクションは $a[x]\theta\langle y\rangle(z)$ 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，θ はアクションの属性，$\langle y\rangle$ は受信者数，(z) はこのアクションによって渡されるメッセージである．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 $a_1[x_1]$ と $a_2[x_2]$ が等しいとは，その基本メッセージ名とポストフィクスが各々等しい場合である $(a_1 = a_2, x_1 = x_2)$．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 $a[x]$ と呼ぶ．π 計算のように通信の制限範囲の動的な変更は表現できないが，ポストフィクスには変数が使えるのでプロセス間の結合を実行時に変更可能である．以下，しばしば，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 ( ) を省略する．

アクションの属性 θ には，マルチキャストと可算ブロードキャストの送受信に対応する $!, !!, ?, ??$ の 4 種類がある．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アクションである．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プロセス数を指定する定数である．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**内部アクション**と呼ぶ (CCS の τ に相当する)．便宜

上，任意の内部アクションを表すための変数として $\tau, \tau', \cdots$ を用いる．観測的な解析を行なうためには，内部アクションは可能な限り無視されるべきである．

2. $a[i]!!\langle x\rangle(m)$ はメッセージ $a[i]$ を可算ブロードキャストするために使われるアクションである．m はこの通信で渡されるメッセージの内容であり，このメッセージの受信者数は変数 x に代入される．

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アクションである．n は正の整数で 2 以上を指定することにより 1 つのアクションで複数のプロセスがこのメッセージを受信することを表せる．この通信で受けとるメッセージの内容は変数 y に代入される．

4. $a[i]??\langle n\rangle(y)$ は可算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アクションである．n 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 の通信方式に同じになるように定義されている．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 つのプロセスが受信できる通信方式である．

送信についても，マルチキャストを $a!\langle$ 定数 $\rangle$，ブロードキャストを $a!\langle$ 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定数が代入されていれば，受信者数に変数を使うことができるからである．例えば，

$$a?\langle 1\rangle(x).b!\langle x\rangle(message).\mathbf{0}$$

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，CCB の記述例を示し，その特徴を説明する．

例 4.3.1 (**マルチキャスト**) 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プロセス $P \stackrel{\text{def}}{=} a!\langle 2\rangle.P'$ と，マルチキャストされたメッセージ a を受信するプロセス

$Q \stackrel{\text{def}}{=} a?.Q'$ について，次の等式が成り立つ．

$$
\begin{aligned}
(P|Q)\backslash\{a\} &\sim \mathbf{0} \\
(P|Q|Q)\backslash\{a\} &\sim \tau.(P'|Q'|Q')\backslash\{a\} \\
(P|Q|Q|Q)\backslash\{a\} &\sim \tau.(P'|Q'|Q'|Q)\backslash\{a\} + \tau.(P'|Q'|Q|Q')\backslash\{a\} + \tau.(P'|Q|Q'|Q')\backslash\{a\}
\end{aligned}
$$

こ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プロセスが存在する．どのプロセスが受信できないかは非決定的である．∎

例 4.3.2 (**可算ブロードキャスト**) メッセージ a を可算ブロードキャストするプロセス $P \stackrel{\text{def}}{=} a!!\langle x\rangle.P'(x)$ と，可算ブロードキャストされたメッセージ a を受信するプロセス $Q \stackrel{\text{def}}{=} a?.Q'$ について，次の等式が成り立つ．

$$
\begin{aligned}
(P|Q)\backslash\{a\} &\sim \tau.(P'(1)|Q')\backslash\{a\} \\
(P|Q|Q)\backslash\{a\} &\sim \tau.(P'(2)|Q'|Q')\backslash\{a\} \\
(P|Q|Q|Q)\backslash\{a\} &\sim \tau.(P'(3)|Q'|Q'|Q')\backslash\{a\}
\end{aligned}
$$

こ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 には受信者数が代入されている．∎

例 4.3.3 (**ポストフィクス**) プロセス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．次の記述は，P_1 からの情報 ‘No1’ を Q_1 へ，P_2 からの情報 ‘No2’ を Q_2 へ中継するプロセス M の例である．

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

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 にその ID を代入する．次に変数 x にメッセージ $in[i]$ によって受けとった内容を代入し，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．

$$
\begin{aligned}
(P_1|P_2|M|Q_1|Q_2)\backslash\{id,in,out\} \sim \quad & \tau.\tau.\tau.(\mathbf{0}|P_2|M|Q_1'(\text{`No1'})|Q_2)\backslash\{id,in,out\} \\
& + \tau.\tau.\tau.(P_1|\mathbf{0}|M|Q_1|Q_2'(\text{`No2'}))\backslash\{id,in,out\}
\end{aligned}
$$

∎

例 4.3.4（**まとめ**）「メッセージを可算ブロードキャストしたプロセスに，そのメッセージを受信した全てのプロセスが自分のプロセス ID(P-ID) を返信する動作」を記述した例を次に示す．

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

ここで，**if** b **then** P は条件分岐を表しており，真偽式 b が真 $(true)$ ならばエージェント P のように振舞う．また，::はリストの結合演算子である．

P_1 はメッセージ a を用いて自分の P-ID id を可算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 に代入し，その P-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 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 を代入して，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 Q_1 を複製する．

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． ∎

4.4 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 の構文と意味を形式的に定義する．CCB では，アクション $a[x]\theta\langle y\rangle(z)$ の x, y, z に見られるように変数が使われているが，Core-CCB はそのような変数をもたないプロセス代数である．CCB の意味は Core-CCB を基にして 4.6節で定義される．

4.4.1 構文

この小節では Core-CCB の構文を定義する．まず，名前の有限集合 $\mathcal{N} = \{a, b, c, \cdots\}$ が与えられていると仮定する．このとき，アクションの集合 Act は次ように与える．

$$Act = \{a\theta^n : a \in \mathcal{N}, \theta \in \Theta, n \in \mathcal{I}\} \cup \{a!^0, a!!^0 : a \in \mathcal{N}\}$$

ここで，$\mathcal{I}$ は正の整数の無限集合 $\{1, 2, \cdots\}$，Θ はアクション属性の集合 $\{!, ?, !!, ??\}$ である．Θ の要素は $\theta, \phi, \cdots$ によって表される．アクションを表すためには変数 $\alpha, \beta, \cdots$ を用いるが，アクションの名前や属性を考慮するときは $a\theta^n, b\phi^m, \cdots$ の形を用いる．次に Act の部分集合として集合 $\mathcal{T}$ を次のように与える．

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

この $\mathcal{T}$ の要素を**内部アクション**と呼び，内部アクションを表すために変数 $\tau, \tau', \tau_1, \cdots$ を用いる．また，名前の集合 $\mathcal{N}$ から $\mathcal{N}$ への関数を**リネイミング関数**と呼び，名前 a を b に変更するリネイミング関数を (b/a) と記述する．

このとき，Core-CCB の構文を次のように定義する．

定義 4.4.1 Core-CCB の**プロセス式** ($E, F, \cdots$ で表す) の集合 $\mathcal{E}_{core}$ は次の式を含む最小の集合である

$$\begin{aligned}
X &: \text{プロセス変数 } (X \in \mathcal{X}_{core}) \\
A &: \text{プロセス定数 } (A \in \mathcal{K}_{core}) \\
\mathbf{0} &: \text{無動作プロセス} \\
\alpha.E &: \text{プレフィクス } (\alpha \in Act) \\
E + F &: \text{選択} \\
E|F &: \text{並行合成} \\
E[f] &: \text{リネイミング } (f : \text{リネイミング関数}) \\
E\backslash L &: \text{制限 } (L \subseteq \mathcal{N})
\end{aligned}$$

ここで，E, F はすでに $\mathcal{E}_{core}$ の要素であるとする．$\mathcal{X}_{core}$ と $\mathcal{K}_{core}$ は各々プロセス変数とプロセス定数の集合であり，与えられているとする．∎

プロセス変数を含まないプロセス式を**プロセス** ($P, Q, R, \cdots$ で表す) と呼び，プロセスの集合を $\mathcal{P}_{core}$ で表す．プロセス定数は**定義式**によって意味を与えられるプロセスであり，実際に全てのプロセス定数 A について，$A \stackrel{\text{def}}{=} P$ の形の定義式があると仮定する ($P \in \mathcal{P}_{core}$)．さらに，P の中で A は**弱ガード されている**と仮定する (CCS の弱ガードの定義 2.4.5に同様である)．この仮定は各プロセスが受信を希望しているメッセージの名前を決定するために必要であり，CBS[43] でも同様の仮定が用いられている．

また，しばしば，$P_1 + P_2 + \cdots + P_n$ を表すために，$\sum_{i\in\{1,\cdots,n\}} P_i$ の記法を用いる．特に $n = 0$ の場合，$\sum_{i\in\emptyset} P_i$ は $\mathbf{0}$ を表す．

4.4.2 意味

まず，プロセス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．直観的には，$mon(E)$ はプロセス式 E が (可算ブロードキャストに対して) 現在受信可能なアクションの名前の集合である．

定義 4.4.2　各プロセス式について，**監視関数** $mon : \mathcal{E}_{core} \to 2^{\mathcal{N}}$ を次のように帰納的に定義する．

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

∎

プロセス定数は弱ガードされているので，この関数 mon は計算可能である．

このとき，Core-CCB の意味を次のように与える．

名前	仮定	$\vdash$	結果
Act		$\vdash$	$\alpha.E \xrightarrow{\alpha} E$
Choice$_1$	$E \xrightarrow{\alpha} E'$	$\vdash$	$E + F \xrightarrow{\alpha} E'$
Choice$_2$	$F \xrightarrow{\alpha} F'$	$\vdash$	$E + F \xrightarrow{\alpha} F'$
Com$_1$	$E \xrightarrow{a\theta^n} E',\ (\theta \in \{!,?\}$ or $a \notin mon(F))$	$\vdash$	$E\|F \xrightarrow{a\theta^n} E'\|F$
Com$_2$	$F \xrightarrow{a\theta^n} F',\ (\theta \in \{!,?\}$ or $a \notin mon(E))$	$\vdash$	$E\|F \xrightarrow{a\theta^n} E\|F'$
Com$_3$	$E \xrightarrow{a\theta^m} E',\ F \xrightarrow{a\phi^n} F',\ \theta \in \{!,\!!\},\ \phi = com(\theta),\ m \geq n$	$\vdash$	$E\|F \xrightarrow{a\theta^{(m-n)}} E'\|F'$
Com$_4$	$E \xrightarrow{a\phi^n} E',\ F \xrightarrow{a\theta^m} F',\ \theta \in \{!,\!!\},\ \phi = com(\theta),\ m \geq n$	$\vdash$	$E\|F \xrightarrow{a\theta^{(m-n)}} E'\|F'$
Com$_5$	$E \xrightarrow{a\theta^m} E',\ F \xrightarrow{a\theta^n} F',\ \theta \in \{?,?\!?\}$	$\vdash$	$E\|F \xrightarrow{a\theta^{(m+n)}} E'\|F'$
Ren	$E \xrightarrow{a\theta^n} E'$	$\vdash$	$E[f] \xrightarrow{f(a)\theta^n} E'[f]$
Res$_1$	$E \xrightarrow{a\theta^n} E',\ a \notin L$	$\vdash$	$E\backslash L \xrightarrow{a\theta^n} E'\backslash L$
Res$_2$	$E \xrightarrow{a\theta^0} E',\ \theta \in \{!,\!!\},\ a \in L$	$\vdash$	$E\backslash L \xrightarrow{a!^0} E'\backslash L$
Rec	$P \xrightarrow{\alpha} P',\ A \stackrel{\text{def}}{=} P$	$\vdash$	$A \xrightarrow{\alpha} P'$

図 4.3: プロセス式上のラベル付遷移の推論規則

定義 4.4.3　Core-CCB の意味は次のラベル付遷移システムによって与えられる．

$$(\mathcal{E}_{core}, Act, \longrightarrow)$$

ここで，遷移の集合 $\longrightarrow$ は図 4.3の推論規則を満たす最小の関係であり，com は次のように定義されるアクションの属性上の関数 $(com : \Theta \to \Theta)$ である．

$$com(!) =?,\ \ com(?) =!,\ \ com(\!!) =?\!?,\ \ com(?\!?) =\!!$$

関数 com は各属性について，通信の対になる属性を返す関数である． ∎

この定義のもとで次の命題が成り立つ．

命題 4.4.1　$a \in mon(E) \iff$ ある n と E' について $E \xrightarrow{a?^n} E'$

証明　$(\Rightarrow)$: $a \in mon(E)$ とする．E の構造に関する帰納法を用いて証明する．最も興味深い $E \equiv E_1|E_2$ の場合のみ示す．他の場合も同様である．$a \in mon(E_1|E_2)$ を仮定すると mon の定義より $a \in mon(E_1) \cup mon(E_2)$ である．よって，次の 3 つの場合が考えられる．

1. $a \in mon(E_1)$ かつ $a \notin mon(E_2)$ の場合：帰納法の仮定より，ある n と E_1' について $E_1 \xrightarrow{a\overset{?}{!}^n} E_1'$ である．$a \notin mon(E_2)$ であるので，$\mathbf{Com}_1$ より $E_1|E_2 \xrightarrow{a\overset{?}{!}^n} E_1'|E_2$ を導ける．

2. $a \notin mon(E_1)$ かつ $a \in mon(E_2)$ の場合：上の場合と対称に $\mathbf{Com}_2$ を用いる．

3. $a \in mon(E_1)$ かつ $a \in mon(E_2)$ の場合：帰納法の仮定より，ある n_1, n_2,E_1', E_2' について $E_1 \xrightarrow{a\overset{?}{!}^{n_1}} E_1'$ かつ $E_2 \xrightarrow{a\overset{?}{!}^{n_2}} E_2'$ である．$\mathbf{Com}_5$ より $E_1|E_2 \xrightarrow{a\overset{?}{!}^{(n_1+n_2)}} E_1'|E_2'$ を導ける．

($\Leftarrow$) : $E \xrightarrow{a\overset{?}{!}^n} E'$ ならば $a \in mon(E)$ を，この遷移を導いた規則の数に関する帰納法を用いて証明する．上記の ($\Rightarrow$) の場合と同様に $E \equiv E_1|E_2$ の場合のみ示す．$E_1|E_2 \xrightarrow{a\overset{?}{!}^n} E'$ を導いた最後の規則について場合分けをする．

1. $\mathbf{Com}_1$ による場合：ある E_1' について，$E_1 \xrightarrow{a\overset{?}{!}^n} E_1'$ かつ $a \notin mon(E_2)$ かつ $E' \equiv E_1'|E_2$ を得る．帰納法の仮定より $a \in mon(E_1)$ である．よって，$a \in mon(E_1) \cup mon(E_2) = mon(E_1|E_2)$ を得る．

2. $\mathbf{Com}_2$ による場合：$\mathbf{Com}_1$ の場合と対称である．

3. $\mathbf{Com}_{3,4}$ による場合：これらの推論規則によって得られる遷移上のアクションの属性は ! か ‼ のみであり，$a\overset{?}{!}^n$ を導くことはない．

4. $\mathbf{Com}_5$ による場合：ある E_1' と E_2' について，$E_1 \xrightarrow{a\overset{?}{!}^{n_1}} E_1'$ かつ $E_2 \xrightarrow{a\overset{?}{!}^{n_2}} E_2'$ かつ $n_1 + n_2 = n$ かつ $E' \equiv E_1'|E_2'$ を得る．帰納法の仮定より $a \in mon(E_1)$ かつ $a \in mon(E_2)$ である．よって，$a \in mon(E_1) \cup mon(E_2) = mon(E_1|E_2)$ を得る．

∎

つまり，推論規則 $\mathbf{Com}_{1,2}$ の条件 $a \notin mon(F)$ は，可算ブロードキャストされたアクション a の受信を F が希望していないことを表している．

4.5 Core-CCB の等価性

Core-CCB に対しても，CCS と同様に観測的な合同関係が必要である．しかし，CCS で定義された**観測合同** (定義 2.4.11) $=_{ccs}$ 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．

$$P_1 =_{ccs} P_2, \qquad P_1|Q \neq_{ccs} P_2|Q$$

ここで，各プロセス P_1, P_2, Q は次のように定義されている．

$$\begin{aligned} P_1 &\stackrel{\text{def}}{=} a?^1.\tau.P_1 \\ P_2 &\stackrel{\text{def}}{=} a?^1.P_2 \\ Q &\stackrel{\text{def}}{=} a!!^0.out0!^1.Q + a!!^1.out1!^1.Q \end{aligned}$$

τ は内部アクションであるので，P_1 と P_2 は観測合同である．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．これは，P_2 は常に a を受信可能であるのに対し，P_1 は τ によって受信できないときがあるためである．そこで，観測合同に最も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．

以下，まず，4.5.1小節では，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として**監視等価**を定義する．次に，4.5.2小節では，監視等価に含まれる最大の合同関係として**監視合同**を定義する．そして，4.5.3小節では，有限プロセスに対する監視合同の健全で完全な公理系を与える．

4.5.1 監視等価

まず，監視等価の定義のために必要な記法を準備する．Act^* を空列 ε を含む有限アクション列の集合とする．$t = \alpha_1 \cdots \alpha_k \in Act^*$ について，もし $E \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_k} E'$ ならば，$E \xrightarrow{t} E'$ と書く．特に，$E \xrightarrow{\varepsilon} E'$ ならば $E' \equiv E$ である．また，アクション列 t に含まれるアクションが全て内部アクション $(t = \tau_1 \cdots \tau_k)$ のとき，$E \xrightarrow{t} E'$ ならば，$E \Longrightarrow E'$ と書く．このとき，次の状態遷移関係を定義する．

定義 4.5.1　$t = \alpha_1 \cdots \alpha_k \in Act^*$ とする．もし，$E \Longrightarrow \xrightarrow{\alpha_1} \Longrightarrow \cdots \Longrightarrow \xrightarrow{\alpha_k} \Longrightarrow E'$ ならば，$E \stackrel{t}{\Longrightarrow} E'$ である． ∎

次に，CCS の**弱双模倣** (定義 2.4.9) に相当する Core-CCB の双模倣を定義する．

定義 4.5.2　プロセス上の二項関係 $\mathcal{S}$ が**監視双模倣** (monitor bisimulation) であるとは，$(P, Q) \in \mathcal{S}$ ならば，任意の $\alpha \in Act$ 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．

(i)　$P \xrightarrow{\alpha} P'$ ならば，ある Q' について，$Q \overset{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $(P', Q') \in \mathcal{S}$ を満たす．

(ii)　$Q \xrightarrow{\alpha} Q'$ ならば，ある P' について，$P \overset{\widehat{\alpha}}{\Longrightarrow} P'$ かつ $(P', Q') \in \mathcal{S}$ を満たす．

(iii)　$a \notin mon(P)$ ならば，ある Q', Q'' について，

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

(iv)　$a \notin mon(Q)$ ならば，ある P', P'' について，

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

ここで，$\widehat{t}$ はアクション列 t から内部アクションを全て除いたアクション列である． ∎

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．この監視双模倣を用いて監視等価を定義する．

定義 4.5.3　もしある監視双模倣 $\mathcal{S}$ において $(P, Q) \in \mathcal{S}$ ならば，プロセス P と Q は**監視等価** (monitor equivalence) であるといい，$P \approx_m Q$ と書く． ∎

次の命題 4.5.1と命題 4.5.2に示されるように，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 4.5.1　$P_1 \approx_m P_2$ ならば，任意の Q について $P_1|Q \approx_m P_2|Q$ である．

証明　次の集合 $\mathcal{S}$ が監視双模倣であることを示す．

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

$(P_1|Q, P_2|Q) \in \mathcal{S}$ とする．すなわち，$P_1 \approx_m P_2$ である．

(i) $P_1|Q \xrightarrow{a\theta^n} P'$ とする．この遷移を導く最後の規則について場合分けする．

1. $\mathbf{Com}_1$ による場合：ある P_1' について，$P_1 \xrightarrow{a\theta^n} P_1'$ かつ ($\theta \in \{!, ?\}$ または $a \notin mon(Q)$) かつ $P' \equiv P_1'|Q$ を得る．仮定 $P_1 \approx_m P_2$ より，ある P_2' について，$P_2 \overset{\widehat{a\theta^n}}{\Longrightarrow} P_2'$ かつ $P_1' \approx_m P_2'$ を得る．この遷移に $\mathbf{Com}_1$ を 0 回以上適用して，$P_2|Q \overset{\widehat{a\theta^n}}{\Longrightarrow} P_2'|Q$ を導く．また，$P_1' \approx_m P_2'$ であるので，$(P_1'|Q, P_2'|Q) \in \mathcal{S}$ を得る．

2. $\mathbf{Com}_2$ による場合：ある Q' について，$Q \xrightarrow{a\theta^n} Q'$ かつ ($\theta \in \{!, ?\}$ または $a \notin mon(P_1)$) かつ $P' \equiv P_1|Q'$ を得る．次の 2 つの場合に分けて証明する．

- $\theta \in \{!, ?\}$ の場合：$\mathbf{Com}_2$ より $P_2|Q \xrightarrow{a\theta^n} P_2|Q'$ を導く．また，$P_1 \approx_m P_2$ であるので，$(P_1|Q', P_2|Q') \in \mathcal{S}$ を得る．
- $a \notin mon(P_1)$ の場合：仮定 $P_1 \approx_m P_2$ より，ある P_2' と P_2'' について，$P_2 \Rightarrow P_2' \Rightarrow P_2''$ かつ $a \notin mon(P_2')$ かつ $P_1 \approx_m P_2''$ である．これらの遷移に $\mathbf{Com}_1$ を 0 回以上，$\mathbf{Com}_2$ を 1 回適用して $P_2|Q \Rightarrow P_2'|Q \xrightarrow{a\theta^n} P_2'|Q' \Rightarrow P_2''|Q'$ を導く．また，$P_1 \approx_m P_2''$ であるので，$(P_1|Q', P_2''|Q') \in \mathcal{S}$ を得る．

3. $\mathbf{Com}_3$ による場合：ある ϕ, n_1, n_2, P_1', Q' について，$P_1 \xrightarrow{a\theta^{n_1}} P_1'$ かつ $Q \xrightarrow{a\phi^{n_2}} Q'$ かつ $\theta \in \{!, ‼\}$ かつ $\phi = com(\theta)$ かつ $n_1 \geq n_2$ かつ $n = n_1 - n_2$ かつ $P' \equiv P_1'|Q'$ を得る．仮定 $P_1 \approx_m P_2$ より，ある P_2' について，$P_2 \overset{\widehat{a\theta^{n_1}}}{\Longrightarrow} P_2'$ かつ $P_1' \approx_m P_2'$ を得る．この遷移と $Q \xrightarrow{a\phi^{n_2}} Q'$ に $\mathbf{Com}_1$ を 0 回以上，$\mathbf{Com}_3$ を 1 回適用して，$P_2|Q \overset{a\theta^{\widehat{(n_1-n_2)}}}{\Longrightarrow} P_2'|Q'$ を導く．また，$P_1' \approx_m P_2'$ であるので，$(P_1'|Q', P_2'|Q') \in \mathcal{S}$ を得る．

4. $\mathbf{Com}_4$ による場合：$\mathbf{Com}_3$ の場合に同様である．

5. $\mathbf{Com}_5$ による場合：ある n_1, n_2, P_1', Q' について，$P_1 \xrightarrow{a\theta^{n_1}} P_1'$ かつ $Q \xrightarrow{a\theta^{n_2}} Q'$ かつ $\theta \in \{?, ⁇\}$ かつ $n = n_1 + n_2$ かつ $P' \equiv P_1'|Q'$ を得る．仮定 $P_1 \approx_m P_2$ より，ある P_2' について，$P_2 \overset{\widehat{a\theta^{n_1}}}{\Longrightarrow} P_2'$ かつ $P_1' \approx_m P_2'$ を得る．この遷移と $Q \xrightarrow{a\theta^{n_2}} Q'$ に $\mathbf{Com}_1$ を 0 回以上，$\mathbf{Com}_5$ を 1 回適用して，$P_2|Q \overset{a\theta^{\widehat{(n_1+n_2)}}}{\Longrightarrow} P_2'|Q'$ を導く．また，$P_1' \approx_m P_2'$ であるので，$(P_1'|Q', P_2'|Q') \in \mathcal{S}$ を得る．

(iii) $a \notin mon(P_1|Q) = mon(P_1) \cup mon(Q)$ とする．すなわち，$a \notin mon(P_1)$ かつ $a \notin mon(Q)$ である．仮定 $P_1 \approx_m P_2$ より，ある P_2' と P_2'' について，$P_2 \Rightarrow P_2' \Rightarrow P_2''$ かつ $a \notin mon(P_2')$ かつ $P_1 \approx_m P_2''$ である．この遷移に $\mathbf{Com}_1$ を 0 回以上適用して，$P_2|Q \Rightarrow P_2'|Q \Rightarrow P_2''|Q$ を導く．このとき，$a \notin mon(P_2') \cup mon(Q) = mon(P_2'|Q)$ である．また，$P_1 \approx_m P_2''$ であるので，$(P_1|Q, P_2''|Q) \in \mathcal{S}$ を得る．

(ii) と (iv) の場合については，各々(i) と (iii) の場合に対称である． ∎

命題 4.5.2 $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$ とする．任意の Q について $P_1|Q \approx P_2|Q$ ならば，$P_1 \approx_m P_2$ である．ここで，$\mathcal{L}(P)$ は P に現れる全てのアクションの名前の集合である．

証明 次の集合 $\mathcal{S}$ が監視双模倣であることを示す．

$$\mathcal{S} = \{(P_1, P_2) : P_1|\mathbf{T} \approx P_2|\mathbf{T}, \mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}\}$$

ここで，$\mathbf{T}$ は次のように定義されるプロセス定数である．

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

$(P_1, P_2) \in \mathcal{S}$ とする．すなわち，$P_1|\mathbf{T} \approx P_2|\mathbf{T}$ かつ $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$ である．

(i) $P_1 \xrightarrow{a\theta^n} P'$ とする．仮定 $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$ より，$b \notin \mathcal{L}(P_1) \cup \mathcal{L}(P_2)$ かつ $b \in \mathcal{N}$ のような b が存在する．$b \in \mathcal{N}$ であるので，$\mathbf{T} \xrightarrow{b?^1} b?^1\mathbf{T}$ を導ける．さらに，$b \notin mon(P_1) \subseteq \mathcal{L}(P_1)$ であるので，$\mathbf{Com}_2$ より，$P_1|\mathbf{T} \xrightarrow{b?^1} P_1|(b?^1.\mathbf{T})$ を導く．すなわち，$P_1|\mathbf{T} \approx P_2|\mathbf{T}$ であるので，ある R_1 について，$P_2|\mathbf{T} \stackrel{b?^1}{\Longrightarrow} R_1$ かつ $P_1|(b?^1.\mathbf{T}) \approx R_1$ を得る．ここで，$b \notin \mathcal{L}(P_2)$ であるので P_2 は $b?^1$ を実行できず，$\mathbf{T}$ は内部アクションを実行できないので $(*)$，ある P_{21} について，$P_2 \Rightarrow P_{21}$ かつ $R_1 \equiv P_{21}|(b?^1.\mathbf{T})$ を得る．

次に，$P_1 \xrightarrow{a\theta^n} P_1'$ かつ $a \notin mon(b?^1.\mathbf{T})$ であるので，$\mathbf{Com}_1$ より $P_1|(b?^1.\mathbf{T}) \xrightarrow{a\theta^n} P_1'|(b?^1.\mathbf{T})$ を導ける．すなわち，$P_1|(b?^1.\mathbf{T}) \approx R_1 \equiv P_{21}|(b?^1.\mathbf{T})$ であるので，ある R_2 について，$P_{21}|(b?^1.\mathbf{T}) \stackrel{\widehat{a\theta^n}}{\Longrightarrow} R_2$ かつ $P_1'|(b?^1.\mathbf{T}) \approx R_2$ を得る．ここで，$(b?^1.\mathbf{T})$ は $a\theta^n$ も内部アクションも実行できないので，ある P_{22} について，$P_{21} \stackrel{\widehat{a\theta^n}}{\Longrightarrow} P_{22}$ かつ $R_2 \equiv P_{22}|(b?^1.\mathbf{T})$ を得る．

さらに，$b?^1.\mathbf{T} \xrightarrow{b?^1} \mathbf{T}$ かつ $b \notin mon(P_1') \subseteq \mathcal{L}(P_1') \subseteq \mathcal{L}(P_1)$ であるので，$\mathbf{Com}_2$ より，$P_1'|(b?^1.\mathbf{T}) \xrightarrow{b?^1} P_1'|\mathbf{T}$ を導く．すなわち，$P_1'|(b?^1.\mathbf{T}) \approx R_2 \equiv P_{22}|(b?^1.\mathbf{T})$ であるので，ある R_3 について，$P_{22}|(b?^1.\mathbf{T}) \stackrel{b?^1}{\Longrightarrow} R_3$ かつ $P_1'|\mathbf{T} \approx R_3$ を得る．上記の $(*)$ と同じ理由で，ある P_2' について，$P_{22} \Rightarrow P_2'$ かつ $R_3 \equiv P_2'|\mathbf{T}$ を得る．

よって，$P_2 \Rightarrow P_{21} \stackrel{\widehat{a\theta^n}}{\Longrightarrow} P_{22} \Rightarrow P_2'$ より $P_2 \stackrel{\widehat{a\theta^n}}{\Longrightarrow} P_2'$ である．また，$P_1'|\mathbf{T} \approx P_2'|\mathbf{T}$ かつ $\mathcal{L}(P_1') \subseteq \mathcal{L}(P_1)$ かつ $\mathcal{L}(P_2') \subseteq \mathcal{L}(P_2)$ であるので，$(P_1', P_2') \in \mathcal{S}$ を得る．

(ii) 上記の (i) の場合に対称である．

(iii) $a \notin mon(P_1)$ とする．$\mathbf{T}$ の定義より，$\mathbf{T} \xrightarrow{a?^1}\xrightarrow{a?^1} \mathbf{T}$ である．$a \notin mon(P_1)$ であるので，$\mathbf{Com}_2$ より $P_1|\mathbf{T} \xrightarrow{a?^1}\xrightarrow{a?^1} P_1|\mathbf{T}$ を導ける．すなわち，仮定 $P_1|\mathbf{T} \approx P_2|\mathbf{T}$ より，ある R' について，$P_2|\mathbf{T} \stackrel{a?^1}{\Longrightarrow}\stackrel{a?^1}{\Longrightarrow} R'$ かつ $P_1|\mathbf{T} \approx R'$ を得る．ここで，(1) $a \in mon(\mathbf{T})$ (すなわち，$\mathbf{Com}_1$ は不可能)，(2) アクション属性が $??$ (すなわち，$\mathbf{Com}_{3,4}$ は不可能)，(3) 受信者数が 1 (すなわち，$\mathbf{Com}_5$ は不可能) であることより，遷移 $P_2|\mathbf{T} \stackrel{a?^1}{\Longrightarrow}\stackrel{a?^1}{\Longrightarrow} R'$ における $a?^1$ による遷移を導く規則は $\mathbf{Com}_2$ のみであ

る．また，$\mathbf{T}$ は内部アクションを実行できないので，ある P_2', P_2'', P_2''' について，$P_2|\mathbf{T} \Longrightarrow P_2'|\mathbf{T} \xrightarrow{a?^1} P_2'|(a?^1.\mathbf{T}) \Longrightarrow P_2''|(a?^1.\mathbf{T}) \xrightarrow{a?^1} P_2'|\mathbf{T} \Longrightarrow P_2'''|\mathbf{T} \equiv R'$ である．すなわち，$\mathbf{Com}_1$ と $\mathbf{Com}_2$ より $P_2 \Longrightarrow P_2' \Longrightarrow P_2'' \Longrightarrow P_2'''$ かつ $a \notin mon(P_2')$ かつ $a \notin mon(P_2'')$ を得る．また，$P_1|\mathbf{T} \approx R' \equiv P_2'''|\mathbf{T}$ かつ $\mathcal{L}(P_2''') \subseteq \mathcal{L}(P_2)$ であるので，$(P_1, P_2''') \in \mathcal{S}$ を得る．

(iv) 上記の (ii) の場合に対称である．

以上のことから，$P_1|\mathbf{T} \approx P_2|\mathbf{T}$ ならば，$P_1 \approx_m P_2$ である．すなわち，任意の Q について $P_1|Q \approx P_2|Q$ ならば，$P_1|\mathbf{T} \approx P_2|\mathbf{T}$ であり，$P_1 \approx_m P_2$ を得る． ∎

次に，CCS の場合 (定義 2.4.3) と同様に，監視双模倣の定義を弱めて，**監視等価な組を除く監視双模倣**を定義する．

定義 4.5.4　プロセス上の二項関係 $\mathcal{S}$ が**監視等価な組を除く監視双模倣** (monitor bisimulation up to monior equivalence) であるとは，$P\mathcal{S}Q$ ならば，任意の $\alpha \in Act$ 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．

(i) $P \stackrel{\alpha}{\Longrightarrow} P'$ ならば，ある Q' について，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $P' \approx_m \mathcal{S} \approx_m Q'$ を満たす．
(ii) $Q \stackrel{\alpha}{\Longrightarrow} Q'$ ならば，ある P' について，$P \stackrel{\widehat{\alpha}}{\Longrightarrow} P'$ かつ $P' \approx_m \mathcal{S} \approx_m Q'$ を満たす．
(iii) $P \Rightarrow P' \Rightarrow P''$ かつ $a \notin mon(P')$ ならば，ある Q', Q'' について，
$Q \Rightarrow Q' \Rightarrow Q'', a \notin mon(Q'), P'' \approx_m \mathcal{S} \approx_m Q''$ を満たす．
(iv) $Q \Rightarrow Q' \Rightarrow Q''$ かつ $a \notin mon(Q')$ ならば，ある P', P'' について，
$P \Rightarrow P' \Rightarrow P'', a \notin mon(P'), P'' \approx_m \mathcal{S} \approx_m Q''$ を満たす．

ここで，$P\mathcal{S}Q$ は $(P, Q) \in \mathcal{S}$ を表し，$P' \approx_m \mathcal{S} \approx_m Q'$ は，$P' \approx_m P''$ かつ $P''\mathcal{S}Q''$ かつ $Q'' \approx_m Q'$ となる P'' と Q'' が存在することを表している． ∎

次の命題は，監視等価を証明するために，監視双模倣の代わりに上記の監視等価な組を除く監視双模倣をみつければ十分であることを示している．

命題 4.5.3 もし $\mathcal{S}$ が監視等価な組を除く監視双模倣であるならば，$\mathcal{S} \subseteq \approx_m$ である．

証明　まず，$\approx_m \mathcal{S} \approx_m$ が監視双模倣であることを示す．$P \approx_m \mathcal{S} \approx_m Q$ とする．すなわち，ある P_1 と Q_1 について，$P \approx_m P_1$ かつ $P_1\mathcal{S}Q_1$ かつ $Q_1 \approx_m Q$ である．

(i) $P \xrightarrow{\alpha} P'$ とする．仮定 $P \approx_m P_1$ より，ある P_1' について，$P_1 \stackrel{\widehat{\alpha}}{\Longrightarrow} P_1'$ かつ $P' \approx_m P_1'$ を得る．次の場合について証明する．

- $\alpha \in \mathcal{T}$(内部アクション) の場合：もし $P_1 \equiv P_1'$ ならば，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q$ かつ $P' \approx_m P_1' \equiv P_1 \mathcal{S} Q_1 \approx_m Q \equiv Q'$ を得る．それ以外 $(P_1 \neq P_1')$ では，ある $\tau \in \mathcal{T}$ について，$P_1 \stackrel{\tau}{\Longrightarrow} P_1'$ であるので，仮定 $P_1 \mathcal{S} Q_1$ より，ある Q_1' で $Q_1 \stackrel{\varepsilon}{\Longrightarrow} Q_1'$ かつ $P_1' \approx_m \mathcal{S} \approx_m Q_1'$ を得る．さらに $Q_1 \approx_m Q$ であるので，ある Q' について，$Q \stackrel{\varepsilon}{\Longrightarrow} Q'$ かつ $Q_1' \approx_m Q'$ を得られる．すなわち，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $P' \approx_m P_1' \approx_m \mathcal{S} \approx_m Q_1' \approx_m Q'$ である．
- $\alpha \notin \mathcal{T}$ の場合：$P_1 \stackrel{\alpha}{\Longrightarrow} P_1'$ であるので，仮定 $P_1 \mathcal{S} Q_1$ より，ある Q_1' について，$Q_1 \stackrel{\alpha}{\Longrightarrow} Q_1'$ かつ $P_1' \approx_m \mathcal{S} \approx_m Q_1'$ を得る．さらに $Q_1 \approx_m Q$ であるので，ある Q' について，$Q \stackrel{\alpha}{\Longrightarrow} Q'$ かつ $Q_1' \approx_m Q'$ を得られる．すなわち，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $P' \approx_m P_1' \approx_m \mathcal{S} \approx_m Q_1' \approx_m Q'$ である．

(iii) $a \notin mon(P)$ とする．仮定 $P \approx_m P_1$ より，ある P_1' と P_1'' について，$P_1 \Rightarrow P_1' \Rightarrow P_1''$ かつ $a \notin mon(P_1')$ かつ $P \approx_m P_1''$ を得る．このとき，$P_1 \mathcal{S} Q_1$ であるので，ある Q_1' と Q_1'' について，$Q_1 \Rightarrow Q_1' \Rightarrow Q_1''$ かつ $a \notin mon(Q_1')$ かつ $P_1'' \approx_m \mathcal{S} \approx_m Q_1''$ を得ることができる．さらに $Q_1 \approx_m Q$ であるので，ある Q' と Q'' について，$Q \Rightarrow Q' \Rightarrow Q''$ かつ $a \notin mon(Q')$ かつ $Q_1'' \approx_m Q''$ も得る．すなわち，$P \approx_m P_1'' \approx_m \mathcal{S} \approx_m Q_1'' \approx_m Q''$ である．

(ii) と (iv) の場合については各々 (i) と (iii) に対称である．すなわち，$\approx_m \mathcal{S} \approx_m$ は監視双模倣である．ここで，$\approx_m$ は最大の監視双模倣であるので，$\approx_m \mathcal{S} \approx_m \subseteq \approx_m$ である．よって，次の関係を得る．

$$\mathcal{S} \quad = \quad \equiv \mathcal{S} \equiv \quad \subseteq \quad \approx_m \mathcal{S} \approx_m \quad \subseteq \quad \approx_m$$

∎

定義 4.5.2と定義 4.5.4を比較すると，遷移の後に対する要求は $P' \mathcal{S} Q'$ から $P' \approx_m \mathcal{S} \approx_m Q'$ に弱められているため，$\mathcal{S}$ に含まれる組を減らすことができる．しかし，(i) の仮定が $P \xrightarrow{\alpha} P'$ から $P \stackrel{\alpha}{\Longrightarrow} P'$ に，(iii) の仮定が $a \notin mon(P)$ から $P \Rightarrow P' \Rightarrow P''$ かつ $a \notin mon(P')$ に弱められているため，より多くの場合について条件 $(i), \cdots, (iv)$ が成り立つかを調べなければならない．そこで，この仮定が定義 4.5.2の仮定と同じになるように，条件 $P' \approx_m \mathcal{S} \approx_m Q'$ を少し強めて，次の命題を与える．

命題 4.5.4 集合 $\mathcal{S}$ は，$P\mathcal{S}Q$ ならば任意の $\alpha \in Act$ と $a \in \mathcal{N}$ について次の条件を満たすとする．

(i) $P \xrightarrow{\alpha} P'$ ならば，ある Q' について，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $P' \sim \mathcal{S} \approx_m Q'$ を満たす．

(ii) $Q \xrightarrow{\alpha} Q'$ ならば，ある P' について，$P \stackrel{\widehat{\alpha}}{\Longrightarrow} P'$ かつ $P' \approx_m \mathcal{S} \sim Q'$ を満たす．

(iii) $a \notin mon(P)$ ならば，ある Q', Q'' について，

$$Q \Rightarrow Q' \Rightarrow Q'', a \notin mon(Q'), P \approx_m \mathcal{S} \approx_m Q''$$ を満たす．

(iv) $a \notin mon(Q)$ ならば，ある P', P'' について，

$$P \Rightarrow P' \Rightarrow P'', a \notin mon(P'), P'' \approx_m \mathcal{S} \approx_m Q$$ を満たす．

このとき，$\mathcal{S}$ は監視等価な組を除く監視双模倣である．ここで，$\sim$ は強等価である．

証明 $P\mathcal{S}Q$ とし，定義 4.5.4の $(i), \cdots, (iv)$ を示す．

(i) まず，$P \Longrightarrow P'$ ならば，ある Q' について，$Q \Longrightarrow Q'$ かつ $P' \sim \mathcal{S} \approx_m Q'$ である $(*)$ ことを，$\Longrightarrow$ の長さに関する帰納法を用いて示す．

- 長さ 0 ならば，$P \equiv P'$ であるので，$Q \Longrightarrow Q$ かつ $P \sim \mathcal{S} \approx_m Q$ である．
- $P \xrightarrow{\tau} P_1 \Longrightarrow P'$ とする．$P\mathcal{S}Q$ かつ $P \xrightarrow{\tau} P_1$ であるので，ある Q_1 について，$Q \Longrightarrow Q_1$ かつ $P_1 \sim \mathcal{S} \approx_m Q_1$ である．すなわち，ある P_2 と Q_2 で $P_1 \sim P_2$ かつ $P_2 \mathcal{S} Q_2$ かつ $Q_2 \approx_m Q_1$ である．$P_1 \sim P_2$ かつ $P_1 \Longrightarrow P'$ であるので，ある P_2' について，$P_2 \Longrightarrow P_2'$ かつ $P' \sim P_2'$ を得る．ここで，$P_2 \mathcal{S} Q_2$ かつ $P_2 \Longrightarrow P_2'$ であるので，帰納法の仮定より，ある Q_2' について，$Q_2 \Longrightarrow Q_2'$ かつ $P_2' \sim \mathcal{S} \approx_m Q_2'$ を得る．さらに，$Q_2 \approx_m Q_1$ かつ $Q_2 \Longrightarrow Q_2'$ であるので，ある Q' について，$Q_1 \Longrightarrow Q'$ かつ $Q_2' \approx_m Q'$ を得る．以上をまとめて，$Q \Longrightarrow Q_1 \Longrightarrow Q'$ かつ $P' \sim P_2' \sim \mathcal{S} \approx_m Q_2' \approx_m Q'$ である．
- よって，$(*)$ が成り立つ．

次に $P \stackrel{\alpha}{\Longrightarrow} P'$ として，ある Q' について，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $P' \sim \mathcal{S} \approx_m Q'$ を示す．遷移 $P \stackrel{\alpha}{\Longrightarrow} P'$ より，ある P_1 と P_1' について，$P \Longrightarrow P_1 \xrightarrow{\alpha} P_1' \Longrightarrow P'$ である．上記の $(*)$ より，ある Q_1 について，$Q \Longrightarrow Q_1$ かつ $P_1 \sim \mathcal{S} \approx_m Q_1$ を得る．このとき，$P_1 \xrightarrow{\alpha} P_1'$ であるので，ある Q_1' について，$Q_1 \stackrel{\widehat{\alpha}}{\Longrightarrow} Q_1'$ かつ $P_1' \sim \mathcal{S} \approx_m Q_1'$ を得ることができる．さらに，$P_1' \Longrightarrow P'$ についても $(*)$ より，ある Q' について，$Q_1' \Longrightarrow Q'$ かつ $P' \sim \mathcal{S} \approx_m Q'$ を得られる．以上をまとめると，$\sim \subseteq \approx_m$ であるので，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $P' \approx_m \mathcal{S} \approx_m Q'$ を得る．

(*iii*) $P \Rightarrow P' \Rightarrow P''$ かつ $a \notin mon(P')$ とする．$P\mathcal{S}Q$ であるので，ある Q_0 について，$Q \Rightarrow Q_0$ かつ $P' \sim \mathcal{S} \approx_m Q_0$ を得る．すなわち，ある P_1 と Q_1 について，$P' \sim P_1$ かつ $P_1\mathcal{S}Q_1$ かつ $Q_1 \approx_m Q_0$ である．ここで，$P' \sim P_1$ かつ $a \notin mon(P')$ より $a \notin P_1$ を得る．また，$P_1\mathcal{S}Q_1$ であるので，ある Q_1' と Q_1'' について，$Q_1 \Rightarrow Q_1' \Rightarrow Q_1''$ かつ $a \notin mon(Q_1')$ かつ $P_1 \approx_m \mathcal{S} \approx_m Q_1''$ を得る．さらに，$P' \sim P_1$ であるので，ある P_1' について，$P_1 \Rightarrow P_1'$ かつ $P'' \sim P_1'$ を得る．ここで，$P_1 \approx_m \mathcal{S} \approx_m Q_1''$ と $P_1 \Rightarrow P_1'$ から，$(*)$ より，ある Q_1''' について，$Q_1'' \Rightarrow Q_1'''$ かつ $P_1' \approx_m \mathcal{S} \approx_m Q_1'''$ を示すことができる．最後に，$Q_1 \approx_m Q_0$ かつ $Q_1 \Rightarrow Q_1' \Rightarrow Q_1'''$ かつ $a \notin mon(Q_1')$ であるので，ある Q' と Q'' について，$Q_0 \Rightarrow Q' \Rightarrow Q''$ かつ $a \notin mon(Q')$ かつ $Q_1''' \approx_m Q''$ を得る．以上をまとめて，$Q \Rightarrow Q_0 \Rightarrow Q' \Rightarrow Q''$ かつ $P'' \sim P_1' \approx_m \mathcal{S} \approx_m Q_1''' \approx_m Q''$ である．

∎

以上，監視等価の証明に，命題 4.5.3や命題 4.5.4を利用することができる．例えば，後の命題 4.5.8の証明には命題 4.5.4を用い，命題 4.5.9の証明には命題 4.5.3を用いる．

4.5.2 監視合同

監視等価は同値関係であるが，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，監視等価に最も弱い条件を付加して合同関係を定義する．まず無数に存在する内部イベントの違いを無視するため，アクション上の同値関係 $\doteq$ を導入する．

定義 4.5.5　アクション等価 $\doteq$ は

$$\doteq = \{(\tau, \tau') : \tau, \tau' \in \mathcal{T}\} \cup \{(\alpha, \alpha) : \alpha \in Act\}$$

のように定義されるアクション上の関係である．

∎

このとき，監視合同を次のように定義する．

定義 4.5.6　もし，任意の $\alpha \in Act$ について，次の 3 つの条件を満たすならば，P と Q は**監視合同** (monitor congruence) であるといい，$P =_m Q$ と書く．

(*i*) $P \xrightarrow{\alpha} P'$ ならば，ある Q' と α' について，$Q \stackrel{\alpha'}{\Longrightarrow} Q'$, $P' \approx_m Q'$, $\alpha \doteq \alpha'$ を満たす．
(*ii*) $Q \xrightarrow{\alpha} Q'$ ならば，ある Q' と α' について，$P \stackrel{\alpha'}{\Longrightarrow} P'$, $P' \approx_m Q'$, $\alpha \doteq \alpha'$ を満たす．
(*iii*) $mon(P) = mon(Q)$

∎

監視合同と監視等価との違いは，$\stackrel{\widehat{\alpha}}{\Longrightarrow}$ ではなく $\stackrel{\alpha'}{\Longrightarrow}$ が要求されていることである．ただし，この要求は最初の遷移に対してのみであり，それ以降は監視等価 $(P' \approx_m Q')$ が要求される．また，監視双模倣の条件 (iii) と (iv) が $mon(P) = mon(Q)$ に強められている．監視合同は次の命題 4.5.5と命題 4.5.6 に示されるとおり，監視等価に含まれ，かつ選択演算子によって保存される最大の関係である．

命題 4.5.5　$P_1 =_m P_2$ ならば，任意の Q について $P_1 + Q =_m P_2 + Q$ である．

証明　$P_1 =_m P_2$ とする．定義 4.5.6 の条件 (i), (ii), (iii) が満たされることを示す．

(i) $P_1 + Q \stackrel{\alpha}{\longrightarrow} P_1'$ とする．この遷移は $\mathbf{Choice}_1$ か $\mathbf{Choice}_2$ によって導かれる．

- $\mathbf{Choice}_1$ による場合：$P_1 \stackrel{\alpha}{\longrightarrow} P_1'$ である．$P_1 =_m P_2$ であるので，ある P_2' と α' について，$P_2 \stackrel{\alpha'}{\Longrightarrow} P_2'$ かつ $P_1' \approx_m P_2'$ かつ $\alpha \doteq \alpha'$ である．遷移 $P_2 \stackrel{\alpha'}{\Longrightarrow} P_2'$ の長さは必ず 1 以上であるので，$\mathbf{Choice}_1$ により $P_2 + Q \stackrel{\alpha'}{\Longrightarrow} P_2'$ を導ける．
- $\mathbf{Choice}_2$ による場合：$Q \stackrel{\alpha}{\longrightarrow} P_1'$ である．すなわち，$\mathbf{Choice}_2$ により $P_2 + Q \stackrel{\alpha}{\longrightarrow} P_1'$ を得る．このとき，$P_2 + Q \stackrel{\alpha}{\Longrightarrow} P_1'$ かつ $P_1' \approx_m P_1'$ かつ $\alpha \doteq \alpha$ である．

(ii) 上記の (i) と対称である．

(iii) 仮定 $P_1 =_m P_2$ より，$mon(P_1) = mon(P_2)$ である．すなわち，$mon(P_1 + Q) = mon(P_1) \cup mon(Q) = mon(P_2) \cup mon(Q) = mon(P_2 + Q)$ を得る．

∎

命題 4.5.6　$\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$ とする．任意の Q について $P_1 + Q \approx_m P_2 + Q$ ならば，$P_1 =_m P_2$ である．ここで，$\mathcal{L}(P)$ は P に現れる全てのアクションの名前の集合である．すなわち，P_1 と P_2 は全ての名前を使い尽くしていないことを仮定している．

証明　$\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$ であるので，$b \notin \mathcal{L}(P_1) \cup \mathcal{L}(P_2)$ のようなある b について，$B \equiv b?^1.\mathbf{0}$ とする．このとき，仮定より $P_1 + B \approx_m P_2 + B$ である．

(i) $P_1 \stackrel{a\theta^n}{\longrightarrow} P_1'$ とする．$\mathbf{Choice}_1$ により $P_1 + B \stackrel{a\theta^n}{\longrightarrow} P_1'$ を導ける．仮定 $P_1 + B \approx_m P_2 + B$ より，ある P_2' について，$P_2 + B \stackrel{\widehat{a\theta^n}}{\Longrightarrow} P_2'$ かつ $P_1' \approx_m P_2'$ を得る．

- $a\theta^n \notin \mathcal{T}$ の場合 $(\widehat{a\theta^n} = a\theta^n)$：$B \equiv b?^1.\mathbf{0}$ は内部アクションを実行できない．また，$a \in \mathcal{L}(P_1)$ かつ $b \notin \mathcal{L}(P_1)$ であるので，B は $a\theta^n$ も実行できない．すなわち，$P_2 + B \stackrel{a\theta^n}{\Longrightarrow} P_2'$ は P_2 によってのみ導かれる．よって，$P_2 \stackrel{a\theta^n}{\Longrightarrow} P_2'$ かつ $P_1' \approx_m P_2'$ を得る．
- $a\theta^n \in \mathcal{T}$ の場合 $(\widehat{a\theta^n} = \varepsilon)$:今，$P_2' \equiv P_2 + B$ と仮定する．このとき，$B \equiv b?^1.\mathbf{0}$ は $b?^1$ を実行できるので，$P_2' \equiv P_2 + B$ も $b?^1$ を実行できる．さらに，$P_1' \approx_m P_2'$ であるので，P_1' もいつかは $b?^1$ を実行できなければならない．しかし，$b \notin \mathcal{L}(P_1') \subseteq \mathcal{L}(P_1)$ であるので，これは矛盾する．すなわち，$P_2' \neq P_2 + B$ であるので，ある τ について，$P_2 + B \stackrel{\tau}{\Longrightarrow} P_2'$ である．B は内部アクションを実行できないので，$P_2 + B \stackrel{\tau}{\Longrightarrow} P_2'$ は P_2 により導かれる．すなわち，$P_2 \stackrel{\tau}{\Longrightarrow} P_2'$ かつ $P_1' \approx_m P_2'$ を得る．ここで，$a\theta^n, \tau \in \mathcal{T}$ より，$a\theta^n \doteq \tau$ である．

(ii) 上記の (i) の場合に対称である．

(iii) $a \notin mon(P_1)$ とする．

- $a = b$ の場合：仮定より，$b \notin \mathcal{L}(P_2)$ である．
- $a \neq b$ の場合：$a \notin mon(P_1) \cup \{b\} = mon(P_1) \cup mon(b?^1.\mathbf{0}) = mon(P_1 + B)$ である．すなわち，仮定 $P_1 + B \approx_m P_2 + B$ より，ある P_2' と P_2'' について，$P_2 + B \Rightarrow P_2' \Rightarrow P_2''$ かつ $a \notin mon(P_2')$ かつ $P_1 + B \approx_m P_2''$ を得る．

  今，ある τ について，$P_2 + B \stackrel{\tau}{\Longrightarrow} P_2' \Rightarrow P_2''$ であると仮定する．B は内部アクションをもたないので，$\mathbf{Choice}_1$ より，$P_2 \stackrel{\tau}{\Longrightarrow} P_2''$ を得る．ここで，$P_1 + B$ は $b?^1$ を実行できるため，$P_1 + B \approx_m P_2''$ により，P_2'' もいつかは $b?^1$ を実行できなければならない．しかし，これは $b \notin \mathcal{L}(P_2'') \subseteq \mathcal{L}(P_2)$ に矛盾する．すなわち，$P_2 + B \equiv P_2'$ であり，$a \notin mon(P_2') = mon(P_2 + B) \supseteq mon(P_2)$ を得る．

よって，$mon(P_2) \subseteq mon(P_1)$ である．同様に $mon(P_1) \subseteq mon(P_2)$ も証明でき，$mon(P_1) = mon(P_2)$ を得る．

∎

今までプロセス上に監視合同を定義してきたが，CCS のときと同様に，プロセス変数を含むプロセス式上に拡張する．

定義 4.5.7 プロセス式 E と F は高々変数 X_i $(i \in I)$ を含むとする．このとき，任意のプロセス P_i $(i \in I)$ について，$E\{P_i/X_i\}_{i\in I} =_m F\{P_i/X_i\}_{i\in I}$ ならば，$E =_m F$ である．ここで，$E\{P_i/X_i\}_{i\in I}$ は，各 $i \in I$ について，E に含まれる全ての変数 X_i へプロセス P_i を代入して得られるプロセスである． ∎

次の命題 4.5.7と命題 4.5.8 に示すように，観測合同は全ての演算子と再帰定義によって保存される．すなわち合同関係である．

命題 4.5.7 $E_1 =_m E_2$ とする．このとき次の等式が成り立つ．

(1) $\alpha.E_1 =_m \alpha.E_2$
(2) $E_1 + F =_m E_2 + F$
(3) $E_1|F =_m E_2|F$
(4) $E_1\backslash L =_m E_2\backslash L$
(5) $E_1[f] =_m E_2[f]$

証明 簡単のため E と F は高々変数 X を含むとし，$P \in \mathcal{P}_{core}$ とする．

(1) (i) $(\alpha.E_1)\{P/X\} \xrightarrow{\alpha'} P_1'$ とする．**Act** より $\alpha = \alpha'$ かつ $P_1' \equiv E_1\{P/X\}$ を得る．一方，**Act** より $(\alpha.E_2)\{P/X\} \xrightarrow{\alpha'} E_2\{P/X\}$ を導く．ここで，$E_1 =_m E_2$ より，$E_1\{P/X\} =_m E_2\{P/X\}$ である．

(iii) 明らかに，$mon((\alpha.E_1)\{P/X\}) = mon((\alpha.E_2)\{P/X\})$ である．

(2) $(E_1 + F)\{P/X\} \equiv E_1\{P/X\} + F\{P/X\}$ である．$E_1\{P/X\} =_m E_2\{P/X\}$ であるので，命題 4.5.5より，$E_1\{P/X\} + F\{P/X\} =_m E_2\{P/X\} + F\{P/X\} \equiv (E_2 + F)\{P/X\}$ を得る．

(3) (i) $(E_1|F)\{P/X\} \equiv E_1\{P/X\}|F\{P/X\} \xrightarrow{a\theta^n} P'$ とする．この遷移を導いた最後の規則について場合分けする．

1. $\mathbf{Com}_1$ による場合：ある P_1' について，$E_1\{P/X\} \xrightarrow{a\theta^n} P_1'$ かつ $(\theta \in \{!, ?\}$ または $a \notin mon(F\{P/X\}))$ かつ $P' \equiv P_1'|F\{P/X\}$ を得る．仮定 $E_1\{P/X\} =_m E_2\{P/X\}$ より，ある P_2' と a' について，$E_2\{P/X\} \stackrel{a'\theta^n}{\Longrightarrow} P_2'$ かつ $P_1' \approx_m P_2'$ かつ $a\theta^n \doteq a'\theta^n$ を得る．ここで，$\theta \neq !$ ならば必ず $a' = a \notin mon(F\{P/X\}))$ である．すなわち，この遷移 $E_2\{P/X\} \stackrel{a'\theta^n}{\Longrightarrow} P_2'$ に $\mathbf{Com}_1$ を 1 回以上適用し

て，$E_2\{P/X\}|F\{P/X\} \stackrel{a'\theta^n}{\Longrightarrow} P_2'|F\{P/X\}$ を導く．ここで，$P_1' \approx_m P_2'$ であるので，命題 4.5.1より $P' \equiv P_1'|F\{P/X\} \approx_m P_2'|F\{P/X\}$ を得る．

2. $\mathbf{Com}_2$ による場合：ある Q' について，$F\{P/X\} \stackrel{a\theta^n}{\longrightarrow} Q'$ かつ $(\theta \in \{!, ?\}$ または $a \notin mon(E_1\{P/X\}))$ かつ $P' \equiv E_1\{P/X\}|Q'$ を得る．仮定 $E_1\{P/X\} =_m E_2\{P/X\}$ より，$mon(E_1\{P/X\}) = mon(E_2\{P/X\})$ であるので，$(\theta \in \{!, ?\}$ または $a \notin mon(E_2\{P/X\}))$ である．遷移 $F\{P/X\} \stackrel{a\theta^n}{\longrightarrow} Q'$ に $\mathbf{Com}_2$ を 1 回適用して，$E_2\{P/X\}|F\{P/X\} \stackrel{a\theta^n}{\longrightarrow} E_2\{P/X\}|Q'$ を導く．ここで，$E_1\{P/X\} =_m E_2\{P/X\}$ であるので，命題 4.5.1より $P' \equiv E_1\{P/X\}|Q' \approx_m E_2\{P/X\}|Q'$ を得る．

3. $\mathbf{Com}_3$ による場合：ある $n_1,\ n_2,\ \phi,\ P_1',\ Q'$ について，$E_1\{P/X\} \stackrel{a\theta^{n_1}}{\longrightarrow} P_1'$ かつ $F\{P/X\} \stackrel{a\phi^{n_2}}{\longrightarrow} Q'$ かつ $\theta \in \{!, !!\}$ かつ $\phi = com(\theta)$ かつ $n_1 \geq n_2$ かつ $n = n_1 - n_2$ かつ $P' \equiv P_1'|Q'$ を得る．ここで，$n_1 \geq n_2 \geq 1$ であるので，$a\theta^{n_1}$ は内部アクションではない．すなわち，仮定 $E_1\{P/X\} =_m E_2\{P/X\}$ より，ある P_2' について，$E_2\{P/X\} \stackrel{a\theta^{n_1}}{\Longrightarrow} P_2'$ かつ $P_1' \approx_m P_2'$ を得る．この遷移に $\mathbf{Com}_1$ を 0 回以上，$\mathbf{Com}_3$ を 1 回適用して，$E_2\{P/X\}|F\{P/X\} \stackrel{a\theta^{(n_1-n_2)}}{\Longrightarrow} P_2'|Q'$ を導く．また，$P_1' \approx_m P_2'$ であるので，命題 4.5.1より $P_1'|Q' \approx_m P_2'|Q'$ を得る．

4. $\mathbf{Com}_4$ による場合：$\mathbf{Com}_3$ の場合に同様である．

5. $\mathbf{Com}_5$ による場合：ある $n_1,\ n_2,\ P_1',\ Q'$ について，$E_1\{P/X\} \stackrel{a\theta^{n_1}}{\longrightarrow} P_1'$ かつ $Q \stackrel{a\theta^{n_2}}{\longrightarrow} Q'$ かつ $\theta \in \{?, ??\}$ かつ $n = n_1 + n_2$ かつ $P' \equiv P_1'|Q'$ を得る．ここで，$\theta \in \{?, ??\}$ であるので，$a\theta^{n_1}$ は内部アクションではない．すなわち，仮定 $E_1\{P/X\} =_m E_2\{P/X\}$ より，ある P_2' について，$E_2\{P/X\} \stackrel{a\theta^{n_1}}{\Longrightarrow} P_2'$ かつ $P_1' \approx_m P_2'$ を得る．この遷移に $\mathbf{Com}_1$ を 0 回以上，$\mathbf{Com}_5$ を 1 回適用して，$E_2\{P/X\}|F\{P/X\} \stackrel{a\theta^{(n_1+n_2)}}{\Longrightarrow} P_2'|Q'$ を導く．また，$P_1' \approx_m P_2'$ であるので，命題 4.5.1より $P_1'|Q' \approx_m P_2'|Q'$ を得る．

(*iii*) 仮定 $E_1\{P/X\} =_m E_2\{P/X\}$ より，$mon(E_1\{P/X\}) = mon(E_2\{P/X\})$ である．すなわち，$mon(E_1\{P/X\}|F\{P/X\}) = mon(E_1\{P/X\}) \cup mon(F\{P/X\}) = mon(E_2\{P/X\}) \cup mon(F\{P/X\}) = mon(E_2\{P/X\}|F\{P/X\})$ を得る．

(4) と (5) については (3) と同様に証明できる． ∎

命題 4.5.8 プロセス式 E_i, F_i $(i \in I)$ は高々変数 X_j $(j \in I)$ を含むとする．このとき，各 $i \in I$ について，$A_i \stackrel{\text{def}}{=} E_i\{A_j/X_j\}_{j\in I}$ かつ $B_i \stackrel{\text{def}}{=} F_i\{B_j/X_j\}_{j\in I}$ かつ $E_i =_m F_i$ ならば，各 $i \in I$ について，$A_i =_m B_i$ である．

証明 簡単のため E と F は高々変数 X を含むとし，$A \stackrel{\text{def}}{=} E\{A/X\}$ かつ $B \stackrel{\text{def}}{=} F\{B/X\}$ かつ $E =_m F$ ならば，$A =_m B$ を示す．このためには，命題 4.5.4より，次の集合 $\mathcal{S}$ が，

$$\mathcal{S} = \{(G\{A/X\}, G\{B/X\}) : G \text{ は高々変数 } X \text{ を含む }\}$$

次の条件を満たすことを示せば十分である．

(1) $a \in mon(G\{A/X\})$ ならば，$a \in mon(G\{B/X\})$ である．

(2) もし $G\{A/X\} \xrightarrow{a\theta^n} P'$ ならば，ある Q' と a' について，$G\{B/X\} \stackrel{a'\theta^n}{\Longrightarrow} Q'$ かつ $P'\mathcal{S} \approx_m Q'$ かつ $a\theta^n \doteq a'\theta^n$ である．

まず，(1) を $a \in mon(G\{A/X\})$ を導いた $G\{A/X\}$ の構造に関する帰納法を用いて証明する．

1. $G \equiv X$ の場合：$G\{A/X\} \equiv X\{A/X\} \equiv A$ より，$a \in mon(A)$ である．ここで，$A \stackrel{\text{def}}{=} E\{A/X\}$ であるので，$a \in mon(E\{A/X\})$ である．すなわち，帰納法の仮定より $a \in mon(E\{B/X\})$ を得る．さらに，$E\{B/X\} =_m F\{B/X\}$ であるので，$a \in mon(F\{B/X\})$ を得る．ここで，$B \stackrel{\text{def}}{=} F\{B/X\}$ より，$a \in mon(B) = mon(G\{B/X\})$ を得る．

2. $G \equiv C$ の場合:すなわち，$a \in G\{A/X\} \equiv C\{A/X\} \equiv C \equiv C\{B/X\} \equiv G\{B/X\}$ である．

3. $G \equiv \alpha.G'$ の場合:$a \in mon(G\{A/X\}) = mon(\alpha.G'\{A/X\}) = mon(\alpha.G'\{B/X\}) = mon(G\{B/X\})$ である．

4. $G \equiv G_1{+}G_2$ の場合:$a \in mon((G_1{+}G_2)\{A/X\}) = mon(G_1\{A/X\}{+}G_2\{A/X\}) = mon(G_1\{A/X\}) \cup mon(G_2\{A/X\})$ であるので，$a \in mon(G_1\{A/X\})$ または $a \in mon(G_2\{A/X\})$ である．帰納法の仮定より $a \in mon(G_1\{B/X\})$ または $a \in mon(G_2\{B/X\})$ である．すなわち，$a \in mon(G_1\{B/X\}) \cup mon(G_2\{B/X\}) = mon(G\{B/X\})$ である．

上記以外の $G_1|G_2$，$G_1[f]$，$G_1\backslash L$ の場合も，$G \equiv G_1 + G_2$ の場合と同様に証明できる．

次に (2) を証明する．$G\{A/X\} \xrightarrow{a\theta^n} P'$ とし，この遷移を導いた規則の数に関する帰納法を用いる．G の構造について場合分けする．

1. $G \equiv X$ の場合:すなわち，$G\{A/X\} \equiv X\{A/X\} \equiv A$ である．すなわち，$A \xrightarrow{a\theta^n} P'$ かつ $A \stackrel{\text{def}}{=} E\{A/X\}$ であるので，**Rec** より $E\{A/X\} \xrightarrow{a\theta^n} P'$ を得る．ここで，帰納法の仮定より，ある Q'' と a'' について，$E\{B/X\} \stackrel{a''\theta^n}{\Longrightarrow} Q''$ かつ $P'\mathcal{S} \approx_m Q''$ かつ $a\theta^n \doteq a''\theta^n$ を得る．すなわち，$E =_m F$ より，$E\{B/X\} =_m F\{B/X\}$ であるので，ある Q' と a' について，$F\{B/X\} \stackrel{a'\theta^n}{\Longrightarrow} Q'$ かつ $Q'' \approx_m Q'$ かつ $a''\theta^n \doteq a'\theta^n$ を得る．さらに，$B \stackrel{\text{def}}{=} F\{B/X\}$ であるので，**Rec** より $B \stackrel{a'\theta^n}{\Longrightarrow} Q'$ を導ける．すなわち，$G\{B/X\} \equiv X\{B/X\} \equiv B \stackrel{a'\theta^n}{\Longrightarrow} Q'$ かつ $P'\mathcal{S} \approx_m Q'' \approx_m Q'$ かつ $a\theta^n \doteq a''\theta^n \doteq a'\theta^n$ である．

2. $G \equiv C$ の場合:すなわち，$G\{B/X\} \equiv C\{B/X\} \equiv C \equiv C\{A/X\} \equiv G\{A/X\} \xrightarrow{a\theta^n} P'$ である．また，$P' \equiv P'\{A/X\}\mathcal{S}P'\{B/X\} \equiv P'$ である．すなわち，$G\{B/X\} \stackrel{a\theta^n}{\Longrightarrow} P'$ かつ $P'\mathcal{S}P'$ かつ $a\theta^n \doteq a\theta^n$ である．

3. $G \equiv \alpha.G'$ の場合：$\alpha.G'\{A/X\} \xrightarrow{a\theta^n} P'$ であるので，**Act** より $\alpha = a\theta^n$ かつ $P' \equiv G'\{A/X\}$ を得る．一方，**Act** より $G\{B/X\} \equiv a\theta^n.(G'\{B/X\}) \xrightarrow{a\theta^n} G'\{B/X\}$ を導く．すなわち，$G\{B/X\} \stackrel{a\theta^n}{\Longrightarrow} G'\{B/X\}$ かつ $P' \equiv G'\{A/X\}\mathcal{S}G'\{B/X\}$ かつ $a\theta^n \doteq a\theta^n$ である．

4. $G \equiv G_1 + G_2$ の場合：$G\{A/X\} \equiv G_1\{A/X\} + G_2\{A/X\} \xrightarrow{a\theta^n} P'$ を導く最後の規則は $\mathbf{Choice}_1$ か $\mathbf{Choice}_2$ である．対称であるので，$\mathbf{Choice}_1$ による場合のみ示す．すなわち，$G_1\{A/X\} \xrightarrow{a\theta^n} P'$ である．ここで，帰納法の仮定より，ある Q' と a' について，$G_1\{B/X\} \stackrel{a'\theta^n}{\Longrightarrow} Q'$ かつ $P'\mathcal{S} \approx_m Q'$ かつ $a\theta^n \doteq a'\theta^n$ である．このとき，$\mathbf{Choice}_1$ より $G\{B/X\} \equiv G_1\{B/X\} + G_2\{B/X\} \stackrel{a'\theta^n}{\Longrightarrow} Q'$ を導ける．

5. $G \equiv G_1|G_2$ の場合：$G\{A/X\} \equiv G_1\{A/X\}|G_2\{A/X\} \xrightarrow{a\theta^n} P'$ を導く最後の規則は $\mathbf{Com}_{1,2,3,4,5}$ のどれかである．ここでは，$\mathbf{Com}_1$ による場合を示す．他の場合も同様である．すなわち，ある P_1' について，$G_1\{A/X\} \xrightarrow{a\theta^n} P_1'$ かつ ($\theta \in \{!, ?\}$ または $a \notin mon(G_2\{A/X\})$) かつ $P' \equiv P_1'|G_2\{A/X\}$ である．ここで，帰納法の仮定より，ある Q_1' と a' について，$G_1\{B/X\} \stackrel{a'\theta^n}{\Longrightarrow} Q_1'$ かつ $P_1'\mathcal{S} \approx_m Q_1'$ かつ $a\theta^n \doteq a'\theta^n$ である．

- $\theta \in \{!, ?\}$ の場合：$\mathbf{Com}_1$ を 1 回以上適用して，$G_1\{B/X\}|G_2\{B/X\} \stackrel{a'\theta^n}{\Longrightarrow} Q_1'|G_2\{B/X\}$ を導く．
- 上記以外 ($a \notin mon(G_2\{A/X\})$) の場合：$\theta \neq !$ であるので，$a\theta^n$ は内部アクションではない．すなわち，$a = a'$ である．また，すでに証明した (1) より，$a \notin mon(G_2\{B/X\})$ である．すなわち，$\mathbf{Com}_1$ を 1 回以上適用して，$G_1\{B/X\}|G_2\{B/X\} \stackrel{a'\theta^n}{\Longrightarrow} Q_1'|G_2\{B/X\}$ を導く．

また，$P_1'\mathcal{S} \approx_m Q_1'$ より，ある Q_1'' について，$P_1'\mathcal{S}Q_1''$ かつ $Q_1'' \approx_m Q_1'$ である．すなわち，ある高々変数 X を含む G_1' について，$P_1' \equiv G_1'\{A/X\}$ かつ $Q_1'' \equiv G_1'\{B/X\}$ である．さらに，$G' \equiv G_1'|G_2$ とおくと，$P' \equiv P_1'|G_2\{A/X\} \equiv G_1'\{A/X\}|G_2\{A/X\} \equiv G'\{A/X\}$，$Q_1''|G_2\{B/X\} \equiv G_1'\{B/X\}|G_2\{B/X\} \equiv G'\{B/X\}$ である．すなわち，$P'\mathcal{S}(Q_1''|G_2\{B/X\})$ である．ここで，$Q_1'' \approx_m Q_1'$ であるので，命題 4.5.1より，$Q_1''|G_2\{B/X\} \approx_m Q_1'|G_2\{B/X\}$ である．すなわち，$P'\mathcal{S}(Q_1''|G_2\{B/X\}) \approx_m Q_1'|G_2\{B/X\}$ を得る．

上記以外の $G_1[f]$，$G_1 \setminus L$ の場合も，$G \equiv G_1|G_2$ の場合と同様に証明できる． ∎

監視合同に対する唯一解の存在を示すためにも，観測合同の場合と同様に次の “逐次性” と “ガード” の概念が必要である (定義 2.4.13と定義 2.4.14に同じ)．

定義 4.5.8 もしプロセス式 E の変数 X を含む全ての部分式が $\alpha.F$ か $\sum_{i\in I} F_i$ か X の形であるならば，X は E において**逐次的**であるという． ∎

定義 4.5.9 もしプロセス式 E に含まれる全ての変数 X が E のある部分式 $\alpha.F$ に含まれ，かつ $\alpha \notin \mathcal{T}$ ならば，X は E において**ガード**されているという． ∎

観測合同 (命題 2.4.16) と同様に，逐次的かつガードされていれば，次の命題が示すように監視合同も唯一解をもつ．

命題 4.5.9 各 $i \in I$ について，プロセス式 E_i は高々変数 X_j $(j \in I)$ を含み，各変数 X_j $(j \in I)$ は E_i において逐次的かつガードされているとする．このとき，各 $i \in I$ について，$P_i =_m E_i\{P_j/X_j\}_{j\in I}$ かつ $Q_i =_m E_i\{Q_j/X_j\}_{j\in I}$ ならば，各 $i \in I$ について，$P_i =_m Q_i$ である．

証明 簡単のため E は高々変数 X を含み，X は E において逐次的かつガードされているとする．このとき，$P =_m E\{P/X\}$ かつ $Q =_m E\{Q/X\}$ ならば，$P =_m Q$ を示す．

このためには，命題 4.5.3より，次の集合 $\mathcal{S}$ が，

$$\mathcal{S} = \{(G\{P/X\}, G\{Q/X\}) : G \text{は高々変数 } X \text{を含み，逐次的である }\}$$

次の条件を満たすことを示せば十分である．

(1) $a \in mon(G\{P/X\})$ ならば，$a \in mon(G\{Q/X\})$ である．

(2) もし $G\{P/X\} \stackrel{a\theta^n}{\Longrightarrow} P'$ ならば，ある Q' と a' について，$G\{Q/X\} \stackrel{a'\theta^n}{\Longrightarrow} Q'$ かつ $P' \approx_m \mathcal{S} \approx_m Q'$ かつ $a\theta^n \doteq a'\theta^n$ である．

(3) もし $G\{P/X\} \Rightarrow P' \Rightarrow P''$ かつ $a \notin mon(P')$ ならば，ある Q' と Q'' について，$G\{Q/X\} \Rightarrow Q' \Rightarrow Q''$ かつ $a \notin mon(Q')$ かつ $P'' \approx_m \mathcal{S} \approx_m Q''$ である．

以下，これらを証明する．

(1) この証明は命題 4.5.8の証明 (1) に同様である．

(2) $G\{P/X\} \stackrel{a\theta^n}{\Longrightarrow} P'$ とする．監視合同は合同関係であるので，$P =_m E\{P/X\}$ より，$G\{P/X\} =_m G\{E\{P/X\}/X\}$ である．すなわち，ある P_1, P_1', P'', a' について，$G\{E\{P/X\}/X\} \Longrightarrow P_1 \stackrel{a'\theta^n}{\longrightarrow} P_1' \Longrightarrow P''$ かつ $P' \approx_m P''$ かつ $a\theta^n \doteq a'\theta^n$ を得る．ここで，X は $G\{E/X\}$ において逐次的かつガードされているので，内部アクションによる遷移を起こした後も逐次的かつガードされている．つまり，$G\{E\{P/X\}/X\} \Longrightarrow P_1$ の遷移に P が影響することはできないため，X が逐次的かつガードされているようなある G_1 で，$G\{E/X\} \Longrightarrow G_1$ かつ $P_1 \equiv G_1\{P/X\}$ を得ることができる．また，$G\{E\{Q/X\}/X\} \Longrightarrow G_1\{Q/X\}$ も可能である．

さらに，X は G_1 において逐次的かつガードされているので，$P_1 \equiv G_1\{P/X\} \stackrel{a'\theta^n}{\longrightarrow} P_1'$ より，X が逐次的であるようなある G_1' について，$G_1 \stackrel{a'\theta^n}{\longrightarrow} G_1'$ かつ $P_1' \equiv G_1'\{P/X\}$ を得られる．すなわち，$G_1\{Q/X\} \stackrel{a'\theta^n}{\longrightarrow} G_1'\{Q/X\}$ も得る．ここで，X は G_1 においてはガードされていないが，再び $G_1'\{P/X\} =_m G_1'\{E\{P/X\}/X\}$ を利用する．すなわち，$P_1' \equiv G_1'\{P/X\} \Longrightarrow P''$ であるので，ある P''' について，$G_1'\{E\{P/X\}/X\} \Longrightarrow P'''$ かつ $P'' \approx_m P'''$ を得る．X は $G_1'\{E/X\}$ において逐次的かつガードされているので，X が逐次的かつガードされているよ

うなある G' について，$G'_1\{E/X\} \Longrightarrow G'$ かつ $P''' \equiv G'\{E\{P/X\}/X\}$ を得る．すなわち，Q を代入して $G'_1\{E\{Q/X\}/X\} \Longrightarrow G'\{Q/X\}$ も得る．また，$G'_1\{Q/X\} =_m G'_1\{E\{Q/X\}/X\}$ であるので，ある Q'' について，$G'_1\{Q/X\} \Longrightarrow Q''$ かつ $G'\{Q/X\} \approx_m Q''$ を得る．

以上をまとめると，$G\{E\{Q/X\}/X\} \Longrightarrow G_1\{Q/X\} \xrightarrow{a'\theta^n} G'_1\{Q/X\} \Longrightarrow Q''$ である．さらに，$G\{Q/X\} =_m G\{E\{Q/X\}/X\}$ であるので，ある Q' と a'' について，$G\{Q/X\} \stackrel{a''\theta^n}{\Longrightarrow} Q'$ かつ $Q'' \approx_m Q'$ かつ $a'\theta^n \doteq a''\theta^n$ を得る．すなわち，$P' \approx_m P'' \approx_m P''' \equiv G'\{E\{P/X\}/X\}\mathcal{S}G'\{E\{Q/X\}/X\} \approx_m Q'' \approx_m Q'$ かつ $a\theta^n \doteq a'\theta^n \doteq a''\theta^n$ である．

(3) $G\{P/X\} \Rightarrow P' \Rightarrow P''$ かつ $a \notin mon(P')$ とする．(2) の証明で示したように，$G\{P/X\} \Rightarrow P'$ ならば，ある Q_1 と，X が逐次的かつガードされているようなある G' について，$G\{Q/X\} \Longrightarrow Q_1$ かつ $P' \approx_m G'\{P/X\}\mathcal{S}G'\{Q/X\} \approx_m Q_1$ である．ここで，$a \notin mon(P')$ かつ $P' \Rightarrow P''$ より，X が逐次的かつガードされているようなある G'' と G''' について，$G'\{P/X\} \Rightarrow G''\{P/X\} \Rightarrow G'''\{P/X\}$ かつ $a \notin mon(G''\{P/X\})$ かつ $P'' \approx_m G'''\{P/X\}$ を得ることができる．すなわち，$G'\{Q/X\} \Rightarrow G''\{Q/X\} \Rightarrow G'''\{Q/X\}$ かつ $a \notin mon(G''\{Q/X\})$ である．最後に，$G'\{Q/X\} \approx_m Q_1$ より，ある Q' と Q'' について，$Q_1 \Rightarrow Q' \Rightarrow Q''$ かつ $a \notin mon(Q')$ かつ $G'''\{Q/X\} \approx_m Q''$ を得る．よって，$G\{Q/X\} \Rightarrow Q_1 \Rightarrow Q' \Rightarrow Q''$ かつ $P'' \approx_m G'''\{P/X\}\mathcal{S}G'''\{Q/X\} \approx_m Q''$ である．

∎

4.5.3 公理系

本小節では，有限プロセスに対する監視合同の健全で完全な公理系を与える．有限プロセスとはプロセス定数を含まない (再帰をもたない) プロセスのことである．公理系とは等式の集合であり，これによって，プロセスの等価性を式変形によって検証できるようになる．

まず，監視合同よりも定義の簡単な強監視合同の公理系について述べる．ここで，強監視合同は次のように強監視等価をもとに定義される合同関係であり，監視合同に含まれる関係である．

定義 4.5.10　プロセス上の二項関係 $\mathcal{S}$ が**強監視双模倣** (strong monitor bisimulation) であるとは，$(P,Q) \in \mathcal{S}$ ならば，任意の $\alpha \in Act$ 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' について，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．
(ii)　$Q \xrightarrow{\alpha} Q'$ ならば，ある P' について，$P \stackrel{\widehat{\alpha}}{\Longrightarrow} P'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．
(iii)　$mon(P) = mon(Q)$

∎

定義 4.5.11　もし，ある強監視双模倣 $\mathcal{S}$ において $(P,Q) \in \mathcal{S}$ ならば，プロセス 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．∎

定義 4.5.12　もし，任意の $\alpha \in Act$ 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' と α' について，$Q \stackrel{\alpha'}{\Longrightarrow} Q'$, $P' \simeq_m Q'$, $\alpha \doteq \alpha'$ を満たす．
(ii) $Q \xrightarrow{\alpha} Q'$ ならば，ある Q' と α' について，$P \stackrel{\alpha'}{\Longrightarrow} P'$, $P' \simeq_m Q'$, $\alpha \doteq \alpha'$ を満たす．
(iii) $mon(P) = mon(Q)$

∎

強監視双模倣は，監視双模倣の条件 $(iii), (iv)$ を，$P \equiv P' \equiv P''$, $Q \equiv Q' \equiv Q''$ に強めた場合に相当する．すなわち，$P \simeq_m Q$ ならば $P \approx_m Q$ である．監視合同であり，強監視合同でない例を次に示す．

$$a‼.(b⁇.\mathbf{0} + \tau.\tau.b⁇.\mathbf{0}) =_m a‼.\tau.b⁇.\mathbf{0}, \quad a‼.(b⁇.\mathbf{0} + \tau.\tau.b⁇.\mathbf{0}) \not\cong_m a‼.\tau.b⁇.\mathbf{0}$$

左辺のプロセス $a‼.(b⁇.\mathbf{0} + \tau.\tau.b⁇.\mathbf{0})$ は a を送信後，b を受信可能な状態から一時的に受信不能な状態を経て再び受信可能な状態になる．一方，右辺のプロセスは一度 b を受信可能な状態になった後は，受信不能な状態になることはない．これらの振舞いは観測的には区別できないため監視等価になるが，強監視等価にはならない．強監視等価は観測的な等価性としては適切ではないが，定義が簡単なため，まず強監視等価の公理系を扱うことは有効である．

次に強監視合同の公理系として $\mathcal{A}_1$ を与える．

定義 4.5.13　2 つの有限プロセス P と Q の等価性が公理系 $\mathcal{A}_1$ から推論されるならば，$\mathcal{A}_1 \vdash P = Q$ と書く．ここで，公理系 $\mathcal{A}_1$ 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 $P = P + \mathbf{0}$

E1 $(\sum_{(i\in I_1)} a_i(\theta_i)^{m_i}.P'_i)|(\sum_{(i\in I_2)} b_i(\phi_i)^{n_i}.Q'_i)$

$= \sum_{(i\in I_1)}\{a_i(\theta_i)^{m_i}.(P'_i|Q) : \theta_i \in \{!,?\} \text{ or } a_i \notin mon(Q)\}$

$+ \sum_{(i\in I_2)}\{b_i(\phi_i)^{n_i}.(P|Q'_i) : \phi_i \in \{!,?\} \text{ or } b_i \notin mon(P)\}$

$+ \sum_{(i\in I_1)}\sum_{(j\in I_2)}\{a_i(\theta_i)^{(m_i-n_j)}.(P'_i|Q'_j) : a_i = b_j, \theta_i \in \{!,\!!\}, \phi_j = com(\theta_i), m_i \geq n_j\}$

$+ \sum_{(i\in I_1)}\sum_{(j\in I_2)}\{b_j(\phi_j)^{(n_j-m_i)}.(P'_i|Q'_j) : a_i = b_j, \phi_j \in \{!,\!!\}, \theta_i = com(\phi_j), n_j \geq m_i\}$

$+ \sum_{(i\in I_1)}\sum_{(j\in I_2)}\{a_i(\theta_i)^{(m_i+n_j)}.(P'_i|Q'_j) : a_i = b_j, \theta_i = \phi_j \in \{?,??\}\}$

E2 $(\sum_{(i\in I)} a_i(\theta_i)^{n_i}.P'_i)\backslash L$

$= \sum_{(i\in I)}\{a_i(\theta_i)^{n_i}.(P'_i\backslash L) : a_i \notin L\} + \sum_{(i\in I)}\{a_i!^0.(P'_i\backslash L) : a_i \in L, \theta_i \in \{!,\!!\}, n_i{=}0\}$

E3 $(\sum_{(i\in I)} a_i(\theta_i)^{n_i}.P'_i)[f] = \sum_{(i\in I)} f(a_i)(\theta_i)^{n_i}.(P'_i[f])$

T1 $\alpha.(\tau.P + P) = \alpha.P$

T2 $P + \alpha.Q = P \quad$ if $\alpha.Q \in taus(P),\ mon(\alpha.Q) \subseteq mon(P)$

T3 $\alpha.(P + \tau.Q) + \alpha.Q = \alpha.(P + \tau.Q)$

T4 $\tau.P = \tau'.P$

ここで，$taus(P)$ はプロセス P の構造に関して帰納的に次のように定義される関数である．

$$taus(P) = \begin{cases} \{\alpha.P'\} & (P \equiv \alpha.P', \alpha \notin \mathcal{T}) \\ \{\alpha.P'\} \cup taus(P') & (P \equiv \alpha.P', \alpha \in \mathcal{T}) \\ taus(P_1) \cup taus(P_2) & (P \equiv P_1 + P_2) \\ \emptyset & (\text{ 上記以外 }) \end{cases}$$

∎

M1-4，**E1-3**，**T1-4** は各々モノイド規則，展開規則，τ 規則である．関数 $taus$ はプロセスから内部アクションで到達できる状態を取り出すために使われる．以下，$\mathcal{A}_1$ が有限プロセスに対する強監視合同の健全で完全な公理系であることを証明する．

まず，標準形と完全標準形を次のように定義する．

定義 4.5.14 もし $P \equiv \sum_{i=1}^{m} \alpha_i.P_i$ であり，P_i も標準形であるならば，P は**標準形**である．また，各 $\alpha_i.P_i$ を P の**選択プロセス**と呼ぶ. ∎

定義 4.5.15 もし次の条件が満たされるならば，P は**完全標準形**である．

(i) $P \equiv \sum_{i=1}^{m} \alpha_i.P_i$，ここで，$P_i$ も完全標準形である．
(ii) もし $P \stackrel{a\theta^n}{\Longrightarrow} P'$ かつ $(\theta \neq ?? \text{ または } a \in mon(P))$ ならば，$P \stackrel{a\theta^n}{\longrightarrow} P'$ である．

∎

次の補題は任意の標準形のプロセスを $\mathcal{A}_1$ によって完全標準形に変換できることを示している．

補題 4.5.10 P は標準形であるとする．もし $P \stackrel{a\theta^n}{\Longrightarrow} P'$ かつ $(\theta \neq ?? \text{ または } a \in mon(P))$ ならば，$\mathcal{A}_1 \vdash P = P + a\theta^n.P'$ である．

証明 P の構造に関する帰納法を用いる．遷移 $P \stackrel{a\theta^n}{\Longrightarrow} P'$ について次の 3 つの場合が考えられる．

1. $a\theta^n.P'$ が P の選択プロセスである場合：このときは，**M1-3** によって，$\mathcal{A}_1 \vdash P = P + a\theta^n.P'$ を得る．

2. $a\theta^n.Q$ が P の選択プロセスであり，$Q \stackrel{\tau}{\Longrightarrow} P'$ の場合：帰納法の仮定より，$\mathcal{A}_1 \vdash Q = Q + \tau.P'$ $(*1)$ を得る．すなわち，次の等式が得られる．

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

3. $P \stackrel{\tau}{\Longrightarrow} Q \stackrel{a\theta^n}{\longrightarrow} Q' \Longrightarrow P'$ の場合：Q は標準形で $Q \stackrel{a\theta^n}{\longrightarrow} Q'$ より，$a\theta^n.Q'$ は Q の選択プロセスである．このとき，関数 $taus$ の定義から，P が標準形であり，$P \stackrel{\tau}{\Longrightarrow} Q$ であるので，$a\theta^n.Q' \in taus(P)$ $(*2)$ を証明できる．さらに，$(\theta \neq ?? \text{ または } a \in mon(a\theta^n.Q'))$ であるので，$a\theta^n.Q' \stackrel{a\theta^n}{\longrightarrow} Q' \Longrightarrow P'$ に対する帰納法の仮定より，$\mathcal{A}_1 \vdash a\theta^n.Q' = a\theta^n.Q' + a\theta^n.P'$ $(*3)$ を得る．また，仮定 $(\theta \neq ?? \text{ または}$

$a \in mon(P)$) より，($\theta \neq ?$? または $mon(a\theta^n.Q') = \{a\} \subseteq mon(P)$) $(*4)$ である．よって，次の等式が得られる．

$$\begin{aligned}
\mathcal{A} \vdash P &= P + a\theta^n.Q' && \text{by } \mathbf{T2}, (*2), (*4) \\
&= P + a\theta^n.Q' + a\theta^n.P' && \text{by } (*3) \\
&= P + a\theta^n.P' && \text{by } \mathbf{T2}, (*2), (*4)
\end{aligned}$$

∎

次に，強監視等価を強監視合同に強めるために有効な命題を与える．

命題 4.5.11 $P \simeq_m Q \iff (P \cong_m Q)$ または $(P \cong_m \tau.Q{+}Q)$ または $(\tau.P{+}P \cong_m Q)$

証明 $(\Leftarrow)$ の場合：強監視合同であれば強監視等価であるので明らか．

$(\Rightarrow)$ の場合：$P \simeq_m Q$ を仮定し，次の 3 つの場合について証明する．

1. ある P' について，$P \xrightarrow{\tau'} P' \simeq_m Q$ $(*1)$ の場合：$P \cong_m \tau.Q + Q$ を示す．

   (i) $P \xrightarrow{\alpha} P'$ とする．$P \simeq_m Q$ であるので，ある Q' について，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $P' \simeq_m Q'$ を得る．もし α が内部アクションであれば，$Q \Rightarrow Q'$ である．すなわち，$\tau.Q + Q \xrightarrow{\tau} Q \Rightarrow Q'$ かつ $\alpha \doteq \tau$ である．それ以外 $(\widehat{\alpha} = \alpha)$ では，$\tau.Q + Q \stackrel{\alpha}{\Longrightarrow} Q'$ である．

   (ii) $\tau.Q + Q \xrightarrow{\alpha} Q'$ とする．この遷移を導く最後の規則について場合分けする．

   (1) $\mathbf{Choice}_1$ による場合：$\tau.Q \xrightarrow{\alpha} Q'$ である．さらに，$\mathbf{Act}$ より $\alpha = \tau$ かつ $Q' \equiv Q$ を得る．このとき，$(*1)$ より $P \xrightarrow{\tau'} P' \simeq_m Q \equiv Q'$ かつ $\tau \doteq \tau'$ である．

   (2) $\mathbf{Choice}_2$ による場合：$Q \xrightarrow{\alpha} Q'$ である．$\alpha \notin \mathcal{T}$ ならば，$P \simeq_m Q$ であるので，$P \stackrel{\alpha}{\Longrightarrow} P''$ かつ $P'' \simeq_m Q'$ を得る．それ以外 $(\alpha \in \mathcal{T})$ とする．このとき，$P' \simeq_m Q$ であるので，ある P'' について，$P' \Longrightarrow P''$ かつ $P'' \simeq_m Q'$ を得る．ここで，$(*1)$ より $P \xrightarrow{\tau'} P' \Rightarrow P''$ かつ $\alpha \doteq \tau'$ である．

   (iii) $P \simeq_m Q$ より，$mon(P) = mon(Q)$ である．$mon(\tau.Q) = \emptyset$ であるので，$mon(P) = mon(\tau.Q + Q)$ を得る．

2. ある Q' について，$Q \xrightarrow{\tau'} Q' \simeq_m P$ $(*2)$ の場合：上記の場合に対称である．

3. それ以外の場合 $(*3)$：$P \cong_m Q$ を示す．

   (i) $P \xrightarrow{\alpha} P'$ とする．$P \simeq_m Q$ であるので，ある Q' について，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $P' \simeq_m Q'$ である．$\alpha \notin \mathcal{T}$ ならば，$Q \stackrel{\alpha}{\Longrightarrow} Q'$ である．それ以外 $(\alpha = \tau' \in \mathcal{T})$ とする．このとき，$Q \Longrightarrow Q'$ であるが，もし $Q \equiv Q'$ ならば，$P \xrightarrow{\tau'} P' \simeq_m Q' \equiv Q$ となり，$(*3)$ に矛盾する．すなわち，ある τ について，$Q \stackrel{\tau}{\Longrightarrow} Q'$ かつ $\alpha \doteq \tau$ である．

   (ii) は (i) に対称であり，(iii) は明らかである．

∎

以上の補題 4.5.10と命題 4.5.11より，$\mathcal{A}_1$ は有限プロセスに対する強監視合同の健全で完全な公理系であることを示す定理 4.5.12を得る．

定理 4.5.12 P と Q を有限プロセスとする．このとき，次の関係が成り立つ．

$$P \cong_m Q \quad \Longleftrightarrow \quad \mathcal{A}_1 \vdash P = Q$$

証明 $(\Leftarrow$: 健全性 $)$ 強監視合同は合同関係であるので，$\mathcal{A}_1$ の各等式が定義 4.5.12の 3 条件を満たすことを証明する．ここでは，**T2**$(\alpha.Q \in taus(P)$ かつ $mon(\alpha.Q) \subseteq mon(P)$ ならば，$P + \alpha.Q \cong_m P)$ の等式のみ示す．

(i) $P + \alpha.Q \xrightarrow{\alpha'} P'$ とする．この遷移が P により導かれたのであれば，明らかに $P \xrightarrow{\alpha'} P'$ かつ $P' \simeq_m P'$ である．それ以外では，$\alpha.Q \xrightarrow{\alpha'} P'$ より，$\alpha = \alpha'$ かつ $P' \equiv Q$ である．ここで，$\alpha.Q \in taus(P)$ より，$P \Longrightarrow \xrightarrow{\alpha} Q$ を得る．

(ii) $P \xrightarrow{\alpha'} P'$ とする．明らかに，$P + \alpha.Q \xrightarrow{\alpha'} P'$ である．

(iii) 条件 $mon(\alpha.Q) \subseteq mon(P)$ より，$mon(P + \alpha.Q) = mon(P)$ である．

$(\Rightarrow$: 完全性 $)$ $P \cong_m Q$ とする．まず，**M4** と **E1-3** と補題 4.5.10より，$\mathcal{A}_1 \vdash P = P_0$ かつ $\mathcal{A}_1 \vdash Q = Q_0$ のような完全標準形のプロセス P_0 と Q_0 が存在する．すなわち，$\mathcal{A}_1$ は健全であるので $P_0 \cong_m Q_0$ である．このとき，$\mathcal{A}_1 \vdash P_0 = Q_0$ を P_0 と Q_0 の深さの和に関する帰納法を用いて証明する．ここで，標準形 P_0 の深さ $(depth(P_0)$ と書く $)$ とは，P_0 が停止するまでに実行できる最長のアクション列の長さである．

深さ 0 の場合 $\mathcal{A}_1 \vdash \mathbf{0} = \mathbf{0}$ は明らかである．深さ 1 以上の場合について，$(a\theta^n.P_0')$ が P_0 の選択プロセスであると仮定する．すなわち，$P_0 \xrightarrow{a\theta^n} P_0'$ である．このとき，$P_0 \cong_m Q_0$ であるので，ある Q_0' と a' について，$Q_0 \stackrel{a'\theta^n}{\Longrightarrow} Q_0'$ かつ $P_0' \simeq_m Q_0'$ かつ $a\theta^n \doteq a'\theta^n$ である．ここで，$\theta \neq ??$ または $a \in mon(P_0) = mon(Q_0)$ であり，Q_0 は完全標準形であるので，$a'\theta^n.Q_0'$ は Q_0 の選択プロセスである．また，$P_0' \simeq_m Q_0'$ であるので，命題 4.5.11 より，$P_0' \cong_m Q_0'$ または $P_0' \cong_m \tau.Q_0' + Q_0'$ または $\tau.P_0' + P_0' \cong_m Q_0'$ である．

(1) $P_0' \cong_m Q_0'$ の場合：P_0' と Q_0' の深さの和は P_0 と Q_0 の深さの和より 2 以上減っているので，帰納法の仮定より，$\mathcal{A}_1 \vdash P_0' = Q_0'$ を得る．すなわち，**T4** により，$\mathcal{A}_1 \vdash a\theta^n.P_0' = a\theta^n.Q_0' = a'\theta^n.Q_0'$ を得る．

(2) $P_0' \cong_m \tau.Q_0' + Q_0'$ の場合：$\tau.Q_0' + Q_0'$ は完全標準形であり，P_0' と $\tau.Q_0' + Q_0'$ の深さの和は P_0 と Q_0 の深さの和より 1 以上減っているので，帰納法の仮定より，$\mathcal{A}_1 \vdash P_0' = \tau.Q_0' + Q_0'$ を得る．すなわち，**T1** と **T4** により，$\mathcal{A}_1 \vdash a\theta^n.P_0' = a\theta^n.(\tau.Q_0' + Q_0') = a\theta^n.Q_0' = a'\theta^n.Q_0'$ を得る．

(3) $\tau.P_0' + P_0' \cong_m Q_0'$ の場合：上記の (2) に対称である．

すなわち，公理系 $\mathcal{A}_1$ によって，P_0 の任意の選択プロセス $(a\theta^n.P_0')$ は，Q_0 のある選択プロセスに等しくなることが証明された．同様に，逆も証明できる．このことから，$\mathcal{A}_1 \vdash P = P_0 = Q_0 = Q$ を得られる． ∎

次に，有限プロセスに対する監視合同の健全で完全な公理系として $\mathcal{A}_2$ を与える．

定義 4.5.16　2 つの有限プロセス P と Q の等価性が公理系 $\mathcal{A}_2$ から推論されるならば，$\mathcal{A}_2 \vdash P = Q$ と書く．ここで，公理系 $\mathcal{A}_2$ は $\mathcal{A}_1$ の **M1-4**，**E1-3**，**T1-4** と次の等式から構成される．

T5　$\alpha.(P + \sum_{i\in I} Q_i' + \tau.(P + \sum_{i\in I} \tau.(Q_i + Q_i'))) = \alpha.(P + \sum_{i\in I} \tau.(Q_i + Q_i'))$

∎

T5 は左辺の $\sum_{i\in I} Q_i'$ に含まれる受信アクション $a??^n$ を，右辺のように内部アクション τ の後に延期できることを表している．本小節のはじめに紹介した監視合同であり強監視合同でない例 $(a!!.(b??.\mathbf{0} + \tau.\tau.b??.\mathbf{0}) =_m a!!.\tau.b??.\mathbf{0})$ は **T5** により導かれる．

以下，$\mathcal{A}_2$ が有限プロセスに対する監視合同の健全で完全な公理系であることを証明する．まず，完全監視標準形を定義する．

定義 4.5.17 もし次の条件が満たされるならば，P は**完全監視標準形**である．

(i) $P \equiv \sum_{i=1}^{m} \alpha_i.P_i$ ここで，P_i も完全監視標準形である．
(ii) もし $P \stackrel{a\theta^n}{\Longrightarrow} P'$ かつ $(\theta \neq ?? \text{ または } a \in mon(P))$ ならば，$P \stackrel{a\theta^n}{\longrightarrow} P'$ である．
(iii) もし $P \Rightarrow P' \Rightarrow P''$ かつ $a \notin mon(P')$ かつ $P \approx_m P''$ ならば，$a \notin mon(P)$ である．

∎

次の補題は完全監視標準形の特性を明確に表している．つまり，完全監視標準形のプロセスはそれと監視等価な全てのプロセスのなかで最も受信可能なアクションが少ないプロセスである．

補題 4.5.13 P_m は完全監視標準形であるとする．このとき，もし $P_m \approx_m P$ ならば，$mon(P_m) \subseteq mon(P)$ である．

証明 $a \notin mon(P)$ とする．$P_m \approx_m P$ であるので，ある P' と P'' について，$P_m \Rightarrow P' \Rightarrow P''$ かつ $a \notin mon(P')$ かつ $P'' \approx_m P$ である．ここで，$P_m \approx_m P \approx_m P''$ であるので，完全監視標準形の条件 (iii) より，$a \notin mon(P_m)$ である． ∎

この完全監視標準形を用いることによって，次のように監視等価 (や監視合同) を強監視等価 (や強監視合同) に強めることができる．

命題 4.5.14 P と Q は完全監視標準形であるとする．このとき，次の関係が成り立つ．

(1) もし $P \approx_m Q$ ならば，$P \simeq_m Q$ である．

(2) もし $P =_m Q$ ならば，$P \cong_m Q$ である．

証明 (1) P と Q の深さの和に関する帰納法を用いる．深さ 0 の場合 $(P \equiv Q \equiv \mathbf{0})$ は明らかである．深さ 1 以上の場合について，$P \approx_m Q$ を仮定する．

(i) $P \stackrel{\alpha}{\longrightarrow} P'$ とする．$P \approx_m Q$ であるので，ある Q' について，$Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $P' \approx_m Q'$ を得る．$depth(P') + depth(Q') + 1 \leq depth(P) + depth(Q)$ であるので，帰納法の仮定より，$P' \simeq_m Q'$ を得る．

(ii) 上記の (i) の場合に対称である．

(iii) $a \notin mon(P)$ とする．$P \approx_m Q$ であるので，ある Q' と Q'' について，$Q \Rightarrow Q' \Rightarrow Q''$ かつ $a \notin mon(Q')$ かつ $P \approx_m Q''$ を得る．ここで，$Q'' \approx_m P \approx_m Q$ かつ Q は完全監視標準形であるので，$a \notin mon(Q)$ を得る．すなわち，$mon(P) \supseteq mon(Q)$ である．対称的な証明により $mon(P) \subseteq mon(Q)$ も得られる．

すなわち，$P \simeq_m Q$ である．

(2) $P =_m Q$ ならば $mon(P) = mon(Q)$ であるので，(1) の場合より簡単である． ∎

次の補題 4.5.15と補題 4.5.16は補題 4.5.17の証明に使われる．

補題 4.5.15 もし $P \Rightarrow P' \Rightarrow P''$ かつ $P \approx_m P''$ ならば，$P \approx_m P'$ である．

証明 (i) $P \xrightarrow{\alpha} Q$ とする．$P' \Rightarrow P'' \approx_m P$ であるので，ある Q' について，$P' \Rightarrow P'' \stackrel{\widehat{\alpha}}{\Longrightarrow} Q'$ かつ $Q \approx_m Q'$ を得る．(ii) $P' \xrightarrow{\alpha} Q$ とすると，$P \Rightarrow P' \xrightarrow{\alpha} Q$ である．(iii) $a \notin mon(P)$ とする．$P' \Rightarrow P'' \approx_m P$ であるので，ある Q と Q' について，$P' \Rightarrow P'' \Rightarrow Q \Rightarrow Q'$ かつ $a \notin mon(Q)$ かつ $Q' \approx_m P$ を得る．(iv) $a \notin mon(P')$ のときは明らか． ∎

補題 4.5.16 もし $P \xrightarrow{\tau} P' \approx_m Q \approx_m P$ かつ $mon(P) = mon(Q)$ ならば，
$P =_m \tau.Q + Q$ である．

証明 (i) $P \xrightarrow{\alpha} Q'$ とする．$P \approx_m Q$ であるので，ある Q'' について，$\tau.Q + Q \xrightarrow{\tau} Q \stackrel{\widehat{\alpha}}{\Longrightarrow} Q''$ かつ $Q' \approx_m Q''$ を得る．(ii) $\tau.Q + Q \xrightarrow{\alpha} Q'$ とする．$\tau.Q \xrightarrow{\alpha} Q'$ の場合は，$\alpha = \tau$ かつ $Q' \equiv Q$ であるので，$P \xrightarrow{\tau} P' \approx_m Q \equiv Q'$ を得る．また，$Q \xrightarrow{\alpha} Q'$ の場合は，$P \xrightarrow{\tau} P' \approx_m Q$ であるので，ある P'' について，$P \xrightarrow{\tau} P' \stackrel{\widehat{\alpha}}{\Longrightarrow} P''$ かつ $P'' \approx_m Q'$ を得る．(iii) 仮定による． ∎

これらの補題を用いて，任意の完全標準形のプロセスは $\mathcal{A}_2$ によって完全監視標準形に変換可能であることを次のように証明できる．

補題 4.5.17 任意の完全標準形のプロセス P と任意の α について，$\mathcal{A}_2 \vdash \alpha.P = \alpha.P_m$ かつ $depth(P_m) \leq depth(P)$ のような，完全監視標準形 P_m が存在する．

証明 P の深さに関する帰納法を用いる．深さ 0 の場合 $(P \equiv \mathbf{0})$ は明らかである．$P \stackrel{\tau}{\Longrightarrow} P' \Rightarrow P''$ かつ $P \approx_m P''$ のような P' と P'' があると仮定する．もし無け

れば，帰納法の仮定より P は容易に完全監視標準形に変換される．この仮定と命題 4.5.15より，$P \approx_m P'$ である．さらに，P は完全標準形であるので，$P \xrightarrow{\tau} P'$ である．ここで，$depth(P') < depth(P)$ であるので，帰納法の仮定より，任意の α について，$\mathcal{A}_2 \vdash \alpha.P' = \alpha.P_m$ かつ $depth(P_m) \leq depth(P')$ のような完全監視標準形 P_m が存在する $(*1)$．さらに，$\mathcal{A}_2$ が監視合同に健全であることは容易に証明できるので，$P' \approx_m P_m$ を得る．すなわち，$P \approx_m P' \approx_m P_m$ であるので，補題 4.5.13より，$mon(P_m) \subseteq mon(P)$ を得る．このとき，P を次の条件を満たす P_1 と P_2 に分割できる：$P \equiv P_1 + P_2$ かつ $mon(P_m) = mon(P_1)$ かつ $mon(P_1) \cap mon(P_2) = \emptyset$ かつ，$P_2 \xrightarrow{a\theta^n}$ ならば $\theta = ?\!?$ である．

まず，$P_1 \approx_m P_m$ を示す．(i) $P_1 \xrightarrow{\alpha} P_1'$ とする．このとき，**Choice**$_1$ より $P_1 + P_2 \xrightarrow{\alpha} P_1'$ を導ける．また，$P_m \approx_m P \equiv P_1 + P_2$ であるので，ある P_m' について，$P_m \xRightarrow{\widehat{\alpha}} P_m'$ かつ $P_1' \approx_m P_m'$ を得る．(ii) $P_m \xrightarrow{\alpha} P_m'$ とする．$P' \approx_m P_m$ であるので，ある P_1'' について，$P' \xRightarrow{\widehat{\alpha}} P_1''$ かつ $P_1'' \approx_m P_m'$ を得る．P_2 は内部アクションを実行できないので，$P \equiv P_1 + P_2 \xrightarrow{\tau} P'$ は $P_1 \xrightarrow{\tau} P'$ によって導かれる．すなわち，$P_1 \xrightarrow{\tau} P' \xRightarrow{\widehat{\alpha}} P_1''$ である．$(iii), (iv)$ 仮定 $mon(P_m) = mon(P_1)$ より明らか．よって，$P_1 \approx_m P_m$ が証明された．

次に，$\mathcal{A}_2 \vdash P_1 = \tau.P_m + P_m$ を証明する．$P_1 \xrightarrow{\tau} P'$ かつ $P' \approx_m P_m \approx_m P_1$ であるので，補題 4.5.16より，$P_1 =_m \tau.P_m + P_m$ を得る．(1) $\alpha.P_1'$ を P_1 の選択プロセスであるとする．すなわち，$P_1 \xrightarrow{\alpha} P_1'$ である．$\tau.P_m + P_m$ は完全標準形であり，$mon(P_m) = mon(P_1)$ であるので，ある P_m' と α' について，$\tau.P_m + P_m \xrightarrow{\alpha'} P_m'$ かつ $\alpha \doteq \alpha'$ かつ $P_1' \approx_m P_m'$ を得る．ここで，$depth(P_1') < depth(P)$ であるので，帰納法の仮定より，$\mathcal{A}_2 \vdash \alpha.P_1' = \alpha.P_{m1}'$ かつ $depth(P_{m1}') \leq depth(P_1')$ のような完全監視標準形 P_{m1}' が存在する．さらに，$P_{m1}' \approx_m P_1' \approx_m P_m'$ であるので，命題 4.5.14より，$P_{m1}' \simeq_m P_m'$ を得る．すなわち，$\alpha.P_{m1}' \cong_m \alpha.P_m' \cong_m \alpha'.P_m'$ であるので，定理 4.5.12より，$\mathcal{A}_2 \supseteq \mathcal{A}_1 \vdash \alpha.P_{m1}' = \alpha'.P_m'$ を得る．以上，P_1 の各選択プロセスは $\mathcal{A}_2$ によって，$\tau.P_m + P_m$ のある選択プロセスに等しくできることが証明された．(2) 対称的に，$\tau.P_m + P_m$ の各選択プロセスも P_1 のある選択プロセスに等しくできる．すなわち，$\mathcal{A}_2 \vdash P_1 = \tau.P_m + P_m$ を証明した $(*2)$．

また，P_2 の仮定 $(mon(P_1) \cap mon(P_2) = \emptyset$，$P_2 \xrightarrow{a\theta^n}$ ならば $\theta = ?\!?\ )$ より，P_2 は $\sum_{i \in I} a_i ?\!?^{n_i}.P_{2i}$ の形をもつ．ここで，任意の $i \in I$ について $a_i \notin mon(P_1)$ である．すなわち，$\alpha.P_2'$ を P_2 の選択プロセスとすると，ある $j \in I$ について，$\alpha = a_j ?\!?^{n_j}$ かつ $P_2 \equiv P_{2j}$ である．$P_2 \xrightarrow{a_j ?\!?^{n_j}} P_{2j}$ であるので，**Choice**$_2$ より $P \equiv P_1 + P_2 \xrightarrow{a_j ?\!?^{n_j}} P_{2j}$ を導く．

このとき，$P \approx_m P_m \approx_m P_1$ かつ $a_j \notin mon(P_1)$ かつ P_1 は完全標準形であるので，ある Q_j と Q_{2j} について，$P_1 \xrightarrow{\tau_j} Q_j \xrightarrow{a_j ?^{n_j}} Q_{2j}$ $(*3)$ かつ $P_{2j} \approx_m Q_{2j}$ を得る．P_{2j} と Q_{2j} の深さは共に P より浅いので，帰納法の仮定より，任意の α について，$\mathcal{A}_2 \vdash \alpha.P_{2j} = \alpha.P_{m2j}$ かつ $\mathcal{A}_2 \vdash \alpha.Q_{2j} = \alpha.Q_{m2j}$ となる完全監視標準形 P_{m2j} と Q_{m2j} が存在する．ここで，$\mathcal{A}_2$ の健全性より $P_{m2j} \approx_m P_{2j} \approx_m Q_{2j} \approx_m Q_{m2j}$ である．すなわち，命題 4.5.14より $P_{m2j} \simeq_m Q_{m2j}$ である．さらに，定理 4.5.12より，任意の α について，$\mathcal{A}_1 \vdash \alpha.P_{m2j} = \alpha.Q_{m2j}$ を得る．以上をまとめて，$\mathcal{A}_2 \vdash \alpha.P_{2j} = \alpha.P_{m2j} = \alpha.Q_{m2j} = \alpha.Q_{2j}$ である $(*4)$.

以上，$(*1)$,$(*2)$,$(*3)$,$(*4)$ により，任意の α について，次の等式を導ける．

$$
\begin{aligned}
&\mathcal{A}_2 \vdash \alpha.P \\
&\quad = \alpha.(P + \tau.P') && \text{by } \mathbf{M3} \\
&\quad = \alpha.(P + \tau.P_m) && \text{by } (*1) \\
&\quad = \alpha.(P + \tau.(P_m + \tau.P_m)) && \text{by } \mathbf{T1} \\
&\quad = \alpha.(P + \tau.P_1) && \text{by } (*2) \\
&\quad = \alpha.(P + \tau.(P_1 + \textstyle\sum \tau_i.(Q_i + a_i ?^{n_i}.Q_{2i}))) && \text{by } \mathbf{M3}, (*3) \\
&\quad = \alpha.(P_1 + \textstyle\sum a_i ?^{n_i}.P_{2i} + \tau.(P_1 + \sum \tau.(Q_i + a_i ?^{n_i}.Q_{2i}))) && \text{by } P \equiv P_1 + P_2, \mathbf{T4} \\
&\quad = \alpha.(P_1 + \textstyle\sum a_i ?^{n_i}.Q_{2i} + \tau.(P_1 + \sum \tau.(Q_i + a_i ?^{n_i}.Q_{2i}))) && \text{by } (*4) \\
&\quad = \alpha.(P_1 + \textstyle\sum \tau.(Q_i + a_i ?^{n_i}.Q_{2i})) && \text{by } \mathbf{T5} \\
&\quad = \alpha.P_1 && \text{by } \mathbf{M3} \\
&\quad = \alpha.(\tau.P_m + P_m) && \text{by } (*2) \\
&\quad = \alpha.P_m
\end{aligned}
$$

ここで，$depth(P_m) \leq depth(P') < depth(P)$ である． ∎

最後に，$\mathcal{A}_2$ が有限プロセスに対して健全で完全な監視合同の公理系であることを示す定理を与える．

定理 4.5.18 P と Q を有限プロセスとする．このとき，次の関係が成り立つ．

$$P =_m Q \quad \Longleftrightarrow \quad \mathcal{A}_2 \vdash P = Q$$

証明 ($\Leftarrow$: 健全性) $\mathcal{A}_2$ の等式 **T5** が監視合同であることを示す．すなわち，

$$P_1 \equiv P + \textstyle\sum_{i \in I} Q'_i + \tau.(P + \sum_{i \in I} \tau.(Q_i + Q'_i)) \approx_m P_2 \equiv P + \sum_{i \in I} \tau.(Q_i + Q'_i)$$

を示す．この等式は $mon(P_2) \subseteq mon(P_1)$ であるため，一般に強監視等価の条件 (iii) は成り立たないが，監視等価の条件 (iv) は，$a \notin mon(P_2)$ ならば $P_1 \xrightarrow{\tau} P_2$ かつ $a \notin mon(P_2)$ であるので成り立つ．$(i), (ii), (iii)$ は簡単である．

($\Rightarrow$: 完全性) $P =_m Q$ とする．まず，**M4**, **E1-3** と補題 4.5.10より，$\mathcal{A}_1 \vdash P = P_0$ かつ $\mathcal{A}_1 \vdash Q = Q_0$ のような完全標準形 P_0 と Q_0 が存在する．$\mathcal{A}_1$ は健全であるので，$P_0 \cong_m P =_m Q \cong_m Q_0$ を得る．深さ 0 の場合 $P_0 \equiv \mathbf{0} \equiv Q_0$ は明らかである．深さ 1 以上とし，$(\alpha.P_0')$ が P_0 の選択プロセスであるとする．すなわち，$P_0 \xrightarrow{\alpha} P_0'$ である．ここで，$P_0 =_m Q_0$ であるので，ある Q_0' と α' について，$Q_0 \stackrel{\alpha'}{\Longrightarrow} Q_0'$ かつ $P_0' \approx_m Q_0'$ かつ $\alpha \doteq \alpha'$ を得る．さらに，Q_0 は完全標準形であり，かつ $mon(P_0) = mon(Q_0)$ であるので，$Q_0 \xrightarrow{\alpha'} Q_0'$ である．また，$P_0' \approx_m Q_0'$ より，$\alpha.P_0' =_m \alpha'.Q_0'$ である．ここで，補題 4.5.17より，$\mathcal{A}_2 \vdash \alpha.P_0' = \alpha.P_m'$ かつ $\mathcal{A}_2 \vdash \alpha'.Q_0' = \alpha'.Q_m'$ のような，完全監視標準形 P_m' と Q_m' が存在する．$\mathcal{A}_2$ は健全であるので，$\alpha.P_m' =_m \alpha.P_0' =_m \alpha'.Q_0' =_m \alpha'.Q_m'$ を得る．さらに，命題 4.5.14より，$\alpha.P_m' \cong_m \alpha'.Q_m'$ を得る．すなわち，定理 4.5.12により，$\mathcal{A}_1 \vdash \alpha.P_m' = \alpha'.Q_m'$ を得る．以上をまとめて，$\mathcal{A}_2 \vdash \alpha.P_0' = \alpha.P_m' = \alpha'.Q_m' = \alpha'.Q_0'$ である．これにより，P_0 の各選択プロセス $(\alpha.P_0')$ は $\mathcal{A}_2$ によって Q_0 のある選択プロセスに等しくなることが証明された．逆も同様である．よって，**M1-4** により，$\mathcal{A}_2 \vdash P_0 = Q_0$ を得られる．すなわち，$\mathcal{A}_2 \vdash P = P_0 = Q_0 = Q$ である． ∎

4.6 CCB の定義

Core-CCB は変数をもたないが，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，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，可算ブロードキャストや値の受渡しを容易に記述できるように 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 $I_{\texttt{max}}$ と記述する．

CCB のシンタックスは次のように定義される．

定義 4.6.1　*CCB* のプロセス式の集合 $\mathcal{E}_{ccb}$ は次の *BNF* 記法に従って定義される．

$$E ::= X \mid \mathbf{0} \mid a[e_1]!\langle c\rangle(e_2).E \mid a[e_1]!!\langle x\rangle(e_2).E \mid a[e]?\langle c\rangle(x).E \mid a[e]??\langle c\rangle(x).E$$
$$\mid E+E \mid E|E \mid E[f] \mid E\backslash L \mid \mathbf{if}\ b\ \mathbf{then}\ E \mid A(e_1,\cdots,e_n)$$

x は変数，b は真偽式，e, c は式である．ただし，c の計算結果は 0 以上の整数である．さらに，式 b, e, c の中の全ての変数は，それらの出現の左側で，束縛されていなければならない． ∎

n 引数をもつ各プロセス定数は，

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

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，プロセス変数も含まないとする．

CCB のプロセス式 $E \in \mathcal{E}_{ccb}$ の意味は，Core-CCB のプロセス式 $\mathcal{B}(E) \in \mathcal{E}_{core}$ 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 4.6.2　関数 $\mathcal{B} : \mathcal{E}_{ccb} \to \mathcal{E}_{core}$ を次のように与える．

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

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，

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

である．さらに，プロセス定数の定義式 $A(x_1, \cdots, x_n) \stackrel{\text{def}}{=} E$ は，各値 $(v_1, \cdots, v_n) \in V^n$ に対して，

$$A_{(v_1,\cdots,v_n)} \stackrel{\text{def}}{=} \mathcal{B}(E\{v_i/x_i\}_{i\in\{1,\cdots,n\}})$$

に変換される．∎

次の CCB のプロセス CHK を用いて，CCB の記述から Core-CCB の記述への変換過程の例を示す．

$$CHK \stackrel{\text{def}}{=} a!!\langle x\rangle.(\mathbf{if}\ x = 0\ \mathbf{then}\ zero!\langle 1\rangle.\mathbf{0} + \mathbf{if}\ x \neq 0\ \mathbf{then}\ nonzero!\langle 1\rangle.\mathbf{0})$$

この CHK はメッセージ a を可算ブロードキャストし，その受信プロセスが無ければメッセージ $zero$ を送信し，有れば $nonzero$ を送信するプロセスである．これは，$\mathcal{B}$ によって次のように Core-CCB の記述に変換される．

$$\begin{aligned}
&\mathcal{B}(CHK)\\
&\quad\equiv \mathcal{B}(a!!\langle x\rangle.(\mathbf{if}\ x = 0\ \mathbf{then}\ zero!\langle 1\rangle.\mathbf{0} + \mathbf{if}\ x \neq 0\ \mathbf{then}\ nonzero!\langle 1\rangle.\mathbf{0}))\\
&\quad\equiv \textstyle\sum_{n\in I_{\mathtt{max}}} a!!^n.\mathcal{B}(\mathbf{if}\ n = 0\ \mathbf{then}\ zero!\langle 1\rangle.\mathbf{0} + \mathbf{if}\ n \neq 0\ \mathbf{then}\ nonzero!\langle 1\rangle.\mathbf{0})\\
&\quad\equiv \textstyle\sum_{n\in I_{\mathtt{max}}} a!!^n.(\mathcal{B}(\mathbf{if}\ n = 0\ \mathbf{then}\ zero!\langle 1\rangle.\mathbf{0}) + \mathcal{B}(\mathbf{if}\ n \neq 0\ \mathbf{then}\ nonzero!\langle 1\rangle.\mathbf{0}))\\
&\quad\equiv a!!^0.(\mathcal{B}(zero!\langle 1\rangle.\mathbf{0}) + \mathbf{0}) + \textstyle\sum_{n\in I^+_{\mathtt{max}}} a!!^n.(\mathbf{0} + \mathcal{B}(nonzero!\langle 1\rangle.\mathbf{0}))\\
&\quad\equiv a!!^0.(zero!^1.\mathbf{0} + \mathbf{0}) + \textstyle\sum_{n\in I^+_{\mathtt{max}}} a!!^n.(\mathbf{0} + nonzero!^1.\mathbf{0})\\
&(\sim a!!^0.zero!^1.\mathbf{0} + \textstyle\sum_{n\in I^+_{\mathtt{max}}} a!!^n.nonzero!^1.\mathbf{0})
\end{aligned}$$

ここで，$I^+_{\mathtt{max}}$ は 1 以上 max 以下の整数の集合である．

4.7 関連研究

既に 4.2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[43] 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) $\not\longrightarrow$」の扱いが重要である．CBS では，**ディスカード** と呼ばれる特殊なアクションによる (正の) 状態遷移が負の状態遷移の代わりに使われている．このディスカードよる状態遷移を使う利点は同期代数 (synchronization algebra)[55] の枠組を適用できることにある．

CCB では負の状態遷移の代わりに監視関数 mon を採用している．監視関数が見積もる集合は，VIABUS において各プロセスが宣言する受信可能な名前の集合に対応し

ている．ただし，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の扱いにある．CCB ではアクションをブロードキャストしたプロセスはそのアクション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．

プロセス代数によるブロードキャストの記述に関する他の研究として [19] があげられる．この研究では，CBS のプロセスを SCCS のプロセス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．ブロードキャストでは負の状態遷移が重要であるが，この SCCS への変換では，プロセス P から現在受信しないアクション名の集合を取り出すために関数 D が定義されている．この関数 D は CCB の監視関数 mon と相補的な関数である $(D(P) = \overline{mon(P)})$．

4.8 おわりに

プロセスを柔軟に結合するために，マルチエージェントモデルをもとにした VIABUS が開発されている．さらに，多様な言語で記述されたプロセスを制御するための共通の通信言語として π 計算 [39] が VIABUS に実装され，その有効性が認められていた [50]．しかし，π 計算は VIABUS の可算ブロードキャストの記述には適していなかった．

本章では，従来のプロセス代数による可算ブロードキャストの記述の問題点を述べ，VIABUS 上に構築されたシステムの解析に適したプロセス代数として CCB を提案した．CCB は，可算ブロードキャストを一つのアクションで表現できる特徴をもつ．また，CCS の観測等価が CCB の並行合成演算子によって保存されないことを指摘し，観測等価に含まれ，かつ並行合成演算子によって保存される最大の関係として監視等価を定義した．さらに，監視等価に含まれる最大の合同関係として監視合同を定義した．そして，有限プロセスに対する監視合同の健全で完全な公理系 $\mathcal{A}_2$ を与え，その性質を明らかにした．

第 5 章

近似解析用空間プロセス代数

5.1 はじめに

4章では，ブロードキャスト通信をもつ並行プロセスの振舞いを明らかにした．ブロードキャスト通信では受信者を指定しないため，拡張性の高いシステムを構築できる利点がある．一方，メッセージの受信範囲は一般にネットワークの構造によって決まり，その受信範囲をメッセージの内容に応じて変えることはできない．例えば，名前 a をもつメッセージが制限 $\backslash\{a\}$ を越えて外側に送信されることはない．

重要なメッセージは広い範囲に，重要でないメッセージは狭い範囲に送信されるように，送信者がメッセージにその重要度示す値を添付して，その受信範囲を決める通信方式がある．例えば，MBone[57] では，その重要度を TTL(Time To Live) で表し，TTL が 31 以下で組織内，TTL が 127 以下で国内全体，TTL が 128 以上で海外と決められている．また，無線通信では，受信範囲はその送信電力によって決められるため，送信電力の大きさがその重要度に相当する．

本章では，重要度によって受信範囲が決まる通信方式をもつ並行プロセスの解析に適したプロセス代数として CCSG (a Calculus of Communicating Systems with Graded spatial actions) を提案する．CCSG では，各アクションにその重要度を示す値として**グレイド**が付加されており，このグレイドが距離とともに減衰することによって，そのアクションの観測可能範囲を制御できる．CCSG の特徴として，観測レベルを低くして，遠くの重要でないアクションを無視した近似的な解析ができることがあげられる．この近似解析は，非常に多くの状態をもつ大規模システムの解析に有効である．本章

では，そのための近似的な等価性として**レベル** $\langle r \rangle$ **等価**を定義し，その性質を明らかにする．ここで，r は観測レベルを表す実数であり，大きいほど詳細な観測を意味する．

以下，5.2節で CCSG の概要を述べ，5.3節で CCSG のアクション，経路，構文，意味を定義する．5.4節では，CCSG のプロセスの振舞いを解析するために，CCSG に適した等価性を定義し，その公理系を与える．また，CCSG のプロセスの振舞いに対する要求を適切に記述できるプロセス論理を与える．5.5節では CCSG による近似解析の例を示す．5.6節では関連研究について述べる．

5.2 CCSG の紹介

CCSG では，各アクションがその重要度を示す**グレイド**とその発生位置を示す**経路**の情報をもっており，遠くの重要でないアクションは観測できないという仮定をもとにした近似解析が可能である．遠くのアクションの重要度を減少させるために，グレイドの**損失**を伴う**ルータ**を用いる．ルータは図 5.1に示すように星型にプロセスと接続される．ノードがプロセスを表し，枝がルータを表している．ルータは名前 a とグレイドの損失値 r をもち，$a\langle r \rangle$ の形で記述される．グレイドの損失値は 0 以上の実数である．例えば図 5.1では，プロセス P_0 と P_1 の間にルータ $a_1\langle 6 \rangle$ が存在し，名前と損失値は各々a_1 と 6 である．2 点間の経路とは，その間にあるルータの列のことである．例えば図 5.1において，P_2 から P_3 までの経路は $(a_2\langle 4 \rangle a_3\langle 1 \rangle)$ として表される．ルータの接続は図 5.2 に示すように階層的に星型に接続することが可能である．

CCSG は，2章で紹介した CCS[38] をもとにしている．CCSG の構文は CCS にほぼ同じであるが，CCSG には経路を表す演算子 @ が加えられている．例えば図 5.1のシステムは，プロセス P_0 を観測位置として，

$$S_0 \equiv P_0|(P_1@a_1\langle 6 \rangle)|(P_2@a_2\langle 4 \rangle)|(P_3@a_3\langle 1 \rangle)$$

のように記述される．ここで，$\equiv$ は構文的に等しいことを表し，$|$ は並行合成演算子である．CCSG で重要なことの一つに観測位置によって記述が変化することがあげられる．例えば，S_0 は P_2，または P_3 を観測位置として次のように書き換えられる．

$$S_2 \equiv P_2|((P_0|(P_1@a_1\langle 6 \rangle)|(P_3@a_3\langle 1 \rangle))@a_2\langle 4 \rangle)$$
$$S_3 \equiv P_3|((P_0|(P_1@a_1\langle 6 \rangle)|(P_2@a_2\langle 4 \rangle))@a_3\langle 1 \rangle)$$

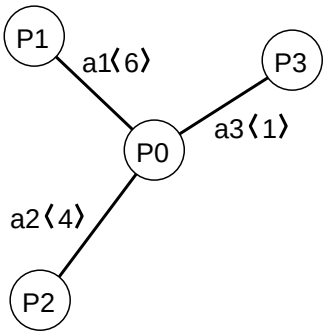

図 5.1: 星型構造の例

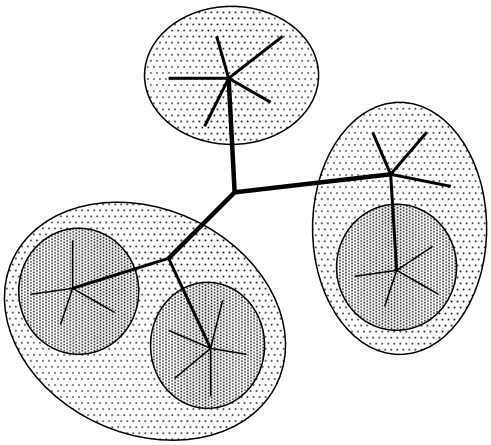

図 5.2: 階層的星型構造

この S_2 は，P_1 と P_3 が各々ルータ $a_1\langle 6\rangle$ と $a_3\langle 1\rangle$ を通して P_0 に接続され，さらに P_0 がルータ $a_2\langle 4\rangle$ を通して P_2 に接続されていることを表している．このように違う位置で観測された結果を関係付けるために，CCSG では**シフト** (s) **等価** $\overset{\sim}{\sim}_{(s)}$ を与える．ここで，s は**左辺の観測位置から右辺の観測位置まで**の経路を表すパラメータである．例えば，P_2 から P_3 への経路は $(a_2\langle 4\rangle a_3\langle 1\rangle)$ であるから，

$$S_2 \overset{\sim}{\sim}_{(a_2\langle 4\rangle a_3\langle 1\rangle)} S_3$$

が成り立つ．特に両辺の観測者の位置が同じである場合，シフト (ε) 等価 $\overset{\sim}{\sim}_{(\varepsilon)}$ は強等価 (定義 2.4.2) $\sim$ に相当する (ε は空列を表す).

CCSG のアクションは，ラベル l，グレイド r，経路 s から構成され，$l\langle r\rangle @ s$ の形で

記述される．ここで使われている @ は前述のプロセス位置を明記する演算子ではなく，アクションの位置を明記するために使われている．共に位置を表し，構文から区別可能であるので同じ記号を用いる．$l\langle r\rangle@s$ は，経路 s で示される位置で生じたラベル l とグレイド r をもつアクションを表している．グレイド r は実数であり，重要なアクションほど大きな値が与えられる．s はアクションが生じた位置からそれが観測 (記述) された位置までの経路である．アクションの生じた位置が観測位置ならば経路 s は空列 ε であり，しばしば $l\langle r\rangle@\varepsilon$ は $l\langle r\rangle$ と略記される．例えば図 5.1 の P_1 の位置で生じたアクション $l\langle 7\rangle$ は，P_2 の位置では $l\langle 7\rangle@(a_1\langle 6\rangle a_2\langle 4\rangle)$ として観測される．

2 点間の経路に含まれる全てのルータの損失値の和を，その 2 点間の**損失距離**と呼ぶ．例えば，図 5.1の P_1 と P_2 の間の損失距離は $(6+4=)10$ である．重要なことは，離れたところで生じたアクションのグレイドは，その間の損失距離だけ減少して観測されることである．この減少して観測される実際に有効となるグレイドを**実効グレイド**と呼ぶ．例えば，$l\langle 7\rangle@(a_1\langle 6\rangle a_2\langle 4\rangle)$ の実効グレイドは $(7-(6+4)=)-3$ である．

CCSG では，2 つのアクションが同期するために,「**2 つのアクションのグレイドの和が，それらの発生位置の間の損失距離以上である**」という条件を満たすことを要求する．例えば図 5.1の P_1 と P_2 の位置で，各々アクション $l\langle 9\rangle$ と $\overline{l}\langle 3\rangle$ が起ったとする．この 2 つのアクションのグレイドの和は $(9+3=)12$ であり，P_1 と P_2 の損失距離は $(6+4=)10$ であるので，この 2 つのアクションは同期可能である．この条件の直観的な意味として無線通信を考えると，送信アクション $\overline{l}\langle 3\rangle$ の 3 は送信電力，受信アクション $l\langle 9\rangle$ の 9 は受信感度とみなすことができる．すなわち，送信電力を大きくするか，または受信感度を高くすることによって，遠く離れていても受信することができる．

5.1節で紹介した Mbone はブロードキャスト通信方式を採用しているが，CCSG の通信方式は 1 対 1 通信である．CCSG では，グレイドによって通信が制御されるプロセスの特性を明らかにすることを目的に，より簡単な通信方式を採用した．Mbone のような通信方式は，前章の CCB と CCSG を統合することによって実現できる．

大規模システムにおいては，全ての動作を理解することは困難である．そこで，まず遠くの重要でないアクションを無視して動作の概要を得ることは，全体を把握するためにも有効である．CCSG では，観測位置で実効グレイドの低いアクションを無視した近似的な解析が可能である．この近似的解析を行なうために，パラメータとして観測のレベル r をもつ**レベル** $\langle r\rangle$ **等価** $=_{\langle r\rangle}$ を与える．レベル $\langle r\rangle$ 等価は,「**観測位置での実効グレイドが** $-r$ **未満であるアクションは観測できない**」という仮定のもとで等

しい関係を表す．また，このような仮定のもとで行なわれる観測を**レベル** $\langle r \rangle$ **観測**と呼ぶ．観測のレベル r が大きいほどグレイドの低いアクションまで観測でき，$=_{\langle\infty\rangle}$ は CCS の観測合同 $=_{ccs}$ (定義 2.4.11) に相当する．例えば次の関係が成り立つ．

$$(l\langle 4\rangle.P)@a\langle 7\rangle \neq_{\langle 3\rangle} (\tau.P)@a\langle 7\rangle$$
$$(l\langle 4\rangle.P)@a\langle 7\rangle =_{\langle 2\rangle} (\tau.P)@a\langle 7\rangle$$

$(l\langle 4\rangle.P)@a\langle 7\rangle$ は，観測位置から経路 $a\langle 7\rangle$ 離れた位置にあるプロセス $(l\langle 4\rangle.P)$ が，アクション $l\langle 4\rangle$ を実行後，プロセス P のように振舞うことを表している．τ は観測できない**内部アクション**である．アクション $l\langle 4\rangle$ は $l\langle 4\rangle@a\langle 7\rangle$ として観測されるため，その実効グレイドは $4-7=-3$ となる．すなわち，-3 以上であるので観測レベル 3 では観測されなければならず，等式は成り立たない．一方，観測レベル 2 では $4-7<-2$ であるので，$l\langle 4\rangle@a\langle 7\rangle$ は内部アクションとみなされ，上の等式が成り立つ．このように，CCSG では遠くのグレイドの低いアクションを内部アクションとみなし，それらを可能な限り無視することによって，システムの振舞いを近似解析することができる．

5.3 CCSG の定義

まず，5.3.1小節で CCSG で用いる各集合を定義する．次に，5.3.2小節でプロセスの位置関係を代数的に扱う方法を示す．そして，5.3.3小節と 5.3.4小節で CCSG の構文と意味を定義する．

5.3.1 準備

まず，**名前**の無限集合 $\mathcal{N}$ が与えられていると仮定し，$\mathcal{N}$ の要素を a で表す．**ルータ**の集合 Ω は，名前の集合 $\mathcal{N}$ と非負の実数の集合 $\mathcal{R}^+$ の直積集合

$$\Omega = \{a\langle r\rangle : a \in \mathcal{N}, r \in \mathcal{R}^+\}$$

として定義され，その要素は ω によって表される．CCSG ではルータの接続方法について「**１つのノードに接続されているルータは全て区別可能である**」ことを仮定する．ここで，二つのルータ $a_1\langle r_1\rangle$,$a_2\langle r_2\rangle$ が区別できないとは，$a_1=a_2$ かつ $r_1=r_2$ の場合である．例えば図 5.3に示されるように，もし P_2 と P_4 が区別できない 2 つのルータ $a_2\langle 4\rangle$ を通して P_0 に接続されているならば，P_2 と P_4 は同じ場所にあると解釈される．

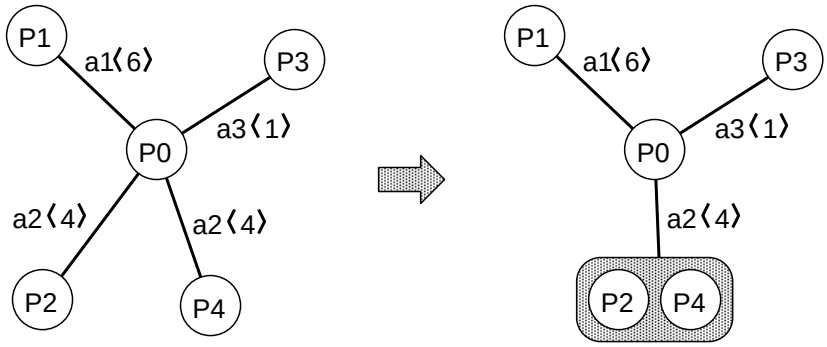

図 5.3: 区別できないルータによる結合の解釈

つまり，P_2 と P_4 の間の経路は $(a_2\langle 4\rangle a_2\langle 4\rangle)$ ではなく空列 ε となる．よって，任意の 2 点間の経路は，区別できないルータを続けることなく表現できる．そこで，**経路**の集合 Ψ を，次のように定義する．

$$\Psi = \{s \in \Omega^* | \mathit{No\text{-}adjacent}(s)\}$$

ここで，Ω^* は有限ルータ列の集合を表し，$\mathit{No\text{-}adjacent}(s)$ は s が 2 つの連続した区別できないルータ列をもたないことを表している．例えば，$(a_2\langle 4\rangle a_2\langle 4\rangle) \notin \Psi$ である．

余名の集合 $\overline{\mathcal{N}}$ を $\{\overline{a} : a \in \mathcal{N}\}$ として与える．ここで，$\bar{\cdot}$ は名前と余名の間の全単射であり，$\overline{\overline{a}} = a$ である．$\mathcal{N}$ と $\overline{\mathcal{N}}$ の和集合 $\mathcal{N} \cup \overline{\mathcal{N}}$ を**ラベル**の集合 $\mathcal{L}$ と呼び，その要素を l で表す．このとき，**アクション**の集合 Act は，ラベルの集合 $\mathcal{L}$ と実数の集合 $\mathcal{R}$ と経路の集合 Ψ の直積集合に内部アクション τ を加えて与えられる．

$$Act = \{l\langle r\rangle @ s : l \in \mathcal{L}, r \in \mathcal{R}, s \in \Psi\} \cup \{\tau\}$$

アクションは α によって表される．表 5.1に集合の要素を表す変数をまとめておく．

5.3.2 経路

図 5.1において，P_1 から P_2 への経路 s_{12} は $(a_1\langle 6\rangle a_2\langle 4\rangle)$ であり，P_2 から P_3 への経路 s_{23} は $(a_2\langle 4\rangle a_3\langle 1\rangle)$ である．このとき，s_{12} と s_{23} の**和**は，P_1 から P_3 への経路 $(a_1\langle 6\rangle a_3\langle 1\rangle)$ であることが期待される．つまり経路の和は，$(a_1\langle 6\rangle a_2\langle 4\rangle a_2\langle 4\rangle a_3\langle 1\rangle)$ のような単なる文字列の結合ではない．

この小節では，経路上に**和**，**差**，**反転**の演算子を定義し，それらに関する命題を与える．2 つの経路 s_1 と s_2 の和とは，s_1 の終点と s_2 の始点を同じ位置にしてできる，s_1

表 5.1: 各集合の要素を表す変数

集合	要素の名前	要素上の変数
$\mathcal{N}$	名前	$a, a', a_1, \cdots$
$\overline{\mathcal{N}}$	余名	$\overline{a}, \overline{a'}, \overline{a_1}, \cdots$
$\mathcal{R}$	実数	$r, r', r_1, \cdots$
$\mathcal{L}$	ラベル	$l, l', l_1, \cdots$
Ω	ルータ	$\omega, \omega', \omega_1, \cdots$
Ψ	経路	$s, s', s_1, \cdots$
Act	アクション	$\alpha, \alpha', \alpha_1, \cdots$

の始点から s_2 の終点までの経路のことであり，$(s_1 \circ s_2)$ と記述される．和の演算子 $\circ$ は次のように，結合点の同じルータを順次消去することによって定義される．

定義 5.3.1 和の演算子 $\circ : \Psi \times \Psi \to \Psi$ を次のように定義する．

$$s_1 \circ s_2 = \begin{cases} s_1' \circ s_2' & (s_1 = s_1'\omega, s_2 = \omega s_2') \\ s_1 s_2 & (\text{ 上記以外 }) \end{cases}$$

∎

図 5.4を用いて経路の計算方法を説明する．図中 ω_i がルータ，s_i が経路を表し，各経路は $s_1{=}\omega_1\omega_2\omega_3\omega_4$, $s_2{=}\omega_4\omega_3\omega_5$, $s_3{=}\omega_1\omega_2\omega_5$ のように与えられている．このとき，s_3 は s_1 と s_2 の和になっており，s_1 と s_2 から次のように求められる．

$$s_1 \circ s_2 = \omega_1\omega_2\omega_3\omega_4 \circ \omega_4\omega_3\omega_5 = \omega_1\omega_2\omega_3 \circ \omega_3\omega_5 = \omega_1\omega_2 \circ \omega_5 = \omega_1\omega_2\omega_5 = s_3$$

次に経路の向きを反転する演算子 rev を定義する．

定義 5.3.2 反転の演算子 $\mathtt{rev} : \Psi \to \Psi$ を次のように定義する．

$$\mathtt{rev}(s) = \begin{cases} \mathtt{rev}(s')\,\omega & (s = \omega s') \\ \varepsilon & (s = \varepsilon) \end{cases}$$

∎

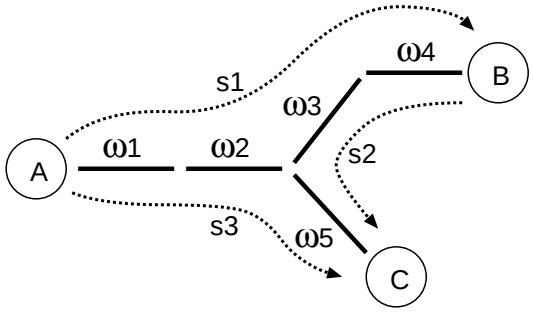

図 5.4: 経路演算の例

最後に差の演算子を定義する．2 つの経路 s_1 と s_2 の差とは，s_1 の終点と s_2 の終点を同じ位置にしてできる，s_1 の始点から s_2 の始点までの経路のことであり，$(s_1 \triangleright s_2)$ と記述される．差の演算子 $\triangleright$ は，和と反転の演算子を用いて定義できる．

定義 5.3.3 差の演算子 $\triangleright : \Psi \times \Psi \to \Psi$ を次のように定義する．

$$s_1 \triangleright s_2 = s_1 \circ \mathtt{rev}(s_2)$$

∎

例えば，図 5.4の s_1 は s_3 と s_2 の差になっており，s_3 と s_2 から次のように求められる．

$$\begin{aligned} s_3 \triangleright s_2 &= s_3 \circ \mathtt{rev}(s_2) = \omega_1\omega_2\omega_5 \circ \mathtt{rev}(\omega_4\omega_3\omega_5) = \omega_1\omega_2\omega_5 \circ \omega_5\omega_3\omega_4 \\ &= \omega_1\omega_2 \circ \omega_3\omega_4 = \omega_1\omega_2\omega_3\omega_4 = s_1 \end{aligned}$$

CCSG では 2 つのアクションの同期の可能性を調べるために，2 点間の損失距離を求めることが重要である．そこで経路の損失距離を求める関数 $loss$ を定義する．

定義 5.3.4 関数 $loss : \Psi \to \mathcal{R}^+$ を次のように定義する．

$$loss(s) \;=\; \begin{cases} r + loss(s') & : (s = a\langle r \rangle\ s') \\ 0 & : (s = \varepsilon) \end{cases}$$

∎

2 つの経路が与えられ，かつそれらの終点が同じであるとき，それらの 2 つの始点の間の損失距離を求めることが，しばしば必要となる．例えば，図 5.4の AB 間の損失距離は s_3 と s_2 を用いて，$loss(s_3 \triangleright s_2)$ によって求められる．

この経路の演算はベクトルの演算に似ており，命題 5.3.1に示すような関係が成り立つ．

命題 5.3.1 任意の $s, s_i \in \Psi$ について，次式が成り立つ．

(1) $s \circ \varepsilon = \varepsilon \circ s = s$

(2) $s \triangleright s = \varepsilon$

(3) $(s_1 \circ s_2) \circ s_3 = s_1 \circ (s_2 \circ s_3)$

(4) $(s_1 \triangleright s_2) \triangleright s_3 = s_1 \triangleright (s_3 \circ s_2)$

(5) $(s_1 \circ s_2) \triangleright s_3 = s_1 \circ (s_2 \triangleright s_3)$

(6) $(s_1 \circ s_2) \triangleright s_3 = s_1 \triangleright (s_3 \triangleright s_2)$

(7) $s_1 \triangleright s_2 = (s_1 \circ s) \triangleright (s_2 \circ s)$

(8) $s_1 \circ s_2 = s_3 \iff s_1 = s_3 \triangleright s_2$

(9) $loss(s_1 \circ s_2) \geq |loss(s_1) - loss(s_2)|$

(10) $loss(s_1 \circ s_2) \leq loss(s_1) + loss(s_2)$

証明 (3) と (9) の証明のみ示す．他は同様に，またはより簡単に証明できる．

(3) s_2 の長さについて場合分けをする．$s_2 = \varepsilon$ の場合は簡単である．s_2 の長さが 2 以上の場合は長さに関する帰納法を用いる．$s_1 = s_1'\omega$ かつ $s_2 = \omega$ かつ $s_3 = \omega s_3'$ の場合が最も重要である．このとき，

$$(s_1 \circ s_2) \circ s_3 = (s_1'\omega \circ \omega) \circ \omega s_3' = (s_1' \circ \varepsilon) \circ \omega s_3' = (s_1'\varepsilon) \circ \omega s_3' = s_1' \circ \omega s_3',$$
$$s_1 \circ (s_2 \circ s_3) = s_1'\omega \circ (\omega \circ \omega s_3') = s_1'\omega \circ (\varepsilon \circ s_3') = s_1'\omega \circ (\varepsilon s_3') = s_1'\omega \circ s_3'.$$

となる．ここで，もし $s_1' = s_1''\omega'$ ならば，$s_1 = s_1''\omega'\omega \in \Psi$ より，$\omega \neq \omega'$ である．同様に，もし $s_3' = \omega'' s_3''$ ならば，$\omega \neq \omega''$ である．よって，

$$s_1' \circ \omega s_3' = s_1'\omega s_3' = s_1'\omega \circ s_3'$$

を得る．これ以外の場合はこの場合より簡単である．

(9) $s_1 = \varepsilon$ か $s_2 = \varepsilon$ の場合は簡単であるので省略する．$s_1 = s_1'\omega$ かつ $s_2 = \omega s_2'$ の場合は次のように s_1 と s_2 の長さの和に関する帰納法を用いる．

$$\begin{aligned} loss(s_1 \circ s_2) &= loss(s_1'\omega \circ \omega s_2') = loss(s_1' \circ s_2') \geq |loss(s_1') - loss(s_2')| \\ &= |loss(s_1') + loss(\omega) - (loss(\omega) + loss(s_2'))| \\ &= |loss(s_1'\omega) - loss(\omega s_2')| = |loss(s_1) - loss(s_2)| \end{aligned}$$

それ以外の場合，つまり $\omega \neq \omega'$ で $s_1 = s_1'\omega$ かつ $s_2 = \omega' s_2'$ の場合は，任意の $s \in \Psi$ について $loss(s) \geq 0$ であるから，次のように直接示される．

$$loss(s_1 \circ s_2) = loss(s_1 s_2) = loss(s_1) + loss(s_2) \geq |loss(s_1) - loss(s_2)|$$

∎

5.3.3 構文

CCS と同様に，プロセスの動作は**プロセス式**によって記述される．まず，プロセス変数の集合 $\mathcal{X}_{ccsg}$ とプロセス定数の集合 $\mathcal{K}_{ccsg}$ が与えられていると仮定する．CCSG のプロセス式 ($E, F, \cdots$ で表す) の集合 $\mathcal{E}_{ccsg}$ は次のように定義される．

定義 5.3.5 プロセス式の集合 $\mathcal{E}_{ccsg}$ は次の式を含む最小の集合である

$$\begin{aligned}
X &: \text{プロセス変数 } (X \in \mathcal{X}_{ccsg}) \\
A &: \text{プロセス定数 } (A \in \mathcal{K}_{ccsg}) \\
\mathbf{0} &: \text{無動作プロセス} \\
\alpha.E &: \text{逐次 } (\alpha \in Act) \\
E + F &: \text{選択} \\
E|F &: \text{並行合成} \\
E[f] &: \text{リラベリング } (f \text{ はリラベリング関数 }) \\
E\backslash^{L}_{\langle r \rangle(s)} &: \text{局所制限 } (r \in \mathcal{R}^+, s \in \Psi, L \subseteq \mathcal{L}) \\
E@s &: \text{経路 } (s \in \Psi)
\end{aligned}$$

ここで，E, F はすでに $\mathcal{E}_{ccsg}$ の要素であるとする． ∎

CCS と同様に，リラベリング関数 f は $f(\bar{l}) = \overline{f(l)}$ を満たすラベルからラベルへの関数であり，ラベル a を b に変更するリラベリング関数を (b/a) と書く．便宜的に $f(l\langle r \rangle @ s) = f(l)\langle r \rangle @ s$ かつ $f(\tau) = \tau$ とし，f をアクションの集合 Act 上に拡張する．この条件にみられるように，ルータの名前を変えることはできない．

プロセス変数を含まないプロセス式を**プロセス**と呼び，プロセスの集合を $\mathcal{P}_{ccsg}$，その要素を $P, Q, R, \cdots$ で表す．プロセス定数は**定義式**によって意味を与えられるプロセスであり，全てのプロセス定数 A について，$A \stackrel{\text{def}}{=} P$ の形の定義式があると仮定する．

以下，各演算子の役割と，プロセス式の位置関係について説明する．

- $\alpha.E$ はアクション α を起こした後は E のように振舞うプロセス式である．プロセス式 $\alpha.E$ とその部分式 E の位置は同じである．$\alpha = l\langle r\rangle@s$ ならば，s はアクションが起こった位置から E の位置までの経路を表している．

- $E + F$ は E か F のように振舞うプロセス式である．この選択は E か F のアクションによって行なわれる．$E + F$, E, F の位置は全て同じである．

- $E|F$ は E と F の並行動作を表す．$E|F$, E, F の位置は全て同じである．

- $E[f]$ は，E の全てのアクションのラベルがリラベリング関数 f によって変更されていることを除いて，E と同じように振舞う．$E[f]$ と E の位置は同じである．

- $E\backslash^{L}_{\langle r\rangle(s)}$ は制限領域の中心における**制限力**が r である局所制限を行なう．s はその制限領域の中心から E の位置までの経路を表している．制限領域の中心から経路 s' 離れた位置で生じたアクションに対する制限力はその損失距離 $loss(s')$ だけ減少しており，その実際に有効になる制限力 $(r - loss(s'))$ を**実効制限力**と呼ぶ．アクションのグレイドの絶対値がその実効制限力以下であり，そのラベルが L に含まれるならば，そのアクションは制限される (5.3.4小節の例参照)．$E\backslash^{L}_{\langle r\rangle(s)}$ と E の位置は同じである．

- $E@s$ は E から $E@s$ までの位置が経路 s だけ離れていることを表す．

例えば $E@s|F@s'$ では，$E@s$ と $F@s'$ と $E@s|F@s'$ が同じ位置にある．つまり，E から $E@s|F@s'$ までの経路は s であり，F から $E@s|F@s'$ までの経路は s' である．さらに，E から F までの経路は $(s \triangleright s')$ によって求められる．

演算子の結合の優先順位は { 局所制限 , リラベリング , 経路 } > 逐次 > 並行合成 > 選択，である．

5.3.4 意味

CCS と同様に，CCSG の意味をラベル付遷移システムによって定義する．

定義 5.3.6 CCSG の意味は次のラベル付遷移システムによって与えられる．

$$(\mathcal{E}_{ccsg}, Act, \longrightarrow)$$

ここで，遷移の集合 $\longrightarrow$ は図 5.5の推論規則を満たす最小の関係である． ∎

名前	仮定	⊢	結果
Act		$\vdash \alpha.E \xrightarrow{\alpha} E$	
Choice$_1$		$E \xrightarrow{\alpha} E' \vdash E + F \xrightarrow{\alpha} E'$	
Choice$_2$		$F \xrightarrow{\alpha} F' \vdash E + F \xrightarrow{\alpha} F'$	
Com$_1$		$E \xrightarrow{\alpha} E' \vdash E\|F \xrightarrow{\alpha} E'\|F$	
Com$_2$		$F \xrightarrow{\alpha} F' \vdash E\|F \xrightarrow{\alpha} E\|F'$	
Com$_3$	$E \xrightarrow{l\langle r_1\rangle@s_1} E',\ F \xrightarrow{\overline{l}\langle r_2\rangle@s_2} F',\ r_1 + r_2 \geq loss(s_1 \triangleright s_2)$	$\vdash E\|F \xrightarrow{\tau} E'\|F'$	
Rel		$E \xrightarrow{\alpha} E' \vdash E[f] \xrightarrow{f(\alpha)} E'[f]$	
Res$_1$	$E \xrightarrow{l\langle r_1\rangle@s_1} E', (l \notin L \cup \overline{L}$ or $\|r_1\| > r_2 - loss(s_1 \triangleright s_2))$	$\vdash E\backslash^L_{\langle r_2\rangle(s_2)} \xrightarrow{l\langle r_1\rangle@s_1} E'\backslash^L_{\langle r_2\rangle(s_2)}$	
Res$_2$		$E \xrightarrow{\tau} E' \vdash E\backslash^L_{\langle r_2\rangle(s_2)} \xrightarrow{\tau} E'\backslash^L_{\langle r_2\rangle(s_2)}$	
Rec	$P \xrightarrow{\alpha} P',\ A \stackrel{\text{def}}{=} P$	$\vdash A \xrightarrow{\alpha} P'$	
Rou$_1$		$E \xrightarrow{l\langle r\rangle@s_1} E' \vdash E@s_2 \xrightarrow{l\langle r\rangle@(s_1 \circ s_2)} E'@s_2$	
Rou$_2$		$E \xrightarrow{\tau} E' \vdash E@s_2 \xrightarrow{\tau} E'@s_2$	

図 5.5: CCSG のプロセス式上のラベル付状態遷移の推論規則

CCS との違いは，**Rou**$_1$, **Com**$_3$, **Res**$_1$ にある．以下，これらの規則について説明する．

Rou$_1$ の E のアクション $l\langle r\rangle@s_1$ は，アクションが生じた位置から E までの経路が s_1 であることを表している．$E@s_2$ は E から経路 s_2 だけ離れているので，このアクションが生じた位置から $E@s_2$ までの経路は $(s_1 \circ s_2)$ となる．

CCS と同様に，**Com**$_3$ は相補的なラベル $(l, \overline{l})$ をもつ 2 つのアクションの同期を表している．ただし，同期条件「グレイドの和が損失距離以上」を考慮して条件 $r_1 + r_2 \geq loss(s_1 \triangleright s_2)$ が付加されている．5.3.2小節で述べたように $loss(s_1 \triangleright s_2)$ は 2 つのアクションの損失距離である．

Res$_1$ の条件 $|r_1| > r_2 - loss(s_1 \triangleright s_2)$ は，グレイド r_1 の絶対値が実効制限力 $(r_2 - loss(s_1 \triangleright s_2))$ より大きいときは，ラベルが L に含まれていても制限できないことを表している．r_2 は s_2 で示される制限領域の中心での制限力であり，その中心からアクションの生じた位置までの損失距離は $loss(s_1 \triangleright s_2)$ である．

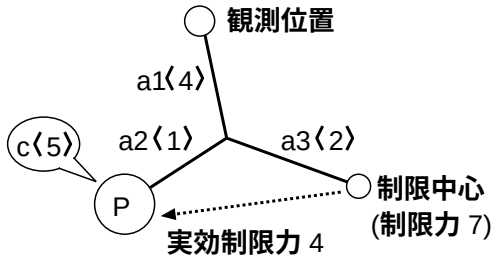

図 5.6: 局所制限の例

局所制限について，次の例を用いて説明する．

$$SYS \equiv (P@s_1)\backslash^{\{c\}}_{\langle 7\rangle(s_2)}, \quad P \equiv (c\langle 5\rangle@\varepsilon).\mathbf{0}$$

ここで，$s_1 = a_2\langle 1\rangle a_1\langle 4\rangle, s_2 = a_3\langle 2\rangle a_1\langle 4\rangle$ である．図 5.6に SYS の位置関係を示す．P はアクション $c\langle 5\rangle$ を起こすと停止するプロセスである．P から SYS までは経路 s_1 離れており，SYS は c の局所制限を行なっている．制限領域の中心から SYS までは経路 s_2 離れており，中心における制限力は 7 である．このとき，P から制限領域の中心までの損失距離は次のように求められる．

$$\begin{aligned} loss(s_1 \triangleright s_2) &= loss(s_1 \circ \mathtt{rev}(s_2)) = loss((a_2\langle 1\rangle a_1\langle 4\rangle) \circ (a_1\langle 4\rangle a_3\langle 2\rangle)) \\ &= loss(a_2\langle 1\rangle a_3\langle 2\rangle) = 3 \end{aligned}$$

よって，P に対する実効制限力は $(7-3=)4$ となり，P の $c\langle 5\rangle$ は制限されない．

5.4 CCSG の等価性

本節では，並行システムの振舞いを局所的に近似解析するために有効な等価性を与える．まず，5.4.1小節では，観測位置の違いを補正する**シフト** (s) **等価**を定義する．次に，5.4.2小節で，CCS の観測等価 (定義 2.4.10) に観測レベルの概念を導入して**レベル** $\langle r\rangle$ **弱等価**を定義する．レベル $\langle r\rangle$ 弱等価は，グレイドの低い遠くのアクションを内部アクションと同じように (可能な限り) 無視し，観測的に区別できない振舞いを等しいとみなす同値関係である．ただし，観測等価と同様にレベル $\langle r\rangle$ 弱等価も選択演算子 $+$ によって保存されない問題がある．そこで，5.4.3小節で，レベル $\langle r\rangle$ 弱等価に含まれ，かつ選択演算子 $+$ によって保存される最大の関係として**レベル** $\langle r\rangle$ **等価**を定義す

る．さらに，5.4.4小節では，有限逐次プロセスに対して健全で完全なレベル $\langle r \rangle$ 等価の公理系 $\mathcal{A}\langle r \rangle$ を与える．また，5.4.5小節では，レベル $\langle r \rangle$ 観測を考慮したプロセス論理の充足関係を定義し，その充足関係とレベル $\langle r \rangle$ 弱等価の関係を明らかにする．

5.4.1 シフト (s) 等価

観測位置によってシステムが異って観測されるため，それらの観測位置の違いを補正するシフト (s) 等価を，次のシフト (s) 双模倣をもとに定義する．

定義 5.4.1 $s \in \Psi$ とする．プロセス上の関係 $\mathcal{S}$ が**シフト** (s) **双模倣**であるとは，$(P,Q) \in \mathcal{S}$ ならば，任意の $l\langle r \rangle @ s' \in Act$ について，次の 4 つの条件が成り立つことである．

(i) $P \xrightarrow{l\langle r \rangle @ s'} P'$ ならば，ある Q' について $Q \xrightarrow{l\langle r \rangle @ (s' \circ s)} Q'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．
(ii) $P \xrightarrow{\tau} P'$ ならば，ある Q' について $Q \xrightarrow{\tau} Q'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．
(iii) $Q \xrightarrow{l\langle r \rangle @ s'} Q'$ ならば，ある P' について $P \xrightarrow{l\langle r \rangle @ (s' \triangleright s)} P'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．
(iv) $Q \xrightarrow{\tau} Q'$ ならば，ある P' について $P \xrightarrow{\tau} P'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．

∎

定義 5.4.2 もし，あるシフト (s) 双模倣 $\mathcal{S}$ において $(P,Q) \in \mathcal{S}$ ならば，プロセス P と Q は**シフト** (s) **等価**であるといい，$P \precsim_{(s)} Q$ と書く． ∎

定義5.4.1の (i) と (iii) の $(s' \circ s)$ と $(s' \triangleright s)$ が，右辺と左辺の観測者の位置の違い s を補正している．$\precsim_{(s)}$ は同値関係ではないが，次のようにパラメータ付きの反射律，対称律，推移律が成り立つ．

命題 5.4.1　次の関係が成り立つ．

(1) $P \precsim_{(\varepsilon)} P$
(2) $P \precsim_{(s)} Q$ ならば $Q \precsim_{(\mathtt{rev}(s))} P$
(3) $P \precsim_{(s_1)} Q$ かつ $Q \precsim_{(s_2)} R$ ならば $P \precsim_{(s_1 \circ s_2)} R$

証明 (1) については，$s \circ \varepsilon = s \triangleright \varepsilon = s$ より明らか．(2) については，$s' \circ \mathtt{rev}(s) = s' \triangleright s$ と $s' \triangleright \mathtt{rev}(s) = s' \circ s$ より明らか．(3) について，次の集合 $\mathcal{S}$ がシフト $(s_1 \circ s_2)$ 双模倣になることを示す．

$$\mathcal{S} = \{(P,R) : \exists Q.\ P \precsim_{(s_1)} Q, Q \precsim_{(s_2)} R\}$$

$(P,R)\in\mathcal{S}$ とする．すなわち，ある Q について，$P \precsim_{(s_1)} Q \precsim_{(s_2)} R$ である．ここでは，最も興味深い，定義 5.4.1 の (iii) を満たすことのみ示す．すなわち，$R \xrightarrow{l\langle r\rangle@s} R'$ とする．$Q \precsim_{(s_2)} R$ より，ある Q' について，$Q \xrightarrow{l\langle r\rangle@(s\triangleright s_2)} Q'$ かつ $Q' \precsim_{(s_2)} R'$ を得る．さらに，$P \precsim_{(s_1)} Q$ より，ある P' について，$P \xrightarrow{l\langle r\rangle@((s\triangleright s_2)\triangleright s_1)} P'$ かつ $P' \precsim_{(s_1)} Q'$ を得る．ここで，命題 5.3.1(4) より $(s\triangleright s_2)\triangleright s_1 = s\triangleright(s_1\circ s_2)$ である．よって，$P \xrightarrow{l\langle r\rangle@(s\triangleright(s_1\circ s_2))} P'$ を得る．また，$P' \precsim_{(s_1)} Q' \precsim_{(s_2)} R'$ より，$(P',R')\in\mathcal{S}$ である． ∎

この命題が示すように，全ての $s\in\Psi$ について $\precsim_{(s)}$ の和集合をとってつくられる関係 $(\bigcup_{s\in\Psi}\precsim_{(s)})$ は同値関係になる．また，$s\circ\varepsilon = s\triangleright\varepsilon = s$ であるので，$\precsim_{(\varepsilon)}$ は CCS の強等価 $\sim$ (定義 2.4.2) に同じである．強等価の経路演算子に関する等式を次に示す．

命題 5.4.2 次の等式が成り立つ．

(1) $((l\langle r\rangle@s).P)@s' \sim (l\langle r\rangle@(s\circ s')).(P@s')$

(2) $(P_1|P_2)@s \sim (P_1@s)|(P_2@s)$

(3) $(P@s_1)@s_2 \sim P@(s_1\circ s_2)$

証明 (2) の証明のみ示す．他の場合も同様である．(2) のために次の集合 $\mathcal{S}$ がシフト (ε) 双模倣であることを示す．

$$\mathcal{S} = \{((P_1|P_2)@s, (P_1@s)|(P_2@s)) : s\in\Psi, P_i\in\mathcal{P}_{ccsg}\}$$

ここでは，重要な $\mathbf{Com}_3$ による同期の場合のみ示す．(ii) $(P_1|P_2)@s \xrightarrow{\tau} P'$ とする．このとき $\mathbf{Rou}_2$ より，ある P'_{12} について，$P_1|P_2 \xrightarrow{\tau} P'_{12}$ かつ $P'\equiv P'_{12}@s$ が得られる．さらに，$\mathbf{Com}_3$ より，ある $l, r_1, r_2, s_1, s_2, P'_1, P'_2$ について，$P_1 \xrightarrow{l\langle r_1\rangle@s_1} P'_1$ かつ $P_2 \xrightarrow{\bar{l}\langle r_2\rangle@s_2} P'_2$ かつ $P'_{12}\equiv P'_1|P'_2$ かつ $P'\equiv(P'_1|P'_2)@s$ かつ $r_1+r_2\geq loss(s_1\triangleright s_2)$ が得られる．P_1 と P_2 について，$\mathbf{Rou}_1$ より $P_1@s \xrightarrow{l\langle r_1\rangle@(s_1\circ s)} P'_1@s$ と $P_2@s \xrightarrow{\bar{l}\langle r_2\rangle@(s_2\circ s)} P'_2@s$ を導ける．また，命題 5.3.1(7) より，$(s_1\circ s)\triangleright(s_2\circ s) = s_1\triangleright s_2$ である．よって，$r_1+r_2\geq loss(s_1\triangleright s_2) = loss((s_1\circ s)\triangleright(s_2\circ s))$．ゆえに，$\mathbf{Com}_3$ より $(P_1@s)|(P_2@s) \xrightarrow{\tau} (P'_1@s)|(P'_2@s)$ を導ける．ここで，$((P'_1|P'_2)@s, (P'_1@s)|(P'_2@s))\in\mathcal{S}$ である．

(iv) 同様にして，$(P_1@s)|(P_2@s) \xrightarrow{\tau} Q'$ ならば，$(P_1|P_2)@s \xrightarrow{\tau} P'$ かつ $(P',Q')\in\mathcal{S}$ のような P' が存在する． ∎

(1) 式と (2) 式は，各々逐次と並行合成への経路の分配則であり，(3) は経路の結合則である．

次の命題は経路演算子 @ を使って観測位置を経路 s' だけ移動したとき，シフト (s) 等価がどのように保存されるかを表している．

命題 5.4.3

(1) $P \precsim_{(s)} Q$ ならば，$P \precsim_{(s\circ s')} Q@s'$

(2) $P \precsim_{(s)} Q$ ならば，$P@\mathtt{rev}(s') \precsim_{(s'\circ s)} Q$

証明 (1) について，次の集合 $\mathcal{S}$ がシフト $(s \circ s')$ 双模倣であることを示す．

$$\mathcal{S} = \{(P, Q@s') : P \precsim_{(s)} Q\}$$

$(P, Q_1@s') \in \mathcal{S}$ とする．すなわち，$P \precsim_{(s)} Q_1$ である．ここでは，定義 5.4.1 の (iii) のみ示す．$Q_1@s' \xrightarrow{a\langle r\rangle@s''} Q'$ とする．$\mathbf{Rou}_1$ より，ある s_1 と Q_1' について，$Q_1 \xrightarrow{a\langle r\rangle@s_1} Q_1'$ かつ $Q' \equiv Q_1'@s'$ かつ $s'' = s_1 \circ s'$ を得る．$P \precsim_{(s)} Q_1$ より，ある P' について，$P \xrightarrow{a\langle r\rangle@(s_1 \triangleright s)} P'$ かつ $P' \precsim_{(s)} Q_1'$ を得る．ここで，命題 5.3.1(8) より，$s_1 = s'' \triangleright s'$ である．さらに，命題 5.3.1(4) より，$s_1 \triangleright s = (s'' \triangleright s') \triangleright s = s'' \triangleright (s \circ s')$ である．それゆえ，$P \xrightarrow{a\langle r\rangle@(s'' \triangleright (s\circ s'))} P'$ である．また，$P' \precsim_{(s)} Q_1'$ より，$(P', Q_1'@s') \in \mathcal{S}$ である．

(2) についても同様である． ∎

次の命題は，$\precsim_{(s)}$ が並行合成演算子 | によって保存されることを示している．

命題 5.4.4 $P_1 \precsim_{(s)} Q_1$ かつ $P_2 \precsim_{(s)} Q_2$ ならば，$P_1|P_2 \precsim_{(s)} Q_1|Q_2$ である．

証明 次の集合 $\mathcal{S}$ がシフト (s) 双模倣であることを示す．

$$\mathcal{S} = \{(P_1|P_2, Q_1|Q_2) : P_1 \precsim_{(s)} Q_1, P_2 \precsim_{(s)} Q_2\}$$

$(P_1|P_2, Q_1|Q_2) \in \mathcal{S}$ とする．ここでは，最も重要な定義 5.4.1 の (ii) の $\mathbf{Com}_3$ による場合のみ示す．$(P_1|P_2) \xrightarrow{\tau} P'$ を仮定する．$\mathbf{Com}_3$ より，ある l, r_1, r_2, s_1, s_2, P_1', P_2' について，$P_1 \xrightarrow{l\langle r_1\rangle@s_1} P_1'$ かつ $P_2 \xrightarrow{\overline{l}\langle r_2\rangle@s_2} P_2'$ かつ $P' \equiv P_1'|P_2'$ かつ $r_1 + r_2 \geq loss(s_1 \triangleright s_2)$ を得る．ここで，各 $i \in \{1,2\}$ について $P_i \precsim_{(s)} Q_i$ より，$Q_1 \xrightarrow{l\langle r_1\rangle@(s_1\circ s)} Q_1'$ かつ $Q_2 \xrightarrow{\overline{l}\langle r_2\rangle@(s_2\circ s)} Q_2'$ かつ $P_i' \precsim_{(s)} Q_i'$ を得る．また，命題 5.3.1(7) より，$r_1 + r_2 \geq loss(s_1 \triangleright s_2) = loss((s_1 \circ s) \triangleright (s_2 \circ s))$ である．ゆえに，$\mathbf{Com}_3$ より，$(Q_1|Q_2) \xrightarrow{\tau} (Q_1'|Q_2')$ を得る．また，$P_1' \precsim_{(s)} Q_1'$ かつ $P_2' \precsim_{(s)} Q_2'$ より，$(P_1'|P_2', Q_1'|Q_2') \in \mathcal{S}$ である． ∎

5.4.2 レベル $\langle r \rangle$ 弱等価

この小節では，観測等価 $\approx$ (定義 2.4.10) に,「観測位置での実効グレイドが $-r$ 未満であるアクションは観測できない」というレベル $\langle r \rangle$ 観測の概念を導入して，レベル $\langle r \rangle$ 弱等価を定義する．まず，必要な記法を準備する．

Act^* を空列 ε を含む有限アクション列の集合とし，その要素を $t, t', t_1, \cdots$ で表す．$t = \alpha_1 \cdots \alpha_n \in Act^*$ について，もし $E \xrightarrow{\alpha_1} \cdots \xrightarrow{\alpha_n} E'$ ならば，$E \xrightarrow{t} E'$ と書く．特に，$E \xrightarrow{\varepsilon} E'$ は $E' \equiv E$ を表す．

次にレベル $\langle r \rangle$ 観測を考慮して，アクションの実効グレイドが $-r$ より低いアクションを無視するための閾値関数を定義する．

定義 5.4.3 関数 $\phi : Act^* \times \mathcal{R} \to Act^*$ を次のように定義する．

$$\phi(t,r) = \begin{cases} (l\langle r' \rangle @ s)\phi(t', r) & (t = (l\langle r' \rangle @ s)t', r' - loss(s) \geq -r) \\ \phi(t', r) & (t = (l\langle r' \rangle @ s)t', r' - loss(s) < -r) \text{ or } (t = \tau t') \\ \varepsilon & (t = \varepsilon) \end{cases}$$

∎

この近似解析で注意しなければならないことは，**「グレイドの高過ぎるアクションの観測は不確定になる」**ことである．直観的には，無視されるほど低いグレイドをもつアクションと同期する可能性があるため，その実行が観測できない場合がある．これは高感度な受信機がノイズを拾ってしまう現象に似ている．具体的には，観測位置から経路 s 離れた位置では，観測レベルは $r - loss(s)$ まで減少しているため，このレベルを越えるアクションは不確定要素をもつ．このことを考慮し，次の関数も定義する．

定義 5.4.4 関数 $\theta : Act^* \times \mathcal{R} \to Act^*$ を次のように定義する．

$$\theta(t,r) = \begin{cases} (l\langle r' \rangle @ s)\theta(t', r) & (t = (l\langle r' \rangle @ s)t', |r'| \leq r - loss(s)) \\ \theta(t', r) & (t = (l\langle r' \rangle @ s)t', |r'| > r - loss(s)) \text{ or } (t = \tau t') \\ \varepsilon & (t = \varepsilon) \end{cases}$$

∎

つまり，レベル $\langle r \rangle$ 観測ではアクションを，常に観測できない**観測不能アクション**，常に観測できる**観測可能アクション**，時々観測できない**観測不確定アクション**の 3 つに分類できる．例えば，レベル $\langle 1 \rangle$ 観測では，$(c\langle -2 \rangle @ \varepsilon)$ は観測不能アクション，$(\overline{c}\langle 0 \rangle @ \varepsilon)$

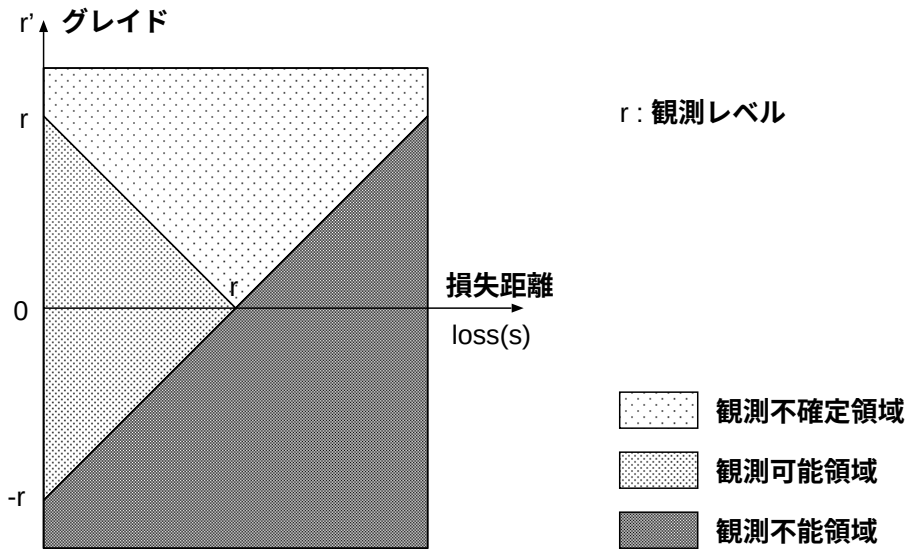

図 5.7: アクション $l\langle r'\rangle@s$ の観測可能性

は観測可能アクション，$(\overline{c}\langle 2\rangle@\varepsilon)$ は観測不確定アクションである．アクション $(\overline{c}\langle 2\rangle@\varepsilon)$ が不確定になるのは，観測不能な $(c\langle -2\rangle@\varepsilon)$ と同期できるためである．一方，$(\overline{c}\langle 0\rangle@\varepsilon)$ は $(c\langle -2\rangle@\varepsilon)$ とは同期できず，不確定にはならない．レベル $\langle r\rangle$ 観測におけるアクションの分類と関数 ϕ, θ の関係は次のとおりである．

$$\begin{aligned}
\alpha \text{ は観測可能} &\iff \phi(\alpha, r) = \alpha,\ \theta(\alpha, r) = \alpha \\
\alpha \text{ は観測不能} &\iff \phi(\alpha, r) = \varepsilon,\ \theta(\alpha, r) = \varepsilon \\
\alpha \text{ は観測不確定} &\iff \phi(\alpha, r) = \alpha,\ \theta(\alpha, r) = \varepsilon
\end{aligned}$$

また，図 5.7に，アクション $l\langle r'\rangle@s$ が r' と s によってどのアクションに分類されるかを示す．近い (損失距離 $loss(s)$ が小さい) アクションはグレイドが低くても観測できるが，遠くなるに従って観測できるアクションは減少する．

次に，各 $r \in \mathcal{R}$ について，状態遷移システム

$$(\mathcal{E}_{ccsg}, Act^*, \Longrightarrow_{\langle r\rangle})$$

を定義する．この $\Longrightarrow_{\langle r\rangle}$ は観測不確定アクションと観測不能アクションを暗に含む状態遷移関係であり，次のように与えられる．

定義 5.4.5 $r \in \mathcal{R}$ とする．$t \in Act^*$ について，$\theta(t, r) = \varepsilon$ かつ $E \xrightarrow{t} E'$ ならば $E \overset{\varepsilon}{\Longrightarrow}_{\langle r\rangle} E'$ である．この $\overset{\varepsilon}{\Longrightarrow}_{\langle r\rangle}$ は $\Longrightarrow_{\langle r\rangle}$ と略記される．さらに $t = \alpha_1 \cdots \alpha_n$ について，$E \Longrightarrow_{\langle r\rangle} \xrightarrow{\alpha_1} \Longrightarrow_{\langle r\rangle} \cdots \Longrightarrow_{\langle r\rangle} \xrightarrow{\alpha_n} \Longrightarrow_{\langle r\rangle} E'$ ならば，$E \overset{t}{\Longrightarrow}_{\langle r\rangle} E'$ である． ∎

この状態遷移関係を用いて，レベル $\langle r\rangle$ 弱等価を次のレベル $\langle r\rangle$ 双模倣をもとに定義する．

定義 5.4.6 $r \in \mathcal{R}$ とする．プロセス上の関係 $\mathcal{S}$ が**レベル** $\langle r\rangle$ **双模倣**であるとは，$(P,Q) \in \mathcal{S}$ ならば，任意の $\alpha \in Act$ について，次の 2 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' について，$Q \overset{\phi(\alpha,r)}{\Longrightarrow}_{\langle r\rangle} Q'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．
(ii) $Q \xrightarrow{\alpha} Q'$ ならば，ある P' について，$P \overset{\phi(\alpha,r)}{\Longrightarrow}_{\langle r\rangle} P'$ かつ $(P',Q') \in \mathcal{S}$ を満たす．

∎

定義 5.4.7 もし，あるレベル $\langle r\rangle$ 双模倣 $\mathcal{S}$ において $(P,Q) \in \mathcal{S}$ ならば，プロセス P と Q は**レベル** $\langle r\rangle$ **弱等価**であるといい，$P \approx_{\langle r\rangle} Q$ と書く． ∎

定義 5.4.6の $\Longrightarrow_{\langle r\rangle}$ の上には θ ではなく ϕ が使われている．これは，グレイドの高いアクションは不確定ではあっても無視されてはならないためである．次の命題 5.4.5に示すように $\approx_{\langle r\rangle}$ は同値関係である．

命題 5.4.5 次の関係が成り立つ．

(1) $P \approx_{\langle r\rangle} P$
(2) $P \approx_{\langle r\rangle} Q$ ならば，$Q \approx_{\langle r\rangle} P$ である．
(3) $P \approx_{\langle r\rangle} Q$ かつ $Q \approx_{\langle r\rangle} R$ ならば，$P \approx_{\langle r\rangle} R$ である．

証明 反射律 (1) と対称律 (2) は容易に示される．推移律 (3) については，まず次の補題 $(*)$ が成り立つことを証明する：

$$P \approx_{\langle r\rangle} Q \text{ かつ } P \Longrightarrow_{\langle r\rangle} P' \text{ ならば，}$$
$$\text{ある } Q' \text{ について，} Q \Longrightarrow_{\langle r\rangle} Q' \text{ かつ } P' \approx_{\langle r\rangle} Q' \text{ である．}$$

$P \approx_{\langle r\rangle} Q$ かつ $P \Longrightarrow_{\langle r\rangle} P'$ とする．すなわち，ある t について，$\theta(t,r) = \varepsilon$ かつ $P \xrightarrow{t} P'$ である．t の長さに関する帰納法を用いる．$t = \varepsilon$ の場合は明らかである．$t = \alpha t_1$ とする．すなわち，ある P_1 について，$\theta(\alpha,r) = \varepsilon$，$\theta(t_1,r) = \varepsilon$，$P \xrightarrow{\alpha} P_1 \xrightarrow{t_1} P'$ である．仮定 $P \approx_{\langle r\rangle} Q$ より，ある Q_1 について，$Q \overset{\phi(\alpha,r)}{\Longrightarrow}_{\langle r\rangle} Q_1$ かつ $P_1 \approx_{\langle r\rangle} Q_1$ を得る．さらに，帰納法の仮定より，ある Q' について，$Q_1 \Longrightarrow_{\langle r\rangle} Q'$ かつ $P' \approx_{\langle r\rangle} Q'$ である．また，θ と ϕ の定義より，$\theta(\alpha,r) = \varepsilon$ ならば $\theta(\phi(\alpha,r),r) = \varepsilon$ であるので，$Q \Longrightarrow_{\langle r\rangle} Q_1$ である．よって，$Q \Longrightarrow_{\langle r\rangle} Q_1 \Longrightarrow_{\langle r\rangle} Q'$ を得る．ゆえに，補題 $(*)$ は証明された．

この補題 $(*)$ を用いて，次の $\mathcal{S}$ がレベル $\langle r\rangle$ 双模倣であることを証明する．

$$\mathcal{S} = \{(P,R) : \exists Q.\ P \approx_{\langle r\rangle} Q \approx_{\langle r\rangle} R\}$$

$(P,R) \in \mathcal{S}$ かつ $P \xrightarrow{\alpha} P'$ とする．$P \approx_{\langle r\rangle} Q$ より，ある Q' について，$Q \overset{\phi(\alpha,r)}{\Longrightarrow}_{\langle r\rangle} Q'$ かつ $P' \approx_{\langle r\rangle} Q'$ を得る．さらに，$Q \approx_{\langle r\rangle} R$ より，補題 $(*)$ を用いて，$R \overset{\phi(\phi(\alpha,r),r)}{\Longrightarrow}_{\langle r\rangle} R'$ かつ $Q' \approx_{\langle r\rangle} R'$ を得られる．ここで，ϕ の定義より $\phi(\alpha,r) = \phi(\phi(\alpha,r),r)$ である．また，$P' \approx_{\langle r\rangle} Q' \approx_{\langle r\rangle} R'$ より，$(P',R') \in \mathcal{S}$ である． ∎

また，次の命題 5.4.6に示すように，レベル $\langle r\rangle$ 弱等価 $\approx_{\langle r\rangle}$ は，観測レベルを上げることによって，より詳細にプロセスの振舞いを区別するようになる．十分にレベル r を高くとれば，観測できないアクションは内部アクション τ のみとなる．特に，レベル $\langle\infty\rangle$ 弱等価は CCS の観測等価 $\approx$ に同じである．

命題 5.4.6　$r \geq r'$ ならば $\approx_{\langle r\rangle} \subseteq \approx_{\langle r'\rangle}$ である．

証明　$r' \leq r$ かつ $P \overset{\phi(\alpha,r)}{\Longrightarrow}_{\langle r\rangle} P'$ ならば，$P \overset{\phi(\alpha,r')}{\Longrightarrow}_{\langle r'\rangle} P'$ であることを示せば十分である．$r' \leq r$ かつ $P \Longrightarrow_{\langle r\rangle} P_1 \xrightarrow{\phi(\alpha,r)} P_2 \Longrightarrow_{\langle r\rangle} P'$ とする．θ の定義より，$\theta(t,r) = \varepsilon$ ならば $\theta(t,r') = \varepsilon$ であるので，$P \Longrightarrow_{\langle r'\rangle} P_1$ と $P_2 \Longrightarrow_{\langle r'\rangle} P'$ を得る．α について次の 3 つの場合分けを行なう．

- $\alpha = l\langle r_1\rangle@s$ かつ $-r' < r_1 - loss(s)$ の場合：$r' \leq r$ より，$\phi(\alpha,r) = \alpha = \phi(\alpha,r')$ である．よって，$P_1 \xrightarrow{\phi(\alpha,r')} P_2$ を得る．

- $\alpha = l\langle r_1\rangle@s$ かつ $-r < r_1 - loss(s) \leq -r'$ の場合：このとき，$\phi(\alpha,r) = \alpha$ かつ $\phi(\alpha,r') = \varepsilon$ である．よって，$P_1 \xrightarrow{\alpha} P_2$ である．また，$\phi(\alpha,r') = \varepsilon$ より，$\theta(\alpha,r') = \varepsilon$ である．すなわち，$P_1 \Longrightarrow_{\langle r'\rangle} P_2$ を得る．ここで，$\phi(\alpha,r') = \varepsilon$ より，$P_1 \overset{\phi(\alpha,r')}{\Longrightarrow}_{\langle r'\rangle} P_2$ である．

- 上記以外の場合：$\phi(\alpha,r) = \varepsilon = \phi(\alpha,r')$ より，$P_1 \xrightarrow{\phi(\alpha,r')} P_2$ を得る．

以上をまとめて $P \overset{\phi(\alpha,r')}{\Longrightarrow}_{\langle r'\rangle} P'$ を得る． ∎

次の命題 5.4.7が示すように，$\approx_{\langle r\rangle}$ は並行合成演算子によって保存され，制限演算子によっても条件付きで保存される．(3) の条件 $(r' + loss(s') \leq r)$ は，直観的には制限領域 (中心までの損失距離 $loss(s')$，半径 r' の内側) が観測不能領域 (半径 r の外側) に入らないことを意味している．

命題 5.4.7 $P_1 \approx_{\langle r \rangle} P_2$ ならば，次の関係が成り立つ．

(1) $\alpha.P_1 \approx_{\langle r \rangle} \alpha.P_2$

(2) $P_1|Q \approx_{\langle r \rangle} P_2|Q$

(3) $P_1\backslash^L_{\langle r' \rangle(s')} \approx_{\langle r \rangle} P_2\backslash^L_{\langle r' \rangle(s')}$ if $r' + loss(s') \leq r$

(4) $P_1@s' \approx_{\langle r' \rangle} P_2@s'$ if $r' + loss(s') \leq r$

証明

(1) $\alpha.P_1 \xrightarrow{\alpha} P_1$ かつ $\alpha.P_2 \xrightarrow{\alpha} P_2$ かつ $P_1 \approx_{\langle r \rangle} P_2$ より明らか．

(2) 次の $\mathcal{S}$ がレベル $\langle r \rangle$ 双模倣であることを示す．

$$\mathcal{S} = \{(P_1|P_0, P_2|P_0) : P_1 \approx_{\langle r \rangle} P_2\}$$

$(P_1|P_0, P_2|P_0) \in \mathcal{S}$ を仮定する．ここでは，$\mathbf{Com_3}$ による同期の場合のみ示す．$P_1|P_0 \xrightarrow{\tau} P'$ とする．$\mathbf{Com_3}$ より，ある $l, r_1, r_0, s_1, s_0, P_1', P_0'$ について，$P_1 \xrightarrow{l\langle r_1 \rangle @ s_1}$ P_1' かつ $P_0 \xrightarrow{\overline{l}\langle r_0 \rangle @ s_0} P_0'$ かつ $P' \equiv P_1'|P_0'$ かつ $r_1 + r_0 \geq loss(s_1 \triangleright s_0)$ $^{(*1)}$ を得る．さらに，$P_1 \approx_{\langle r \rangle} P_2$ より，ある P_2' について，$P_2 \xRightarrow{\phi(l\langle r_1 \rangle @ s_1, r)}_{\langle r \rangle} P_2'$ かつ $P_1' \approx_{\langle r \rangle} P_2'$ である．次の 2 つの場合に分けて示す．

- $r_1 < loss(s_1) - r$ $^{(*2)}$ の場合：次のようにして，$|r_0| \geq r_0 > r - loss(s_0)$ $^{(*3)}$ を得る．

$$\begin{aligned} r_0 - (r - loss(s_0)) &\geq loss(s_1 \triangleright s_0) - r_1 - (r - loss(s_0)) && \text{by } (*1) \\ &> loss(s_1 \triangleright s_0) - (loss(s_1) - loss(s_0)) && \text{by } (*2) \\ &\geq loss(s_1 \triangleright s_0) - |loss(s_1) - loss(s_0)| \\ &= loss(s_1 \circ \mathtt{rev}(s_0)) - |loss(s_1) - loss(\mathtt{rev}(s_0))| \\ &\geq 0 && \text{by 命題 5.3.1(9)} \end{aligned}$$

  $(*3)$ より，$\theta(\overline{l}\langle r_0 \rangle @ s_0, r) = \varepsilon$ なので，$P_0 \Longrightarrow_{\langle r \rangle} P_0'$ である．また，$(*2)$ より，$\phi(l\langle r_1 \rangle @ s_1, r) = \varepsilon$ なので，$P_2 \Longrightarrow_{\langle r \rangle} P_2'$ である．$\mathbf{Com_1}$ と $\mathbf{Com_2}$ より，それらは $P_2|P_0 \Longrightarrow_{\langle r \rangle} P_2'|P_0'$ を導く．ここで，$\phi(\tau, r) = \varepsilon$ より，$P_2|P_0 \xRightarrow{\phi(\tau,r)}_{\langle r \rangle} P_2'|P_0'$ である．また，$P_1' \approx_{\langle r \rangle} P_2'$ より $(P_1'|P_0', P_2'|P_0') \in \mathcal{S}$ である．

- それ以外の場合 (すなわち，$r_1 \geq loss(s_1) - r$)．このとき，$\phi(l\langle r_1 \rangle @ s_1, r) = l\langle r_1 \rangle @ s_1$ である．$\mathbf{Com_1}$ と $\mathbf{Com_3}$ より，それらは $P_2|P_0 \Longrightarrow_{\langle r \rangle} P_2'|P_0'$ を導く．

ここで，$\theta(\tau, r) = \varepsilon$ より，$P_2|P_0 \Longrightarrow_{\langle r\rangle} Q'$ を得る．さらに，$\phi(\tau, r) = \varepsilon$ より，$P_2|P_0 \stackrel{\phi(\tau,r)}{\Longrightarrow}_{\langle r\rangle} P_2'|P_0'$ である．また，$P_1' \approx_{\langle r\rangle} P_2'$ より $(P_1'|P_0', P_2'|P_0') \in \mathcal{S}$ である．

(3) まず，次の補題 $(**)$ を証明する．

$$P \Longrightarrow_{\langle r\rangle} P' \text{ かつ } r' + loss(s') \leq r \text{ ならば，} P\backslash^L_{\langle r'\rangle(s')} \Longrightarrow_{\langle r\rangle} P'\backslash^L_{\langle r'\rangle(s')} \text{ である．}$$

$P \Longrightarrow_{\langle r\rangle} P'$ かつ $r' + loss(s') \leq r$ $^{(*4)}$ とする．$P \Longrightarrow_{\langle r\rangle} P'$ より，$\theta(t, r) = \varepsilon$ かつ $P \stackrel{t}{\longrightarrow} P'$ となる t が存在する．t の長さに関する帰納法を用いる．長さ 0 の場合 $(t = \varepsilon)$ は明らか．長さ 1 以上の場合 $(t = \alpha t')$，すなわち $P \stackrel{\alpha}{\longrightarrow} P_1 \stackrel{t'}{\longrightarrow} P'$ の場合を示す．仮定より，$\theta(t', r) = \varepsilon$ かつ $\theta(\alpha, r) = \varepsilon$ である．帰納法の仮定より $P_1\backslash^L_{\langle r'\rangle(s')} \Longrightarrow_{\langle r\rangle} P'\backslash^L_{\langle r'\rangle(s')}$ を得る．また，$\theta(\alpha, r) = \varepsilon$ であるので，α は内部アクション τ か，または $|r_1| > r - loss(s_1)$ $^{(*5)}$ のような $\alpha = l_1\langle r_1\rangle@s_1$ である．$\alpha = \tau$ の場合は明らかであるので $\alpha = l_1\langle r_1\rangle@s_1$ の場合のみ示す．このとき，次のようにして $|r_1| > r' - loss(s_1 \triangleright s')$ $^{(*6)}$ を得る．

$$\begin{aligned}
&|r_1| - (r' - loss(s_1 \triangleright s')) \\
&\quad > r - loss(s_1) - r' + loss(s_1 \triangleright s') && \text{by } (*5) \\
&\quad \geq r - loss(s_1) - (r - loss(s')) + loss(s_1 \triangleright s') && \text{by } (*4) \\
&\quad = -loss(s_1) + loss(s') + loss(s_1 \triangleright s') \\
&\quad \geq -loss(s_1) + loss(s') + |loss(s_1) - loss(s')| && \text{by 命題 5.3.1(9)} \\
&\quad \geq 0
\end{aligned}$$

よって，$\mathbf{Res}_1$ と $(*6)$ より，$P \stackrel{l_1\langle r_1\rangle@s_1}{\longrightarrow} P_1$ から，$P\backslash^L_{\langle r'\rangle(s')} \stackrel{l_1\langle r_1\rangle@s_1}{\longrightarrow} P_1\backslash^L_{\langle r'\rangle(s')}$ を導ける．ここで，$\theta(l_1\langle r_1\rangle@s_1, r) = \theta(\alpha, r) = \varepsilon$ より，$P\backslash^L_{\langle r'\rangle(s')} \Longrightarrow_{\langle r\rangle} P_1\backslash^L_{\langle r'\rangle(s')}$ を得る．よって，補題 $(**)$ が証明された．

この補題 $(**)$ を用いて，次の $\mathcal{S}$ がレベル $\langle r\rangle$ 双模倣であることを示す．

$$\mathcal{S} = \{(P_1\backslash^L_{\langle r'\rangle(s')}, P_2\backslash^L_{\langle r'\rangle(s')}) : P_1 \approx_{\langle r\rangle} P_2, r' + loss(s') \leq r\}$$

$(P_1\backslash^L_{\langle r'\rangle(s')}, P_2\backslash^L_{\langle r'\rangle(s')}) \in \mathcal{S}$ かつ $P_1\backslash^L_{\langle r'\rangle(s')} \stackrel{\alpha}{\longrightarrow} P_1'\backslash^L_{\langle r'\rangle(s')}$ とする．この遷移を導く最後の規則は $\mathbf{Res}_1$ か $\mathbf{Res}_2$ であるが，$\mathbf{Res}_2$ の場合は簡単であるので，$\mathbf{Res}_1$ の場合を示す．つまり，ある l, r_1, s_1 について，$P_1 \stackrel{l\langle r_1\rangle@s_1}{\longrightarrow} P_1'$ かつ $\alpha = l\langle r_1\rangle@s_1$

かつ ($l \notin L \cup \overline{L}$ または $|r_1| > r' - loss(s_1 \triangleright s')$) である．ここで，$P_1 \approx_{\langle r \rangle} P_2$ より，ある P_{21}, P_{22}, P_2' について，$P_2 \Longrightarrow_{\langle r \rangle} P_{21} \xrightarrow{\phi(l\langle r_1 \rangle @ s_1, r)} P_{22} \Longrightarrow_{\langle r \rangle} P_2'$ かつ $P_1' \approx_{\langle r \rangle} P_2'$ を得る．仮定 $r' + loss(s') \leq r$ より，上記の補題 $(**)$ によって，$P_2 \backslash^L_{\langle r' \rangle (s')} \Longrightarrow_{\langle r \rangle} P_{21} \backslash^L_{\langle r' \rangle (s')}$ かつ $P_{22} \backslash^L_{\langle r' \rangle (s')} \Longrightarrow_{\langle r \rangle} P_2' \backslash^L_{\langle r' \rangle (s')}$ を得る．また，$l \notin L \cup \overline{L}$ または $|r_1| > r' - loss(s_1 \triangleright s')$ であるので，$\mathbf{Res}_2$ より，$P_{21} \backslash^L_{\langle r' \rangle (s')} \xrightarrow{\phi(l\langle r_1 \rangle @ s_1, r)} P_{22} \backslash^L_{\langle r' \rangle (s')}$ を得る．ここで，$P_1' \approx_{\langle r \rangle} P_2'$ より，$(P_1' \backslash^L_{\langle r' \rangle (s')}, P_2' \backslash^L_{\langle r' \rangle (s')}) \in \mathcal{S}$ である．

(4) 上記の (3) の場合に同様である．

∎

5.4.3 レベル $\langle r \rangle$ 等価

レベル $\langle r \rangle$ 弱等価は観測等価と同様に選択演算子 $+$ によって保存されない問題がある．この小節では，レベル $\langle r \rangle$ 弱等価に含まれ，かつ選択演算子 $+$ によって保存される最大の同値関係として，**レベル $\langle r \rangle$ 等価**を与える．先ず，アクション上の関係 $\trianglerighteq_{\langle r \rangle} (\subseteq Act \times Act)$ を次のように定義する．

定義 5.4.8 $r \in \mathcal{R}$ とする．**レベル $\langle r \rangle$ 代用** $\trianglerighteq_{\langle r \rangle}$ は次のように定義されるアクション上の関係である．

$$\trianglerighteq_{\langle r \rangle} = \{(\alpha, \alpha) : \alpha \in Act\} \cup \{(\alpha, \alpha') : \alpha, \alpha' \in Act, \phi(\alpha, r) = \theta(\alpha', r) = \varepsilon\}$$

∎

$(\alpha \trianglerighteq_{\langle r \rangle} \alpha')$ は，レベル $\langle r \rangle$ 観測において，アクション α の代わりにアクション α' を対応させてもよいことを表している．例えば，次の関係は $\trianglerighteq_{\langle r \rangle}$ の特徴をよく表している．

$$\tau \trianglerighteq_{\langle 1 \rangle} (c\langle -2 \rangle @ \varepsilon), \quad (c\langle -2 \rangle @ \varepsilon) \trianglerighteq_{\langle 1 \rangle} \tau$$
$$\tau \trianglerighteq_{\langle 1 \rangle} (c\langle 2 \rangle @ \varepsilon) \quad , \qquad (c\langle 2 \rangle @ \varepsilon) \not\trianglerighteq_{\langle 1 \rangle} \tau$$

レベル $\langle 1 \rangle$ 観測では，$(c\langle -2 \rangle @ \varepsilon)$ は観測不能アクションであり，$(c\langle 2 \rangle @ \varepsilon)$ は観測不確定アクションである．観測不能アクションは内部アクション τ に同等であるが，観測不確定アクションは同等ではない．観測不確定アクションは (観測不能アクションと同期して) 内部アクションとして動作することがあるが，逆はできない．それは，観測不確定アクションは観測可能アクションと同期することもできるためである．

次にレベル $\langle r \rangle$ 等価を定義する．

定義 5.4.9 $r \in \mathcal{R}$ とする．もし，任意の $\alpha \in Act$ について，次の 2 つの条件を満たすならば，P と Q は**レベル** $\langle r \rangle$ **等価**であるといい，$P =_{\langle r \rangle} Q$ と書く．

(i) $P \xrightarrow{\alpha} P'$ ならば，ある Q', α' について $Q \stackrel{\alpha'}{\Longrightarrow}_{\langle r \rangle} Q'$, $P' \approx_{\langle r \rangle} Q'$, $\alpha \trianglerighteq_{\langle r \rangle} \alpha'$ を満たす．
(ii) $Q \xrightarrow{\alpha} Q'$ ならば，ある P', α' について $P \stackrel{\alpha'}{\Longrightarrow}_{\langle r \rangle} P'$, $P' \approx_{\langle r \rangle} Q'$, $\alpha \trianglerighteq_{\langle r \rangle} \alpha'$ を満たす．

∎

レベル $\langle r \rangle$ 弱等価 $\approx_{\langle r \rangle}$ との違いは，レベル $\langle r \rangle$ 等価 $=_{\langle r \rangle}$ では最初の各アクションについては，代用可能なアクションによって対応されなければならないことである．レベルが高くなるに従って，代用できるアクションの選択の幅は狭くなり，レベル $\langle \infty \rangle$ 等価 $=_{\langle \infty \rangle}$ は観測合同 $=_{ccs}$ と同じになる．

命題 5.4.8 と命題 5.4.9 に示されるように，レベル $\langle r \rangle$ 等価は，レベル $\langle r \rangle$ 弱等価に含まれ，かつ選択演算子 $+$ によって保存される最大の同値関係である．

命題 5.4.8 $P_1 =_{\langle r \rangle} P_2$ ならば，任意の R について，$P_1 + R =_{\langle r \rangle} P_2 + R$ である．

証明 $P_1 =_{\langle r \rangle} P_2$ かつ $P_1 + R \xrightarrow{\alpha} P'$ とする．この遷移を導く次の 2 つの場合を考える．

- $\mathbf{Choice}_1$ より，$P_1 + R \xrightarrow{\alpha} P'$ から $P_1 \xrightarrow{\alpha} P'$ を得る．$P_1 =_{\langle r \rangle} P_2$ であるので，ある Q' と α' について，$P_2 \stackrel{\alpha'}{\Longrightarrow}_{\langle r \rangle} Q'$ かつ $P' \approx_{\langle r \rangle} Q'$ かつ $\alpha \trianglerighteq_{\langle r \rangle} \alpha'$ である．遷移列 $P_2 \stackrel{\alpha'}{\Longrightarrow}_{\langle r \rangle} Q'$ の長さは必ず 1 以上であるので，$\mathbf{Choice}_1$ より，$P_2 + R \stackrel{\alpha'}{\Longrightarrow}_{\langle r \rangle} Q'$ を導ける．

- $\mathbf{Choice}_2$ より，$P_1 + R \xrightarrow{\alpha} P'$ から $R \xrightarrow{\alpha} P'$ を得る．よって，$\mathbf{Choice}_2$ より，$P_2 + R \xrightarrow{\alpha} P'$ を導く．すなわち，$P_2 + R \stackrel{\alpha}{\Longrightarrow}_{\langle r \rangle} P'$ でもある．ここで，$\alpha \trianglerighteq_{\langle r \rangle} \alpha$ である．

∎

命題 5.4.9 $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$ とする．任意の Q について $P_1 + Q \approx_{\langle r \rangle} P_2 + Q$ ならば，$P_1 =_{\langle r \rangle} P_2$ である．ここで，$\mathcal{L}(P)$ は P に現れる全てのアクションの名前の集合である．

証明 $\mathcal{L}(P_1) \cup \mathcal{L}(P_2) \neq \mathcal{N}$ であるので，$c \notin \mathcal{L}(P_1) \cup \mathcal{L}(P_2)$ のような $c \in \mathcal{N}$ が存在する．$C \stackrel{\text{def}}{=} (c\langle r_0 \rangle @ \varepsilon).C$ とすると，仮定より $P_1 + C \approx_{\langle r \rangle} P_2 + C$ である．ここで，$r_0 \geq -r$ とする．このとき，$P_1 =_{\langle r \rangle} P_2$ を証明するために，$P_1 \xrightarrow{\alpha} P_1'$ とする．$\mathbf{Choice}_1$

より，$P_1 + C \xrightarrow{\alpha} P_1'$ を導く．$P_1 + C \approx_{\langle r\rangle} P_2 + C$ であるので，ある P_2' について，$P_2 + C \stackrel{\phi(\alpha,r)}{\Longrightarrow}_{\langle r\rangle} P_2'$ かつ $P_1' \approx_{\langle r\rangle} P_2'$ を得る．

もし α が，$r_1 \geq loss(s_1) - r$ のような $(l_1\langle r_1\rangle @ s_1)$ ならば，$\phi(l_1\langle r_1\rangle @ s_1, r) = l_1\langle r_1\rangle @ s_1$ となる．このとき，$P_2 \stackrel{\alpha}{\Longrightarrow}_{\langle r\rangle} P_2'$ かつ $P_1' \approx_{\langle r\rangle} P_2'$ かつ $\alpha \trianglerighteq_{\langle r\rangle} \alpha$ は簡単に得られる．

それ以外の場合 $(\phi(\alpha, r) = \varepsilon)$ を考える．つまり，$P_2 + C \Longrightarrow_{\langle r\rangle} P_2'$ である．先ず，$P_2' \not\equiv P_2 + C$ を背理法を用いて示すために $P_2' \equiv P_2 + C$ を仮定する．このとき，$r_0 \geq -r$ より，$\phi(c\langle r_0\rangle @ \varepsilon, r) = c\langle r_0\rangle @ \varepsilon$ であるので，P_2' はアクション $(c\langle r_0\rangle @ \varepsilon)$ を起こせなければならない．さらに，$P_1' \approx_{\langle r\rangle} P_2'$ より，P_1' もいつかは $(c\langle r_0\rangle @ \varepsilon)$ を起こせなければならない．しかし，$c \notin \mathcal{L}(P_1') \subseteq \mathcal{L}(P_1)$ より，これは不可能である．よって $P_2' \not\equiv P_2 + C$ を得る．つまり，$\theta(\alpha', r) = \varepsilon$ かつ $P_2 + C \stackrel{\alpha'}{\Longrightarrow}_{\langle r\rangle} P_2'$ のような α' が存在する．ここで，もし，$P_2 + C \stackrel{\alpha'}{\Longrightarrow}_{\langle r\rangle} P_2'$ の遷移が C によって導かれた場合，P_2' は C となるため，再び P_1' はいつかは $(c\langle r_0\rangle @ \varepsilon)$ を起こせなければならなくなる．よって，$P_2 + C \stackrel{\alpha'}{\Longrightarrow}_{\langle r\rangle} P_2'$ の遷移は P_2 によって導かれなければならない．すなわち，$P_2 \stackrel{\alpha'}{\Longrightarrow}_{\langle r\rangle} P_2'$ かつ $P_1' \approx_{\langle r\rangle} P_2'$ を得る．ここで，$\phi(\alpha, r) = \varepsilon = \theta(\alpha', r)$ より，$\alpha \trianglerighteq_{\langle r\rangle} \alpha'$ である． ∎

レベル $\langle r\rangle$ 等価は，制限演算子や経路演算子によって保存されないため，合同関係ではない．ただし，レベル $\langle r\rangle$ 弱等価のための命題 5.4.7は，レベル $\langle r\rangle$ 等価に置き換えても成り立ち，レベル $\langle r\rangle$ 等価も制限演算子や経路演算子によって条件付きでならば保存される．

5.4.4 公理系 $\mathcal{A}\langle r\rangle$

この小節で，レベル $\langle r\rangle$ 等価 $=_{\langle r\rangle}$ と観測合同 $=_{ccs}$ を比較するために，有限逐次プロセスのための公理系を与える．有限逐次プロセスは逐次演算子と選択演算子から構成されるプロセスであり，その構文は次のように定義される．

定義 5.4.10 有限逐次プロセスの集合 $\mathcal{P}_{seq}(\subset \mathcal{P}_{ccsg})$ は *BNF* 記法を用いて次のように定義される．

$$P \quad ::= \quad \mathbf{0} \;\Big|\; \alpha.P \;\Big|\; P + P$$

ここで，$\alpha \in Act$ である． ∎

レベル $\langle r\rangle$ 等価は，逐次と選択演算子によって保存されるので，$\mathcal{P}_{seq}$ に対しては合同関係である．$\mathcal{P}_{seq}$ に対するレベル $\langle r\rangle$ 等価の公理系として，次のように $\mathcal{A}\langle r\rangle$ を与える．

定義 5.4.11 2 つの有限逐次プロセス P と Q の等価性が公理系 $\mathcal{A}\langle r\rangle$ から推論されるならば，$\mathcal{A}\langle r\rangle \vdash P = Q$ と書く．ここで，公理系 $\mathcal{A}\langle r\rangle$ 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	$P = P + \mathbf{0}$	
T1	$\alpha.\tau.P = \alpha.P$	
T2	$P + \tau.P = \tau.P$	
T3	$\alpha.(P + \tau.Q) + \alpha.Q = \alpha.(P + \tau.Q)$	
G1$\langle r\rangle$	$(l\langle r'\rangle@s).P = \tau.P$	if $r' < -(r - loss(s))$
G2$\langle r\rangle$	$(l\langle r'\rangle@s).P = (l\langle r'\rangle@s).P + \tau.P$	if $r' > r - loss(s)$

∎

公理系 $\mathcal{A}\langle r\rangle$ の等式 **M1-4**，**T1-3** は観測合同のための公理系 (定義 2.4.16) と同じである．等式 **G1**$\langle r\rangle$ と **G2**$\langle r\rangle$ は，各々観測不能アクションと観測不確定アクションのための等式である．

以下，$\mathcal{A}\langle r\rangle$ がレベル $\langle r\rangle$ 等価の健全で完全な公理系であることを証明する．まず，次の標準形を定義する．ここで，$\sum_{i=1}^{m} P_i$ は $P_1 + P_2 + \cdots + P_m$ の略記である．

定義 5.4.12 もし次の条件が満たされるならば，P は**レベル** $\langle r\rangle$ **標準形**である．

(i) $P \equiv \sum_{i=1}^{m} \alpha_i.P_i$，ここで各 P_i もレベル $\langle r\rangle$ 標準形である．
(ii) $P \xrightarrow{l\langle r'\rangle@s} P'$ ならば，$r' \geq -(r - loss(s))$ である．
(iii) $P \xrightarrow{l\langle r'\rangle@s} P'$ かつ $r' > r - loss(s)$ ならば，$P \xrightarrow{\tau} P'$ である．

∎

条件 (ii) は内部アクション τ を除く全ての観測不能アクションは起こることができないことを表し，(iii) は全ての観測不確定アクションは内部アクションによってバイパスされなければならないことを表している．

次の命題 5.4.10に示すように，レベル $\langle r\rangle$ 標準形によって，プロセス間の関係をレベル $\langle r\rangle$ 弱等価 (レベル $\langle r\rangle$ 等価) から観測等価 (観測合同) へ強めることができる．

命題 5.4.10 P と Q はレベル $\langle r\rangle$ 標準形であるとする．

(1) $P \approx_{\langle r\rangle} Q$ ならば，$P \approx Q$ である．

(2) $P =_{\langle r\rangle} Q$ ならば，$P =_{ccs} Q$ である．

証明 まず，次の補題 $(*)$ を証明する：

$(*)$ P はレベル $\langle r\rangle$ 標準形であるとする．このとき，$P \xrightarrow{\alpha} P'$ かつ $\theta(\alpha, r) = \varepsilon$ ならば，$P \xrightarrow{\tau} P'$ である．

$\theta(\alpha, r) = \varepsilon$ より，α は τ か，または $|r'| > loss(s') - r$ のような $(l\langle r'\rangle @ s')$ でなければならない．もし $\alpha = \tau$ ならば，明らかに $P \xrightarrow{\tau} P'$ である．$\alpha = (l\langle r'\rangle @ s')$ ならば，レベル $\langle r\rangle$ 標準形の条件 (ii) により，$r' > loss(s') - r$ でなければならない．すなわち，レベル $\langle r\rangle$ 標準形の条件 (iii) より，$P \xrightarrow{\tau} P'$ を得る．

(1) P と Q の深さの和に関する帰納法を用いる．深さ 0 の場合 $(P \equiv Q \equiv \mathbf{0})$ は明らかである．深さ 1 以上の場合について，$P \xrightarrow{\alpha} P'$ とする．$P \approx_{\langle r\rangle} Q$ であるので，ある Q' について，$Q \xRightarrow{\phi(\alpha,r)}_{\langle r\rangle} Q'$ かつ $P' \approx_{\langle r\rangle} Q'$ である．P' と Q' の深さの和は P と Q の深さの和より 1 以上浅いので，帰納法の仮定より $P' \approx Q'$ を得る．遷移 $Q \xRightarrow{\phi(\alpha,r)}_{\langle r\rangle} Q'$ について，ある Q_1 と Q_1' について，$Q \Longrightarrow_{\langle r\rangle} Q_1 \xrightarrow{\phi(\alpha,r)} Q_1' \Longrightarrow_{\langle r\rangle} Q'$ であるので，補題 $(*)$ より，$Q \Longrightarrow Q_1$ かつ $Q_1' \Longrightarrow Q'$ を得る．ここで，$\phi(\alpha, r) = \varepsilon$ ならば，レベル $\langle r\rangle$ 標準形の (ii) より，$\alpha = \tau$ であり，$Q_1 \xrightarrow{\widehat{\tau}} Q_1' \equiv Q_1$ を得る．それ以外では，$\phi(\alpha, r) = \alpha \neq \tau$ であり，$Q_1 \xrightarrow{\widehat{\alpha}} Q_1'$ を得る．すなわち，$Q \xRightarrow{\widehat{\alpha}} Q'$ かつ $P' \approx Q'$ より，$P \approx Q$ を得る．

(2) 上記の (1) と同様である．

∎

次の補題は，公理系 $\mathcal{A}\langle r\rangle$ によって任意のプロセスをレベル $\langle r\rangle$ 標準形に変換できることを示している．

補題 5.4.11 任意の $P \in \mathcal{P}_{seq}$ について，$\mathcal{A}\langle r\rangle \vdash P = P'$ のようなレベル $\langle r\rangle$ 標準形 P' が存在する．

証明 レベル $\langle r\rangle$ 標準形の条件 $(ii)(iii)$ を満たさない次の 2 つの場合がある．

- $P \xrightarrow{l\langle r'\rangle@s} P'$ かつ $r' < -(r - loss(s))$ の場合：**G1**$\langle r\rangle$ により，この遷移を $P \xrightarrow{\tau} P'$ に置き換えることによって条件 (ii) を満たすことができる．
- $P \xrightarrow{l\langle r'\rangle@s} P'$ かつ $r' > r - loss(s)$ かつ $P \overset{\tau}{\not\longrightarrow} P'$ の場合：**G2**$\langle r\rangle$ により，$P \xrightarrow{\tau} P'$ を追加し，条件 (iii) を満たすことができる．

∎

最後に，$\mathcal{A}\langle r\rangle$ が有限プロセスに対するレベル $\langle r\rangle$ 等価の健全で完全な公理系であることを示す定理を与える．

定理 5.4.12 $P, Q \in \mathcal{P}_{seq}$ とする．このとき，

$$P =_{\langle r\rangle} Q \iff \mathcal{A}\langle r\rangle \vdash P = Q$$

証明 ($\Leftarrow$：健全性) 各等式がレベル $\langle r\rangle$ 等価に対して成り立つことを示す．等式 **M1-4**，**T1-3** については観測合同であり，観測合同ならばレベル $\langle r\rangle$ 等価である．**G1**$\langle r\rangle$ については，$r' < -(r - loss(s))$ ならば $(l\langle r'\rangle@s) \trianglerighteq_{\langle r\rangle} \tau$ かつ $\tau \trianglerighteq_{\langle r\rangle} (l\langle r'\rangle@s)$ より，$(l\langle r'\rangle@s).P =_{\langle r\rangle} \tau.P$ を得る．**G2**$\langle r\rangle$ については，$r' > r - loss(s)$ ならば $\tau \trianglerighteq_{\langle r\rangle} (l\langle r'\rangle@s)$ より，$(l\langle r'\rangle@s).P =_{\langle r\rangle} (l\langle r'\rangle@s).P + \tau.P$ を得る．

($\Rightarrow$：完全性) 補題 5.4.11 より，$\mathcal{A}\langle r\rangle \vdash P = P'$ かつ $\mathcal{A}\langle r\rangle \vdash Q = Q'$ のような，レベル $\langle r\rangle$ 標準形 P' と Q' が存在する．$\mathcal{A}\langle r\rangle$ の健全性により，$P' =_{\langle r\rangle} Q'$ を得る．よって，命題 5.4.10 より，$P' =_{ccs} Q'$ を得る．ここで，2章の命題 2.4.18 より，$\mathcal{A}_{ccs} \vdash P' = Q'$ である．最後に，$\mathcal{A}_{ccs} \subset \mathcal{A}\langle r\rangle$ より，$\mathcal{A}\langle r\rangle \vdash P = P' = Q' = Q$ を得る． ∎

5.4.5 プロセス論理

2.5節で紹介したように，双模倣関係をもとにした同値関係をプロセス論理によって特徴付けることができる．本小節では，レベル $\langle r\rangle$ 観測を考慮したプロセス論理の充足関係を定義し，その充足関係とレベル $\langle r\rangle$ 弱等価の関係を明らかにする．

レベル $\langle r\rangle$ 弱等価のためプロセス論理は，観測等価のためのプロセス論理 (定義 2.5.3) にほぼ同じであるが，観測可能なアクションが観測レベルに依存するため，ここでは各レベルごとに次のように与える．

定義 5.4.13 $r \in \mathcal{R}$ とする．**仕様** ($S, T, \cdots$ で表す) の集合 $\mathcal{PL}\langle r\rangle$ は次の式を含む最小の集合である．

$$\begin{aligned} \langle\langle\varepsilon\rangle\rangle S &: \text{可能} \\ \langle\langle\alpha\rangle\rangle S &: \text{可能 } (\alpha \in Act\langle r\rangle) \\ \neg S &: \text{否定} \\ \textstyle\bigwedge_{i\in I} S_i &: \text{合接 } (I \text{ はインデックス集合 }) \end{aligned}$$

ここで，S, S_i はすでに $\mathcal{PL}\langle r\rangle$ の要素であるとする．また，$Act\langle r\rangle$ はレベル $\langle r\rangle$ 観測のもとで観測可能なアクションの集合であり，次のように与えられる．

$$Act\langle r\rangle = \{l\langle r'\rangle @ s \in Act : r' - loss(s) \geq -r\}$$

∎

この仕様によりプロセスの性質を記述する．レベル $\langle r\rangle$ 観測のもとで，プロセス P が仕様 $S \in \mathcal{PL}\langle r\rangle$ を満たすことを $P \models_{\langle r\rangle} S$ と書き，次のように与える．

定義 5.4.14 レベル $\langle r\rangle$ 観測における充足関係 $\models_{\langle r\rangle} \subseteq \mathcal{P}_{ccsg} \times \mathcal{PL}\langle r\rangle$ は式の構造について帰納的に次のように与えられる．

$$\begin{aligned} &(1)\ P \models_{\langle r\rangle} \langle\langle\varepsilon\rangle\rangle S \ \Leftrightarrow\ \exists P'.\ (P \Longrightarrow_{\langle r\rangle} P',\ P' \models_{\langle r\rangle} S) \\ &(2)\ P \models_{\langle r\rangle} \langle\langle\alpha\rangle\rangle S \ \Leftrightarrow\ \exists P'.\ (P \overset{\alpha}{\Longrightarrow}_{\langle r\rangle} P',\ P' \models_{\langle r\rangle} S,\ \alpha \in Act\langle r\rangle) \\ &(3)\ P \models_{\langle r\rangle} \neg S \ \Leftrightarrow\ P \not\models_{\langle r\rangle} S \\ &(4)\ P \models_{\langle r\rangle} \textstyle\bigwedge_{i\in I} S_i \ \Leftrightarrow\ \forall i \in I.\ P \models_{\langle r\rangle} S_i \end{aligned}$$

∎

次の定理に示すように，P と Q がレベル $\langle r\rangle$ 弱等価でないことと，P と Q を区別する仕様が存在することは同じである．

定理 5.4.13 $P \not\approx_{\langle r\rangle} Q \iff \exists S \in \mathcal{PL}\langle r\rangle.\ (P \models_{\langle r\rangle} S, P \not\models_{\langle r\rangle} S)$

証明 $(\Leftarrow)$：対偶「$P \approx_{\langle r\rangle} Q$ ならば，全ての S について，$P \models_{\langle r\rangle} S \iff Q \models_{\langle r\rangle} S$」を S の構造に関する帰納法を用いて証明する．$P \approx_{\langle r\rangle} Q$ かつ $P \models_{\langle r\rangle} S$ とする．

1. $S \equiv \langle\langle\varepsilon\rangle\rangle S'$ の場合：$P \models_{\langle r\rangle} S$ より，ある P' について，$P \Longrightarrow_{\langle r\rangle} P'$ かつ $P' \models_{\langle r\rangle} S'$ である．このとき，$P \approx_{\langle r\rangle} Q$ であるので，命題 5.4.5の証明で示した補題 $(*)$ より，ある Q' について，$Q \Longrightarrow_{\langle r\rangle} Q'$ かつ $P' \approx_{\langle r\rangle} Q'$ である．ここで，帰納法の仮定より，$Q' \models_{\langle r\rangle} S'$ を得る．すなわち，$Q \models_{\langle r\rangle} \langle\langle\varepsilon\rangle\rangle S' \equiv S$ である．

2. $S \equiv \langle\langle\alpha\rangle\rangle S'$ の場合：$P \models_{\langle r\rangle} S$ より，ある P' について，$P \stackrel{\alpha}{\Longrightarrow}_{\langle r\rangle} P'$ かつ $P' \models_{\langle r\rangle} S'$ である．α は観測可能 $(\alpha \in Act\langle r\rangle)$ であるので，ある Q' について，$Q \stackrel{\alpha}{\Longrightarrow}_{\langle r\rangle} Q'$ かつ $P' \approx_{\langle r\rangle} Q'$ を容易に得ることができる．ここで，帰納法の仮定より，$Q' \models_{\langle r\rangle} S'$ を得る．すなわち，$Q \models_{\langle r\rangle} \langle\langle\alpha\rangle\rangle S' \equiv S$ である．

3. $S \equiv \neg S'$ の場合：$P \not\models_{\langle r\rangle} S'$ であるので，帰納法の仮定より，$Q \not\models_{\langle r\rangle} S'$ を得る．すなわち，$Q \models_{\langle r\rangle} \neg S' \equiv S$ である．

4. $S \equiv \bigwedge_{i\in I} S_i$ の場合：全ての $i \in I$ について，$P \models_{\langle r\rangle} S_i$ であるので，帰納法の仮定より，$Q \models_{\langle r\rangle} S_i$ を得る．すなわち，$Q \models_{\langle r\rangle} \bigwedge_{i\in I} S_i \equiv S$ である．

$(\Rightarrow)$：対偶「全ての S について $P \models_{\langle r\rangle} S \iff Q \models_{\langle r\rangle} S$ ならば，$P \approx_{\langle r\rangle} Q$」を証明する．そのために，次の $\mathcal{S}$ がレベル $\langle r\rangle$ 双模倣であることを示す．

$$\mathcal{S} = \{(P,Q) : \forall S \in \mathcal{PL}\langle r\rangle.\, (P \models_{\langle r\rangle} S \iff Q \models_{\langle r\rangle} S)\}$$

背理法を用いるために，ある $(P,Q) \in \mathcal{S}$ と $P \xrightarrow{\alpha} P'$ について，$Q \xrightarrow{\phi(\alpha,r)}_{\langle r\rangle} Q_i$ のような全ての Q_i $(i \in I)$ で，$(P',Q_i) \notin \mathcal{S}$ であると仮定する．各 $i \in I$ について $(P',Q_i) \notin \mathcal{S}$ より，$P' \models_{\langle r\rangle} S_i$ かつ $Q_i \not\models_{\langle r\rangle} S_i$ のような $S_i \in \mathcal{PL}\langle r\rangle$ が存在する．

- $\alpha \in Act\langle r\rangle$ の場合：すなわち，$\phi(\alpha,r) = \alpha$ である．このとき，$P \stackrel{\alpha}{\Longrightarrow}_{\langle r\rangle} P'$ と，全ての $i \in I$ について，$Q \stackrel{\alpha}{\Longrightarrow}_{\langle r\rangle} Q_i$ を得る．よって，$S \equiv \langle\langle\alpha\rangle\rangle(\bigwedge_{i\in I} S_i)$ とすると，$P \models_{\langle r\rangle} S$ かつ $Q \not\models_{\langle r\rangle} S$ となるが，これは $(P,Q) \in \mathcal{S}$ の仮定に反する．

- $\alpha \notin Act\langle r\rangle$ の場合：すなわち，$\phi(\alpha,r) = \varepsilon$ である．このとき，$P \Longrightarrow_{\langle r\rangle} P'$ と，全ての $i \in I$ について，$Q \Longrightarrow_{\langle r\rangle} Q_i$ を得る．よって，$S \equiv \langle\langle\varepsilon\rangle\rangle(\bigwedge_{i\in I} S_i)$ とすると，$P \models_{\langle r\rangle} S$ かつ $Q \not\models_{\langle r\rangle} S$ となるが，これは $(P,Q) \in \mathcal{S}$ の仮定に反する．

∎

さらに，次の記法を導入する．

$$\mathtt{tt} \equiv \bigwedge_{i\in\emptyset} S_i, \quad \mathtt{ff} \equiv \neg\mathtt{tt}, \quad [\![\alpha]\!]S \equiv \neg\langle\langle\alpha\rangle\rangle\neg S$$

tt は全てのプロセスによって満たされる仕様であり，ff は満たすことができない仕様である．また，$\langle\langle\alpha\rangle\rangle S$ は，(0 個以上の観測不能アクションの後に)α を実効でき，その後に (0 個以上の観測不能アクションの後に) S を満たすことができることを要求する．一方，$[\![\alpha]\!]S$ は，(0 個以上の観測不能アクションの後に) α を実効するならば，その後は (0 個以上の観測不能アクションを実効後は全て) 必ず S を満たすことを要求する．

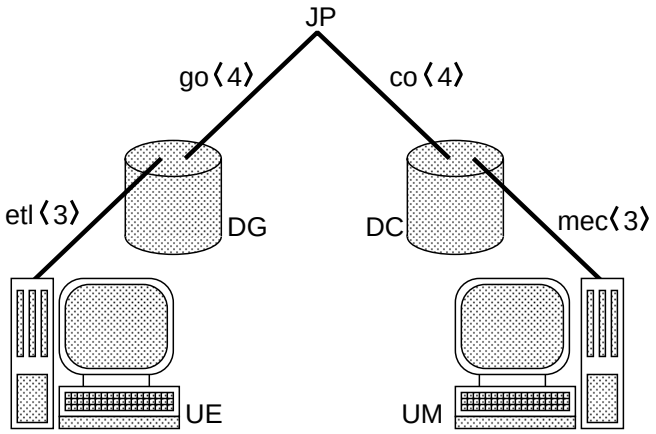

図 5.8: デッドロックをもつシステムの例

5.5 デッドロックをもつ並行システムの例

図 5.8の並行システムを用いて CCSG による解析例を示す．図中，DG と DC は各々国と企業が所有するデータベース，UE は国立研究所 ETL の端末機，UM は株式会社 MEC の端末機であり，各々階層的に 4 つのルータを通して接続されている．このシステムは図中 JP の位置を観測位置として次のように記述される．

$$
\begin{aligned}
SYS &\stackrel{\text{def}}{=} ((GO@go\langle 4\rangle)|(CO@co\langle 4\rangle))\backslash^{\{lk_1,lk_2,ul_1,ul_2\}}_{\langle 18\rangle(\varepsilon)} \\
GO &\stackrel{\text{def}}{=} DG|(UE@etl\langle 3\rangle) \\
CO &\stackrel{\text{def}}{=} DC|(UM@mec\langle 3\rangle)
\end{aligned}
$$

ここで，局所制限の制限力が 18 である理由については後で説明する．

DG と DC へのアクセスにはロックが必要であるとする．簡単のためデータ操作は省略する．UE と UM は，各々コマンド ac_1, ac_2 を受けると DG と DC のロックを試みる．ただし，各端末は近い方からロックを行なうとする．このとき，各プロセスは次のように記述される．ここで，空経路 ε とグレイド $\langle 0\rangle$ の記述は省略されている (例えば，ac_1 は $ac_1\langle 0\rangle@\varepsilon$，$\overline{lk_1}\langle 3\rangle$ は $\overline{lk_1}\langle 3\rangle@\varepsilon$ の略記である)．

$$
\begin{aligned}
UE &\stackrel{\text{def}}{=} ac_1.\overline{lk_1}\langle 3\rangle.\overline{lk_2}\langle 11\rangle.su_1.\overline{ul_2}\langle 11\rangle.\overline{ul_1}\langle 3\rangle.UE \\
UM &\stackrel{\text{def}}{=} ac_2.\overline{lk_2}\langle 3\rangle.\overline{lk_1}\langle 11\rangle.su_2.\overline{ul_1}\langle 11\rangle.\overline{ul_2}\langle 3\rangle.UM \\
DG &\stackrel{\text{def}}{=} lk_1.ul_1.DG \\
DC &\stackrel{\text{def}}{=} lk_2.ul_2.DC
\end{aligned}
$$

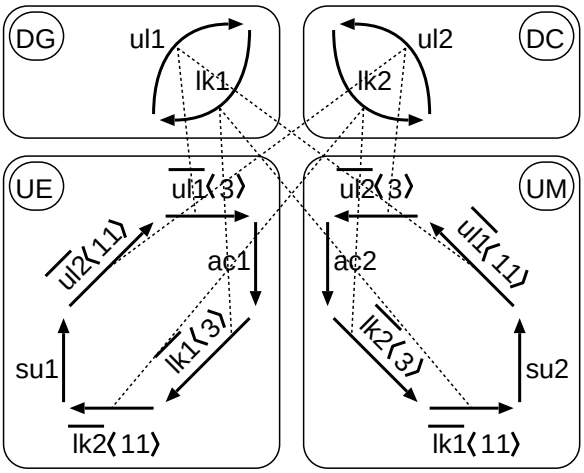

図 5.9: SYS の状態遷移図

コマンド ac の入力とロック成功 su を知らせる出力は端末に局所的なアクションであるので，それらのグレイドは 0 に設定してある．ロック lk とアンロック ul はその要求側 (端末側) に十分なグレイドを設定してある．例えば $\overline{lk_2}\langle 11\rangle$ のグレイド 11 は UE と DC の損失距離 $(3+4+4)$ である．上の SYS で使われている制限力 18 は，JP から損失距離 7 離れた $\overline{lk_i}\langle 11\rangle$ と $\overline{ul_i}\langle 11\rangle$ を制限するために必要な最小値である．

図 5.9に各プロセスの振舞いを状態遷移図で示す．各プロセスは独立にアクションを起こすことができるが，点線で結ばれた 2 つのアクションは同期することを表している．図 5.9にみられるように，並行プロセスの全体の動作を理解することは，逐次プロセスの動作に比べて困難である．そこで，SYS の動作と観測合同な逐次プロセス SP を次に与える $(SYS =_{ccs} SP)$.

$$SP \stackrel{\text{def}}{=} ac_1@s_1.R_1 + ac_2@s_2.R_2$$
$$R_1 \stackrel{\text{def}}{=} \tau.(\tau.(su_1@s_1.SP + ac_2@s_2.su_1@s_1.R_2) + ac_2@s_2.O_{12}) + ac_2@s_2.(\tau.O_{12} + \tau.O_{21})$$
$$R_2 \stackrel{\text{def}}{=} \tau.(\tau.(su_2@s_2.SP + ac_1@s_1.su_2@s_2.R_1) + ac_1@s_1.O_{21}) + ac_1@s_1.(\tau.O_{12} + \tau.O_{21})$$
$$O_{12} \stackrel{\text{def}}{=} \tau.su_1@s_1.R_2 + \tau.\mathbf{0}$$
$$O_{21} \stackrel{\text{def}}{=} \tau.su_2@s_2.R_1 + \tau.\mathbf{0}$$

ここで，$s_1 = etl\langle 3\rangle go\langle 4\rangle$，$s_2 = mec\langle 3\rangle co\langle 4\rangle$ である．

図 5.10に SP の状態遷移図を示す．この SP は並行合成演算子を含まず，SYS の振舞いを明確に表しており，SYS はデッドロック状態 $\mathbf{0}$ をもつことが分かる．しかし，SP は簡単な SYS に対してさえやや複雑である．この原因は 2 つの端末のアクション

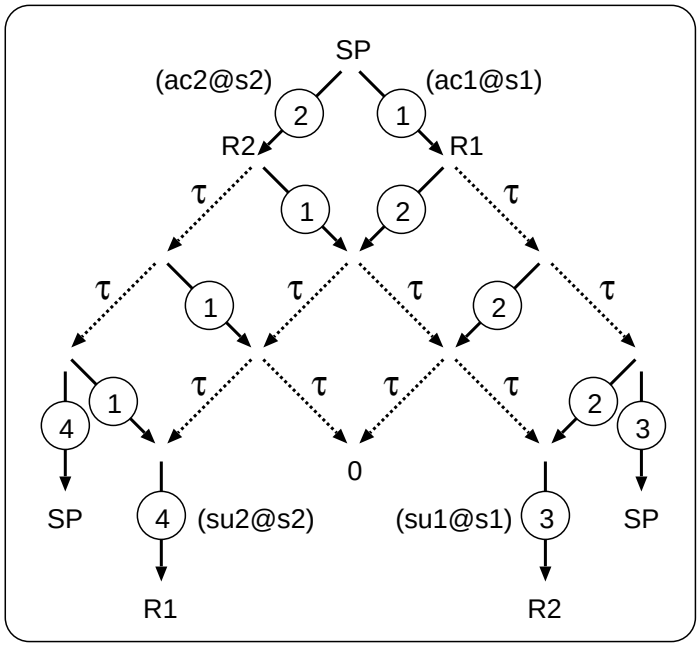

図 5.10: SP の状態遷移図

が独立に起こせるためである．この SP は端末の増加とともに急激に複雑になる．

一方，ETL の利用者は，しばしば ETL の端末操作のみ可能で，遠くの重要でないアクションには興味をもたないことがある．このような解析に対して，シフト (s) 等価とレベル $\langle r \rangle$ 等価を有効に使うことができる．まず，観測位置を

$$SYS \precsim_{(s_e)} SYS@s_e$$

によって JP から ETL に移動する．ここで，s_e は JP から ETL までの経路で，$s_e{=}go\langle 4\rangle etl\langle 3\rangle$ である．そして，$SYS@s_e$ の動作を表す仕様として SP_{ETL} を

$$SP_{ETL} \stackrel{\text{def}}{=} ac_1.(\tau.su_1.SP_{ETL} + \tau.\mathbf{0})$$

と定義すると，次の等式が成り立つ．

$$SYS@s_e =_{\langle r\rangle} \tau.SP_{ETL}$$
$$SYS@s_e \approx_{\langle r\rangle} SP_{ETL}$$

ここで，r は 14 未満の実数である．14 とは ETL-MEC 間の損失距離であり，これにより MEC での端末操作を無視することができる．$=_{\langle r\rangle}$ の右辺の $\tau.SP_{ETL}$ の τ は，MEC

の端末操作による観測できないアクションに対応するために最初の 1 回だけ必要である．SP_{ETL} から，デッドロックが ac_1 の直後に起こる可能性があるが，su_1 の直後には起こらないことを読みとることができる．

次に，SYS の動作の ETL の位置における，プロセス論理による検証例を示す．まず，0 以上の任意の r について，

$$SYS@s_e \models_{\langle r \rangle} \langle\langle ac_1 \rangle\rangle \langle\langle su_1 \rangle\rangle \mathtt{tt}$$

が成り立つ．ac_1 を起こした後，デッドロックを起こす可能性はあるが，成功 su_1 することもあるので真である．また，0 以上 14 未満の r について，

$$SYS@s_e \models_{\langle r \rangle} \langle\langle ac_1 \rangle\rangle [\![su_1]\!] \mathtt{ff} \qquad (**)$$

が成り立つ．この $[\![su_1]\!]\mathtt{ff}$ を満たすプロセスとは，su_1 を起こせるならば，その後は必ず $\mathtt{ff}$ を満たすプロセスである．しかし，$\mathtt{ff}$ を満たすプロセスは存在しないので，$[\![su_1]\!]\mathtt{ff}$ を満たすプロセスとは su_1 を起こせないプロセスのことである．これは，SP にみられるデッドロック $\mathbf{0}$ に相当する．

ここで重要なことは，14 以上の r については $(**)$ は成り立たないことである．つまり，ac_2 のような遠くの重要でないアクションを考慮しなければ，su_1 のような重要なアクションが起こせなくなる可能性をみつけることができない．例えば，より多くの重要でないアクションが関係している場合は，その全てを考慮しなければ，デッドロックをみつけることができないかも知れない．この例はレベル r を 14 より低くすることによって，重要なアクション su_1 が実行できなくなる可能性を示しており，観測精度を落した近似的な充足性 $\models_{\langle r \rangle}$ を用いることによって，重要なアクションの実行可能性を調べることができる近似解析の有効性を示した例である．

5.6 関連研究

空間の概念をもつプロセス代数としては，CCS を基に [6, 30, 42]，ACP[4] を基に [5] 等が提案されている．

文献 [5] の特徴は時間と空間の統一的な表現にある．例えば，アクションの伝達速度を考慮して離れた位置にあるプロセス間通信の可能性を検証できる．CCSG には時間の概念は無いが，CCSG の目的は重要度が減衰する通信の表現とその近似解析にあり，

そのためにグレイドや損失が導入されている．[5] にはグレイドの概念は無く，近似的な解析は行なわれていない．また，CCSG に TCCS[41] のような時間の概念を導入し，各ルータに遅延時間を明記することによって，伝達速度を表現することも考えられる．

CCS 系の拡張 [6, 30, 42] では，アクションの生じた位置を考慮することにより，観測等価よりも詳細にプロセスを解析することを可能にしている．例えば，[6] の**位置等価** (location equivalence) では，$P_1 \equiv (a.\mathbf{0}|b.\mathbf{0})$ と $P_2 \equiv (a.b.\mathbf{0} + b.a.\mathbf{0})$ は位置等価にはならない．これは，次の位置付状態遷移にみられるように，P_1 の a と b の位置は独立であるのに対し，P_2 の a と b の位置は独立でないためである．

$$P_1 \xrightarrow[u_1]{a} (u_1 :: \mathbf{0})|(b.\mathbf{0}) \xrightarrow[u_2]{b} (u_1 :: \mathbf{0})|(u_2 :: \mathbf{0}),$$
$$P_2 \xrightarrow[u_1]{a} u_1 :: b.\mathbf{0} \xrightarrow[u_1u_2]{b} u_1u_2 :: \mathbf{0}.$$

ここで，$\xrightarrow[u]{l}$ の u は l の位置を表している．つまり，P_2 の位置付状態遷移の b の位置 u_1u_2 は，a の位置 u_1 に依存していることを表している．この位置付状態遷移の位置情報 u は自動的に付加され，逐次動作 (インターリーブ) と並行動作を明確に区別するために使われる．このような概念は，真の並行性 (true concurrency) と呼ばれている．また，位置付状態遷移 $\xrightarrow[u]{l}$ を一般の状態遷移 $\xrightarrow{l}$ によって解釈する方法が [45] に提案されている．

一方，CCSG では，必ずしも逐次動作と並行動作を区別しない．例えば，次の Q_1 と Q_2 はレベル $\langle\infty\rangle$ 等価となる．

$$Q_1 \equiv (a.\mathbf{0})@s_1|(b.\mathbf{0})@s_2,$$
$$Q_2 \equiv (a@s_1).(b@s_2).\mathbf{0} + (b@s_2).(a@s_1).\mathbf{0}$$

すなわち，CCSG は真の並行プロセス代数ではない．CCSG において，位置 (経路) は損失距離を求めるために導入されており，真の並行性 [6, 30, 42] のために導入された位置とは目的が異なる．

5.7 おわりに

本章では，CCS にアクションの重要性を明記するためにグレイド，プロセス間の空間的距離を考慮するために経路を導入して，重要度が減衰する通信をもつシステムを記述できるプロセス代数として CCSG を提案した．CCSG におけるプロセスの等価性

としては，観測位置を移動するためにシフト (s) 等価，遠くの重要でないアクションを無視するためにレベル $\langle r \rangle$ 弱等価とレベル $\langle r \rangle$ 等価を与えた．これらにより，CCSG では，任意の観測位置における近似的な (局所的な) 解析が可能になっている．

さらに，レベル $\langle r \rangle$ 等価と観測合同の違いを明確にするために，有限逐次プロセスに対するレベル $\langle r \rangle$ 等価の健全で完全な公理系 $\mathcal{A}\langle r \rangle$ を与えた．また，レベル $\langle r \rangle$ 観測を考慮したプロセス論理の充足関係を定義し，その充足関係とレベル $\langle r \rangle$ 弱等価の関係を明らかにした．

5.5節では，システム SYS と局所的仕様 SP_{ETL} との近似的な関係を示した．SP_{ETL} は全体の仕様 SP に比べて簡単であり，ETL における SYS の動作を理解するために有効である．このように，CCSG の利点は，システムの設計者が自分に必要なアクションを取り出し，遠方の重要でないアクションを無視することによって，可読性の高い仕様書を記述できることにある．

第 6 章

仕様合成用プロセス代数

6.1 はじめに

5章では，遠くの重要でない情報を無視して近似解析できるプロセス代数として CCSG を提案した．CCSG では複数の観測位置で観測レベルを下げることによって，完全な仕様から複数の不完全 (局所的) な仕様を得ることができる．本章では，この逆の手続きに着目する．すなわち，複数の不完全な仕様から完全な仕様を合成する方法を検討する．大規模システム開発では，一つの完全な仕様の代わりに複数の柔軟 (不完全) な仕様が与えられることがある．このとき，それら複数の柔軟な仕様からより詳細な一つの仕様を合成し，さらにその詳細な仕様を満たすシステムを合成することは重要である．

完全な仕様とは，その仕様を満たすシステムの振舞いが一意に定まるような仕様であり，柔軟 (不完全) な仕様とは，振舞いの異なる複数のシステムによって満たされる仕様である．CCSG では近似的な同値関係 $=_{\langle r \rangle}$ を用いて柔軟な仕様を表現していたが，この柔軟な仕様はシステムが減衰通信をもつ場合に限られていた．本章では，論理演算子 (合接，離接，最小不動点) をプロセス代数に導入して，仕様自体を柔軟に記述する方法をとる．これによって，より一般的なシステムに対して本手法を適用できる．

柔軟な仕様を記述するために，**プロセス論理** [38, 52] の演算子がプロセス代数に導入されてきた．例えば，Larsen は実行してもしなくてもよいアクションを表現できるように，二種類の遷移 (要求遷移 $\longrightarrow_{\Box}$ と許可遷移 $\longrightarrow_{\diamond}$) を提案し [32]，その遷移をもとに柔軟な仕様が記述できるプロセス代数 modal CCS[33] を与えた．この modal CCS に

は，二つの仕様の共通部分を取り出す合接演算子 $\wedge$ も定義されている．さらに，Steen 等 [47, 48] は Larsen の二種類の遷移に離接遷移 $\mapsto$ を加え，離接演算子 $\vee$ も表現できるプロセス代数 PSL を提案した．

プロセス論理のみでも柔軟な仕様を記述できるが，プロセス論理演算子で拡張されたプロセス代数では，並行演算子と論理演算子の両方を用いて並行動作に対する柔軟な仕様を直接表現できる利点がある．また，この拡張されたプロセス代数では柔軟な仕様から実行可能なシステムまで同じ枠組 (拡張プロセス代数) で記述できるため，仕様の記述からシステムの記述への変換が容易である．

本章では，プロセス論理の演算子をもつプロセス代数として μLOTOS を提案する．従来のプロセス論理の演算子をもつプロセス代数 [33, 47, 48] と比較して，μLOTOS では**最小不動点**を表現できる特徴をもつ．すなわち，μLOTOS ではライブネス特性のような「**いつか**は実行される特性」を表現できる．また，μLOTOS にも複数の柔軟な仕様に共通する要求に相当する仕様を合成するために合接演算子が定義されている．さらに，与えられた仕様を満たすシステムが存在するか (充足可能性) を判定する方法と，その仕様から逐次システムを合成する方法を与える．

以下，まず，6.2節で μLOTOS の概要を述べ，6.3節で μLOTOS の構文と意味を定義する．6.4節でプロセスの仕様に対する充足性を定義する．次に，6.5節で μLOTOS によって最小不動点がどのように表現されるかを示す．そして，6.6節では仕様の充足可能性の帰納的な定義を示す．これにより，仕様の状態数が有限であれば，その充足可能性を判定できる．さらに，仕様が充足可能であれば，その仕様を満たす逐次プロセスを合成する方法を示す．6.7節で関連研究と μLOTOS を比較する．

尚，他の章では CCS の記法をもとにしているが，本章では従来研究 [47, 48] と本研究を比較し易いようにプロセス代数 LOTOS[56] の記法を用いている．ただし，本章で得られる成果は LOTOS に特化したものではなく，ラベル付遷移システムをもとにしたプロセス代数に広く適用可能である．

6.2 μLOTOS の紹介

柔軟な仕様を記述するために，Steen 等は LOTOS に離接演算子 $\vee$ を導入して拡張することを提案した [47]．これは (プロセス) 論理の離接演算子と同様に，もし P_1 が仕様 S_1 を満たすプロセスであり，かつ P_2 が仕様 S_2 を満たすプロセスであるならば，仕

様 $S_1 \vee S_2$ はプロセス P_1 または P_2 として実装できることを表している．例えば，仕様 $(a;\mathbf{stop} \vee b;\mathbf{stop})$ は a か b が実行されることを要求しているので，プロセス $(a;\mathbf{stop})$ または $(b;\mathbf{stop})$ として実装される．ここで，; は逐次演算子であり，**stop** は無動作プロセス (無動作仕様) である．

離接演算子を含む仕様を次の例のように，再帰的に定義することもできる．

$$AB \stackrel{\text{def}}{=} a; AB \vee b;\mathbf{stop}$$

この仕様 AB は，繰り返しアクション a を実行するか，またはアクション b を実行して停止することを要求している．例えば，次に示す全てのプロセスによって AB は満たされる $(n \geq 0)$.

$$PA_\infty \stackrel{\text{def}}{=} a; PA_\infty, \quad PAB_n \stackrel{\text{def}}{=} \underbrace{a;\cdots;a}_{n\text{ 個}};b;\mathbf{stop}$$

ここで注目すべきは，PA_∞ も AB を満たすことである．すなわち，仕様 AB は必ずしも b がいつかは実行されることを要求していない．実際，上記の離接付 LOTOS[47] では，いつかは実行される特性を記述することはできない．

これに対し，いつかは実行される特性 (**ライブネス特性**) が要求されることがある．本章で提案する μLOTOS は，ライブネス特性を表現できるように上記の離接付 LOTOS を拡張したプロセス代数である．μLOTOS では，**不安定演算子** $\triangleleft$ によって**不安定状態**をつくりだし,「全ての状態はいつかは安定状態に到達できなければならない」という条件を加えて，ライブネス特性を表現する．例えば，上記の仕様 AB の例については，次のように不安定状態を明記することによって，いつかは b が実行される要求を記述することができる．

$$AB' \stackrel{\text{def}}{=} \triangleleft a; AB' \vee b;\mathbf{stop}$$

ここで，$(\triangleleft a; AB')$ は不安定状態であり，a を選択し続ける限り安定状態に到達することはできない．すなわち，任意の $n \geq 0$ について上記の PAB_n は AB' を満たすが，PA_∞ は AB' を満たさない．

次に，ファイル操作に対する μLOTOS の仕様の例 $FILE$ を紹介する．

$$FILE \stackrel{\text{def}}{=} open; OPENED$$
$$OPENED \stackrel{\text{def}}{=} \triangleleft(write; OPENED \times read; OPENED) \times close; FILE$$

ここで， $S \times T$ は次のような仕様の略記法である．

$$S \times T \equiv S \vee T \vee (S \,[\!]\, T)$$

ここで，$[]$ は LOTOS の選択演算子であり，CCS の $+$ と同じである．すなわち，例えば，仕様 $(a; \mathbf{stop} \times b; \mathbf{stop})$ は，a かつ/または b が実行されることを要求する．

仕様 $FILE$ は，$open$ の後に仕様 $OPENED$ を満たすことを要求し，$OPENED$ は，$write$ や $read$ の後は再び $OPENED$ を満たし，$close$ の後は $FILE$ を満たすことを要求している．$OPENED$ では，少なくとも $write$ か $read$ か $close$ が実行されることが要求されているが，$write$ か $read$ を選択し続けると安定状態に到達できないため，いつかは $close$ が選択されることが要求される．すなわち，仕様 $FILE$ は $open$ の後に $write$ や $read$ を実行してもよいが，いつかは $close$ が実行されなければならないことを要求している．

例えば，$FILE$ は次のプロセス $READ$ や $UPDATE$ として実装される．

$$READ \stackrel{\text{def}}{=} open; read; close; READ$$
$$UPDATE \stackrel{\text{def}}{=} open; read; (close; UPDATE \,[]\, write; close; UPDATE)$$

プロセス $READ$ は $open$ の後に $read$ を実行して $close$ する動作を繰り返す．また，プロセス $UPDATE$ は $open$ の後に $read$ を実行して $close$ することもできるが，$write$ してから $close$ することもできる．どちらのプロセスも仕様 $FILE$ を満たしている．

一方，次のプロセス $READLOOP$ は $close$ できないため，$FILE$ を満たしてはいない．

$$READLOOP \stackrel{\text{def}}{=} open; LOOP$$
$$LOOP \stackrel{\text{def}}{=} read; LOOP$$

ただし，上記の仕様 $FILE$ から不安定演算子 $\triangleleft$ を取り除いた仕様を $FILE'$ とすると，$READLOOP$ は $FILE'$ を満たしている．このことは，いつかは $close$ が実行されることが，不安定演算子によって保証されていることを示している．

6.3 仕様の定義

前節では μLOTOS の特徴を例をもとに説明した．本節では μLOTOS を形式的に定義する．以下，6.3.1小節と 6.3.2 小節で，μLOTOS の構文と意味を与える．

6.3.1 構文

まず，**名前**の有限集合 $\mathcal{N}$ が与えられていると仮定する．このとき，**アクション**の集合 Act を $Act = \mathcal{N} \cup \{\tau\}$ により与え，その要素を $\alpha, \beta, \cdots$ で表す．ここで，τ は内部ア

クションである ($\tau \notin \mathcal{N}$)．また，**仕様定数** (または**定数**と呼ぶ) の集合 $\mathcal{K}_\mu$ と**仕様変数** (または**変数**と呼ぶ) の集合 $\mathcal{X}_\mu$ が与えられているとする．定数を $A, B, \cdots$ で表し，変数を $X, Y, \cdots$ で表す．

次に μLOTOS の仕様式 ($M, N, \cdots$ で表す) の定義を与える．

定義 6.3.1 仕様式の集合 Spx は次の式を含む最小の集合である．

$$
\begin{array}{rcl}
\mathbf{stop} & : & \text{無動作仕様} \\
A & : & \text{仕様定数 } (A \in \mathcal{K}_\mu) \\
X & : & \text{仕様変数 } (X \in \mathcal{X}_\mu) \\
\alpha; M & : & \text{逐次 } (\alpha \in Act) \\
\triangleleft M & : & \text{不安定} \\
M_1 \,[]\, M_2 & : & \text{選択} \\
M_1 \,|[G]|\, M_2 & : & \text{並行合成 } (G \subseteq \mathcal{N}) \\
M_1 \wedge M_2 & : & \text{合接} \\
\bigvee_{i \in I} M_i & : & \text{離接 } (I \text{ はインデックス集合 })
\end{array}
$$

ここで，M, M_i はすでに Spx の要素であるとする．演算子の結合の優先順位は，{ 逐次, 不安定 } > 並行合成 > 選択 > 合接 > 離接，である． ∎

離接 $\bigvee_{i \in I} M_i$ は $\bigvee\{M_i : i \in I\}$，または $\bigvee\{M : C(M)\}$ とも書かれる．ここで C は，条件 C を満たす仕様式の集合が $\{M_i : i \in I\}$ となるような条件である．また，$\bigvee_{i \in \{1,2\}} M_i$ を $M_1 \vee M_2$ とも書く．特別な場合として，$I = \emptyset$ のときは，$\bigvee_{i \in \emptyset} M_i$ を `ff` と書く．`ff` は満たすことができない仕様である．また，選択演算子についても，次の記法を用いる．

$$
\sum\{M_1, \cdots, M_n\} \equiv \begin{cases} \mathbf{stop} & (n = 0) \\ M_1 \,[]\, \cdots \,[]\, M_n & (n \geq 1) \end{cases}
$$

選択演算子 $[]$ と離接演算子 $\vee$ の直観的な違いは次のように説明できる：$M \,[]\, N$ は M または N のように振舞うかを実行時に決定するのに対し，$M \vee N$ は M または N を満たすように実装するかを設計時に決定する．よって，離接演算子は仕様記述のみに使われ，実装後には残らない．

LOTOS の並行合成演算子 $|[G]|$ は名前の部分集合 G に含まれるアクションを同期して実行し，それ以外のアクションを独立に実行する．この並行合成演算子は三つ以上のアクションを同期することも可能である．

仕様式 M に含まれる仕様変数の集合を $Var(M)$ と書く．すなわち，$Var(M)$ は M の構造に対して帰納的に次のように与えられる．

$$\begin{aligned} Var(A) &= \emptyset, \\ Var(X) &= \{X\}, \\ Var(\mathbf{stop}) &= \emptyset, \\ Var(\alpha; M) &= Var(M), \\ Var(\triangleleft M) &= Var(M), \\ Var(M \ op \ N) &= Var(M) \cup Var(N), \\ Var(\bigvee_{i\in I} M_i) &= \bigcup_{i\in I} Var(M_i) \end{aligned}$$

ここで，op は演算子を表し $op \in \{[], |[G]|, \wedge\}$ である．

仕様とは，仕様変数を含まない仕様式 M $(Var(M) = \emptyset)$ のことである．仕様の集合を Sp と記述し，その要素を $S, T, \cdots$ により表す．また，仕様定数は定義式によって意味を与えられる仕様であり，全ての仕様定数 $A \in \mathcal{K}_\mu$ について，$A \stackrel{\text{def}}{=} S$ の形の定義式があるとする．ここで，仕様 S は再び仕様定数を含むことができ，再帰要求を表現できる．本章では，再帰定義は $A \stackrel{\text{def}}{=} a; A$ のように逐次演算子によってガード されているとし，$A \stackrel{\text{def}}{=} A [] a; S$ のような仕様は扱わない．

不安定演算子 $\triangleleft$ は不安定状態を明記するために使われる．不安定でない状態が安定状態である．安定状態は通常の LOTOS の状態に相当し，不安定演算子 $\triangleleft$ を取り除けば，μLOTOS は [47] の離接付 LOTOS と同じである．

実行可能なプロセスも μLOTOS によって記述される．プロセスは，離接，合接，不安定演算子を含まない μLOTOS の記述であり，基本的な LOTOS のプロセスである．すなわち，プロセスの集合 Pr は次のように定義される Sp の部分集合である．

$$P ::= A \ \Big| \ \mathbf{stop} \ \Big| \ \alpha; P \ \Big| \ P [] P \ \Big| \ P |[G]| P$$

ここで，$A \in \mathcal{K}'_\mu \subseteq \mathcal{K}_\mu$, $\alpha \in Act$, $G \subseteq \mathcal{N}$ である．各 $A \in \mathcal{K}'_\mu$ については，$A \stackrel{\text{def}}{=} P$ $(P \in Pr)$ の定義式があるとする．以下，プロセスを $P, Q, \cdots$ で表す．

6.3.2 意味

まず，安定状態と不安定状態を区別でき，離接遷移をもつラベル付遷移システム μLTS を定義する．

名前	仮定	⊢	結果
$\mathbf{Act}_S$		⊢	$\alpha; M \in Stb$
$\mathbf{Stop}_S$		⊢	$\mathbf{stop} \in Stb$
$\mathbf{Ch}_{S1}$	$M \in Stb$	⊢	$M \,[]\, N \in Stb$
$\mathbf{Ch}_{S2}$	$N \in Stb$	⊢	$M \,[]\, N \in Stb$
$\mathbf{Par}_{S1}$	$M \in Stb$	⊢	$M \,\|[G]\|\, N \in Stb$
$\mathbf{Par}_{S2}$	$N \in Stb$	⊢	$M \,\|[G]\|\, N \in Stb$
$\mathbf{Con}_S$	$M \in Stb$	⊢	$M \wedge N \in Stb$
$\mathbf{Rec}_S$	$A \stackrel{\text{def}}{=} S,\ S \in Stb$	⊢	$A \in Stb$

図 6.1: 安定状態集合 Stb の推論規則

定義 6.3.2 μLTS は次の 5 つ組である．

$$\langle St_1, St_2, Lb, \mapsto, \rightarrow \rangle,$$

ここで，St_1 は状態の集合，St_2 は安定状態の集合 $(St_2 \subseteq St_1)$，Lb はラベルの集合，$\mapsto \subseteq St_1 \times St_1$ は離接遷移の集合，$\rightarrow \subseteq St_1 \times Lb \times St_1$ はラベル付遷移の集合である．以下，しばしば，$(s, s') \in \mapsto$ を $s \mapsto s'$ と書き，$(s, \alpha, s') \in \rightarrow$ を $s \stackrel{\alpha}{\longrightarrow} s'$ と書く． ∎

直観的には，各状態 $s \in St_1$ に対して，$s \mapsto s'$ のようなある s' について，全てのラベル付遷移 $s' \stackrel{\alpha}{\longrightarrow} s''$ が実行できることを要求している．μLOTOS の意味は μLTS により次のように定義される．

定義 6.3.3 プロセス式の意味は次の μLTS により与えられる．

$$(Spx, Stb, Act, \mapsto, \longrightarrow)$$

ここで，安定状態の集合 Stb は図 6.1を満たす最小の集合であり，離接遷移 $\mapsto$ とラベル付遷移の集合 $\longrightarrow$ は各々，図 6.2と図 6.3を満たす最小の集合である． ∎

μLOTOS では，**有限状態**をもつ仕様式のみ扱うとする．例えば，$A \stackrel{\text{def}}{=} a; A$ は有限状態をもつが，$A_i \stackrel{\text{def}}{=} a_i; A_{i+1}$ は無限状態をもつ．

図 6.1に $\triangleleft M$ のための規則がないことから，不安定演算子 $\triangleleft$ を用いて不安定状態を記述することができる．例えば，$(\triangleleft a; \mathbf{stop} \,[]\, \triangleleft b; \mathbf{stop})$ は不安定状態である．安定性を定

名前	仮定	$\vdash$	結果
$\mathbf{Var}_\vee$		$\vdash$	$X \mapsto X$
$\mathbf{Stop}_\vee$		$\vdash$	$\mathbf{stop} \mapsto \mathbf{stop}$
$\mathbf{Act}_\vee$		$\vdash$	$\alpha; M \mapsto \alpha; M$
$\mathbf{Unst}_\vee$	$M \mapsto M'$	$\vdash$	$\lhd M \mapsto \lhd M'$
$\mathbf{Ch}_\vee$	$M \mapsto M',\ N \mapsto N'$	$\vdash$	$M \,[]\, N \mapsto M' \,[]\, N'$
$\mathbf{Par}_\vee$	$M \mapsto M',\ N \mapsto N'$	$\vdash$	$M \,\|[G]\|\, N \mapsto M' \,\|[G]\|\, N'$
$\mathbf{Con}_\vee$	$M \mapsto M',\ N \mapsto N'$	$\vdash$	$M \wedge N \mapsto M' \wedge N'$
$\mathbf{Dis}_\vee$	$M_j \mapsto M',\ j \in I$	$\vdash$	$\bigvee_{i \in I} M_i \mapsto M'$
$\mathbf{Rec}_\vee$	$S \mapsto S',\ A \stackrel{\mathrm{def}}{=} S$	$\vdash$	$A \mapsto S'$

図 6.2: 離接遷移 $\mapsto$ の推論規則

名前	仮定	$\vdash$	結果
$\mathbf{Act}$		$\vdash$	$\alpha; M \xrightarrow{\alpha} M$
$\mathbf{Unst}$	$M \xrightarrow{\alpha} M'$	$\vdash$	$\lhd M \xrightarrow{\alpha} M'$
$\mathbf{Ch}_1$	$M \xrightarrow{\alpha} M'$	$\vdash$	$M \,[]\, N \xrightarrow{\alpha} M'$
$\mathbf{Ch}_2$	$N \xrightarrow{\alpha} N'$	$\vdash$	$M \,[]\, N \xrightarrow{\alpha} N'$
$\mathbf{Par}_1$	$M \xrightarrow{\alpha} M',\ \alpha \notin G$	$\vdash$	$M \,\|[G]\|\, N \xrightarrow{\alpha} M' \,\|[G]\|\, N$
$\mathbf{Par}_2$	$N \xrightarrow{\alpha} N',\ \alpha \notin G$	$\vdash$	$M \,\|[G]\|\, N \xrightarrow{\alpha} M \,\|[G]\|\, N'$
$\mathbf{Par}_3$	$M \xrightarrow{\alpha} M',\ N \xrightarrow{\alpha} N',\ \alpha \in G$	$\vdash$	$M \,\|[G]\|\, N \xrightarrow{\alpha} M' \,\|[G]\|\, N'$
$\mathbf{Con}_1$	$M \xrightarrow{\alpha} M', N' \equiv \bigvee\{N'' : N \xrightarrow{\alpha} N''\},\ M \in Stb$	$\vdash$	$M \wedge N \xrightarrow{\alpha} N' \wedge M'$
$\mathbf{Con}_2$	$N \xrightarrow{\alpha} N', M' \equiv \bigvee\{M'' : M \xrightarrow{\alpha} M''\},\ M \in Stb$	$\vdash$	$M \wedge N \xrightarrow{\alpha} N' \wedge M'$
$\mathbf{Con}_3$	$M \xrightarrow{\alpha} M', N' \equiv \bigvee\{N'' : N \xrightarrow{\alpha} N''\},\ M \notin Stb$	$\vdash$	$M \wedge N \xrightarrow{\alpha} M' \wedge N'$
$\mathbf{Con}_4$	$N \xrightarrow{\alpha} N', M' \equiv \bigvee\{M'' : M \xrightarrow{\alpha} M''\},\ M \notin Stb$	$\vdash$	$M \wedge N \xrightarrow{\alpha} M' \wedge N'$
$\mathbf{Rec}$	$A \stackrel{\mathrm{def}}{=} S,\ S \xrightarrow{\alpha} S'$	$\vdash$	$A \xrightarrow{\alpha} S'$

図 6.3: ラベル付遷移集合 $\rightarrow$ の推論規則

義するときに注意すべきは，合接仕様 $M \wedge N$ の安定性である．その理由は，M と N が各々安定状態に到達可能でも，それらが同時に安定状態に到達できるとは限らないためである．すなわち，$M \wedge N$ の安定条件として,「M と N の両方が安定」では強す

ぎ,「M か N の一方が安定」では弱すぎる．そこで，図 6.1の規則 $\mathbf{Con}_S$ と図 6.3の規則 $\mathbf{Con}_{1,2}$ にみられるように，まず M だけの安定性を考慮し，M が安定状態に到達したら M と N を入れ換えて，次に N の安定性を考慮するようにしている．すなわち，$M \wedge N$ の安定性は，交互に M と N によって決められる．

また，規則 $\mathbf{Con}_1$ の離接仕様 $N' \equiv \bigvee\{N'' : N \xrightarrow{\alpha} N''\}$ も重要である．直観的に，規則 $\mathbf{Con}_1$ は，もし $M \xrightarrow{\alpha} M'$ ならば，$N \xrightarrow{\alpha} N'$ のような**ある** N' について，α の実行後に $N' \wedge M'$ が満たされることを要求している．次の簡単な規則 $\mathbf{Con}'_1$ と比較することによって規則 $\mathbf{Con}_1$ の必要性が理解される．

$$\mathbf{Con}'_1 \quad M \xrightarrow{\alpha} M',\ N \xrightarrow{\alpha} N',\ M \in Stb \vdash \quad M \wedge N \xrightarrow{\alpha} N' \wedge M'$$

この規則 $\mathbf{Con}'_1$ は，直観的に，もし $M \xrightarrow{\alpha} M'$ ならば，$N \xrightarrow{\alpha} N'$ のような**全ての** N' について，α の実行後に $N' \wedge M'$ が満たされることを要求している．すなわち，$\mathbf{Con}'_1$ では適切に合接演算子を定義できない．

離接遷移は図 6.2の $\vee$ のための規則 $\mathbf{Dis}_\vee$ にみられるように，離接演算子 $\vee$ を分解するために使われる．直観的に，プロセス P が仕様 S を満たすとは，$S \mapsto S'$ のようなある S' において P が S' を満たすことである．例えば，次の飲物の自動販売機の仕様 VM は，離接遷移 $VM \mapsto COF$ と $VM \mapsto TEA$ をもつので，珈琲を販売するプロセス COF か，または紅茶を販売する TEA によって実装される．

$$VM \equiv COF \vee TEA, \quad COF \equiv coin; coffee; \mathbf{stop}$$
$$TEA \equiv coin; tea; \mathbf{stop}$$

さらに，離接演算子を次のように階層的に記述できる．

$$S \stackrel{\text{def}}{=} a; \mathbf{stop} \vee b; (c; \mathbf{stop} \vee d; \mathbf{stop})$$

このとき，$S \mapsto S' \equiv b; (c; \mathbf{stop} \vee d; \mathbf{stop})$ の離接遷移によって S' を選択できる．この後は $S' \mapsto S'$ であり，c か d の選択は b のラベル付遷移 $\xrightarrow{b}$ の後に行われる．つまり，二回以上続けて離接遷移を実行する必要はない．形式的には，次の命題 6.3.1が成り立つ．ここで，Spx_0 は離接遷移によって状態を変えない仕様式の集合

$$Spx_0 = \{M : \forall M'.\ \text{if } M \mapsto M' \text{ then } M' \equiv M\}$$

である．

命題 6.3.1 $M \mapsto M'$ ならば，$M' \in Spx_0$ である．

証明 $M \mapsto M'$ を導いた規則の数に関する帰納法を用いて、$M \mapsto M' \mapsto M''$ ならば $M' \equiv M''$ を証明する．$M \mapsto M'$ を導いた最後の規則について場合わけする．

1. $\mathbf{Act}_\vee$ による場合：ある α と M_1 について，$M \equiv M' \equiv M'' \equiv \alpha; M$ である．

2. $\mathbf{Rec}_\vee$ による場合：ある S について，$M \equiv A$ かつ $S \mapsto M'$ かつ $A \stackrel{\mathrm{def}}{=} S$ である．$S \mapsto M'$ に対して帰納法の仮定より，$M' \equiv M''$ を得る．

3. $\mathbf{Unst}_\vee$ による場合：ある M_1 と M_1' について，$M_1 \mapsto M_1'$ かつ $M \equiv \triangleleft M_1$ かつ $M' \equiv \triangleleft M_1'$ である．さらに，$M' \equiv \triangleleft M_1' \mapsto M''$ より，ある M_1'' について，$M_1' \mapsto M_1''$ かつ $M'' \equiv \triangleleft M_1''$ である．ここで，帰納法の仮定より，$M_1' \equiv M_1''$ を得る．すなわち，$M' \equiv \triangleleft M_1' \equiv \triangleleft M_1'' \equiv M''$ である．

4. $\mathbf{Ch}_\vee$ による場合：ある M_1,M_2,M_1',M_1' について，$M_1 \mapsto M_1'$ かつ $M_2 \mapsto M_2'$ かつ $M \equiv M_1 [] M_2$ かつ $M' \equiv M_1' [] M_2'$ である．さらに，$M' \equiv M_1' [] M_2' \mapsto M''$ より，ある M_1'' と M_2'' について，$M_1' \mapsto M_1''$ かつ $M_2' \mapsto M_2''$ かつ $M'' \equiv M_1'' [] M_2''$ である．ここで，帰納法の仮定より，$M_1' \equiv M_1''$ かつ $M_2' \equiv M_2''$ を得る．すなわち，$M' \equiv M_1' [] M_2' \equiv M_1'' [] M_2'' \equiv M''$ である．

5. $\mathbf{Dis}_\vee$ による場合：ある M_j $(j \in I)$ について，$M_j \mapsto M'$ かつ $M \equiv \bigvee_{i \in I} M_i$ である．ここで，帰納法の仮定より $M' \equiv M''$ を得る．

他の規則による場合も同様に示せる． ∎

以下，Spx_0 の要素を $M_0, N_0, \cdots$ により表す．また，仕様変数を含まない M_0 の集合を Sp_0 (i.e. $Sp_0 = Sp \cap Spx_0$) と書き，Sp_0 の要素を $S_0, T_0, \cdots$ で表す．

6.4 充足性

本節では，μLOTOS におけるプロセス P の仕様 S に対する充足性 $P \models_\mu S$ を定義する．まず，離接付 LOTOS[47] の次の充足性 $\models$ について説明する．

定義 6.4.1 充足関係 $\models$ は次の条件を満たす最大の関係である：$P \models S$ ならば，$S \mapsto S_0$ のようなある S_0 について，全ての $\alpha \in Act$ について次の 2 つの条件が成り立つ：

(i) もし $P \xrightarrow{\alpha} P'$ ならば，ある S' について，$S_0 \xrightarrow{\alpha} S'$ かつ $P' \models S'$ である．
(ii) もし $S_0 \xrightarrow{\alpha} S'$ ならば，ある P' について，$P \xrightarrow{\alpha} P'$ かつ $P' \models S'$ である．

∎

この充足性 $\models$ は，(i) と (ii) を満たす S_0 が存在することを要求しており，これにより，一つの仕様が複数のプロセスによって満たされることが可能となる．命題 6.3.1に示したように，S_0 はこれ以上 $\mapsto$ によって分解されないが，ラベル付遷移 $S_0 \xrightarrow{\alpha} S'$ の後は，再び分解される可能性がある．それゆえ，充足性の定義は帰納的である

離接付 LOTOS の $\models$ では，仕様 S_0 を $\{S_0 : S \mapsto S_0\}$ から自由に選択できた．これに対し，μLOTOS ではこの選択を不安定演算子 $\triangleleft$ によって制御できる．重要なことは，次に定義するように，S がいつかは安定状態に到達できるように S_0 を選択しなければならないことである．

定義 6.4.2 $\mathcal{R} \subseteq Pr \times Sp$ とする．$\mathcal{R} \subseteq \theta(\mathcal{R})$ ならば，$\mathcal{R}$ は**充足部分集合**である．ここで，$\theta(\mathcal{R}) \subseteq Pr \times Sp$ は，任意の関係 $\mathcal{R} \subseteq Pr \times Sp$ について，次のように定義される．

• $(P, S) \in \theta^{(0)}(\mathcal{R})$ iff $S \mapsto S_0 \in Stb$ のようなある S_0 で，全ての $\alpha \in Act$ について，

(i) もし $P \xrightarrow{\alpha} P'$ ならば，ある S' で $S_0 \xrightarrow{\alpha} S'$ かつ $(P', S') \in \mathcal{R}$ である．
(ii) もし $S_0 \xrightarrow{\alpha} S'$ ならば，ある P' で $P \xrightarrow{\alpha} P'$ かつ $(P', S') \in \mathcal{R}$ である．

• $(P, S) \in \theta^{(n+1)}(\mathcal{R})$ iff $S \mapsto S_0 \notin Stb$ のようなある S_0 で，全ての $\alpha \in Act$ について，

(i) もし $P \xrightarrow{\alpha} P'$ ならば，ある m, S' で $S_0 \xrightarrow{\alpha} S'$, $(P', S') \in \theta^{(m)}(\mathcal{R})$, $m \leq n$ である．
(ii) もし $S_0 \xrightarrow{\alpha} S'$ ならば，ある m, P' で $P \xrightarrow{\alpha} P'$, $(P', S') \in \theta^{(m)}(\mathcal{R})$, $m \leq n$ である．

• $\theta(\mathcal{R}) = \bigcup_{n \geq 0} \theta^{(n)}(\mathcal{R})$.

∎

定義 6.4.3 ある充足部分集合 $\mathcal{R}$ において $(P, S) \in \mathcal{R}$ ならば，P は S を満たすといい，$P \models_\mu S$ と書く．また，S を満たす全てのプロセスの集合を $Proc(S)$ と書く．すなわち，$Proc(S) = \{P \in Pr : P \models_\mu S\}$ である．

∎

定義 6.4.2より，離接遷移をもたない仕様を満たすプロセスは存在しない．すなわち，$\mathtt{ff} \equiv \bigvee_{i \in \emptyset} M_i$ は満たすことができない仕様である．このとき，図 6.2の規則より，$S \,[\!]\, \mathtt{ff}$

や $S \wedge \mathtt{ff}$ も離接遷移をもたないので，満たすことができない仕様である．一方，全てのプロセスによって満たされる仕様 $\mathtt{tt}$ は仕様定数として次のように定義される．

$$\mathtt{tt} \stackrel{\mathrm{def}}{=} \bigvee\{\sum\{\alpha; \mathtt{tt} : \alpha \in \mathcal{A}\} : \mathcal{A} \subseteq Act\}$$

この仕様 $\mathtt{tt}$ は，$\mathcal{A}$ に含まれるアクションを実行することを要求する．ただし，任意のプロセス P に対して，$\mathcal{A}$ を P が今実行できるアクションの集合になるように選ぶことができるので，P は $\mathtt{tt}$ を満たすことができる．この証明は命題 6.4.4(1) で与えられる．

次に充足部分集合の例を示す．

例 6.4.1 次の仕様 SAB では，$(b; SAB)$ だけが安定状態である．

$$SAB \stackrel{\mathrm{def}}{=} \triangleleft a; SAB \vee b; SAB$$

すなわち，SAB は (a が間に実行されるかもしれないが) 常にいつかは b が実行されなければならないことを要求している．例えば，次のプロセス PR はこの要求を満たしている．

$$PR \stackrel{\mathrm{def}}{=} a; a; b; PR$$

実際に次の $\mathcal{R}$ が充足部分集合であるので，$PR \models_\mu SAB$ を得る．

$$\mathcal{R} = \{(PR, SAB), (a; b; PR, SAB), (b; PR, SAB)\}$$

ここで，$(PR, SAB) \in \theta^{(2)}(\mathcal{R})$, $(a; b; PR, SAB) \in \theta^{(1)}(\mathcal{R})$, $(b; PR, SAB) \in \theta^{(0)}(\mathcal{R})$ である．

∎

充足関係 $\models_\mu$ は最大の充足部分集合であり，次の命題 6.4.1と命題 6.4.2 が成り立つ．

命題 6.4.1 $\models_\mu = \theta(\models_\mu)$

証明 関係 $\models'_\mu$ を $\models'_\mu = \theta(\models_\mu)$ と定義する．$\models_\mu$ は充足部分集合であるので，$\models_\mu \subseteq \theta(\models_\mu)$ である．すなわち，$\models_\mu \subseteq \models'_\mu$ である．$\theta(\models_\mu)$ の定義より，$\models_\mu \subseteq \models'_\mu$ であるので，$\theta(\models_\mu) \subseteq \theta(\models'_\mu)$ を得る．よって，$\models'_\mu = \theta(\models_\mu) \subseteq \theta(\models'_\mu)$ であるので，$\models'_\mu$ は充足部分集合である．すなわち，$\models_\mu$ は最大の充足部分集合であるので，$\models'_\mu \subseteq \models_\mu$ である． ∎

命題 6.4.2 $P \models_\mu S \iff S \mapsto S_0$ のようなある S_0 で，全ての $\alpha \in Act$ について，

(i_μ) もし $P \xrightarrow{\alpha} P'$ ならば，ある S' について，$S_0 \xrightarrow{\alpha} S'$ かつ $P' \models_\mu S'$ である．
(ii_μ) もし $S_0 \xrightarrow{\alpha} S'$ ならば，ある P' について，$P \xrightarrow{\alpha} P'$ かつ $P' \models_\mu S'$ である．

証明 $(\Rightarrow)$ の場合：$P \models_\mu S$ より，ある n' で $(P,S) \in \theta^{(n')}(\models_\mu)$ である．$n'=0$ の場合，$(i_\mu),(ii_\mu)$ は明らか．$n'=n+1$ $(n \geq 0)$ の場合は，ある S_0 で，$S \mapsto S_0 \notin Stb$ かつ

(i) もし $P \xrightarrow{\alpha} P'$ ならば，ある m, S' で $S_0 \xrightarrow{\alpha} S'$, $(P',S') \in \theta^{(m)}(\models_\mu)$, $m \leq n$ である．
(ii) もし $S_0 \xrightarrow{\alpha} S'$ ならば，ある m, P' で $P \xrightarrow{\alpha} P'$, $(P',S') \in \theta^{(m)}(\models_\mu)$, $m \leq n$ である．

ここで，$(P',S') \in \theta^{(m)}(\models_\mu) \subseteq \theta(\models_\mu) = \models_\mu$ より，$(i_\mu),(ii_\mu)$ を得る．

$(\Leftarrow)$ の場合：ある S_0 について，$S \mapsto S_0$ かつ $(i_\mu),(ii_\mu)$ を仮定する．

- $S_0 \in Stb$ の場合：$(P,S) \in \theta^{(0)}(\models_\mu)$ である．
- $S_0 \notin Stb$ の場合：$(i_\mu),(ii_\mu)$ の各 $P' \models_\mu S'$ について，$(P',S') \in \theta^{(m)}(\models_\mu)$ のような m が存在する．さらに，このような全ての m に対して $m \leq n$ となる n が存在する．すなわち，$(P,S) \in \theta^{(n+1)}(\models_\mu)$ である．

よって，$(P,S) \in \theta(\models_\mu) = \models_\mu$ である． ∎

次の命題に示すように，充足性 $\models_\mu$ はプロセス演算子 (逐次，選択，並行合成) によって保存される．

命題 6.4.3 $\alpha \in Act$, $G \subseteq \mathcal{N}$ とする．また，各 $i \in \{1,2\}$ について，$P_i \in Pr$, $S_i \in Sp$, $P_i \models_\mu S_i$ とする．このとき次の関係が成り立つ．

(1) $\alpha; P_1 \models_\mu \alpha; S_1$
(2) $P_1 [] P_2 \models_\mu S_1 [] S_2$
(3) $P_1 |[G]| P_2 \models_\mu S_1 |[G]| S_2$

証明 (1) と (2) の証明は (3) よりも簡単である．ここでは (3) を証明するため，次の集合 $\mathcal{R}$ が充足部分集合であることを示す．

$$\mathcal{R} = \{(P_1 |[G]| P_2, S_1 |[G]| S_2) : P_1 \models_\mu S_1, P_2 \models_\mu S_2\}$$

$(P_1 |[G]| P_2, S_1 |[G]| S_2) \in \mathcal{R}$ とする．すなわち，各 $i \in \{1,2\}$ について，$P_i \models_\mu S_i$ であるので，ある n_i' について，$(P_i,S_i) \in \theta^{(n_i')}(\models_\mu)$ である．この $n_1'+n_2'$ に関する帰納法を用いて，$(P_1 |[G]| P_2, S_1 |[G]| S_2) \in \theta(\mathcal{R})$ を証明する．$n_1'=0$ または $n_2'=0$ の場合は，$n_1' \geq 1$ かつ $n_2' \geq 1$ の場合より簡単である．$n_1'=n_1+1$ かつ $n_2'=n_2+1$ とする

$(n_i \geq 0)$．各 $i \in \{1,2\}$ について，$(P_i, S_i) \in \theta^{(n_i+1)}(\models_\mu)$ であるので，ある S_{0i} について，$S_i \mapsto S_{0i} \notin Stb$ かつ $(P_i, S_{0i}) \in \theta^{(n_i+1)}(\models_\mu)$ を得る．$\mathbf{Par}_\vee$ より $S_1 \,||[G]||\, S_2 \mapsto S_{01} \,||[G]||\, S_{02}$ を導く．ここで，$S_{01} \notin Stb$ かつ $S_{02} \notin Stb$ であるので，$\mathbf{Par}_{S1,2}$ より，$S_{01} \,||[G]||\, S_{02} \notin Stb$ である．

(i) $P_1 \,||[G]||\, P_2 \xrightarrow{\alpha} P'_{12}$ とする．この遷移を導く最後の規則は $\mathbf{Par}_{1,2,3}$ のどれかである．ここでは $\mathbf{Par}_3$ による場合を示す．他の場合も同様である．すなわち，ある P'_1 と P'_2 について，$P_1 \xrightarrow{\alpha} P'_1$ かつ $P_2 \xrightarrow{\alpha} P'_2$ かつ $P'_{12} \equiv P'_1 \,||[G]||\, P'_2$ かつ $\alpha \in G$ である．ここで，各 $i \in \{1,2\}$ について，$(P_i, S_{0i}) \in \theta^{(n_i+1)}(\models_\mu)$ であるので，ある m_i と S'_i について，$S_{0i} \xrightarrow{\alpha} S'_i$ かつ $(P'_i, S'_i) \in \theta^{(m_i)}(\models_\mu)$ かつ $m_i \leq n_i$ である．すなわち，$\alpha \in G$ であるので，$\mathbf{Par}_3$ より，$S_{01} \,||[G]||\, S_{02} \xrightarrow{\alpha} S'_1 \,||[G]||\, S'_2$ を導く．また，各 $i \in \{1,2\}$ について，$(P'_i, S'_i) \in \theta^{(m_i)}(\models_\mu)$ かつ $m_i \leq n_i$ であるので，帰納法の仮定より，$(P'_1 \,||[G]||\, P'_2, S'_1 \,||[G]||\, S'_2) \in \theta(\mathcal{R})$ を得る．

(ii) は (i) に対称である． ∎

次の命題 6.4.4は，論理演算子が期待される性質をもつことを示している．

命題 6.4.4 $P \in Pr$, $S_i \in Sp$ とする．このとき次の関係が成り立つ．

(1) $P \models_\mu \mathtt{tt}$

(2) $P \not\models_\mu \mathtt{ff}$

(3) $P \models_\mu S_1 \wedge S_2 \iff P \models_\mu S_1$ かつ $P \models_\mu S_2$

(4) $P \models_\mu S_1 \vee S_2 \iff P \models_\mu S_1$ または $P \models_\mu S_2$

(5) $P \models_\mu \triangleleft S \iff P \models_\mu S$

証明

(1) 集合 $\mathcal{R} = \{(P, \mathtt{tt}) : P \in Pr\}$ が充足部分集合であることを示す．$(P, \mathtt{tt}) \in \mathcal{R}$ とする．ここで，$\mathcal{A}_P = \{\alpha : P \xrightarrow{\alpha}\}$ とし，$S_0 \equiv \{\alpha; \mathtt{tt} : \alpha \in \mathcal{A}_P\}$ とすると，$\mathtt{tt} \mapsto S_0$ である．このとき，P と S_0 が定義 6.4.2の条件 (i) と (ii) を満たすことは容易に証明できる．

(2) $\mathtt{ff} \not\mapsto$ であるので，定義 6.4.2の条件を満たすことはない．

(3) $(\Rightarrow)$ の場合：次の集合 $\mathcal{R}$ が充足部分集合であることを証明する．

$$\mathcal{R} = \{(P, S_1) : \exists S_2.\ P \models_\mu S_1 \wedge S_2\} \cup \{(P, S_1) : \exists S_2.\ P \models_\mu S_2 \wedge S_1\}$$

まず，次の補題 $(*1), (*2)$ を n に関する帰納法を用いて証明する．

$(*1)$ $(P, S_1 \wedge S_2) \in \theta^{(n)}(\models_\mu)$ ならば，$(P, S_1) \in \theta^{(n)}(\mathcal{R})$ である．

$(*2)$ $(P, S_2 \wedge S_1) \in \theta^{(n)}(\models_\mu)$ ならば，$(P, S_1) \in \theta(\mathcal{R})$ である．

$(*1)$ **の証明**：$(P, S_1 \wedge S_2) \in \theta^{(n')}(\models_\mu)$ とする．$n' = 0$ の場合は $n' = n + 1$ $(n \geq 0)$ の場合より簡単であるので，$n' = n + 1$ とする．すなわち，ある T_0 について，$S_1 \wedge S_2 \mapsto T_0 \notin Stb$ かつ $(P, T_0) \in \theta^{(n+1)}(\models_\mu)$ である．ここで，$\mathbf{Con}_\vee$ より，ある S_{01} と S_{02} について，$S_1 \mapsto S_{01}$ かつ $S_2 \mapsto S_{02}$ かつ $T_0 \equiv S_{01} \wedge S_{02}$ を得る．さらに，$S_{01} \wedge S_{02} \equiv T_0 \notin Stb$ であるので，$\mathbf{Con}_S$ より，$S_{01} \notin Stb$ である．

(i) $P \xrightarrow{\alpha} P'$ とする．$(P, T_0) \in \theta^{(n+1)}(\models_\mu)$ であるので，ある m と T_1' について，$T_0 \xrightarrow{\alpha} T_1'$ かつ $(P', T_1') \in \theta^{(m)}(\models_\mu)$ かつ $m \leq n$ である．$T_0 \notin Stb$ であるので，この遷移は $\mathbf{Con}_3$ か $\mathbf{Con}_4$ によって導かれる．$\mathbf{Con}_3$ による場合は $\mathbf{Con}_4$ による場合より簡単なので，$\mathbf{Con}_4$ による場合を示す．すなわち，ある S_{02}' について，$S_{02} \xrightarrow{\alpha} S_2'$ かつ $T_1' \equiv S_{11} \wedge S_2'$ である．ここで，$S_{11} \equiv \bigvee\{S_{11}' : S_{01} \xrightarrow{\alpha} S_{11}'\}$ である．

また，$(P', T_1') \in \theta^{(m)}(\models_\mu)$ であるので，ある T_0' について，$T_1' \equiv S_{11} \wedge S_2' \mapsto T_0'$ かつ $(P', T_0') \in \theta^{(m)}(\models_\mu)$ を得る．すなわち，$\mathbf{Con}_\vee$ より，ある S_{01}' と S_{02}' について，$S_{11} \mapsto S_{01}'$ かつ $S_2' \mapsto S_{02}'$ かつ $T_0' \equiv S_{01}' \wedge S_{02}'$ を得る．ここで，$S_{11} \equiv \bigvee\{S_{11}' : S_{01} \xrightarrow{\alpha} S_{11}'\} \mapsto S_{01}'$ は $\mathbf{Dis}_\vee$ によって導かれるので，ある S_1' について，$S_{01} \xrightarrow{\alpha} S_1' \mapsto S_{01}'$ でなければならない．すなわち，$S_1' \mapsto S_{01}'$ かつ $S_2' \mapsto S_{02}'$ であるので，$\mathbf{Con}_\vee$ より，$S_1' \wedge S_2' \mapsto T_0' \equiv S_{01}' \wedge S_{02}'$ である．このとき，$S_1' \wedge S_2' \mapsto T_0'$ かつ $(P', T_0') \in \theta^{(m)}(\models_\mu)$ であるので，$(P', S_1' \wedge S_2') \in \theta^{(m)}(\models_\mu)$ を得る．ここで，$m \leq n$ であるので，帰納法の仮定より，$(P', S_1') \in \theta^{(m)}(\mathcal{R})$ を得る．以上まとめて，$S_{01} \xrightarrow{\alpha} S_1'$ かつ $(P', S_1') \in \theta^{(m)}(\mathcal{R})$ かつ $m \leq n$ である．

(ii) $S_{01} \xrightarrow{\alpha} S_1'$ とする．このとき，上記の (i) の場合より簡単に，$P \xrightarrow{\alpha} P'$ かつ $(P', S_1') \in \theta^{(m)}(\mathcal{R})$ かつ $m \leq n$ を得る．

よって $(P, S_1) \in \theta^{(n+1)}(\mathcal{R})$ が得られた．補題 $(*1)$ の証明完了．

$(*2)$ **の証明**：$S_2 \wedge S_1$ がいつかは安定状態に到達して $S_1' \wedge S_2'$ になることと，上記の補題 $(*1)$ を利用する．$(P, S_2 \wedge S_1) \in \theta^{(n')}(\models_\mu)$ とする．$n' = 0$ の場合が重要で

あり，$n' \geq 1$ の場合は $(*1)$ の証明に同様であるので，$(P, S_1 \wedge S_2) \in \theta^{(0)}(\models_\mu)$ とする．すなわち，ある T_0 について，$S_1 \wedge S_2 \mapsto T_0 \in Stb$ かつ $(P, T_0) \in \theta^{(0)}(\models_\mu)$ である．ここで，$\mathbf{Con}_\vee$ より，ある S_{01} と S_{02} について，$S_1 \mapsto S_{01}$ かつ $S_2 \mapsto S_{02}$ かつ $T_0 \equiv S_{02} \wedge S_{01}$ を得る．さらに，$S_{02} \wedge S_{01} \equiv T_0 \in Stb$ であるので，$\mathbf{Con}_S$ より $S_{02} \in Stb$ である．

(*i*) $P \xrightarrow{\alpha} P'$ とする．$(P, T_0) \in \theta^{(0)}(\models_\mu)$ であるので，ある T_1' について，$T_0 \xrightarrow{\alpha} T_1'$ かつ $P' \models_\mu T_1'$ である．$T_0 \in Stb$ であるので，この遷移は $\mathbf{Con}_1$ か $\mathbf{Con}_2$ によって導かれる．$\mathbf{Con}_1$ による場合は上記補題 $(*1)$ の $\mathbf{Con}_4$ による場合に同様である．ここでは $\mathbf{Con}_2$ による場合のみ示す．すなわち，ある S_{01}' について，$S_{01} \xrightarrow{\alpha} S_1'$ かつ $T_1' \equiv S_1' \wedge S_{21}$ である．ここで，$S_{21} \equiv \bigvee\{S_{21}' : S_{02} \xrightarrow{\alpha} S_{21}'\}$ である．すなわち，$P' \models_\mu T_1' \equiv S_1' \wedge S_{21}$ であるので，ある m について，$(P', S_1' \wedge S_{21}) \in \theta^{(m)}(\models_\mu)$ である．すなわち，上記の補題 $(*1)$ より，$(P', S_1') \in \theta^{(m)}(\mathcal{R})$ である．以上まとめて，$S_{01} \xrightarrow{\alpha} S_1'$ かつ $(P', S_1') \in \theta(\mathcal{R})$ である．

(*ii*) $S_{01} \xrightarrow{\alpha} S_1'$ とする．このとき，上記の (i) の場合より簡単に，$P \xrightarrow{\alpha} P'$ かつ $(P', S_1') \in \theta(\mathcal{R})$ を得る．

よって $(P, S_1) \in \theta(\mathcal{R})$ が得られた．補題 $(*2)$ の証明完了．

次に，$\mathcal{R}$ が充足部分集合であることを示す．$(P, S_1) \in \mathcal{R}$ とする．すなわち，ある S_2 について，$P \models_\mu S_1 \wedge S_2$ または $P \models_\mu S_2 \wedge S_1$ である．さらに，ある n について，$(P, S_1 \wedge S_2) \in \theta^{(n)}(\models_\mu)$ または $(P, S_2 \wedge S_1) \in \theta^{(n)}(\models_\mu)$ である．よって，上記の補題 $(*1), (*2)$ より，$(P, S_1) \in \theta(\mathcal{R})$ を得る．

$(\Leftarrow)$ の場合：次の集合 $\mathcal{R}$ が充足部分集合であることを証明する．

$$\mathcal{R} = \{(P, S_1 \wedge S_2) : P \models_\mu S_1, P \models_\mu S_2\}$$

この証明は上記の $(\Rightarrow)$ の場合より簡単である．

(4) $(\Rightarrow)$ の場合：$P \models_\mu S_1 \vee S_2$ とする．このとき，ある S_0 について，$S_1 \vee S_2 \mapsto S_0$ かつ $P \models_\mu S_0$ である．ここで，$\mathbf{Dis}_\vee$ により，$S_1 \mapsto S_0$ または $S_2 \mapsto S_0$ を得る．すなわち，$P \models_\mu S_1$ または $P \models_\mu S_2$ を得る．

($\Leftarrow$) の場合:$P \models_\mu S_1$ とする．$\mathbf{Dis}_\vee$ より，ある S_0 について，$S_1 \mapsto S_0$ かつ $P \models_\mu S_0$ である．ここで，$\mathbf{Dis}_\vee$ により，$S_1 \vee S_2 \mapsto S_0$ を導く．すなわち，$P \models_\mu S_1 \vee S_2$ である．

(5) 規則 $\mathbf{Unst}_\vee$ より，$\triangleleft S \mapsto \triangleleft S_0$ ならば，そのときに限り $S_0 \mapsto S_0$ である．また，規則 **Unst** より，$\triangleleft S_0 \xrightarrow{\alpha} S'$ ならば，そのときに限り $S_0 \xrightarrow{\alpha} S'$ である．よって命題 6.4.2より，関係 $(P \models_\mu \triangleleft S \iff P \models_\mu S)$ が得られる．

∎

命題 6.4.4の適用例を次に示す．

例 6.4.2 二つ仕様 *SAB* と *SBA* を次のように対称に定義する．

$$SAB \stackrel{\text{def}}{=} \triangleleft a; SAB \vee b; SAB,$$
$$SBA \stackrel{\text{def}}{=} a; SBA \vee \triangleleft b; SBA$$

仕様 *SAB* は例 6.4.1で使われた仕様であり，*SBA* は (b が間に実行されるかもしれないが) 常にいつかは a が実行されなければならないことを要求している．すなわち，仕様 $SAB \wedge SBA$ は，常にいつかは a と b が実行されなければならないことを要求している．すなわち，例 6.4.1で与えたプロセス ($PR \stackrel{\text{def}}{=} a; a; b; PR$) は *SBA* も満たしている ($PR \models_\mu SAB,\ PR \models_\mu SBA$)．よって，命題 6.4.4より，次の関係を得る．

$$PR \models_\mu SAB \wedge SBA$$

仕様 $SAB \wedge SBA$ の状態遷移図を図 6.4に示す．ff は満たすことができないので，この図の点線で書かれた四角の外側に出ている遷移は無視できる．図中，◯ は安定状態を表している．この図は，a と b がいつかは実行されることが交互に要求されることを示している．

∎

次に，仕様上に半順序 $\sqsubseteq_\mu$ と同値関係 $\cong_\mu$ を定義する．この半順序 $S \sqsubseteq_\mu T$ は，もしあるプロセスが S を満たすならば，それは T も満たすことを表している．すなわち，S は T の詳細化された仕様である．この $\sqsubseteq_\mu$ を**詳細順序**と呼ぶ．

定義 6.4.4 $S, T \in Sp$ とする．

(1) $S \sqsubseteq_\mu T \iff Proc(S) \subseteq Proc(T)$
(2) $S \cong_\mu T \iff Proc(S) = Proc(T)$

∎

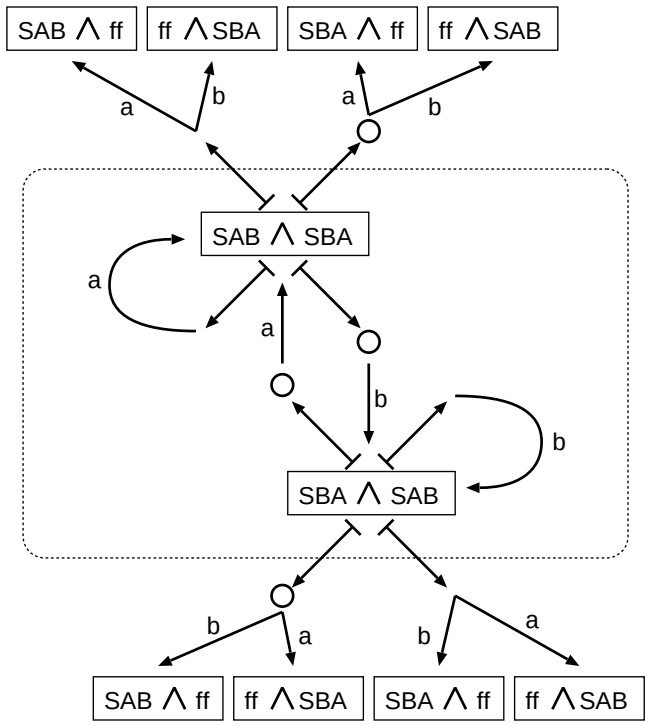

図 6.4: 仕様 *SAB* ∧ *SBA* の状態遷移図 (◯ : 安定状態)

次の命題は，詳細順序 $\sqsubseteq_\mu$ が逐次的な演算子 (逐次，選択，合接，離接，不安定) によって保存されることを示している．このとき，$S \sqsubseteq_\mu T$ かつ $T \sqsubseteq_\mu S$ ならば $S \cong_\mu T$ であるので，$\cong_\mu$ も逐次的な演算子によって保存される．

命題 6.4.5 $S_1 \sqsubseteq_\mu T_1$ かつ $S_2 \sqsubseteq_\mu T_2$ とする．このとき，次の関係が成り立つ．

(1) $\alpha; S_1 \sqsubseteq_\mu \alpha; T_1$
(2) $S_1 [] S_2 \sqsubseteq_\mu T_1 [] T_2$
(3) $\mathtt{ff} \sqsubseteq_\mu S_1 \sqsubseteq_\mu \mathtt{tt}$
(4) $S_1 \wedge S_2 \sqsubseteq_\mu T_1 \wedge T_2$
(5) $S_1 \vee S_2 \sqsubseteq_\mu T_1 \vee T_2$
(6) $\triangleleft S_1 \sqsubseteq_\mu \triangleleft T_1$

証明 (1) は簡単である．

(2) $P \models_\mu S_1 [] S_2$ とする．すなわち，$\mathbf{Ch}_\vee$ より，ある S_{01} と S_{02} について，$P \models_\mu S_{01} [] S_{02}$ かつ $S_1 \mapsto S_{01}$ かつ $S_2 \mapsto S_{02}$ である．ここで，各 $i \in \{1, 2\}$ について，P_i を

次のように与える．

$$P_i \equiv \sum\{\alpha; P' : \exists S'.\ S_{0i} \xrightarrow{\alpha} S', P \xrightarrow{\alpha} P', P' \models_\mu S'\}$$

このとき，$P \sim P_1 [] P_2$ かつ $P_1 \models_\mu S_1$ かつ $P_2 \models_\mu S_2$ を次のように証明できる．ここで，$\sim$ は強等価 (定義 2.4.2) である．

- $P \sim P_1 [] P_2$ の証明：(i) $P \xrightarrow{\alpha} P'$ とする．$P \models_\mu S_{01} [] S_{02}$ と **Ch** より，ある $i \in \{1,2\}$ と S' について，$S_{0i} \xrightarrow{\alpha} S'$ かつ $P' \models_\mu S'$ を得る．すなわち，**Ch** と **Act** より，$P_i \equiv \sum\{\alpha'; P'' : \exists S''.\ S_{0i} \xrightarrow{\alpha'} S'', P \xrightarrow{\alpha'} P'', P'' \models_\mu S''\} \xrightarrow{\alpha} P'$ を導く．さらに，**Ch** より，$P_1 [] P_2 \xrightarrow{\alpha} P'$ を導ける．(ii) については明らかである．

- $P_1 \models_\mu S_1$ の証明：(i) $P_1 \equiv \sum\{\alpha'; P'' : \exists S''.\ S_{0i} \xrightarrow{\alpha'} S'', P \xrightarrow{\alpha'} P'', P'' \models_\mu S''\} \xrightarrow{\alpha} P'$ とする．この遷移は **Ch** と **Act** より導かれるので，ある S' について，$S_{01} \xrightarrow{\alpha} S'$ かつ $P \xrightarrow{\alpha} P'$ かつ $P' \models_\mu S'$ を得る．

  (ii) $S_{01} \xrightarrow{\alpha} S'$ とする．**Ch** より $S_{01} [] S_{02} \xrightarrow{\alpha} S'$ を導く．ここで，$P \models_\mu S_{01} [] S_{02}$ より，ある P' について，$P \xrightarrow{\alpha} P'$ かつ $P' \models_\mu S'$ を得る．すなわち，**Ch** と **Act** より，$P_1 \equiv \sum\{\alpha'; P'' : \exists S''.\ S_{0i} \xrightarrow{\alpha'} S'', P \xrightarrow{\alpha'} P'', P'' \models_\mu S''\} \xrightarrow{\alpha} P'$ を導く．

ここで，$P_1 \models_\mu S_1 \sqsubseteq_\mu T_1$ かつ $P_2 \models_\mu S_2 \sqsubseteq_\mu T_2$ より，$P_1 \models_\mu T_1$ かつ $P_2 \models_\mu T_2$ を得る．すなわち，命題 6.4.3(2) より，$P_1 [] P_2 \models_\mu T_1 [] T_2$ を得る．さらに，$P \sim P_1 [] P_2$ より，$P \models_\mu T_1 [] T_2$ は容易に得られる．

$(3), (4), (5), (6)$ は命題 6.4.4を用いて容易に証明できる．例として (3) の証明を与える．

$$Proc(S_1 \wedge S_2) = Proc(S_1) \cap Proc(S_2) \subseteq Proc(T_1) \cap Proc(T_2) = Proc(T_1 \wedge T_2).$$

∎

6.5 不動点

本節で不動点に対する性質を明らかにする．等式 $X \cong_\mu M$ の解は，X に代入したときに各々を満たすプロセスの集合が同じ $(S \cong_\mu M\{S/X\})$ になる仕様 S である．命題 2.4.16や命題 4.5.9に示したように，観測合同や監視等価は唯一解をもつが，μLOTOS

の同値関係 $\cong_\mu$ の解は唯一には定まらない．例えば，次の 2 つの仕様 S_{max}, S_{min} は共に等式 $X \cong_\mu a; X \vee b; \mathbf{stop}$ の解である．

$$S_{max} \stackrel{\text{def}}{=} a; S_{max} \vee b; \mathbf{stop}$$
$$S_{min} \stackrel{\text{def}}{=} \lhd a; S_{min} \vee \lhd b; \mathbf{stop}$$

このとき，例えば，プロセス $A \stackrel{\text{def}}{=} a; A$ は S_{max} を満たすが，S_{min} を満たさない．すなわち，$S_{max} \ncong_\mu S_{min}$ である．

このように複数存在する解のなかで，($\sqsubseteq_\mu$ について) 最大の解が**最大不動点**であり，最小の解が**最小不動点**である．これらは，次のように定義される．ここで，$M\{N/X\}$ は M の変数 X に仕様式 N を代入して得られる仕様式である．

定義 6.5.1 $Var(M) \subseteq \{X\}$ かつ $T \cong_\mu M\{T/X\}$ とする．

1. T は $X \cong_\mu M$ の最大不動点である

   $\iff$ $S \cong_\mu M\{S/X\}$ のような全ての S について，$S \sqsubseteq_\mu T$ である．

2. T は $X \cong_\mu M$ の最小不動点である

   $\iff$ $S \cong_\mu M\{S/X\}$ のような全ての S について，$T \sqsubseteq_\mu S$ である．

∎

以下，逐次仕様を対象として最大不動点と最小不動点が μLOTOS においてどのように表現されるかを述べる．まず，CCS の逐次性 (定義 2.4.13) とガード (定義 2.4.5) の概念を修正して μLOTOS に対して定義する．

定義 6.5.2 $X \in \mathcal{X}_\mu$，$M \in Spx$ とする．

1. X は M において逐次的である $\iff$ $X \in Var(M')$ のような M の全ての部分式 M' について，M' が，$\alpha; M''$ か，$\lhd M''$ か，$M''_1 \,[]\, M''_2$ か，$\bigvee_{i \in I} M''_i$ か，または X の形をもつ．

2. X は M においてガードされている $\iff$ プロセス式 M に含まれる全ての変数 X が M のある部分式 $\alpha.M'$ に含まれる．

∎

例えば，仕様式 $(a;X) \vee (c;\mathbf{stop}\,|[\emptyset]|\,d;\mathbf{stop})\,[]\,(i;X)$ において，変数 X は逐次的であり，ガードされている．尚，逐次性の定義に $M_1 \wedge M_2$ が含まれることが期待されるが，$M_1 \wedge M_2$ を含めて最大不動点と最小不動点を表現することは非常に難しく，本節では変数をもつ $M_1 \wedge M_2$ は省略して議論する．

次に，最大不動点と最小不動点の表現に必要な仕様上の関数 f_ν と f_μ を定義する．

定義 6.5.3 仕様式上の関数 $f_\nu : Spx \to Spx$ と $f_\mu : Spx \to Spx$ を M の構造に対して帰納的に次のように与える．

$$f_\nu(M) \equiv \begin{cases} \mathbf{stop} & (M \equiv \mathbf{stop}) \\ f_\nu(S) & (M \equiv A, A \stackrel{\text{def}}{=} S) \\ X & (M \equiv X) \\ \alpha; f_\nu(M') & (M \equiv \alpha; M', Var(M') \neq \emptyset) \\ \alpha; M' & (M \equiv \alpha; M', Var(M') = \emptyset) \\ f_\nu(M') & (M \equiv \triangleleft M') \\ f_\nu(M_1)\,[]\,f_\nu(M_2) & (M \equiv M_1\,[]\,M_2) \\ M_1\ op\ M_2 & (M \equiv M_1\ op\ M_2) \\ \bigvee_{i\in I} f_\nu(M_i) & (M \equiv \bigvee_{i\in I} M_i) \end{cases}$$

$$f_\mu(M) \equiv \begin{cases} \triangleleft\mathbf{stop} & (M \equiv \mathbf{stop}) \\ f_\mu(S) & (M \equiv A, A \stackrel{\text{def}}{=} S) \\ X & (M \equiv X) \\ \triangleleft\,\alpha; f_\mu(M') & (M \equiv \alpha; M', Var(M') \neq \emptyset) \\ \triangleleft\,\alpha; M' & (M \equiv \alpha; M', Var(M') = \emptyset) \\ \triangleleft\, f_\mu(M') & (M \equiv \triangleleft M') \\ f_\mu(M_1)\,[]\,f_\mu(M_2) & (M \equiv M_1\,[]\,M_2) \\ \triangleleft\,(M_1\ op\ M_2) & (M \equiv M_1\ op\ M_2) \\ \bigvee_{i\in I} f_\mu(M_i) & (M \equiv \bigvee_{i\in I} M_i) \end{cases}$$

ここで，op は演算子を表し $op \in \{|[G]|, \wedge\}$ である． ∎

これら関数 f_ν, f_μ の性質を明らかにするために，補題 6.5.1と補題 6.5.2と補題 6.5.3を与える．補題 6.5.1は，変数を含む仕様式の状態を関数 f_ν と f_μ が各々安定状態と不安定状態にすることを表している．また，補題 6.5.2と補題 6.5.3は，関数 f_ν と f_μ が離接遷移とラベル付遷移によって維持されることを示している．

補題 6.5.1 $M_0 \in Spx_0$，$Var(M_0) \subseteq \{X\}$，かつ X は M_0 において逐次的であり，ガードされているとする．

(1) $X \in Var(M_0)$ ならば，$f_\nu(M_0) \in Stb$ である．
(2) $f_\mu(M_0) \notin Stb$ である．

証明

(1) $X \in Var(M_0)$ とし，M_0 の構造に関する帰納法を用いて証明する．X は M_0 において逐次的であり，ガードされているので，$M_0 \equiv X$ と $M_0 \equiv M_{01} \,||[G]||\, M_{02}$ と $M_0 \equiv M_{01} \wedge M_{02}$ の場合はない．

- $M_0 \equiv \alpha; M'$ かつ $Var(M') \neq \emptyset$ の場合：$\mathbf{Act}_S$ より $f_\nu(\alpha; M') \equiv \alpha; f_\nu(M') \in Stb$ である．
- $M_0 \equiv \alpha; M'$ かつ $Var(M') = \emptyset$ の場合：$\mathbf{Act}_S$ より $f_\nu(\alpha; M') \equiv \alpha; M' \in Stb$ である．
- $M_0 \equiv \triangleleft M_0'$ の場合：帰納法の仮定より $f_\nu(\triangleleft M_0') \equiv f_\nu(M_0') \in Stb$ を得る．
- $M_0 \equiv M_{01} \,[]\, M_{02}$ の場合：$X \in Var(M_0) = Var(M_{01}) \cup Var(M_{02})$ より，$X \in Var(M_{01})$ か $X \in Var(M_{02})$ である．
  - $X \in Var(M_{01})$ の場合：帰納法の仮定より $f_\nu(M_{01}) \in Stb$ を得る．すなわち，$\mathbf{Ch}_{S1}$ より $f_\nu(M_{01} \,[]\, M_{02}) \equiv f_\nu(M_{01}) \,[]\, f_\nu(M_{02}) \in Stb$ である．
  - $X \in Var(M_{02})$ の場合：上記の場合に対称である．

(2) M_0 の構造に関する帰納法を用いて証明する．

- $M_0 \equiv \mathbf{stop}$, $\alpha; M'$, $\triangleleft M_0'$ の場合は明らかに $f_\mu(M_0) \notin Stb$ である．
- $M_0 \equiv A$ かつ $A \stackrel{\text{def}}{=} S$ の場合：$f_\mu(M_0) \equiv f_\mu(S)$ である．帰納法の仮定より，$f_\mu(S) \notin Stb$ である．
- $M_0 \equiv M_{01} \,[]\, M_{02}$ の場合：$f_\mu(M_{01} \,[]\, M_{02}) \equiv f_\mu(M_{01}) \,[]\, f_\mu(M_{02})$ である．帰納法の仮定より，$f_\mu(M_{01}) \notin Stb$ かつ $f_\mu(M_{02}) \notin Stb$ である．すなわち，$\mathbf{Ch}_{S1,2}$ より $f_\nu(M_{01}) \,[]\, f_\nu(M_{02}) \notin Stb$ である．

∎

補題 6.5.2 $M \in Spx$，$Var(M) \subseteq \{X\}$，かつ X は M において逐次的であるとする．

(1) $f_\nu(M) \mapsto N_0$ ならば，ある M_0 について，$N_0 \equiv f_\nu(M_0)$ である．

(2) $f_\mu(M) \mapsto N_0$ ならば，ある M_0 について，$N_0 \equiv f_\mu(M_0)$ である．

証明

(1) M の構造に関する帰納法を用いる．

- $M \equiv \mathbf{stop}$ の場合：$\mathbf{Stop}_\vee$ より $f_\nu(\mathbf{stop}) \equiv \mathbf{stop} \mapsto \mathbf{stop} \equiv N_0 \equiv f_\nu(\mathbf{stop})$ である．
- $M \equiv A$ かつ $A \stackrel{\text{def}}{=} S$ の場合：$f_\nu(A) \equiv f_\nu(S) \mapsto N_0$ である．帰納法の仮定より，ある M_0 について，$N_0 \equiv f_\nu(M_0)$ を得る．
- $M \equiv X$ の場合：$\mathbf{Var}_\vee$ より $f_\nu(X) \equiv X \mapsto X \equiv N_0 \equiv f_\nu(X)$ である．
- $M \equiv \alpha; M'$ かつ $Var(M') \neq \emptyset$ の場合：$\mathbf{Act}_\vee$ より $f_\nu(\alpha; M') \equiv \alpha; f_\nu(M') \mapsto \alpha; f_\nu(M') \equiv N_0 \equiv f_\nu(\alpha; M')$ を得る．
- $M \equiv \alpha; M'$ かつ $Var(M') = \emptyset$ の場合：$\mathbf{Act}_\vee$ より $f_\nu(\alpha; M') \equiv \alpha; M' \mapsto \alpha; M' \equiv N_0 \equiv f_\nu(\alpha; M')$ を得る．
- $M \equiv \triangleleft M'$ の場合：$f_\nu(\triangleleft M') \equiv f_\nu(M') \mapsto N_0$ である．帰納法の仮定より，ある M_0 について，$N_0 \equiv f_\nu(M_0)$ を得る．
- $M \equiv M_1 \,[]\, M_1$ の場合：$f_\nu(M_1 \,[]\, M_2) \equiv f_\nu(M_1) \,[]\, f_\nu(M_2) \mapsto N_0$ である．$\mathbf{Ch}_\vee$ より，ある N_{01} と N_{02} について，$f_\nu(M_1) \mapsto N_{01}$ かつ $f_\nu(M_2) \mapsto N_{02}$ かつ $N_0 \equiv N_{01} \,[]\, N_{02}$ である．帰納法の仮定より，各 $i \in \{1,2\}$ について，ある M_{0i} について，$N_{0i} \equiv f_\nu(M_{0i})$ を得る．すなわち，$N_0 \equiv N_{01} \,[]\, N_{02} \equiv f_\nu(M_{01}) \,[]\, f_\nu(M_{02}) \equiv f_\nu(M_{01} \,[]\, M_{02})$ である．
- $M \equiv M_1 \,|[G]|\, M_1$ の場合：$f_\nu(M_1 \,|[G]|\, M_2) \equiv M_1 \,|[G]|\, M_2 \mapsto N_0$ である．$\mathbf{Par}_\vee$ より，ある N_{01} と N_{02} について，$M_1 \mapsto N_{01}$ かつ $M_2 \mapsto N_{02}$ かつ $N_0 \equiv N_{01} \,|[G]|\, N_{02}$ である．すなわち，$N_0 \equiv N_{01} \,|[G]|\, N_{02} \equiv f_\nu(N_{01} \,|[G]|\, N_{02})$ である．
- $M \equiv M_1 \wedge M_1$ の場合：$M_1 \,|[G]|\, M_1$ の場合に同様である．
- $M \equiv \bigvee_{i \in I} M_i$ の場合：$f_\nu(\bigvee_{i \in I} M_i) \equiv \bigvee_{i \in I} f_\nu(M_i) \mapsto N_0$ である．$\mathbf{Dis}_\vee$ より，ある $j \in I$ について，$f_\nu(M_j) \mapsto N_0$ である．帰納法の仮定より，ある M_0 について，$N_0 \equiv f_\nu(M_0)$ を得る．

(2) M の構造に関する帰納法を用いる．

- $M \equiv \mathbf{stop}$ の場合：$\mathbf{Unst}_\vee$ と $\mathbf{Stop}_\vee$ より $f_\nu(\mathbf{stop}) \equiv \triangleleft\mathbf{stop} \mapsto \triangleleft\mathbf{stop} \equiv N_0 \equiv f_\nu(\mathbf{stop})$ である．
- $M \equiv A$ かつ $A \stackrel{\mathrm{def}}{=} S$ の場合：$f_\mu(A) \equiv f_\mu(S) \mapsto N_0$ である．帰納法の仮定より，ある M_0 について，$N_0 \equiv f_\mu(M_0)$ を得る．
- $M \equiv X$ の場合：$\mathbf{Var}_\vee$ より $f_\nu(X) \equiv X \mapsto X \equiv N_0 \equiv f_\nu(X)$ である．
- $M \equiv \alpha; M'$ かつ $Var(M') \neq \emptyset$ の場合：$\mathbf{Unst}_\vee$ と $\mathbf{Act}_\vee$ より $f_\nu(\alpha; M') \equiv \triangleleft\alpha; f_\nu(M') \mapsto \triangleleft\alpha; f_\nu(M') \equiv N_0 \equiv f_\nu(\alpha; M')$ を得る．
- $M \equiv \alpha; M'$ かつ $Var(M') = \emptyset$ の場合：$\mathbf{Unst}_\vee$ と $\mathbf{Act}_\vee$ より $f_\nu(\alpha; M') \equiv \triangleleft\alpha; M' \mapsto \triangleleft\alpha; M' \equiv N_0 \equiv f_\nu(\alpha; M')$ を得る．
- $M \equiv \triangleleft M'$ の場合：$f_\nu(\triangleleft M') \equiv \triangleleft f_\nu(M') \mapsto N_0$ である．$\mathbf{Unst}_\vee$ より，ある N_0' で，$f_\nu(M') \mapsto N_0'$ かつ $N_0 \equiv \triangleleft N_0'$ である．帰納法の仮定より，ある M_0' で，$N_0' \equiv f_\nu(M_0')$ を得る．すなわち，$N_0 \equiv \triangleleft N_0' \equiv \triangleleft f_\nu(M_0') \equiv f_\nu(\triangleleft M_0')$ である．
- $M \equiv M_1 \,[]\, M_2$ の場合：上記の (1) の $M \equiv M_1 \,[]\, M_1$ の場合に同様である．
- $M \equiv M_1 \,|[G]|\, M_2$ の場合：$f_\nu(M_1 \,|[G]|\, M_2) \equiv \triangleleft(M_1 \,|[G]|\, M_2) \mapsto N_0$ である．$\mathbf{Unst}_\vee$ と $\mathbf{Par}_\vee$ より，ある N_{01} と N_{02} について，$M_1 \mapsto N_{01}$ かつ $M_2 \mapsto N_{02}$ かつ $N_0 \equiv \triangleleft(N_{01} \,|[G]|\, N_{02})$ である．すなわち，$N_0 \equiv \triangleleft(N_{01} \,|[G]|\, N_{02}) \equiv f_\nu(N_{01} \,|[G]|\, N_{02})$ である．
- $M \equiv M_1 \wedge M_2$ の場合：$M_1 \,|[G]|\, M_2$ の場合に同様である．
- $M \equiv \bigvee_{i \in I} M_i$ の場合：上記の (1) の $M \equiv \bigvee_{i \in I} M_i$ の場合に同様である．

∎

補題 6.5.3 $M_0 \in Spx_0$，$Var(M_0) \subseteq \{X\}$，かつ X は M_0 において逐次的であるとする．

(1) $f_\nu(M_0) \stackrel{\alpha}{\longrightarrow} N'$ かつ $Var(N') \neq \emptyset$ ならば，ある M' について $N' \equiv f_\nu(M')$ である．

(2) $f_\mu(M_0) \stackrel{\alpha}{\longrightarrow} N'$ かつ $Var(N') \neq \emptyset$ ならば，ある M' について $N' \equiv f_\mu(M')$ である．

証明

(1) $f_\nu(M_0) \stackrel{\alpha}{\longrightarrow} N'$ かつ $Var(N') \neq \emptyset$ とする．このとき，$Var(M_0) \neq \emptyset$ である．M_0 の構造に関する帰納法を用いる．X は M_0 において逐次的であるので，次の 3 つの場合が考えられる．

- $M_0 \equiv \alpha'; M''$ かつ $Var(M'') \neq \emptyset$ の場合：$f_\nu(\alpha'; M') \equiv \alpha'; f_\nu(M'') \xrightarrow{\alpha} N'$ である．**Act** より，$\alpha = \alpha'$ かつ $N' \equiv f_\nu(M'')$ を得る．
- $M_0 \equiv \triangleleft M_0'$ の場合：$f_\nu(\triangleleft M_0') \equiv f_\nu(M_0') \xrightarrow{\alpha} N'$ である．帰納法の仮定より，ある M' について，$N' \equiv f_\nu(M')$ を得る．
- $M_0 \equiv M_{01} [] M_{01}$ の場合：$f_\nu(M_{01} [] M_{02}) \equiv f_\nu(M_{01}) [] f_\nu(M_{02}) \xrightarrow{\alpha} N'$ である．この遷移を導く最後の規則は $\mathbf{Ch}_1$ か $\mathbf{Ch}_2$ である．対称であるので $\mathbf{Ch}_1$ による場合を示す．すなわち，$f_\nu(M_{01}) \xrightarrow{\alpha} N'$ である．よって，帰納法の仮定より，ある M' について，$N' \equiv f_\nu(M')$ を得る．

(2) $f_\mu(M_0) \xrightarrow{\alpha} N'$ かつ $Var(N') \neq \emptyset$ とする．このとき，$Var(M_0) \neq \emptyset$ である．M_0 の構造に関する帰納法を用いる．

- $M_0 \equiv \alpha'; M''$ かつ $Var(M'') \neq \emptyset$ の場合：$f_\mu(\alpha'; M') \equiv \triangleleft\alpha'; f_\nu(M'') \xrightarrow{\alpha} N'$ である．**Unst** と **Act** より，$\alpha = \alpha'$ かつ $N' \equiv f_\nu(M'')$ を得る．
- $M_0 \equiv \triangleleft M_0'$ の場合：$f_\nu(\triangleleft M_0') \equiv \triangleleft f_\nu(M_0') \xrightarrow{\alpha} N'$ である．ここで，**Unst** より，$f_\nu(M_0') \xrightarrow{\alpha} N'$ を得る．すなわち，帰納法の仮定より，ある M' について，$N' \equiv f_\nu(M')$ を得る．
- $M_0 \equiv M_{01} [] M_{01}$ の場合：(1) の $M_{01} [] M_{01}$ の場合と同様である．

∎

また，次の命題 6.5.4と命題 6.5.5は，関数 f_ν と f_μ の不動点に関する性質を表している．直観的に，命題 6.5.4は関数 f_ν と f_μ による変換が等式の解を保存することを保証し，命題 6.5.5 は f_ν によって定義されたループは無限であり，f_μ によって定義されたループは有限であることを示している．

命題 6.5.4 任意の $S \in Sp$ について，次の等式が成り立つ．

(1) $M\{S/X\} \cong_\mu f_\nu(M)\{S/X\}$
(2) $M\{S/X\} \cong_\mu f_\mu(M)\{S/X\}$

証明

(1) M の構造に関する帰納法を用いて証明する．

- $M \equiv \mathbf{stop}, X, \alpha; M'\ (Var(M') = \emptyset), M_1 \,|[G]|\, M_2, M_1 \wedge M_2$ の場合：$f_\nu(M) \equiv M$ より明らか．
- $M \equiv \alpha; M'$ かつ $Var(M') \neq \emptyset$ の場合：$f_\nu(M) \equiv f_\nu(\alpha; M') \equiv \alpha; f_\nu(M')$ である．帰納法の仮定より，$M'\{S/X\} \cong_\mu f_\nu(M')\{S/X\}$ を得る．さらに，命題 6.4.5(1) より，$M\{S/X\} \equiv \alpha; M'\{S/X\} \cong_\mu \alpha; f_\nu(M')\{S/X\} \equiv f_\nu(M)\{S/X\}$ を得る．
- $M \equiv \triangleleft M'$ の場合：$f_\nu(M) \equiv f_\nu(\triangleleft M') \equiv f_\nu(M')$ である．帰納法の仮定より，$M'\{S/X\} \cong_\mu f_\nu(M')\{S/X\}$ を得る．さらに，命題 6.4.4(5) より，$M\{S/X\} \equiv \triangleleft M'\{S/X\} \cong_\mu f_\nu(M')\{S/X\} \equiv f_\nu(M)\{S/X\}$ を得る．
- $M \equiv M_1 \,[]\, M_2$ の場合：$f_\nu(M) \equiv f_\nu(M_1 \,[]\, M_2) \equiv f_\nu(M_1) \,[]\, f_\nu(M_2)$ である．帰納法の仮定より，各 $i \in \{1,2\}$ について $M_i\{S/X\} \cong_\mu f_\nu(M_i)\{S/X\}$ を得る．さらに，命題 6.4.5(2) より，$M\{S/X\} \equiv M_1\{S/X\} \,[]\, M_2\{S/X\} \cong_\mu f_\nu(M_1)\{S/X\} \,[]\, f_\nu(M_2)\{S/X\} \equiv f_\nu(M)\{S/X\}$ を得る．
- $M \equiv \bigvee_{i\in I} M_i$ の場合：上記の $M_1 \,[]\, M_2$ の場合と同様に命題 6.4.5(5) を用いる．

(2) 上記の (1) の場合と同様に帰納法を用いる．f_μ では，$\alpha; M', M_1 \,|[G]|\, M_2, M_1 \wedge M_2$ の場合に不安定演算子 $\triangleleft$ が追加されるが，これは命題 6.4.4(5) により無視できる．例として，$M \equiv \alpha; M'$ かつ $Var(M') = \emptyset$ の場合を示す．このとき，$f_\mu(M) \equiv f_\mu(\alpha; M') \equiv \triangleleft\alpha; f_\mu(M')$ である．帰納法の仮定より，$M'\{S/X\} \cong_\mu f_\mu(M')\{S/X\}$ を得る．また，命題 6.4.4(5) より，$\triangleleft\alpha; f_\mu(M')\{S/X\} \cong_\mu \alpha; f_\mu(M')\{S/X\}$ である．すなわち，命題 6.4.5(1) より，$M\{S/X\} \equiv \alpha; M'\{S/X\} \cong_\mu \alpha; f_\mu(M')\{S/X\} \cong_\mu \triangleleft\alpha; f_\mu(M')\{S/X\} \equiv f_\mu(M)\{S/X\}$ を得る．

∎

命題 6.5.5 $Var(M) = \{X\}$，$A_\nu \stackrel{\text{def}}{=} f_\nu(M)\{A_\nu/X\}$，$A_\mu \stackrel{\text{def}}{=} f_\mu(M)\{A_\mu/X\}$ とし，X は M において逐次的であり，ガードされているとする．このとき，次の関係が成り立つ．

(1) $P \models_\mu A_\nu \iff$ 任意の $n \geq 0$ について，$P \models_\mu M^{\langle n\rangle}\{\mathtt{tt}/X\}$
(2) $P \models_\mu A_\mu \iff$ ある $n \geq 0$ について，$P \models_\mu M^{\langle n\rangle}\{\mathtt{ff}/X\}$

ここで，$M^{\langle n\rangle}$ は M から帰納的に次のように定義される仕様式である．

$$M^{\langle 0\rangle} \equiv X, \quad M^{\langle n+1\rangle} \equiv M\{M^{\langle n\rangle}/X\}.$$

証明 変数 X は仕様式 M において逐次的であるので，命題 6.4.5より，$S_1 \sqsubseteq_\mu S_2$ ならば，$M\{S_1/X\} \sqsubseteq_\mu M\{S_2/X\}$ であることを利用する．以下，本証明では，M の変数 X に仕様 S を代入して得られる仕様 $M\{S/X\}$ の記述を $M(S)$ と略記する．

(1) ($\Rightarrow$) の場合：まず，任意の n' について，$A_\nu \sqsubseteq_\mu M^{\langle n' \rangle}(\mathtt{tt})$ を n' に関する帰納法を用いて証明する．$n' = 0$ の場合は $A_\nu \sqsubseteq_\mu \mathtt{tt}$ より明らかである．$n' = n+1$ の場合は帰納法の仮定より，$A_\nu \sqsubseteq_\mu M^{\langle n \rangle}(\mathtt{tt})$ であるので，$M(A_\nu) \sqsubseteq_\mu M(M^{\langle n \rangle}(\mathtt{tt})) \equiv M^{\langle n+1 \rangle}(\mathtt{tt})$ を得る．さらに，命題 6.5.4(1) より，$A_\nu \cong_\mu f_\nu(M)(A_\nu) \cong_\mu M(A_\nu)$ を得る．すなわち，$P \models_\mu A_\nu$ ならば，任意の n' について，$P \models_\mu M^{\langle n' \rangle}(\mathtt{tt})$ である．

($\Leftarrow$) の場合：まず，充足性を次のように帰納的に定義する：

$$\models_\mu^{(0)} = \mathit{Pr} \times \mathit{Sp}, \quad \models_\mu^{(n+1)} = \theta(\models_\mu^{(n)}).$$

このとき，$\models_\mu = \bigcap_{n \geq 0} \models_\mu^{(n)}$ を証明できる．そこで，次の補題 $(*1)$ を n' に関する帰納法を用いて証明する．

$(*1)$ $P \models_\mu f_\nu(N)(f_\nu(M)^{\langle n' \rangle}(\mathtt{tt}))$ ならば，$P \models_\mu^{(n')} f_\nu(N)(A_\nu)$ である．ここで，$\mathit{Var}(N) \subseteq \{X\}$，かつ X は N において逐次的である．

$n' = 0$ のときは定義より明らかである．$n' = n+1$ $(n \geq 0)$ とする．

仮定 $P \models_\mu f_\nu(N)(f_\nu(M)^{\langle n+1 \rangle}(\mathtt{tt}))$ より，ある T_0 で，$f_\nu(N)(f_\nu(M)^{\langle n+1 \rangle}(\mathtt{tt})) \mapsto T_0$ かつ $P \models_\mu T_0$ である．ここで，$f_\nu(N)(f_\nu(M)(f_\nu(M)^{\langle n \rangle}(\mathtt{tt}))) \mapsto T_0$ であり，X は $f_\nu(N)(f_\nu(M))$ においてガードされているので，あるガードされている N_0' について，$f_\nu(N)(f_\nu(M)) \mapsto N_0'$ かつ $T_0 \equiv N_0'(f_\nu(M)^{\langle n \rangle}(\mathtt{tt}))$ を得る．さらに，$f_\nu(N)(f_\nu(M)(A_\nu)) \mapsto N_0'(A_\nu)$ も得られる．また，$X \in \mathit{Var}(f_\nu(M))$ であるので，$f_\nu(N)(f_\nu(M)) \equiv f_\nu(N(M))$ を示せる．すなわち，$f_\nu(N(M)) \mapsto N_0'$ であるので，補題 6.5.2(1) より，あるガードされている N_0 について，$N_0' \equiv f_\nu(N_0)$ である．

- $X \notin \mathit{Var}(N_0')$ の場合：$P \models_\mu T_0 \equiv N_0'(f_\nu(M)^{\langle n \rangle}(\mathtt{tt})) \equiv N_0' \equiv N_0'(A_\nu)$ である．すなわち，$f_\nu(N)(f_\nu(M)(A_\nu)) \mapsto N_0'(A_\nu)$ であるので $P \models_\mu f_\nu(N)(f_\nu(M)(A_\nu))$ を得る．さらに，$A_\nu \stackrel{\mathrm{def}}{=} f_\nu(M)(A_\nu)$ より，$P \models_\mu f_\nu(N)(A_\nu)$ を得る．ここで，$\models_\mu \subseteq \models_\mu^{(n+1)}$ より，$P \models_\mu^{(n+1)} f_\nu(N)(A_\nu)$ である．
- $X \in \mathit{Var}(N_0')$ の場合：このとき，$X \in \mathit{Var}(N_0)$ かつ X は N_0 においてガードされているので，補題 6.5.1(1) より，$f_\nu(N_0) \in \mathit{Stb}$ である．すなわち，$f_\nu(N_0)(A_\nu) \in \mathit{Stb}$ である．

(i) $P \xrightarrow{\alpha} P'$ とする．$P \models_\mu T_0 \equiv f_\nu(N_0)(f_\nu(M)^{\langle n\rangle}(\mathtt{tt}))$ より，ある T' について，$f_\nu(N_0)(f_\nu(M)^{\langle n\rangle}(\mathtt{tt})) \xrightarrow{\alpha} T'$ かつ $P' \models_\mu T'$ を得る．$f_\nu(N_0)$ はガードされているので，ある N'' について，$f_\nu(N_0) \xrightarrow{\alpha} N''$ かつ $T' \equiv N''(f_\nu(M)^{\langle n\rangle}(\mathtt{tt}))$ を得る．さらに，$f_\nu(N_0)(A_\nu) \xrightarrow{\alpha} N''(A_\nu)$ も得られる．

- $X \notin Var(N'')$ の場合：$P' \models_\mu T' \equiv N''(f_\nu(M)^{\langle n\rangle}(\mathtt{tt})) \equiv N'' \equiv N''(A_\nu)$ である．すなわち，$P \models_\mu^{(n)} N''(A_\nu)$ である．
- $X \in Var(N'')$ の場合：補題 6.5.3(1) より，ある N' について，$N'' \equiv f_\nu(N')$ を得る．よって，$P' \models_\mu f_\nu(N')(f_\nu(M)^{\langle n\rangle}(\mathtt{tt}))$ であるので，帰納法の仮定より，$P' \models_\mu^{(n)} f_\nu(N')(A_\nu) \equiv N''(A_\nu)$ を得る．

(ii) 上記の (i) の場合と同様である．

すなわち，$P \models_\mu^{(n+1)} f_\nu(N)(f_\nu(M)(A_\nu)) \cong_\mu f_\nu(N)(A_\nu)$ を得る．よって，補題 $(*1)$ は証明された．

今，$P \models_\mu^{(n)} M^{\langle n\rangle}(\mathtt{tt})$ とする．命題 6.5.4(1) より，$P \models_\mu^{(n)} f_\nu(M)^{\langle n\rangle}(\mathtt{tt})$ である．さらに，補題 $(*1)$ において $N \equiv X$ として，$P \models_\mu^{(n)} A_\nu$ を得る．すなわち，任意の $n \geq 0$ について $P \models_\mu^{(n)} M^{\langle n\rangle}(\mathtt{tt})$ ならば，$P \models_\mu A_\nu$ を得る．

(2) $(\Leftarrow)$ の場合：まず，任意の n' について，$M^{\langle n'\rangle}(\mathtt{ff}) \sqsubseteq_\mu A_\mu$ を n' に関する帰納法を用いて証明する．$n' = 0$ の場合は $\mathtt{ff} \sqsubseteq_\mu A_\mu$ より明らかである．$n' = n+1$ の場合は帰納法の仮定より，$M^{\langle n\rangle}(\mathtt{ff}) \sqsubseteq_\mu A_\mu$ であるので，$M^{\langle n+1\rangle}(\mathtt{ff}) \equiv M(M^{\langle n\rangle}(\mathtt{ff})) \sqsubseteq_\mu M(A_\mu)$ を得る．さらに，命題 6.5.4(1) より，$A_\mu \cong_\mu f_\mu(M)(A_\mu) \cong_\mu M(A_\mu)$ を得る．すなわち，ある n' について $P \models_\mu M^{\langle n'\rangle}(\mathtt{ff})$ ならば，$P \models_\mu A_\mu$ である．

$(\Rightarrow)$ の場合：まず，次の補題 $(*2)$ を n' に関する帰納法を用いて証明する．

$(*2)$ $(P, f_\mu(N)(f_\mu(M)(A_\mu))) \in \theta^{(n')}(\models_\mu)$ ならば，$P \models_\mu f_\mu(N)(f_\mu(M)^{\langle n'+1\rangle}(\mathtt{ff}))$ である．ここで，$Var(N) \subseteq \{X\}$，かつ X は N において逐次的である．

- $n' = 0$ の場合：$(P, f_\mu(N)(f_\mu(M)(A_\mu))) \in \theta^{(0)}(\models_\mu)$ であるので，ある S_0 について，$f_\mu(N)(f_\mu(M)(A_\mu)) \mapsto S_0 \in Stb$ かつ $P \models_\mu S_0$ を得る．$f_\mu(N)(f_\mu(M))$ はガードされているので，あるガードされた N_0' について，$f_\mu(N)(f_\mu(M)) \mapsto N_0'$ かつ $S_0 \equiv N_0'(A_\mu) \in Stb$ を得る．また，$X \in Var(f_\mu(M))$ であるので，$f_\mu(N)(f_\mu(M)) \equiv f_\mu(N(M))$ を示せる．すなわち，$f_\mu(N(M)) \mapsto N_0'$ であるので，補題 6.5.2(2) より，あるガードされた N_0 について，$N_0' \equiv f_\mu(N_0)$

を得る．また，補題 6.5.1(2) より，$f_\mu(N_0) \notin Stb$ である．しかし，これは $f_\mu(N_0)(A_\mu) \equiv S_0 \in Stb$ に矛盾する．すなわち，$n' = 0$ の場合はない．

- $n' = 1$ の場合:$(P, f_\mu(N)(f_\mu(M)(A_\mu))) \in \theta^{(1)}(\models_\mu)$ であるので，ある S_0 について，$f_\mu(N)(f_\mu(M)(A_\mu)) \mapsto S_0$ かつ $(P, S_0) \in \theta^{(1)}(\models_\mu)$ を得る．$n' = 0$ の場合と同様に，あるガードされた N_0 について，$f_\mu(N)(f_\mu(M)) \equiv f_\mu(N(M)) \mapsto f_\mu(N_0)$ かつ $S_0 \equiv f_\mu(N_0)(A_\mu)$ を得る．すなわち，$f_\mu(N)(f_\mu(M)^{\langle 2\rangle}(\mathtt{ff})) \equiv f_\mu(N)(f_\mu(M)(f_\mu(M)^{\langle 1\rangle}(\mathtt{ff}))) \mapsto f_\mu(N_0)(f_\mu(M)^{\langle 1\rangle}(\mathtt{ff}))$ である．

  (*i*) $P \xrightarrow{\alpha} P'$ とする．$(P, f_\mu(N_0)(A_\mu)) \in \theta^{(1)}(\models_\mu)$ であるので，ある S' について，$f_\mu(N_0)(A_\mu) \xrightarrow{\alpha} S'$ かつ $(P', S') \in \theta^{(0)}(\models_\mu)$ である．ここで，X は $f_\mu(N_0)$ においてガードされているので，この遷移は A_μ に依存しない．すなわち，ある N'' について，$f_\mu(N_0) \xrightarrow{\alpha} N''$ かつ $S' \equiv N''(A_\mu)$ である．さらに，$f_\mu(N_0)(f_\mu(M)^{\langle 1\rangle}(\mathtt{ff})) \xrightarrow{\alpha} N''(f_\mu(M)^{\langle 1\rangle}(\mathtt{ff}))$ を得る．

  - $X \notin Var(N'')$ の場合：$S' \equiv N''(A_\mu) \equiv N'' \equiv N''(f_\mu(M)^{\langle 1\rangle}(\mathtt{ff}))$ である．すなわち，$(P', S') \equiv (P', N''(f_\mu(M)^{\langle 1\rangle}(\mathtt{ff}))) \in \theta^{(0)}(\models_\mu)$ である．
  - $X \in Var(N'')$ の場合：補題 6.5.3(2) より，ある N' について，$N'' \equiv f_\mu(N')$ である．すなわち，$(P', f_\mu(N')(A_\mu)) \in \theta^{(0)}(\models_\mu)$ となるが，これは $n' = 0$ の場合に示したように不可能である．よって，必ず $X \notin Var(N'')$ である．

  (*ii*) $f_\mu(N_0)(f_\mu(M)^{\langle 1\rangle}(\mathtt{ff})) \xrightarrow{\alpha} T'$ とする．(*i*) の場合と同様に，ある N'' について，$f_\mu(N_0) \xrightarrow{\alpha} N''$ かつ $T' \equiv N''(f_\mu(M)^{\langle 1\rangle}(\mathtt{ff}))$ を得る．すなわち，$f_\mu(N_0)(A_\mu) \xrightarrow{\alpha} N''(A_\mu)$ である．このとき，ある P' について，$P \xrightarrow{\alpha} P'$ かつ $(P', T') \in \theta^{(0)}(\models_\mu)$ を得られる．

  すなわち，$(P, f_\mu(N)(f_\mu(M)^{\langle 1+1\rangle}(\mathtt{ff}))) \in \theta^{(1)}(\models_\mu)$ である．

- $n' = n+1$ $(n \geq 1)$ の場合：$n' = 0$ の場合と同様に，あるガードされた N_0 で，$f_\mu(N)(f_\mu(M)) \equiv f_\mu(N(M)) \mapsto f_\mu(N_0)$ かつ $(P, f_\mu(N_0)(A_\mu)) \in \theta^{(n+1)}(\models_\mu)$ を得る．すなわち，$f_\mu(N)(f_\mu(M)^{\langle n+2\rangle}(\mathtt{ff})) \equiv f_\mu(N)(f_\mu(M)(f_\mu(M)^{\langle n+1\rangle}(\mathtt{ff}))) \mapsto f_\mu(N_0)(f_\mu(M)^{\langle n+1\rangle}(\mathtt{ff}))$ である．

  (*i*) $P \xrightarrow{\alpha} P'$ とする．$(P, f_\mu(N_0)(A_\mu)) \in \theta^{(n+1)}(\models_\mu)$ であるので，ある m と S' について，$f_\mu(N_0)(A_\mu) \xrightarrow{\alpha} S'$ かつ $(P', S') \in \theta^{(m)}(\models_\mu)$ かつ $m \leq n$ である．すなわち，X は $f_\mu(N_0)$ においてガードされているので，あ

る N'' について，$f_\mu(N_0) \xrightarrow{\alpha} N''$ かつ $S' \equiv N''(A_\mu)$ である．さらに，$f_\mu(N_0)(f_\mu(M)^{\langle n+1\rangle}(\mathtt{ff})) \xrightarrow{\alpha} N''(f_\mu(M)^{\langle n+1\rangle}(\mathtt{ff}))$ である．

- $X \notin Var(N'')$ の場合：$(P', S') \in \theta^{(m)}(\models_\mu)$ であるので，$P' \models_\mu S' \equiv N''(A_\mu) \equiv N'' \equiv N''(f_\mu(M)^{\langle n+1\rangle}(\mathtt{ff}))$ である．
- $X \in Var(N'')$ の場合：補題 6.5.3(2) より，ある N' について，$N'' \equiv f_\mu(N')$ を得る．すなわち，$(P', f_\mu(N')(A_\mu)) \in \theta^{(m)}(\models_\mu)$ である．ここで，$A_\mu \stackrel{\text{def}}{=} f_\mu(M)(A)$ より，$(P', f_\mu(N')(f_\mu(M)(A))) \in \theta^{(m)}(\models_\mu)$ である．よって，帰納法の仮定より，$P' \models_\mu f_\mu(N')(f_\mu(M)^{\langle m+1\rangle}(\mathtt{ff})) \equiv N''(f_\mu(M)^{\langle m+1\rangle}(\mathtt{ff}))$ を得る．さらに，$m \leq n$ であるので，$P' \models_\mu N''(f_\mu(M)^{\langle m+1\rangle}(\mathtt{ff})) \sqsubseteq_\mu N''(f_\mu(M)^{\langle n+1\rangle}(\mathtt{ff}))$ を得られる．

(ii) $f_\mu(N_0)(f_\mu(M)^{\langle n+1\rangle}(\mathtt{ff})) \xrightarrow{\alpha} T'$ とする．(i) の場合と同様に，ある P' と N' について，$P \xrightarrow{\alpha} P'$ かつ $T' \equiv N'(f_\mu(M)^{\langle n+1\rangle}(\mathtt{ff}))$ かつ $P' \models_\mu N'(f_\mu(M)^{\langle n+1\rangle}(\mathtt{ff}))$ を得られる．

すなわち，$P \models_\mu N(M^{\langle n+2\rangle}(\mathtt{ff}))$ である．よって，補題 $(*2)$ は証明された．

今，$P \models_\mu A_\mu$ とする．$A_\mu \stackrel{\text{def}}{=} f_\mu(M)(A_\mu)$ より，$P \models_\mu f_\mu(M)(A_\mu)$ である．すなわち，ある n について，$(P, f_\mu(M)(A_\mu)) \in \theta^{(n)}(\models_\mu)$ である．ここで，補題 $(*2)$ において $N \equiv X$ として，$P \models_\mu f_\mu(M)^{\langle n+1\rangle}(\mathtt{ff})$ を得る．さらに，命題 6.5.4(2) より，$P \models_\mu M^{\langle n+1\rangle}(\mathtt{ff})$ を得る．

∎

次の例を用いて命題 6.5.5を説明する．

$$M \equiv (\triangleleft a; X \,[]\, b; \mathbf{stop}) \vee c; \mathbf{stop}$$

$$SA_{max} \stackrel{\text{def}}{=} f_\nu(M)\{SA_{max}/X\} \equiv (a; SA_{max} \,[]\, b; \mathbf{stop}) \vee c; \mathbf{stop}$$
$$SA_{min} \stackrel{\text{def}}{=} f_\mu(M)\{SA_{min}/X\} \equiv (\triangleleft a; SA_{min} \,[]\, \triangleleft b; \mathbf{stop}) \vee \triangleleft c; \mathbf{stop}$$

$$A_{max} \stackrel{\text{def}}{=} a; A_{max} \,[]\, b; \mathbf{stop}$$
$$A_{min} \stackrel{\text{def}}{=} a; c; \mathbf{stop} \,[]\, b; \mathbf{stop}$$

このとき，任意の $n \geq 0$ について $A_{max} \models_\mu M^{\langle n\rangle}\{\mathtt{tt}/X\}$ かつ $A_{min} \models_\mu M^{\langle n\rangle}\{\mathtt{tt}/X\}$ であるので，命題 6.5.5(1) より，$A_{max} \models_\mu SA_{max}$ かつ $A_{min} \models_\mu SA_{max}$ である．これに対し，$A_{max} \models_\mu M^{\langle n\rangle}\{\mathtt{ff}/X\}$ となるような n は存在しないので $A_{max} \not\models_\mu SA_{min}$ である．

一方，$A_{min} \models_\mu M^{\langle 2\rangle}\{\mathtt{ff}/X\}$ であるので $A_{min} \models_\mu SA_{min}$ が成り立つ．次の定理に示すように，SA_{max} と SA_{min} は，各々等式 $X \cong_\mu M$ の最大不動点と最小不動点である．

定理 6.5.6 $Var(M) = \{X\}$，$A_\nu \stackrel{\rm def}{=} f_\nu(M)\{A_\nu/X\}$，$A_\mu \stackrel{\rm def}{=} f_\mu(M)\{A_\mu/X\}$ とし，X は M において逐次的であり，ガードされているとする．

(1) A_ν は $X \cong_\mu M$ の最大不動点である．

(2) A_μ は $X \cong_\mu M$ の最小不動点である．

証明 命題6.5.4より，$A_\nu \cong_\mu f_\nu(M)\{A_\nu/X\} \cong_\mu M\{A_\nu/X\}$，$A_\mu \cong_\mu f_\mu(M)\{A_\mu/X\} \cong_\mu M\{A_\mu/X\}$ である．すなわち，A_ν と A_μ は共に $X \cong_\mu M$ の解である．以下，各々が最大と最小の解であることを示す．

(1) $S \cong_\mu M\{S/X\}$ かつ $P \models_\mu S$ とする．X は M において逐次的であるので，任意の $n \geq 0$ について次の関係が成り立つ．

$$P \models_\mu S \cong_\mu M\{S/X\} \cong_\mu \cdots \cong_\mu M^{\langle n\rangle}\{S/X\} \sqsubseteq_\mu M^{\langle n\rangle}\{\mathtt{tt}/X\}$$

すなわち，命題 6.5.5(1) より $P \models_\mu A_\nu$ を得る．これは，$S \cong_\mu M\{S/X\}$ のような任意の S について，$S \sqsubseteq_\mu A_\nu$ であることを意味している．

(2) $S \cong_\mu M\{S/X\}$ かつ $P \models_\mu A_\mu$ とする．このとき，命題 6.5.5(2) より，ある n で，$P \models_\mu M^{\langle n\rangle}\{\mathtt{ff}/X\}$ を得る．すなわち，次の関係が成り立つ．

$$P \models_\mu M^{\langle n\rangle}\{\mathtt{ff}/X\} \sqsubseteq_\mu M^{\langle n\rangle}\{S/X\} \cong_\mu \cdots \cong_\mu M\{S/X\} \cong_\mu S$$

これは，$S \cong_\mu M\{S/X\}$ のような任意の S について，$A_\mu \sqsubseteq_\mu S$ であることを意味している．

∎

本節の残りで,「アクション β は必ずいつかは実行される」というライブネス特性について形式的に説明する．プロセス P がこのライブネス特性を満たすとき $P \stackrel{\beta}{\rightsquigarrow}$ とかく．これは，次のように定義できる．

定義 6.5.4 $\beta \in Act$ とする．集合 $\stackrel{\beta}{\rightsquigarrow} \subseteq Pr$ は帰納的に次のように与えられる．

(1) $P \stackrel{\beta}{\rightsquigarrow}_{(0)}$ iff $act(P) = \{\beta\}$

(2) $P \stackrel{\beta}{\rightsquigarrow}_{(n+1)}$ iff 全ての $P' \in deri_\beta(P) \neq \emptyset$ について，ある m で $P' \stackrel{\beta}{\rightsquigarrow}_{(m)}$ かつ $m \leq n$

(3) $P \stackrel{\beta}{\rightsquigarrow}$ iff ある n で，$P \stackrel{\beta}{\rightsquigarrow}_{(n)}$

ここで，$deri_\beta(P) = \{P' : \exists\alpha.\, P \xrightarrow{\alpha} P', \alpha \neq \beta\}$，$act(P) = \{\alpha : \exists P'.\, P \xrightarrow{\alpha} P'\}$ である．∎

条件 $deri_\beta(P) \neq \emptyset$ はプロセスが β を実行する前に停止しないことを要求している．次のプロセス PE_1 と PE_2 について，

$$PE_1 \equiv a;(b;e;f;\mathbf{stop} \,[]\, e;\mathbf{stop}) \,[]\, c;d;e;\mathbf{stop},$$
$$PE_2 \equiv a;(b;f;\mathbf{stop} \,[]\, e;\mathbf{stop}) \,[]\, c;d;e;\mathbf{stop}$$

PE_1 は必ずいつか e を実行するため，$PE_1 \overset{e}{\rightsquigarrow}$ である．一方，PE_2 は abf と実行したときに e を実行しないで停止するため，$PE_2 \overset{e}{\not\rightsquigarrow}$ である．

このライブネス特性は次の命題に示されるように，μLOTOS によって仕様定数 $LIVE_\beta$ として表現できる．この $LIVE_\beta$ は $X \cong_\mu M_\beta$ の最小不動点である．

命題 6.5.7 $\beta \in Act$ とする．仕様定数 $LIVE_\beta$ を次のように定義する．

$$LIVE_\beta \overset{\text{def}}{=} f_\mu(M_\beta)\{LIVE_\beta/X\}$$

ここで，仕様式 M_β は次のように与えられる．

$$M_\beta \equiv \bigvee\{N_{(\beta,\mathcal{A})} : \mathcal{A} \subseteq Act, \mathcal{A} \neq \emptyset\},$$
$$N_{(\beta,\mathcal{A})} \equiv \sum\{\alpha; X : \beta \neq \alpha \in \mathcal{A}\} \,[]\, \sum\{\beta; \mathtt{tt} : \beta \in \mathcal{A}\}.$$

このとき，次の関係が成り立つ

$$P \models_\mu LIVE_\beta \iff P \overset{\beta}{\rightsquigarrow}$$

証明 ($\Rightarrow$) の場合：$P \models_\mu LIVE_\beta$ かつ $LIVE_\beta \overset{\text{def}}{=} f_\mu(M_\beta)\{LIVE_\beta/X\}$ であるので，命題 6.5.5(2) より，ある $n' \geq 0$ について，$P \models_\mu M_\beta^{\langle n' \rangle}\{\mathtt{ff}/X\}$ である．このとき，$P \overset{\beta}{\rightsquigarrow}_{(n'-1)}$ を n' に関する帰納法を用いて証明する．$n' = 0$ ならば $M_\beta^{\langle n' \rangle}\{\mathtt{ff}/X\} \equiv \mathtt{ff}$ であるので，$n' = 0$ の場合はない．$n' = n+1$ $(n \geq 0)$ とする．このとき，ある S_0 について，$M_\beta^{\langle n' \rangle}\{\mathtt{ff}/X\} \equiv M_\beta\{M_\beta^{\langle n \rangle}\{\mathtt{ff}/X\}/X\} \mapsto S_0$ かつ $P \models_\mu S_0$ を得る．この遷移は $\mathbf{Rec}_\vee$, $\mathbf{Act}_\vee$, $\mathbf{Ch}_\vee$, $\mathbf{Dis}_\vee$ により導かれるので，ある $\mathcal{A}$ で，$S_0 \equiv N_{(\beta,\mathcal{A})}\{M_\beta^{\langle n \rangle}\{\mathtt{ff}/X\}/X\}$ かつ $\mathcal{A} \neq \emptyset$ である．

- $n = 0$ の場合：$P \models_\mu S_0 \equiv N_{(\beta,\mathcal{A})}\{\mathtt{ff}/X\}$ である．もし $\beta \neq \alpha \in \mathcal{A}$ のような α が存在するならば，$N_{(\beta,\mathcal{A})}\{\mathtt{ff}/X\} \xrightarrow{\alpha} \mathtt{ff}$ であるので，ある P' について，$P \xrightarrow{\alpha} P'$ かつ $P' \models_\mu \mathtt{ff}$ となり矛盾する．すなわち，$\mathcal{A} \neq \emptyset$ であるので，$\mathcal{A} = \{\beta\}$ である．これは，$act(P) = \{\beta\}$ を導く．よって $P \overset{\beta}{\rightsquigarrow}_{(0)}$ である．

- $n \geq 1$ の場合：$\mathcal{A} = \{\beta\}$ ならば $P \overset{\beta}{\rightsquigarrow}_{(0)}$ であるので，$\mathcal{A} - \{\beta\} \neq \emptyset$ とする．これは，$deri_\beta(P) \neq \emptyset$ を導く．今，$P' \in deri_\beta(P)$ とする．すなわち，ある α について，$P \xrightarrow{\alpha} P'$ かつ $\alpha \neq \beta$ である．ここで，$P \models_\mu S_0 \equiv N_{(\beta,\mathcal{A})}\{M_\beta^{\langle n \rangle}\{\mathtt{ff}/X\}/X\}$ であるので，$S_0 \xrightarrow{\alpha} M_\beta^{\langle n \rangle}\{\mathtt{ff}/X\}$ かつ $P' \models_\mu M_\beta^{\langle n \rangle}\{\mathtt{ff}/X\}$ を得る．よって，帰納法の仮定より，$P' \overset{\beta}{\rightsquigarrow}_{(n-1)}$ を得る．すなわち，全ての $P' \in deri_\beta(P) \neq \emptyset$ について，ある m で $P' \overset{\beta}{\rightsquigarrow}_{(m)}$ かつ $m \leq n-1$ であるので，$P \overset{\beta}{\rightsquigarrow}_{(n)}$ を得る．

($\Leftarrow$) の場合：上記の ($\Rightarrow$) の場合と同様に，$P \overset{\beta}{\rightsquigarrow}_{(n)}$ ならば，n に関する帰納法を用いて $P \models_\mu M_\beta^{\langle n+1 \rangle}\{\mathtt{ff}/X\}$ を証明できる． ∎

前述の例 PE_1 と PE_2 について，$PE_1 \overset{e}{\rightsquigarrow}$ かつ $PE_2 \overset{e}{\not\rightsquigarrow}$ であるので，命題 6.5.7より，$PE_1 \models_\mu LIVE_e$ かつ $PE_2 \not\models_\mu LIVE_e$ である．

6.6 充足可能性

複数の柔軟な仕様が与えられたとき，その仕様の無矛盾性を判定し，それら全てを満たすプロセスを合成することは重要である．μLOTOS は合接演算子 $\wedge$ をもつので，複数の仕様 $S_1, \cdots, S_n$ の無矛盾性は，合接仕様 $S_1 \wedge \cdots \wedge S_n$ の充足可能性の判定に置き換えられる．ここで，仕様 S が充足可能とはそれを満たすプロセスが存在することである ($Proc(S) \neq \emptyset$)．

すなわち，もし $P \models_\mu S$ のようなプロセス P をみつけることができれば，S は充足可能である．しかし，全ての $P \in Pr$ について $P \models_\mu S$ を判定することは，Pr が無限集合なので不可能である．そこで，充足可能性の帰納的な定義を与える．

定義 6.6.1 仕様の集合 Sat を次のように定義する．

$$Sat = \bigcup\{\mathcal{S} \ : \ \mathcal{S} \text{ は充足可能部分集合 } \} \subseteq Sp$$

ここで，$\mathcal{S} \subseteq \Theta(\mathcal{S}) \subseteq Sp$ ならば $\mathcal{S}$ は**充足可能部分集合**である．ここで，$\Theta(\mathcal{S})$ は任意の $\mathcal{S} \subseteq Sp$ について，次のように帰納的に定義される．

1. $S \in \Theta^{(0)}(\mathcal{S})$ iff 次の条件を満たす S_0 が存在する：$S \mapsto S_0 \in Stb$ かつ，任意の $\alpha \in Act$ について，もし $S_0 \xrightarrow{\alpha} S'$ ならば $S' \in \mathcal{S}$ である．

2. $S \in \Theta^{(n+1)}(\mathcal{S})$　iff　次の条件を満たす S_0 が存在する：$S \mapsto S_0$ かつ，任意の $\alpha \in Act$ について，もし $S_0 \xrightarrow{\alpha} S'$ ならば，ある m について，$S' \in \Theta^{(m)}(\mathcal{S})$ かつ $m \leq n$ である．

3. $\Theta(\mathcal{S}) = \bigcup_{n \geq 0} \Theta^{(n)}(\mathcal{S})$

∎

充足関係 $\models_\mu$ の定義 6.4.2と同様に，$S \in Sat$ ならば，S は安定状態か停止状態に到達できなければならない．この集合 Sat に対して期待通り次の命題が成り立つ．

命題 6.6.1 $S \in Sp$ とする．このとき，

$$S \in Sat \iff P \models_\mu S \text{ となる } P \text{ が存在する．}$$

証明 $(\Rightarrow)$ の場合：次の集合 $\mathcal{R}$ が充足部分集合であることを示す．

$$\mathcal{R} = \{(\mathbf{Pr}(S_0), S) : S \mapsto S_0 \in Sat\}$$

ここで，プロセス $\mathbf{Pr}(S_0)$ は仕様 $S_0 \in Sp_0$ から次のように定義される定数である．

$$\mathbf{Pr}(S_0) \stackrel{\text{def}}{=} \sum\{\alpha; \mathbf{Pr}(S_0') : \exists S', S_0 \xrightarrow{\alpha} S', S_0' \in Min(S')\}$$

$$Min(S) = \{S_0 : S \mapsto S_0 \in Sat, |S_0| = |S|\}$$
$$|S| = \min\{n : S \in \Theta^{(n)}(Sat)\}$$

ここで，S の深さ $|S|$ は S が安定状態か停止状態に到達するまでの遷移の回数の最小値である．まず，次の補題 $(*)$ を n' に関する帰納法を用いて証明する．

$(*)$　もし $S \mapsto S_0$ かつ $S_0 \in \Theta^{(n')}(Sat)$ ならば，$(\mathbf{Pr}(S_0), S) \in \theta^{(n')}(\mathcal{R})$ である．

$n' = 0$ の場合は $n' = n + 1$ $(n \geq 0)$ の場合より簡単である．$S \in \Theta^{(n+1)}(Sat)$ とする．

(i) $\mathbf{Pr}(S_0) \xrightarrow{\alpha} P'$ とする．**Ch** と **Act** より，ある S' と S_0' について，$S_0 \xrightarrow{\alpha} S'$ かつ $S_0' \in Min(S')$ かつ $P' \equiv \mathbf{Pr}(S_0')$ である．すなわち，$S_0 \in \Theta^{(n+1)}(Sat)$ であるので，ある m について，$S' \in \Theta^{(m)}(Sat)$ かつ $m \leq n$ を得る．ここで，$S_0' \in Min(S')$ より，$S' \mapsto S_0'$ かつ $S_0' \in Sat$ かつ $|S_0'| = |S'|$ である．すなわち，ある m' について，$S_0' \in \Theta^{(m')}(Sat)$ かつ $m' \leq m$ である．以上，$S' \mapsto S_0' \in \Theta^{(m')}(Sat)$ かつ $m' \leq m \leq n$ であるので，帰納法の仮定より，$(\mathbf{Pr}(S_0'), S') \in \theta^{(m')}(\mathcal{R})$ を得る．

(ii) $S_0 \xrightarrow{\alpha} S'$ とする．(i) の場合と同様に，ある S_0' と m' について，$\mathbf{Pr}(S_0) \xrightarrow{\alpha} \mathbf{Pr}(S_0')$ かつ $(\mathbf{Pr}(S_0'), S') \in \theta^{(m')}(\mathcal{R})$ かつ $m' \leq n$ を得る．

すなわち，$(\mathbf{Pr}(S_0), S) \in \theta^{(n+1)}(\mathcal{R})$ である．よって，補題 $(*)$ は証明された．

今，$(\mathbf{Pr}(S_0), S) \in \mathcal{R}$ とする．すなわち，$S \mapsto S_0 \in Sat$ である．さらに，ある n について，$S_0 \in \Theta^{(n)}(Sat)$ である．よって，補題 $(*)$ より，$(\mathbf{Pr}(S_0), S) \in \theta^{(n)}(\mathcal{R})$ を得る．すなわち，$\mathcal{R} \subseteq \theta(\mathcal{R})$ であるので，$\mathcal{R}$ は充足部分集合である．

$(\Leftarrow)$ の場合：仕様がプロセスによって満たされるためには，つねに $S \mapsto S_0$ のような S_0 が存在し，いつかは安定状態か停止状態に到達できなければならない．これらのことを考慮し，集合 $\{S : \exists P.\ P \models_\mu S\}$ が充足可能部分集合であることを証明できる． ∎

命題 6.6.1の証明の中に，仕様からそれを満たすプロセスを合成する方法が $\mathbf{Pr}(S)$ として与えられている．これにより，複数の仕様 $S_1, \cdots, S_2$ が与えられたとき，それらの無矛盾性は $S_1 \wedge \cdots \wedge S_2 \in Sat$ によって判定でき，それら全てを満たすプロセスは $\mathbf{Pr}(S_0)$ によって合成できる．ここで，$S_1 \wedge \cdots \wedge S_2 \mapsto S_0 \in Sat$ である．

例 6.6.1 仕様 *SAB* と *SBA* を再び用いる．例 6.4.2に示したように，$SAB \wedge SBA$ を満たすプロセス *PR* が存在する．すなわち，$SAB \wedge SBA$ は充足可能である．実際に，次の $\mathcal{S}$ が充足可能部分集合であることを証明できるので，$SAB \wedge SBA \in Sat$ である．

$$\mathcal{S} = \{(SAB \wedge SBA), (SBA \wedge SAB)\}$$

さらに，各仕様の深さは次のように得られる．

$$\begin{aligned} &|SAB \wedge SBA| = 0, && |SBA \wedge SAB| = 0, \\ &|b; SAB \wedge \triangleleft b; SBA| = 0, && |a; SBA \wedge \triangleleft a; SAB| = 0, \\ &|\triangleleft a; SAB \wedge a; SBA| = 1, && |\triangleleft b; SBA \wedge b; SAB| = 1. \end{aligned}$$

すなわち，プロセスを次のように合成することができる．

$$\begin{aligned} PAB &\equiv \mathbf{Pr}(b; SAB \wedge \triangleleft b; SBA) \stackrel{\text{def}}{=} b; PBA \\ PBA &\equiv \mathbf{Pr}(a; SBA \wedge \triangleleft a; SAB) \stackrel{\text{def}}{=} a; PAB \end{aligned}$$

この *PAB* は交互に繰り返し a と b を実行するプロセスである．ここで，$SAB \wedge SBA \mapsto (b; SAB \wedge \triangleleft b; SBA) \in Sat$ であるので，命題 6.6.1の証明より，$PAB \models_\mu SAB \wedge SBA$ が得られる．さらに，命題 6.4.4より，$PAB \models_\mu SAB$ かつ $PAB \models_\mu SBA$ である．すなわち，*PAB* は *SAB* と *SBA* の共通のプロセスである． ∎

6.7 関連研究

仕様の統合や詳細化については，既にいくつかの方法が提案されている [8, 46, 33]. Brinksma[8] はマルチラベルをもつ並行合成演算子を提案した．この演算子は LOTOS で合接演算子を効果的に実現するために使われる．Steen 等 [46] は合接演算子や結合演算子を提案し，仕様の合成に適用した．また，Larsen 等 [33] は合接演算子を柔軟な仕様に対して定義した．しかしながら，これらの演算子では離接演算子や最小不動点が考慮されていないため，そのまま μLOTOS で利用することはできない．

論理的な要求からプロセスを合成するアルゴリズムも提案されている [29, 36]．木村等はプロセス論理 (μ 計算の部分計算) の記述 (要求) からプロセス代数 (CCS) の記述 (プロセス) を合成する方法を提案した [29]．しかしこの部分計算は離接演算子を含んでいなかった．また，Manna 等は命題時相論理 (PTL) から振舞いのグラフを合成する方法を提案した [36]．しかし，合成されるグラフは必ずしも要求の共通部分を正確には表していないことが指摘されている．

6.8 おわりに

本章では，論理的な演算子 (合接，離接，最大不動点，最小不動点) をもつプロセス代数として μLOTOS を提案した．これらの演算子によって，柔軟な仕様から実行可能なプロセスまで，同じ枠組 (μLOTOS) で記述できる利点がある．従来のプロセス代数と比較して最小不動点を表現できる特徴があり,「いつかは実行される」ライブネス特性を記述することができる．

また，仕様の充足可能性の帰納的な定義を与えた．仕様の状態数が有限であれば，その充足可能性を判定することができる．さらに，充足可能な仕様に対しては，それを満たすプロセスを合成する方法を与えた．これにより，複数の設計者が記述した複数の柔軟な仕様の無矛盾性を判定し，それら全てを満たすプロセスを合成することが可能である．

プロセス論理の演算子をもつプロセス代数の特徴として，並行合成演算子と離接演算子を用いて，並行動作に対する柔軟な仕様を記述できることがあげられる．μLOTOS も並行合成演算子 $||[G]||$ と離接演算子 $\vee$ をもち，これらの性質を調べてきた．例えば，命題 6.4.3(3) に示すように充足性は並行合成演算子によって保存されるため，各プロ

セスに分離して充足性を判定することができる．しかし，並行合成演算子と離接演算子が混在する場合，いくつかの重要な特性が成り立たない問題も明らかになった．

例えば，詳細順序 $\sqsubseteq_\mu$ は並行合成演算子によって保存されない．その例を示す．まず，仕様 S_1, S_2, S_3 を次のように与える．

$$S_1 \equiv a; (b; \mathbf{stop} \vee c; \mathbf{stop})$$
$$S_2 \equiv a; b; \mathbf{stop} \vee a; c; \mathbf{stop}$$
$$S_3 \equiv a; b; \mathbf{stop} \,[]\, a; c; \mathbf{stop}$$

S_1 は a の後に b か c が実行されることを要求しているので，S_1 を満たすプロセスは ab か ac を実行できるため，S_2 も満たす．すなわち，$S_1 \sqsubseteq_\mu S_2$ である．しかし，S_1 と S_2 に S_3 を並行合成すると，次の詳細順序は成り立たない．

$$S_1 \,|[a,b,c]|\, S_3 \sqsubseteq_\mu S_2 \,|[a,b,c]|\, S_3$$

例えば，プロセス $P \equiv a; b; \mathbf{stop} \,[]\, a; c; \mathbf{stop}$ は，$S_1 \,|[a,b,c]|\, S_3$ を満たすが，$S_2 \,|[a,b,c]|\, S_3$ を満たさない．

$$P \models_\mu S_1 \,|[a,b,c]|\, S_3$$
$$P \not\models_\mu S_2 \,|[a,b,c]|\, S_3$$

これは，$S_2 \,|[a,b,c]|\, S_3$ では，次のようにデッドロックすることがあるためである．

$$\begin{aligned} S_2 \,|[a,b,c]|\, S_3 &\mapsto a; b; \mathbf{stop} \,|[a,b,c]|\, S_3 \xrightarrow{a} b; \mathbf{stop} \,|[a,b,c]|\, c; \mathbf{stop} \\ &\mapsto b; \mathbf{stop} \,|[a,b,c]|\, c; \mathbf{stop} \not\longrightarrow \end{aligned}$$

これに対し，$S_1 \,|[a,b,c]|\, S_3$ では，上記のようなデッドロックが生じないように a を実行後に b か c を選ぶことができる．

$$\begin{aligned} S_1 \,|[a,b,c]|\, S_3 &\mapsto S_1 \,|[a,b,c]|\, S_3 \xrightarrow{a} (b; \mathbf{stop} \vee c; \mathbf{stop}) \,|[a,b,c]|\, c; \mathbf{stop} \\ &\mapsto c; \mathbf{stop} \,|[a,b,c]|\, c; \mathbf{stop} \xrightarrow{c} \mathbf{stop} \,|[a,b,c]|\, \mathbf{stop} \end{aligned}$$

上記の例 (S_1, S_2, S_3) に見られるように，この詳細順序が保存されない問題は μLOTOS の不安定演算子 $\triangleleft$ とは無関係であり，並行合成演算子と離接演算子の基本的な問題である．柔軟な仕様において並行性を適切に扱うためには，並行合成演算子に代わる何らかの方法が必要であると考える．これについては次の章で検討する．

第 7 章

分散システム合成用プロセス代数

7.1 はじめに

分散システムでは複数のプロセスが互いに通信しながら並行に実行されており，その全体の動作を設計時に予測することは容易ではない．本研究の最終的な目標は仕様から分散システムを合成する方法を確立することである．そのためにはシステムが仕様を満たしているかを検証できる形式的な枠組が必要である．

6章では，(プロセス) 論理演算子をプロセス代数に導入し，並行動作に対する柔軟な仕様を記述できる μLOTOS を提案した．そして，仕様の充足可能性を判定し，プロセスを合成する方法を与えた．ただし，この合成されるプロセスは逐次的である．さらに，6.8節で述べたように，並行合成演算子と離接演算子が混在する場合は，詳細順序が保存されない等の問題も明らかになった．本章では，並行合成演算子の代わりに**真の並行性**を考慮したプロセス論理を用いて，並行動作に対する柔軟な仕様を記述する．

2.5節で説明したように，プロセス論理 [38, 52] では，可能性演算子や離接演算子 $\vee$ を用いて柔軟に仕様を記述し，必然性演算子と合接演算子 $\wedge$ によって仕様を追加 (詳細化) できる．ただし，基本的なプロセス論理では並行性を明記できないため，並行システム (例: $a|b$) を合成することを期待しても，それに振舞いが等価な逐次システム (例: $a.b + b.a$) が合成される可能性がある．これは，CCS 等の基本的なプロセス代数では，並行動作と逐次動作を明確に区別しないためである ($a|b = a.b + b.a$)．

アクションの位置や因果関係を考慮して，真の並行性 (true concurrency) を表現できるプロセス代数 [7, 6, 13, 14, 30, 42] が提案されており，そのためのプロセス論理 [6] も

与えられている．このような真の並行プロセス代数では，並行動作と逐次動作は明確に区別されるため $(a|b \neq a.b + b.a)$，真の並行性を考慮したプロセス論理で仕様を記述すれば，この仕様から適切に並行システム (分散システム) を合成できると考える．しかし，真の並行性を考慮したプロセス論理の充足可能性は議論されていなかった．与えられた仕様を満たすシステムを合成するには，先ずその仕様を満たすことができるかを判定することが重要である．

本章では，分散システムを記述するための真の並行プロセス代数 *DS*@ と，その仕様を記述するためのプロセス論理 *SP*@ を定義し，停止性は保証されていないが，仕様 (*SP*@ の記述) の充足可能性を判定するアルゴリズムと，その仕様から分散システム (*DS*@ の記述) を合成する方法を与える．さらに，そのような充足可能性を判定する停止性が保証されたアルゴリズムは存在しないことを示す．すなわち，*SP*@ で記述された仕様の充足可能性は決定不能である．この結果は，仕様が有限状態をもち，離接演算子をもたず，通信に対する要求がない場合でも成り立つ．*DS*@ は非常に簡単な真の並行プロセス代数であり，この結果は他の多くの真の並行プロセス代数に対しても成り立つ．

以下，7.2節で，真の並行プロセス代数 *DS*@ とプロセス論理 *SP*@ の概要を述べ，7.3節と 7.4節で各々*DS*@ と *SP*@ を定義する．次に，7.5節では，仕様の充足可能性を判定し，それを満たす分散システムを合成する方法を与える．そして，7.6節で仕様の充足可能性が決定不能であることを証明する．7.7節では，従来研究と本研究を比較する．

7.2 *DS*@ と *SP*@ の紹介

DS@ は CCS にプロセス名を明記する名前付演算子 @ を導入し，アクションがどのプロセスで実行されたかを観測可能にしたプロセス代数である．このような考え方は既に Krishnan[30] によって提案された Distributed CCS[30] 等によって採用されているが，本研究の目的は新しいプロセス代数やプロセス論理の提案ではない．本研究の目的は，仕様から分散システムを合成する方法を与えることであり，まずは簡単な真の並行プロセス代数を用いて，その合成方法の基礎的な性質を明らかにすることが重要である．そこで，非常に簡単ではあるが，真の並行性を表現できるプロセス代数として *DS*@ を定義している．

図 7.1のシステムを例に *DS*@ について説明する．このシステムは制御装置 $(cntl)$ のボ

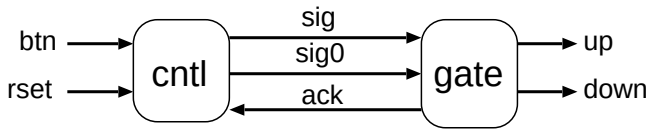

図 7.1: 遮断機の遠隔制御の例

タン (btn) を押すごとに，踏切装置 $(gate)$ の遮断機を上げたり (up)，下げたり $(down)$ する例である．また，リセット $(rset)$ により，遮断機を必ず上げた状態にすることができる．このシステムは $DS@$ を用いて次の SYS のように記述できる．

$$
\begin{aligned}
SYS &\equiv net \triangleright (CT@cntl \mid UP@gate) \\
CT &\stackrel{\text{def}}{=} btn.sig.ack.CT + rset.sig_0.ack.CT \\
UP &\stackrel{\text{def}}{=} sig.down.ack.DN + sig_0.ack.UP \\
DN &\stackrel{\text{def}}{=} sig.up.ack.UP + sig_0.up.ack.UP
\end{aligned}
$$

$$
\begin{aligned}
net = \{&btn\{cntl\}, rset\{cntl\}, up\{gate\}, down\{gate\}, \\
&sig\{cntl, gate\}, sig_0\{cntl, gate\}, ack\{cntl, gate\}\}
\end{aligned}
$$

ここで，$|, \text{‘.’}, +$ は CCS と同様に並行合成演算子，逐次演算子，選択演算子である．また，$\triangleright$ はこのシステムが環境 net のもとで動作することを表す**環境演算子**である．直観的に環境 net は各プロセスの接続状況を表しており，この例ではプロセス $cntl$ と $gate$ は各々アクション $btn, rset$ と $up, down$ を独立に実行でき，アクション sig, sig_0, ack を通して同期 (通信) できることを表している．一方，$sig\{cntl\}$ などは net に含まれないので，sig, sig_0, ack を各プロセスが独立に実行することはできない．環境演算子は CCS の制限演算子に相当する．

CT は制御装置の動作，UP と DN は踏切装置の動作を表し，SYS はシステムの構造を表す．sig と sig_0 は制御装置からの要求を踏切装置に伝え，ack は要求された動作の完了を返信するアクションである．UP と DN は各々遮断機が上がっている状態と下がっている状態に対応し，sig, sig_0 に応じて遮断機の上げ下げを行う．

CCS との違いは**名前付演算子** @ によってプロセス名を明記できることである．例えば，$CT@cntl$ は $cntl$ という名前のプロセスは CT のように振舞うことを表す．また，アクションがどのプロセスで実行されたかを観測することができる．例えば，up はプロセス $gate$ によって実行されるので，$up\{gate\}$ として観測される．一方，sig はプロセス $cntl$ と $gate$ の同期によって実行されるので，$sig\{cntl, gate\}$ として観測される．

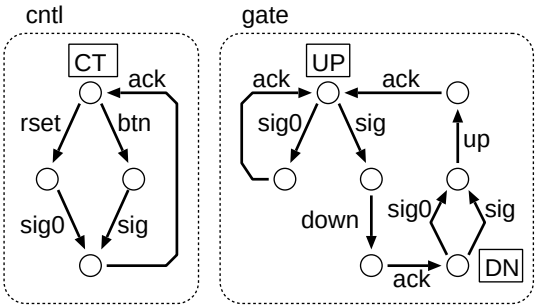

図 7.2: 各プロセス $(cntl, gate)$ の状態遷移図

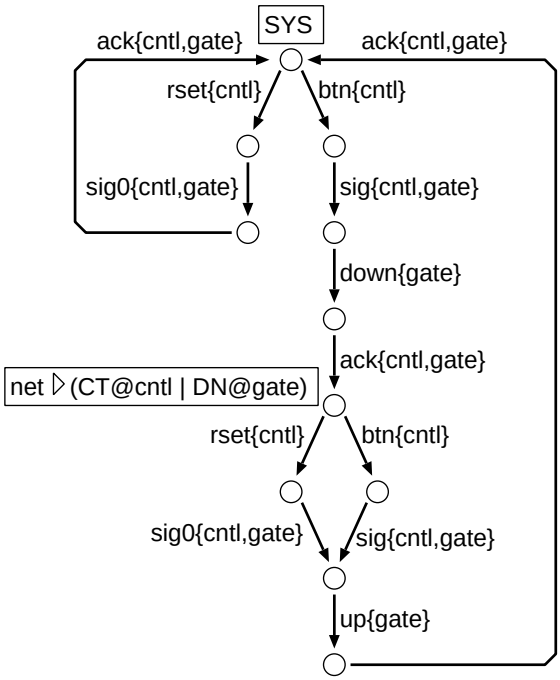

図 7.3: 分散システム SYS の状態遷移図

図 7.2と図 7.3に各プロセス $(cntl, gate)$ の状態遷移図と分散システム SYS の状態遷移図を示す.

次にプロセス論理 $SP@$ について説明する．分散システムに対する要求はプロセス論理 $SP@$ によって記述される．例えば，図 7.1のシステムでは「ボタン (btn) を押すごとに，遮断機を下げる動作 $(down)$ と上げる動作 (up) を繰り返えす」ことが要求され

る．このための仕様は $SP@$ によって次のように記述される．

$$SP_1 \stackrel{\text{def}}{=} \langle btn\{cntl\}\rangle \stackrel{net}{\rightsquigarrow} \langle down\{gate\}\rangle \stackrel{net}{\rightsquigarrow} SP_2$$
$$SP_2 \stackrel{\text{def}}{=} \langle btn\{cntl\}\rangle \stackrel{net}{\rightsquigarrow} \langle up\{gate\}\rangle \stackrel{net}{\rightsquigarrow} SP_1$$

ここで，$\langle\cdot\rangle$ は可能性演算子である．例えば，仕様 $\langle btn\{cntl\}\rangle S$ は，プロセス $cntl$ がアクション btn を実行でき，その後に仕様 S を満たすことができるを要求している．$SP@$ はこのように，**各アクションをどのプロセスが実行するかを指定できる特徴をもつ**．また，$\stackrel{net}{\rightsquigarrow} S$ は仕様 S を満たす前に一回の通信を許可することを表しており，ここでは次の略記法によって与えられる．

$$\stackrel{net}{\rightsquigarrow} S \equiv S \vee \langle sig\{cntl, gate\}\rangle S \vee \langle sig_0\{cntl, gate\}\rangle S \vee \langle ack\{cntl, gate\}\rangle S$$

より柔軟に n 回の通信を許すように記述することもできる．

一方，システムが安全であるためには，単にアクションが実行できることを要求するだけでなく，危険な振舞いは実行しないように制限する必要がある．例えば,「常に，ボタン (btn) を押した後，遮断機が上がる (up) か下がる $(down)$ までは次のボタン (btn) は押せない」ことが安全上必要である．この仕様は $SP@$ によって次のように記述される．

$$SN_1 \stackrel{\text{def}}{=} [btn\{cntl\}]SN_2 \wedge [net - \{btn\{cntl\}\}]SN_1$$
$$SN_2 \stackrel{\text{def}}{=} [btn\{cntl\}]\mathtt{ff} \wedge$$
$$[\{up\{gate\}, down\{gate\}\}]SN_1 \wedge$$
$$[net - \{btn\{cntl\}, up\{gate\}, down\{gate\}\}]SN_2$$

ここで，$[\cdot]$ は必然性演算子である．例えば，仕様 $[btn\{cntl\}]S$ は，もしプロセス $cntl$ がアクション btn を実行するならば，その後は必ず仕様 S を満たすことを要求している．また，$\mathcal{A}$ をアクションの集合 $\{\alpha_1\Psi_1, \cdots, \alpha_n\Psi_n\}$ とすると，$[\mathcal{A}]S$ は $[\alpha_1\Psi_1]S \wedge \cdots \wedge [\alpha_n\Psi_n]S$ の略記である．例えば，上記の $[net - \{btn\{cntl\}\}]SN_1$ はプロセス $cntl$ が btn を実行するまで SN_1 を満たし続けることを要求している．そして，btn が実行された後は，$gate$ が up か $down$ を実行するまで，(偽 $\mathtt{ff}$ は満たせないので)$cntl$ は btn を実行しないことが要求される．

分散システム ($DS@$ の記述) の仕様 ($SP@$ の記述) に対する充足関係は形式的に与えられ，上記のシステム SYS は仕様 $SP_1 \wedge SN_1$ を満たす ($SYS \models SP_1 \wedge SN_1$ と書く) ことが証明できる．

7.3 分散システムの定義

前節で紹介したプロセス代数 $DS@$ を定義する．まず，**アクション名**の集合 $AN = \{a, b, \cdots\}$ と**プロセス名**の集合 $PN = \{p, q, \cdots\}$ が与えられているとし，AN の要素を $\alpha, \beta, \gamma, \cdots$，$PN$ の要素を $\psi, \varphi, \cdots$，PN の部分集合を $\Psi, \Phi, \Theta, \cdots$ で表す．**分散アクション**はアクション名とプロセス名の集合の組であり，その集合は

$$Act = \{\alpha\Psi : \alpha \in AN, \Psi \subseteq PN, 1 \leq \|\Psi\| \leq 2\}$$

である．ここで，$\|\Psi\|$ は集合 Ψ の要素数を表す．直観的に，$\alpha\{\psi\}$ はプロセス ψ で実行されたアクション α を表し，$\alpha\{\psi, \varphi\}$ はアクション α を通してプロセス ψ と φ が同期したことを表す．

さらに，**プロセス定数** ($A, \cdots$ で表す) の集合 $Cons^P$ が与えられているとする．このとき，$DS@$ の構文は次のように与えられる．

定義 7.3.1 プロセス ($P, Q, \cdots$ で表す) の集合 Pr は次の式を含む最小の集合である．

$$\begin{aligned} A &: \text{プロセス定数 } (A \in Cons^P), \\ \alpha.P &: \text{逐次 } (\alpha \in AN), \\ \textstyle\sum_{i\in I} P_i &: \text{選択 } (I \text{ はインデックス集合 }), \end{aligned}$$

ここで，P, P_i はすでに Pr の要素である．また，**分散システム** ($D, E, \cdots$ で表す) の集合 Ds は次の式を含む最小の集合である．

$$\begin{aligned} P@\psi &: \text{名前付 } (P \in Pr, \psi \in PN) \\ D|E &: \text{並行合成 } (Pn(D) \cap Pn(E) = \emptyset) \\ \mathcal{E} \triangleright D &: \text{環境 } (\mathcal{E} \subseteq Act) \end{aligned}$$

ここで，D, E は既に Ds の要素である．プロセスに固有の名前を割り当てるために関数 $Pn : Ds \to 2^{PN}$ は帰納的に次のように与えられる：$Pn(P@\psi) = \{\psi\}$, $Pn(D|E) = Pn(D) \cup Pn(E)$, $Pn(\mathcal{E} \triangleright D) = Pn(D)$．演算子の結合の優先順位は (逐次 > 選択 > 名前付 > 並行合成 > 環境) である． ∎

以下，各演算子について簡単に説明する．

プロセス $\alpha.P$ はアクション α を実行し，その後は P のように振舞うプロセスである．プロセス $\sum_{i\in I} P_i$ はある $j \in I$ について P_j のように振舞う．$I = \{1, 2\}$ の場合は，

$\sum_{i\in\{1,2\}} P_i$ の代わりに $P_1 + P_2$ の記法も用いる．すなわち，$P_1 + P_2$ は P_1 か P_2 のように振舞うプロセスである．プロセス定数は定義式によって意味を与えられるプロセスであり，実際に全ての定数 $A \in Cons^P$ について，$A \stackrel{\text{def}}{=} P$ の定義式があるとする．ここで，P は再び定数を含むことができ，再帰動作を表現できる．また，無動作プロセス $\mathbf{0}$ は，$\mathbf{0} \stackrel{\text{def}}{=} \sum_{i\in\emptyset} E_i$ によって定義されるプロセス定数として与えられる．

名前付演算子 @ はプロセスに名前を付ける演算子であり，$P@\psi$ は名前 ψ のプロセスは P のように振舞うことを表している．名前 ψ をもつプロセスをプロセス ψ と呼び，Ψ に含まれる名前をもつ複数のプロセスをまとめてプロセス Ψ と呼ぶ．

$D|E$ では D と E は独立にアクションを実行でき，かつ同じ名前のアクションを同期して実行することができる．ここで，プロセス名は (アクションごとではなく) プロセスごとに与えられるので，分散システム $(a.\mathbf{0}@p|b.\mathbf{0}@q)$ の振舞いは $(a@p.b@q.\mathbf{0}+b@q.a@p.\mathbf{0})$ に展開できないことが重要である．すなわち，$DS@$ では並行動作と逐次動作を明確に区別する．

環境 $\mathcal{E}$ は実行可能なアクションの集合を表し，直観的には各チャンネルの接続状況を表している．例えば，$\{in\{p\}, out\{q\}, sync\{p,q\}\}$ はプロセス p と q が各々アクション in と out を実行でき，$sync$ によって同期できる環境を表している．また，全ての実行可能なアクションの集合を得るために次の関数 $env : Ds \to 2^{Act}$ が有効である．

$$
\begin{aligned}
env(P@\psi) &= \{\alpha\{\psi\} : \alpha \in AN\} \\
env(D|E) &= \{\alpha\{\psi,\varphi\} : \alpha\{\psi\} \in env(D), \alpha\{\varphi\} \in env(E)\} \cup env(D) \cup env(E) \\
env(\mathcal{E} \triangleright D) &= \mathcal{E} \cap env(D)
\end{aligned}
$$

$env(D)$ は分散システム D が実行可能な全ての分散アクションを含んでいる．

また，次のように定義される関数 $pn : 2^{Act} \to 2^{PN}$ が環境からプロセス名を抜き出すために使われる．

$$pn(\mathcal{E}) = \{\psi : \exists \alpha\Psi \in \mathcal{E}.\, \psi \in \Psi\}$$

次にプロセスと分散システムの意味をラベル付遷移システムにより定義する．

定義 7.3.2 プロセスと分散システムの意味は各々ラベル付遷移システム

$$\langle Pr, AN, \dot{\to}\rangle, \quad \langle Ds, Act, \to\rangle$$

により与えられる．ここで，遷移の集合 $\dot{\to}$ と $\to$ は各々図 7.4と図 7.5の推論規則を満たす最小の集合である． ∎

名前	仮定	⊢	結果
Act		⊢	$\alpha.P \xrightarrow{\alpha\cdot} P$
Choice$_j$	$P_j \xrightarrow{\alpha\cdot} P', j \in I$	⊢	$\sum_{i\in I} P_i \xrightarrow{\alpha\cdot} P'$
Rec	$A \stackrel{\text{def}}{=} P,\ P \xrightarrow{\alpha\cdot} P'$	⊢	$A \xrightarrow{\alpha\cdot} P'$

図 7.4: プロセスのための遷移集合 $\dot{\to}$ の推論規則

名前	仮定	⊢	結果
Name	$P \xrightarrow{\alpha\cdot} P'$	⊢	$P@\psi \xrightarrow{\alpha\{\psi\}} P'@\psi$
Com$_1$	$D \xrightarrow{\alpha\Psi} D'$	⊢	$D\|E \xrightarrow{\alpha\Psi} D'\|E$
Com$_2$	$E \xrightarrow{\alpha\Psi} E'$	⊢	$D\|E \xrightarrow{\alpha\Psi} D\|E'$
Com$_3$	$D \xrightarrow{\alpha\{\psi\}} D',\ E \xrightarrow{\alpha\{\varphi\}} E'$	⊢	$D\|E \xrightarrow{\alpha\{\psi,\varphi\}} D'\|E'$
Env	$D \xrightarrow{\alpha\Psi} D', \alpha\Psi \in \mathcal{E}$	⊢	$\mathcal{E} \triangleright D \xrightarrow{\alpha\Psi} \mathcal{E} \triangleright D'$

図 7.5: 分散システムのための遷移集合 $\dot{\to}$ の推論規則

名前付演算子によってアクションにプロセス名が付加されることを除いて，基本的に図 7.4と図 7.5の規則は CCS の規則 (図 2.3) に同じである．

DS@ は抽象的なアクション (CCS では τ) をもたないが，図 7.5の同期の規則 **Com**$_3$ にみられるように，3つ以上のアクションは同期できないので，各 $\alpha\{\psi,\varphi\}$ が制御できない内部アクションに相当する．*DS*@ と *SP*@ で抽象的なアクションを導入していない理由は，充足可能性についての議論を簡単にすることと，そのような抽象的なアクションは略記法によって代用できるためである．これについては 7.4節の最後に述べる．

7.4 仕様の定義

本節では *DS*@ に対する仕様を記述するためのプロセス論理として *SP*@ を定義する．*SP*@ は 2.5節で紹介した基本的なプロセス論理にプロセス名と再帰要求とコストを導

入したプロセス論理である．コストは，離接演算子をもつ仕様 $S_1 \vee S_2$ のどちらを満たすようにシステムを合成するかを決定するための目安である．例えば S_1 と S_2 の両方が充足可能であり，S_1 の方がコストが低いならば，$S_1 \vee S_2$ からは S_1 を満たすようにシステムは合成される．

SP@ についても，**仕様定数** (または単に**定数**と呼ぶ) の集合 *Cons* が与えられていると仮定する．このとき，*SP*@ の構文を次のように与える．

定義 7.4.1 仕様 ($S, T, \cdots$ で表す) の集合 Sp は次の式を含む最小の集合である

$$\begin{aligned}
\texttt{tt} &: \text{真}, \\
\texttt{ff} &: \text{偽}, \\
A &: \text{定数 } (A \in \mathit{Cons}), \\
\langle \alpha\Psi \rangle S &: \text{可能 } (\alpha\Psi \in \mathit{Act}), \\
[\alpha\Psi] S &: \text{必然 } (\alpha\Psi \in \mathit{Act}), \\
S \wedge T &: \text{合接}, \\
S \vee T &: \text{離接}, \\
r :: S &: \text{コスト } (r : \text{正の実数}),
\end{aligned}$$

ここで，S, T はすでに Sp の要素である．演算子の結合順位を次のように定める：
{ 可能, 必然 , コスト } $>$ 合接 $>$ 離接. ∎

直観的に，仕様 $\langle \alpha\Psi \rangle S$ は，プロセス Ψ がアクション α を実行でき，その後に仕様 S を満たすことができることを要求している．一方，仕様 $[\alpha\Psi]S$ は，プロセス Ψ がアクション α を実行するならば，その後は必ず仕様 S を満たすことを要求している．

仕様定数は定義式によって意味を与えられる仕様であり，全ての定数 $A \in \mathit{Cons}$ について $A \stackrel{\text{def}}{=} S$ の定義式があるとする．仕様 S は定数 A を含むことができ，再帰的な要求を表現できる．この再帰要求は最大不動点に相当する．

コスト演算子は主に通信コストを表現するために使われる．例えば，次の仕様は，プロセス p_1 と p_2 の間の通信コスト (5) が，プロセス p_1 と p_3 の間の通信コスト (3) よりも高いことを表している．

$$5 :: \langle c\{p_1, p_2\} \rangle S \vee 3 :: \langle c\{p_1, p_3\} \rangle S$$

システムを合成するときは，コストがなるべく低くなるように，離接仕様から仕様を選ぶことが要求される．

次に，$SP@$ の意味を与えるために**要求ラベル付遷移システム** (RLTS) を定義する．

定義 7.4.2 *RLTS* は組 $\langle St, Lb, \mapsto, \rightarrow_{\Diamond}, \rightarrow_{[]} \rangle$ である．ここで，St は**状態** (**仕様**) の集合，Lb は**ラベル**の集合，$\mapsto \subseteq St \times St$ は**離接遷移**，$\longrightarrow_{\Diamond} \subseteq St \times Lb \times St$ は**可能遷移**，$\longrightarrow_{[]} \subseteq St \times Lb \times St$ は**必然遷移**である．以下，$\langle\rangle$ か $[]$ を変数 ξ で表し，$\rightarrow_{\Diamond}$ か $\rightarrow_{[]}$ を $\rightarrow_{\xi}$ と書き，$(s, s') \in \mapsto$ を $s \mapsto s'$，$(s, a, s') \in \longrightarrow_{\xi}$ を $s \xrightarrow{a}_{\xi} s'$ と書く． ∎

RLTS では各遷移が要求を表しており，仕様 $s \in St$ が満たされるとは，$s \mapsto s'$ のようなある s' について，全ての要求 $s' \xrightarrow{a}_{\xi} s''$ が満たされることである．すなわち，$s \not\mapsto$ のような s は満たすことができず，ある s' について $s \mapsto s' \not\rightarrow_{\xi}$ となる s は必ず満たされる．ここで，可能遷移 $s' \xrightarrow{a}_{\Diamond} s''$ は a を実行でき，その後は s'' を満たすことができることを要求する．一方，必然遷移 $s' \xrightarrow{a}_{[]} s''$ は a を実行するならば，その後は必ず s'' を満たすことを要求する．

次に仕様の意味を RLTS により定義する．

定義 7.4.3 仕様の意味は要求ラベル付遷移システム

$$\langle Sp, Act, \mapsto, \longrightarrow_{\Diamond}, \longrightarrow_{[]} \rangle$$

により与えられる．ここで，遷移の集合 $\mapsto$ と $\longrightarrow_{\xi}$ は各々図 7.6と図 7.7の推論規則を満たす最小の集合である． ∎

6章の離接遷移と同様に，$S \mapsto S' \mapsto S''$ ならば，$S' \equiv S''$ を証明できる (命題 6.3.1)．これは，離接遷移を 2 回以上続けて実行する必要はないことを意味している．以下，離接遷移直後の仕様の集合を Sp_0 で表し，その要素を $S_0, T_0, \cdots$ で表す．すなわち，Sp_0 は次のように与えられる: $Sp_0 = \{S_0 : \exists S \in Sp.\ S \mapsto S_0\}$．

次に分散システムの仕様に対する充足性を定義する．

定義 7.4.4 集合 $\mathcal{R} \subseteq Ds \times Sp$ が**充足部分集合**であるとは，$(D, S) \in \mathcal{R}$ ならば，ある S_0 について，$S \mapsto S_0$ かつ任意の $\alpha\Psi$ と S' に対して次の 2 条件が成り立つことである．

(i) もし $S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ ならば，ある D' について，$D \xrightarrow{\alpha\Psi} D'$ かつ $(D', S') \in \mathcal{R}$ である．
(ii) もし $S_0 \xrightarrow{\alpha\Psi}_{[]} S'$ かつ $D \xrightarrow{\alpha\Psi} D'$ ならば，$(D', S') \in \mathcal{R}$ である．

このとき，ある充足部分集合 $\mathcal{R}$ において $(D, S) \in \mathcal{R}$ ならば，D は S を満たすといい，$D \models S$ と書く． ∎

名前	仮定	⊢	結果
$\mathbf{True}_{\vee}$		$\vdash$	$\mathtt{tt} \mapsto \mathtt{tt}$
$\mathbf{Pos}_{\vee}$		$\vdash$	$\langle\alpha\Psi\rangle S \mapsto \langle\alpha\Psi\rangle S$
$\mathbf{Nec}_{\vee}$		$\vdash$	$[\alpha\Psi]S \mapsto [\alpha\Psi]S$
$\mathbf{Con}_{\vee}$	$S \mapsto S_0, T \mapsto T_0$	$\vdash$	$S \wedge T \mapsto S_0 \wedge T_0$
$\mathbf{Dis}_{1\vee}$	$S \mapsto S_0$	$\vdash$	$S \vee T \mapsto S_0$
$\mathbf{Dis}_{2\vee}$	$T \mapsto T_0$	$\vdash$	$S \vee T \mapsto T_0$
$\mathbf{Rec}_{\vee}$	$A \stackrel{\text{def}}{=} S,\ S \mapsto S_0$	$\vdash$	$A \mapsto S_0$
$\mathbf{Cos}_{\vee}$	$S \mapsto S_0$	$\vdash$	$r::S \mapsto r::S_0$

図 7.6: 離接遷移集合 $\mapsto$ の推論規則

名前	仮定	⊢	結果
Pos		$\vdash$	$\langle\alpha\Psi\rangle S \xrightarrow{\alpha\Psi}_{\Diamond} S$
Nec		$\vdash$	$[\alpha\Psi]S \xrightarrow{\alpha\Psi}_{\Box} S$
$\mathbf{Con}_1$	$S_0 \xrightarrow{\alpha\Psi}_{\xi} S'$	$\vdash$	$S_0 \wedge T_0 \xrightarrow{\alpha\Psi}_{\xi} S'$
$\mathbf{Con}_2$	$T_0 \xrightarrow{\alpha\Psi}_{\xi} T'$	$\vdash$	$S_0 \wedge T_0 \xrightarrow{\alpha\Psi}_{\xi} T'$
Cos	$S_0 \xrightarrow{\alpha\Psi}_{\xi} S'$	$\vdash$	$r::S_0 \xrightarrow{\alpha\Psi}_{\xi} S'$

図 7.7: 要求遷移集合 $\longrightarrow_{\xi}$ の推論規則

離接遷移 $\mapsto$ は 6章で紹介した離接遷移と同じ役割を果たし，定義 7.4.4は，$S \mapsto S_0$ かつ $(i), (ii)$ を満たすような S_0 が存在することを要求している．ここで，(i) は D が分散アクション $\alpha\Psi$ を実行でき，その後に S' を満たすことができることを要求し，(ii) は，もし D が分散アクション $\alpha\Psi$ を実行するならば，その後は必ず S' を満たすことを要求している．

次の命題 7.4.1はプロセス論理の基本的な性質が $SP@$ に対しても成り立つことを示している．

命題 7.4.1 次の関係が成り立つ.

(1) $D \models \mathtt{tt},\ D \not\models \mathtt{ff}$

(2) $D \models \langle\alpha\Psi\rangle S \quad \Leftrightarrow\ \exists D'.\ (D \xrightarrow{\alpha\Psi} D'$ かつ $D' \models S)$

(3) $D \models [\alpha\Psi]S \quad \Leftrightarrow\ \forall D'.\ (D \xrightarrow{\alpha\Psi} D'$ ならば $D' \models S)$

(4) $D \models S \wedge T \quad \Leftrightarrow\ D \models S$ かつ $D \models T$

(5) $D \models S \vee T \quad \Leftrightarrow\ D \models S$ または $D \models T$

(6) $D \models r::S \quad \Leftrightarrow\ D \models S$

(7) $D \models A \quad \Leftrightarrow\ \exists S.\ (D \models S$ かつ $A \stackrel{\mathrm{def}}{=} S)$

証明

(1) $\mathtt{tt} \mapsto \mathtt{tt} \not\longrightarrow_{\xi}$ であるので，tt は全ての分散システムによって満たされる．また，$\mathtt{ff} \not\mapsto$ であるので，ff を満たす分散システムは存在しない.

(2) $\langle\alpha\Psi\rangle S \xrightarrow{\alpha\Psi}_{\Diamond} S$ であるので，定義 7.4.4の (i) より，ある D' について，$D \xrightarrow{\alpha\Psi} D'$ かつ $D' \models S$ を得る.

(3) $[\alpha\Psi]S \xrightarrow{\alpha\Psi}_{[]} S$ であるので，$D \xrightarrow{\alpha\Psi} D'$ ならば，定義 7.4.4の (ii) より，$D' \models S$ を得る.

(4) $(\Rightarrow)$ の場合：$D \models S \wedge T$ ならば $D \models S$ を証明する．$\mathbf{Con}_{\vee}$ より，ある S_0 と T_0 について，$S \mapsto S_0$ かつ $T \mapsto T_0$ かつ $D \models S_0 \wedge T_0$ である．(i) $S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ とする．$\mathbf{Con}_1$ より，$S_0 \wedge T_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ を導く．すなわち，$D \models S_0 \wedge T_0$ であるので，ある D' について，$D \xrightarrow{\alpha\Psi} D'$ かつ $D' \models S'$ を得る．(ii) $S_0 \xrightarrow{\alpha\Psi}_{[]} S'$ かつ $D \xrightarrow{\alpha\Psi} D'$ とする．$\mathbf{Con}_1$ より，$S_0 \wedge T_0 \xrightarrow{\alpha\Psi}_{[]} S'$ を導く．すなわち，$D \models S_0 \wedge T_0$ であるので，$D' \models S'$ を得る．よって，$D \models S$ である.

$(\Leftarrow)$ の場合：上記の $(\Rightarrow)$ の場合と逆向きに証明できる.

(5) $(\Rightarrow)$ の場合：$D \models S \vee T$ とする．$\mathbf{Dis}_{1\vee}$ か $\mathbf{Dis}_{2\vee}$ より，ある S_0 について，$S \mapsto S_0$ かつ $D \models S_0$，または，ある T_0 について，$T \mapsto T_0$ かつ $D \models T_0$ である．すなわち，$D \models S$ または $D \models T$ である.

$(\Leftarrow)$ の場合：上記の $(\Rightarrow)$ の場合と逆向きに証明できる.

(6) と (7) については，その意味の定義より明らかである. ∎

7.3節でも述べたように，*DS*@ と *SP*@ は抽象的なアクション (CCS では τ) をもっていない．抽象的な同期アクション τ を導入しない理由は，そのようなアクションは次の略記法で代用できるためである．

$$\langle\tau\rangle S \equiv \bigvee\{\langle\alpha\Psi\rangle S : \alpha\Psi \in Act, \|\Psi\| = 2\}$$
$$[\tau]S \equiv \bigwedge\{[\alpha\Psi]S : \alpha\Psi \in Act, \|\Psi\| = 2\}$$

ここで，$\bigvee\{S_1, \cdots, S_n\} = S_1 \vee \cdots \vee S_n$，$\bigwedge\{S_1, \cdots, S_n\} = S_1 \wedge \cdots \wedge S_n$ である．

SP@ は原始的な仕様記述言語であり，現実的な仕様を直接 *SP*@ で記述することは容易ではないが，これは略記法を定義することで解決できる．例えば，分散アクション $a\Psi$ は $b\Phi$ の前に実行されるという依存関係 $(a\Psi \prec b\Phi)$ は次のように定義できる．

$$(a\Psi \prec b\Phi) \stackrel{\text{def}}{=} \bigwedge\{[\alpha\Theta](a\Psi \prec b\Phi) : \alpha\Theta \in Act - \{a\Psi, b\Phi\}\} \wedge [b\Phi]\mathtt{ff}$$

これは，$a\Psi$ と $b\Phi$ 以外の分散アクションを実行した場合は再び $(a\Psi \prec b\Phi)$ を満たすことを要求し，$a\Psi$ が実行されるまでは $b\Phi$ は実行できないことを要求している．

また，分散アクション $a\Psi$ の前に同期 1 回を許す可能要求 $\langle\!\langle\alpha\Psi\rangle\!\rangle S$ と，同期 1 回後も $a\Psi$ に対する要求が有効な必然要求 $[[\alpha\Psi]]S$ は次のように定義できる．

$$\langle\!\langle\alpha\Psi\rangle\!\rangle S \equiv \langle\alpha\Psi\rangle S \vee \langle\tau\rangle\langle\alpha\Psi\rangle S$$
$$[[\alpha\Psi]]S \equiv [\alpha\Psi]S \wedge [\tau][\alpha\Psi]S$$

ここでは，1 回に限定しているがより柔軟な仕様を記述することは可能である．このような略記法を用意することによって，システムの設計者は同期や注目しないアクションを無視した柔軟な仕様を記述することができるようになる．

7.5 充足可能性の判定法

与えられた仕様を満たすシステムを合成するために，その仕様の充足可能性 (それを満たすシステムが存在するか) を判定することが重要である．また，分散システムでは振舞いだけでなく環境 (ネットワークの結合状況など) も要求されることがある．そこで，環境 $\mathcal{E}$ のもとで仕様 S を満たす分散システムが存在するかを判定するために次のように条件付きの充足可能性を定義する．

定義 7.5.1 $\mathcal{E} \subseteq Act$ とする．仕様 S が **$\mathcal{E}$-充足可能**とは $D \models S$ かつ $env(D) = \mathcal{E}$ のような分散システム D が存在することである．∎

本節では$\mathcal{E}$-充足可能性を判定し，分散システムを合成する方法を与える．まず，7.5.1小節で充足可能性判定の基礎を説明し，$SP@$ の充足可能性の判定の難しさについて述べる．次に 7.5.2 小節で判定に必要な記法を定義し，7.5.3小節で判定アルゴリズムを与え，その正当性を示す．

7.5.1 充足可能性判定の基礎

プロセス論理における充足可能性は，各必然要求 $[a\{\psi\}]S$ を適切に各可能要求 $\langle a\{\psi\}\rangle S$ に分配して $\langle a\{\psi\}\rangle(S \wedge T)$ に書換え，可能要求によって偽 $\mathtt{ff}$ に到達できるか調べることによって判定できる．例えば，次の仕様 S_1 の可能要求 $\langle a\{p\}\rangle S_{11}$ は，プロセス p がアクション a を実行でき，その後に S_{11} を満たすことができることを要求しているが，同時に必然要求 $[a\{p\}]S_{12}$ が，p が a を実行するならば，その後に必ず S_{12} を満たすことを要求しているため，実際には S_{11} だけでなく S_{12} も満たさなければならない．

$$S_1 \equiv \langle a\{p\}\rangle S_{11} \wedge [a\{p\}]S_{12}$$
$$S_1' \equiv \langle a\{p\}\rangle(S_{11} \wedge S_{12}) \wedge [a\{p\}]S_{12}$$

つまり，S_1 の要求は S_1' の要求と同じになる．これが，必然要求 $[a\{p\}]S_{12}$ の可能要求 $\langle a\{p\}\rangle S_{11}$ への分配である．ここで，$S_{12} \equiv \mathtt{ff}$ ならば，S_1' は可能要求 $\langle a\{p\}\rangle$ の後に $\mathtt{ff}$ に到達できるので充足不能であり，S_1 も充足不能となる．

すなわち，充足可能性を判定するためには必然要求の可能要求への分配の範囲が重要である．例えば，次の S_2 の必然要求 $[a\{p\}]\mathtt{ff}$ は可能要求 $\langle a\{p\}\rangle$ に分配されるため S_2 は充足不能となるが，S_3 の $[a\{p\}]\mathtt{ff}$ は $\langle a\{p\}\rangle$ に分配されないため S_3 は充足可能である (例: $b.a.\mathbf{0}@p \models S_3$)．

$$S_2 \equiv \langle a\{p\}\rangle\mathtt{tt} \wedge [a\{p\}]\mathtt{ff}$$
$$S_3 \equiv \langle b\{p\}\rangle\langle a\{p\}\rangle\mathtt{tt} \wedge [a\{p\}]\mathtt{ff}$$

このように，プロセス名が一つであれば，合接演算子により結合された同じ深さの可能要求に分配すればよく，分配範囲を見積もることができる．しかし，プロセス名が複数あるときは，その分配範囲を見積もることは困難である[1]．例えば，次の S_4 では，必然要求 $[a\{p\}]\mathtt{ff}$ は可能要求 $\langle a\{p\}\rangle\mathtt{tt}$ に分配されるため充足不能となる．

$$S_4 \equiv \langle b\{q\}\rangle\langle a\{p\}\rangle\mathtt{tt} \wedge [a\{p\}]\mathtt{ff}$$

[1]実際には分配の範囲を適切に見積もることは不可能である．もし可能であれば充足可能性は決定可能であるが，7.6節で示すように充足可能性は決定不能である．

これは，プロセスの独立性によって，分散システムが分散アクション $b\{q\}$ を実行してもプロセス p の状態は変化しないためである．すなわち，$b\{q\}$ の実行後にプロセス p が a を実行できるならば，$b\{q\}$ の実行前にも p は a を実行できることになる．つまり，$[a\{p\}]\mathtt{ff}$ に矛盾する．

一般に，あるプロセス p に対する必然要求は他のプロセス q に対する要求を越えてプロセス p に対する可能要求に分配される．例えば，次の仕様 S_5 において，プロセス p に対する必然要求 $[a\{p\}]\mathtt{ff}$ はプロセス q に対する要求 $\langle b\{q\}\rangle[c\{q\}]\langle d\{q\}\rangle$ を越えて可能要求 $\langle a\{p\}\rangle\mathtt{tt}$ に分配される可能性がある．

$$S_5 \equiv \langle a\{p\}\rangle\mathtt{tt} \wedge \langle b\{q\}\rangle([c\{q\}]\langle d\{q\}\rangle[a\{p\}]\mathtt{ff} \wedge S_{51})$$

ただし，この仕様 S_5 は必ずも充足不能であるとは限らない．それは S_{51} が $c\{q\}$ を要求しなければ，必然要求 $[c\{q\}]$ の後の要求 $\langle d\{q\}\rangle[a\{p\}]\mathtt{ff}$ は無視されるためである．ただし，S_{51} が $c\{q\}$ を要求するかしないかは決定不能である．

7.5.2 準備

前節で説明したように複数のプロセス名があるときは，必然要求の分配範囲を見積もることは困難である．そこで，各要求の依存関係を明確にするために，プロセス名を次元とする配列を用意し，その配列に要求を割り当てる方法をとる．本小節ではそのための記法を準備する．

各 $\Psi \subseteq PN$ について，**ポインタ** ($u, v, \cdots$ で表す) の集合 Pnt_Ψ，**配列遷移** ($t, s, \cdots$ で表す) の集合 Trn_Ψ，**配列** ($\rho, \sigma, \cdots$ で表す) の集合 Ary_Ψ を次のように定義する．

$$Pnt_\Psi = \begin{cases} \{\emptyset\} & (\Psi = \emptyset) \\ \{\{(\psi; i)\} : i \in \mathcal{I}\} & (\Psi = \{\psi\}) \\ \{u \cup v : u \in Pnt_\Phi, v \in Pnt_\Theta\} & (\Psi = \Phi \cup \Theta, \Phi \cap \Theta = \emptyset) \end{cases}$$

$$Trn_\Psi = \{t : t \subseteq \mathcal{I} \times AN \times \Psi \times \mathcal{I}\}$$

$$Ary_\Psi = \{\rho : \rho \subseteq Pnt_\Psi \times Sp\}$$

ここで，$\mathcal{I}$ は整数の集合である．直観的に，ポインタの各要素 $(\psi; i)$ はプロセス ψ の一状態を表し，i はその状態を指すための ID である．配列遷移の各要素 (i, α, ψ, i') は α によるプロセス ψ の状態 i から状態 i' への遷移を表す．ポインタ $u = \{(\psi_n; i_n) : n \in N\}$ は各プロセス ψ_n の状態が i_n である分散システムの一状態を表す．配列の各要素 (u, S)

はポインタ u が示す分散システムの状態が S を満たすべきであることを表す．また，集合 Ary^0_Ψ(その要素を $\rho_0, \sigma_0, \cdots$ で表す) を Ary_Ψ の部分集合として，次のように与える．

$$Ary^0_\Psi = \{\rho_0 : \rho_0 \subseteq Pnt_\Psi \times Sp_0\}$$

ポインタ $u[v] \in Pnt_\Psi$ はポインタ $u \in Pnt_\Psi$ の一部を別のポインタ $v \in Pnt_\Phi$ で置き換えて得られるポインタであり，次のように定義される．

$$u[v] = \{(\psi; i) \in u : \psi \notin \Phi\} \cup \{(\psi; i) \in v : \psi \in \Psi\}.$$

例えば，ポインタ u と v を $u = \{(p_1; 1), (p_2; 2), (p_3; 3)\}$，$v = \{(p_2; 4), (p_5; 5)\}$ とすると，$u[v] = \{(p_1; 1), (p_2; 4), (p_3; 3)\}$ である．また，ポインタ u におけるプロセス ψ の状態 ID を $u(\psi)$ と書く．例えば，$u = \{(p; 3), (q; 4)\}$ とすると $u(p)=3$ である．

このとき，$\mathcal{E}$-充足可能性の判定に重要な概念を定義する．

定義 7.5.2 $\mathcal{E} \subseteq Act$, $\rho_0 \in Ary^0_{pn(\mathcal{E})}$, $t \in Trn_{pn(\mathcal{E})}$ とする．組 (ρ_0, t) が **$\mathcal{E}$-閉遷移付配列**であるとは，各 $(u, S_0) \in \rho_0$ について，次の 2 条件が成り立つことである．

(i) もし $S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ ならば，ある $v \in Pnt_\Psi$ について，$\alpha\Psi \in \mathcal{E}$，
$(\forall \psi \in \Psi.\, (u(\psi), \alpha, \psi, v(\psi)) \in t)$，かつ，ある S'_0 で $S' \mapsto S'_0$ かつ $(u[v], S'_0) \in \rho_0$

(ii) もし $S_0 \xrightarrow{\alpha\Psi}_{[]} S'$，$v \in Pnt_\Psi$，$\alpha\Psi \in \mathcal{E}$，かつ
$(\forall \psi \in \Psi.\, (u(\psi), \alpha, \psi, v(\psi)) \in t)$ ならば，ある S'_0 で $S' \mapsto S'_0$ かつ $(u[v], S'_0) \in \rho_0$

∎

例えば，次の (ary, trn) は (net)-閉遷移付配列である．

$$ary = \{((0,0), S_{01}), ((1,0), S_{02}), ((3,0), S_{03}), ((0,2), S_{04}), ((1,2), S_{05})\},$$
$$trn = \{(0, a, p, 1), (0, c, p, 3), (3, d, p, 0), (0, b, q, 2), (2, c, q, 0), (2, e, q, 0)\}$$
$$net = \{a\{p\}, b\{q\}, c\{p, q\}, d\{p\}, e\{q\}\}$$

ここで，ary 中の各 (i, j) はポインタ $\{(p; i), (q; j)\}$ の略記であり，各 S_i は次のように定義されている．

$$S_1 \stackrel{\text{def}}{=} S_{01} \qquad S_{01} \equiv \langle a\{p\}\rangle S_2 \wedge [b\{q\}]S_3 \wedge [d\{p\}]\mathtt{ff}$$
$$S_2 \stackrel{\text{def}}{=} S_{02} \qquad S_{02} \equiv \langle b\{q\}\rangle S_{05}$$
$$S_3 \stackrel{\text{def}}{=} S_{03} \vee S_{04} \qquad S_{03} \equiv \langle d\{p\}\rangle S_1$$
$$S_{04} \equiv 1::\langle c\{p, q\}\rangle S_{03}$$
$$S_{05} \equiv \langle e\{q\}\rangle S_2$$

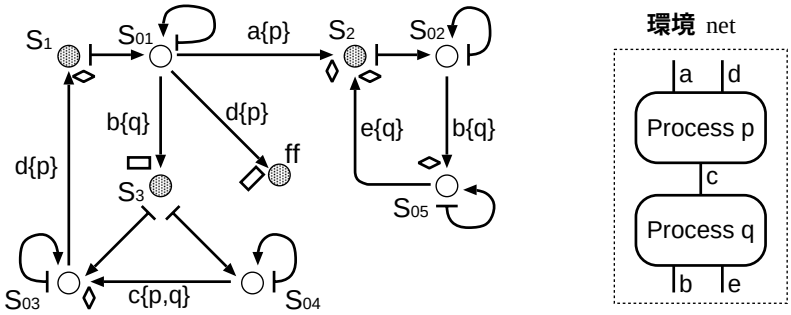

図 7.8: 仕様 S_1 の要求グラフと環境 net

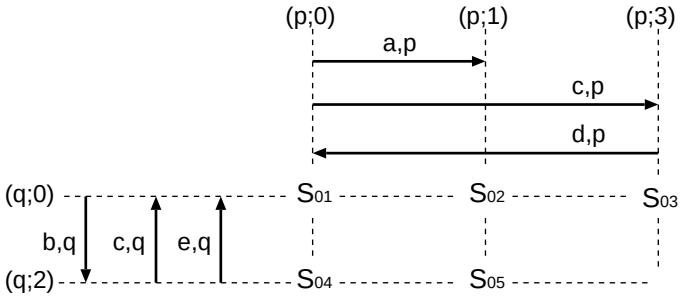

図 7.9: (net)-閉遷移配列 (ary, trn)

図 7.8は S_1 の要求グラフと環境 net を表している．また，配列 ary と遷移 trn は図 7.9のように図示できる．この遷移付配列 (ary, trn) が閉じていることは，各 $((i,j), S_0) \in ary$ が定義 7.5.2の条件 (i),(ii) を満たしていることから確認できる．例えば，$((0,0), S_{01})$ では，$S_{01} \xrightarrow{a\{p\}}_{\emptyset} S_2$ に対して，$(0,a,p,1) \in trn$，$S_2 \mapsto S_{02}$，$((1,0), S_{02}) \in ary$ である．また，$((0,2), S_{04})$ では，$S_{04} \xrightarrow{c\{p,q\}}_{\emptyset} S_{03}$ に対して，$(0,c,p,3), (2,c,q,0) \in trn$，$S_{03} \mapsto S_{03}$，$((3,0), S_{03}) \in ary$ である．

次の命題 7.5.1と命題 7.5.2は，仕様が $\mathcal{E}$-充足可能であるための必要条件と十分条件を示している．すなわち，仕様 S が $\mathcal{E}$-充足可能であることと，ある S_0 について $S \mapsto S_0$ かつ ρ_0 が S_0 を含むような $\mathcal{E}$-閉遷移付配列 (ρ_0, t) が存在することは同じである．

命題 7.5.1 もし $D \models S$ ならば，ある $(u, S_0) \in \rho_0$ について $S \mapsto S_0$ のような，$env(D)$-閉遷移付配列 (ρ_0, t) が存在する．

証明 $D_{init} \models S_{init}$ とし，$\mathcal{E} = env(D_{init})$ かつ $\Theta = \{\psi_k : k \in K\} = pn(\mathcal{E})$ とおく．こ

のとき，遷移付配列 (ρ_0, t) を次のように与える．

$$\rho_0 = \{(u, S_0) : \exists D \in Ds(D_{init}).\, D \models S_0 \in Sp_0(S_{init}), \forall(\psi_k; i) \in u.\, D \triangleleft \psi_k \equiv P_{ki}\} \in Ary_\Theta$$
$$t = \{(i, \alpha, \psi_k, i') : P_{ki} \xrightarrow{\alpha\cdot} P_{ki'}\}$$

ここで，$Ds(D_{init}) \subseteq Ds$ と $Sp_0(S_{init}) \subseteq Sp_0$ は，各々D_{init} と S_{init} から到達可能な全ての分散システムと仕様の集合であり，次のように与えられる．

$$D' \in Ds(D_{init}) \iff \exists(\alpha_i\Psi_i, D_i).\, D_{init} \xrightarrow{\alpha_1\Psi_1} D_1 \xrightarrow{\alpha_2\Psi_2} \cdots \xrightarrow{\alpha_n\Psi_n} D'$$
$$S_0 \in Sp_0(S_{init}) \iff \exists(\alpha_i\Psi_i, \xi_i, S_i, S_{0i}).\, S_{init} \mapsto S_{01} \xrightarrow[\xi_1]{\alpha_1\Psi_1} S_1 \mapsto \cdots \xrightarrow[\xi_n]{\alpha_n\Psi_n} S_n \mapsto S_0$$

また，$D \triangleleft \psi$ は分散システムから取り出されたプロセス ψ であり，次のように与えられる．

$$D \triangleleft \psi = \begin{cases} P & (D \equiv P@\psi) \\ D_i \triangleleft \psi & (D \equiv D_1|D_2, \psi \in Pn(D_i), i \in \{1,2\}) \\ D' \triangleleft \psi & (D \equiv \mathcal{E} \triangleright D') \end{cases}$$

さらに，P_{ki} は D_{init} に含まれるプロセス ψ_k が到達可能な状態の一つであり，各 $k \in K$ に対して次のように与えられる．

$$\{D \triangleleft \psi_k : D \in Ds(D_{init})\} = \{P_{ki} : i \in I_k\}$$

ここで，I_k のサイズはプロセス ψ_k が到達可能な状態数である．

まず，ある $(u, S_0) \in \rho_0$ について，$S_{init} \mapsto S_0$ であることを示す．$D_{init} \models S_{init}$ であるので，ある S_0 について，$S_{init} \mapsto S_0$ かつ $D_{init} \models S_0$ である．また，$D_{init} \in Ds(D_{init})$ であるので，各 $k \in K$ について，$D_{init} \triangleleft \psi_k \equiv P_{ki_k}$ となる $i_k \in I_k$ が存在する．そこで，$u = \{(\psi_k, i_k) : k \in K\}$ とおくと，$D_{init} \models S_0 \in Sp_0(S_{init})$，かつ各 $(\psi_k, i_k) \in u$ について $D_{init} \triangleleft \psi_k \equiv P_{ki_k}$ であるので，$(u, S_0) \in \rho_0$ を得る．

次に，(ρ_0, t) が $\mathcal{E}$-閉遷移付配列であることを示す．$(u, S_0) \in \rho_0$ とする．ここで，各 $k \in K$ について，i_k を $\{(\psi_k, i_k) : k \in K\} = u$ となるようにおく．$(u, S_0) \in \rho_0$ であるので，ある $D \in Ds(D_{init})$ について，$D \models S_0 \in Sp_0(S_{init})$，かつ各 $(\psi_k, i_k) \in u$ について，$D \triangleleft \psi_k \equiv P_{ki_k}$ である．以下，定義 7.5.2の条件 $(i), (ii)$ を満たすことを示す．

(i) $S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ とする．$D \models S_0$ であるので，ある D' について，$D \xrightarrow{\alpha\Psi} D'$ かつ $D' \models S'$ を得る．すなわち，$\alpha\Psi \in env(D) = env(D') = \mathcal{E}$ かつ $D' \in Ds(D_{init})$ である．このとき，各 $k \in K$ について，$D' \triangleleft \psi_k \equiv P_{ki'_k}$ となる i'_k が存在する．

ここで，$D \xrightarrow{\alpha\Psi} D'$ を導いた規則は **Name**, **Com**, **Env** より，各 $\psi_k \in \Psi$ について，$P_{ki_k} \xrightarrow{\alpha\cdot} P_{ki'_k}$ であり，各 $\psi_k \in \Theta - \Psi$ について，$i_k = i'_k$ である．そこで，$v = \{(\psi_k, i'_k) : \psi_k \in \Psi\} \in Pnt_\Psi$ とおくと，$u[v] = \{(\psi_k, i'_k) : \psi_k \in \Psi\} \cup \{(\psi_k, i_k) : \psi_k \in \Theta - \Psi\} = \{(\psi_k, i'_k) : k \in K\}$ を得る．また，$D' \models S'$ であるので，ある S'_0 について，$S' \mapsto S'_0$ かつ $D' \models S'_0 \in Sp_0(S_{init})$ を得る．すなわち，$D' \in Ds(D_{init})$ かつ $D' \models S'_0 \in Sp_0(S_{init})$ かつ各 $(\psi_k, i'_k) \in u[v]$ について $D' \triangleleft \psi_k \equiv P_{ki'_k}$ であるので，$(u[v], S'_0) \in \rho_0$ を得る．また，各 $\psi_k \in \Psi$ について，$P_{ki_k} \xrightarrow{\alpha\cdot} P_{ki'_k}$ であるので，$(u(\psi_k), \alpha, \psi_k, v(\psi_k)) = (i_k, \alpha, \psi_k, i'_k) \in t$ である．

(*ii*) $S_0 \xrightarrow{\alpha\Psi}_{\mathbb{I}} S'$，$v \in Pnt_\Psi$，$\alpha\Psi \in \mathcal{E}$，かつ各 $\psi_k \in \Psi$ について $(u(\psi_k), \alpha, \psi_k, v(\psi_k)) \in t$ であるとする．各 $k \in K$ について，$\psi_k \in \Psi$ ならば $v = \{(\psi_k, i'_k) : \psi_k \in \Psi\} \in Pnt_\Psi$，$\psi_k \in \Theta - \Psi$ ならば $i_k = i'_k$ となるように i'_k をおく．このとき，各 $\psi_k \in \Psi$ について，$(u(\psi_k), \alpha, \psi_k, v(\psi_k)) = (i_k, \alpha, \psi, i'_k) \in t$ であるので，$P_{ki_k} \xrightarrow{\alpha\cdot} P_{ki'_k}$ である．ここで，$\alpha\Psi \in \mathcal{E} = env(D)$ であるので，**Name**, **Com**, **Env** より，$D \xrightarrow{\alpha\Psi} D'$ かつ各 $k \in K$ について，$D' \triangleleft \psi_k \equiv P_{ki'_k}$ となる $D' \in Ds(D_{init})$ が存在する．すなわち，$D \models S_0$ かつ $S_0 \xrightarrow{\alpha\Psi}_{\mathbb{I}} S'$ であるので，$D' \models S'$ を得る．さらに，$D' \models S'$ であるので，ある S'_0 について，$S' \mapsto S'_0$ かつ $D' \models S'_0 \in Sp_0(S_{init})$ を得る．また，$u[v] = \{(\psi_k, i'_k) : \psi_k \in \Psi\} \cup \{(\psi_k, i_k) : \psi_k \in \Theta - \Psi\} = \{(\psi_k, i'_k) : k \in K\}$ である．すなわち，$(u[v], S'_0) \in \rho_0$ である．

∎

命題 7.5.2 (ρ_0, t) は $\mathcal{E}$-閉遷移付配列であるとする．もし $(u, S_0) \in \rho_0$ かつ $S \mapsto S_0$ ならば，$\mathbf{Ds}^t_{\mathcal{E}}(u) \models S$ である．ここで，分散システム $\mathbf{Ds}^t_{\mathcal{E}}(u)$ は次のように定義される：

$$\mathbf{Ds}^t_{\mathcal{E}}(u) \equiv \mathcal{E} \triangleright \prod\{\mathbf{Pr}^t_\psi(u(\psi))@\psi : \psi \in pn(\mathcal{E})\}$$
$$\mathbf{Pr}^t_\psi(i) \stackrel{\text{def}}{=} \sum\{\alpha.\mathbf{Pr}^t_\psi(i') : (i, \alpha, \psi, i') \in t\}$$

ここで，$\prod\{D_i : i \in \{1, \cdots, n\}\} \equiv D_1|\cdots|D_n$ である．

証明 次の集合 $\mathcal{R}$ が充足部分集合であることを示す．

$$\mathcal{R} = \{(\mathbf{Ds}^t_{\mathcal{E}}(u), S) : \exists S_0.\, S \mapsto S_0, (u, S_0) \in \rho_0\}$$

$(\mathbf{Ds}^t_{\mathcal{E}}(u), S) \in \mathcal{R}$ とする．すなわち，ある S_0 について，$S \mapsto S_0$ かつ $(u, S_0) \in \rho_0$ である．

(i) $S_0 \xrightarrow{\alpha\Psi}_{\langle\rangle} S'$ とする．(ρ_0, t) は $\mathcal{E}$-閉遷移付配列であるので，ある $v \in Pnt_\Psi$ と S_0' について，$\alpha\Psi \in \mathcal{E}$，$S' \mapsto S_0'$，$(u[v], S_0') \in \rho_0$，かつ各 $\psi' \in \Psi$ で，$(u(\psi'), \alpha, \psi, v(\psi')) \in t$ を得る．$\|\Psi\| = 1$ の場合は $\|\Psi\| = 2$ の場合より簡単であるので，$\Psi = \{\psi_1, \psi_2\}$ とする $(\psi_1 \neq \psi_2)$．すなわち，各 $k \in \{1,2\}$ について，$(u(\psi_k), \alpha, \psi_k, v(\psi_k)) \in t$ である．よって，**Act**, **Choice**, **Rec** より，各 $k \in \{1,2\}$ について，$\mathbf{Pr}^t_{\psi_k}(u(\psi_k)) \xrightarrow{\alpha\cdot} \mathbf{Pr}^t_{\psi_k}(v(\psi_k))$ を導ける．一方，$\psi' \in pn(\mathcal{E}) - \Psi$ ならば，$u(\psi') = u[v](\psi')$ である．よって，**Name** と $\mathbf{Com}_{1,2,3}$ より，$\prod\{\mathbf{Pr}^t_{\psi'}(u(\psi'))@\psi' : \psi' \in pn(\mathcal{E})\} \xrightarrow{\alpha\{\psi_1,\psi_2\}} \prod\{\mathbf{Pr}^t_{\psi'}(u[v](\psi'))@\psi' : \psi' \in pn(\mathcal{E})\}$ を導く．さらに，$\alpha\{\psi_1, \psi_2\} = \alpha\Psi \in \mathcal{E}$ であるので，**Env** より，$\mathbf{Ds}^t_{\mathcal{E}}(u) \xrightarrow{\alpha\Psi} \mathbf{Ds}^t_{\mathcal{E}}(u[v])$ を導ける．また，$S' \mapsto S_0'$ かつ $(u[v], S_0') \in \rho_0$ であるので，$(\mathbf{Ds}^t_{\mathcal{E}}(u[v]), S') \in \mathcal{R}$ である．

(ii) $S_0 \xrightarrow{\alpha\Psi}_{[\!]} S'$ かつ $\mathbf{Ds}^t_{\mathcal{E}}(u) \xrightarrow{\alpha\Psi} D'$ とする．$\mathbf{Ds}^t_{\mathcal{E}}(u) \xrightarrow{\alpha\Psi} D'$ を導いた最後の規則は **Env** であるので，ある D'' について，$\prod\{\mathbf{Pr}^t_{\psi'}(u(\psi'))@\psi' : \psi' \in pn(\mathcal{E})\} \xrightarrow{\alpha\Psi} D''$ かつ $\alpha\Psi \in \mathcal{E}$ かつ $D' \equiv \mathcal{E} \triangleright D''$ を得る．上記の (i) の場合と同様に，$\Psi = \{\psi_1, \psi_2\}$ の場合を示す．この遷移は **Name** と $\mathbf{Com}_{1,2,3}$ によって導かれるので，ある P_1' と P_2' について，$\mathbf{Pr}^t_{\psi_1}(u(\psi_1)) \xrightarrow{\alpha\cdot} P_1'$ かつ $\mathbf{Pr}^t_{\psi_2}(u(\psi_2)) \xrightarrow{\alpha\cdot} P_2'$ かつ $D'' \equiv \prod\{\mathbf{Pr}^t_{\psi'}(u(\psi'))@\psi' : \psi' \in pn(\mathcal{E}) - \{\psi_1, \psi_2\}\}|P_1'@\psi_1|P_2'@\psi_2$ を得る．さらに，**Act**, $\mathbf{Choice}_{1,2}$, **Rec** より，各 $k \in \{1,2\}$ ついて，$P_k' \equiv \mathbf{Pr}^t_{\psi_k}(i_k')$ かつ $(u(\psi_k), \alpha, \psi_k, i_k') \in t$ となる i_k' がある．ここで，$v = \{(\psi_1; i_1'), (\psi_2; i_2')\}$ とおくと，各 $k \in \{1,2\}$ ついて，$(u(\psi_k), \alpha, \psi_k, v(\psi_k)) \in t$ である．すなわち，$(u, S_0) \in \rho_0$ かつ $S_0 \xrightarrow{\alpha\Psi}_{[\!]} S'$ かつ $\alpha\Psi \in \mathcal{E}$ であり，(ρ_0, t) は $\mathcal{E}$-閉遷移付配列であるので，ある S_0' について，$S' \mapsto S_0'$ かつ $(u[v], S_0') \in \rho_0$ を得る．また，$D' \equiv \mathcal{E} \triangleright D'' \equiv \mathcal{E} \triangleright \prod\{\mathbf{Pr}^t_{\psi'}(u[v](\psi'))@\psi' : \psi' \in pn(\mathcal{E})\} \equiv \mathbf{Ds}^t_{\mathcal{E}}(u[v])$ である．よって，$S' \mapsto S_0'$ かつ $(u[v], S_0') \in \rho_0$ であるので，$(D', S') \equiv (\mathbf{Ds}^t_{\mathcal{E}}(u[v]), S') \in \mathcal{R}$ を得る．

∎

命題 7.5.2は仕様を満たす分散システムの合成方法を与えている．例えば，前述の図 7.8の仕様 S_1 について，$S_1 \mapsto S_{01}$ かつ $((0,0), S_{01}) \in ary$ であり，(ary, trn) は (net)-閉遷移付配列であるので，命題 7.5.2より，次の分散システム $\mathbf{Ds}^{trn}_{net}(0,0)$ は S_1 を満たす．

$$\begin{aligned}
\mathbf{Ds}^{trn}_{net}(0,0) &\equiv net \triangleright (\mathbf{Pr}^{trn}_p(0)@p|\mathbf{Pr}^{trn}_q(0)@q) \\
\mathbf{Pr}^{trn}_p(0) &\stackrel{\text{def}}{=} a.\mathbf{0} + c.d.\mathbf{Pr}^{trn}_p(0) \\
\mathbf{Pr}^{trn}_q(0) &\stackrel{\text{def}}{=} b.(c.\mathbf{Pr}^{trn}_q(0) + e.\mathbf{Pr}^{trn}_q(0))
\end{aligned}$$

7.5.3 アルゴリズム

各環境 $\mathcal{E} \subseteq Act$ について，アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ を図 7.10のように与える．$\mathbf{Sat}^{\mathcal{E}}$ は与えられた仕様 S を満たす (環境が $\mathcal{E}$ の) 分散システムが存在するか (S の $\mathcal{E}$-充足可能性) を判定するアルゴリズムである．入力は判定される仕様であり，出力は組 (b, ρ_0, t) である．ここで，$b \in \{\mathtt{tt}, \mathtt{ff}\}$，$\rho_0 \in Ary^0_{pn(\mathcal{E})}$，$t \in Trn_{pn(\mathcal{E})}$ である．本小節では，アルゴリズム $\mathbf{Sat}^{\mathcal{E}}(S)$ の出力 b が真 $\mathtt{tt}$ ならば，そのときに限り S は $\mathcal{E}$-充足可能性であることを証明する．また，このとき，(ρ_0, t) は $\mathcal{E}$-閉遷移付配列となる．

充足可能性を判定するときに問題になるのは，(1) 離接を含む仕様 $S \vee T$ のどちらを選択して充足可能性を判定するか，(2) 再帰要求をどこで畳み込むかである．本アルゴリズムでは，離接と再帰を処理するためにバックトラックを用いている．すなわち，ある程度予測して離接の選択や再帰の畳み込みを行い，その予測が適切でなかった場合は戻って判定しなおす方法をとっている．以下，まずアルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ について説明し，その適用例を示す．次に，アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ を試験的に実装したことを報告する．そして，アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ の出力が適切であることを示す定理を与える．

アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ の概要

アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ は $\mathbf{Dis}^{\mathcal{E}}$, $\mathbf{Rec}^{\mathcal{E}}$, $\mathbf{Pos}^{\mathcal{E}}$, $\mathbf{Nec}^{\mathcal{E}}$ の 4 つのサブアルゴリズムから構成されている．サブアルゴリズムに対する入力組 $(\rho_0, t, \sigma(\text{ または } \sigma_0))$ の意味は次のとおりである: 配列 ρ_0 にはすでに可能遷移を処理された仕様が含まれており，配列 σ (または σ_0) には離接遷移 (または可能遷移) がまだ処理されていない仕様が含まれている．そして，t には ρ_0 に含まれている仕様の可能遷移によって生成された遷移が含まれている．出力の意味は $\mathbf{Sat}^{\mathcal{E}}$ と同じである．以下，各サブアルゴリズムについて説明する．ここで，$(\sharp_n)$ は図 7.10の行を指している．

$\mathbf{Dis}^{\mathcal{E}}$ は，もし処理すべき仕様がないのであれば $\mathtt{tt}$ を出力し $(\sharp_1)$，もし離接遷移をもたない仕様があれば $\mathtt{ff}$ を出力する $(\sharp_2)$．さもなければ，$\mathbf{Dis}^{\mathcal{E}}$ は $\mathcal{E}$-充足可能な仕様の集合 Γ_0 から (順序 $\ll_D$ に関して) 最小の配列 σ_0 を選択する $(\sharp_{3,4,5})$．ここで，$\longmapsto\!\!\!\!\!\rightarrow$ は，離接遷移 $\mapsto$ を配列上に拡張した遷移集合であり，次の 3 条件を満たす最小の関係である．

(1) $\emptyset \longmapsto\!\!\!\!\!\rightarrow \emptyset$ である．
(2) $S \mapsto S_0$ ならば $\{(u, S)\} \longmapsto\!\!\!\!\!\rightarrow \{(u, S_0)\}$ である．
(3) $\rho \longmapsto\!\!\!\!\!\rightarrow \rho_0$ かつ $\sigma \longmapsto\!\!\!\!\!\rightarrow \sigma_0$ かつ $\rho \cap \sigma = \emptyset$ ならば $\rho \cup \sigma \longmapsto\!\!\!\!\!\rightarrow \rho_0 \cup \sigma_0$ である．

$\mathbf{Sat}^{\mathcal{E}}(S) = \mathbf{Dis}^{\mathcal{E}}(\emptyset, \emptyset, \sigma)$, where $\sigma = \{\{((\psi; 0) : \psi \in pn(\mathcal{E})\}, S)\}$.

$\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma) = (b, \rho_0', t')$, where
- if $\sigma = \emptyset$, then $(b, \rho_0', t') := (\mathtt{tt}, \rho_0, t)$, ($\sharp_1$)
- else if $\sigma \not\mapsto\!\!\!\!\rightarrow$, then $(b, \rho_0', t') := (\mathtt{ff}, \rho_0, t)$, ($\sharp_2$)
- else $(b, \rho_0', t') := \mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0 - \rho_0)$, where
  - $\sigma_0 \in \Gamma_0 := \{\sigma_0' : \sigma \mapsto\!\!\!\!\rightarrow \sigma_0'\}$, ($\sharp_3$)
  - $(check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0 - \rho_0)) = \mathtt{tt}$ or $(\forall \sigma_0' \in \Gamma_0.\ \sigma_0' \ll_D \sigma_0))$, ($\sharp_4$)
  - $(\forall \sigma_0' \in \Gamma_0$ such that $\sigma_0' \ll_D \sigma_0.\ check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0' - \rho_0)) = \mathtt{ff})$. ($\sharp_5$)

$\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0) = \mathbf{Pos}^{\mathcal{E}}(g(\rho_0), g(t), g(\sigma_0) - g(\rho_0))$, where
- $g \in G := \{g' : \forall(\psi, i).\ (g'(\psi, i) = i$ or $\exists(u, S_0) \in \sigma_0.\ \exists u'.$
  $(u', S_0) \in \rho_0 \cup \sigma_0, u(\psi) = i, u'(\psi) = g'(\psi, i))\}$, ($\sharp_6$)
- $(check(\mathbf{Pos}^{\mathcal{E}}(g(\rho_0), g(t), g(\sigma_0) - g(\rho_0))) = \mathtt{tt}$ or $(\forall g' \in G.\ g' \ll_R g))$, ($\sharp_7$)
- $(\forall g' \in G$ such that $g' \ll_R g.\ check(\mathbf{Pos}^{\mathcal{E}}(g'(\rho_0), g'(t), g'(\sigma_0) - g'(\rho_0))) = \mathtt{ff})$. ($\sharp_8$)

$\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0) = (b, \rho_0', t')$, where
- if $\{\alpha\Psi : \exists(u, S_0) \in \sigma_0.\ \exists S'.\ S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'\} - \mathcal{E} \neq \emptyset$, then $(b, \rho_0', t') = (\mathtt{ff}, \rho_0, t)$, ($\sharp_9$)
- else $(b, \rho_0', t') = \mathbf{Nec}^{\mathcal{E}}(\rho_0 \cup \sigma_0, t \cup t'', \sigma)$, where
  - $t'' := \{(u(\psi), \alpha, \psi, v(\psi)) : \exists S'.\ \exists \Psi.\ (u, \alpha\Psi, v, S') \in new, \psi \in \Psi\}$, ($\sharp_{10}$)
  - $\sigma := \{(u[v], S') : \exists \alpha\Psi.\ (u, \alpha\Psi, v, S') \in new\}$, ($\sharp_{11}$)
  - $new := \{(u, \alpha\Psi, v, S') : \exists S_0.\ (u, S_0) \in \sigma_0, S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S',$
    $v = \{(\psi; newid(\rho_0 \cup \sigma_0, u, \alpha\Psi, S')) : \psi \in \Psi\}\}$. ($\sharp_{12}$)

$\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma) = \mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma \cup \sigma'')$, where
- $\sigma' := \{(u[v], S') : \exists S_0.\ \exists \alpha\Psi \in \mathcal{E}.\ (u, S_0) \in \rho_0, S_0 \xrightarrow{\alpha\Psi}_{[\!]} S', v \in Pnt_{\Psi},$
  $(\forall \psi \in \Psi.\ (u(\psi), \alpha, \psi, v(\psi)) \in t)\}$, ($\sharp_{13}$)
- $\sigma'' := \{(u, S') \in \sigma' : \forall S_0'$ such that $S' \mapsto S_0'.\ (u, S_0') \notin \rho_0\}$. ($\sharp_{14}$)

図 7.10: $\mathcal{E}$-充足可能性を判定するためのアルゴリズム $\mathbf{Sat}^{\mathcal{E}}$

もし Γ_0 に含まれる全ての σ_0' が $\mathcal{E}$-充足不能ならば，最大の σ_0' が選択される ($\sharp_4$)．この順序 $\ll_D$ については，$\mathbf{Rec}^{\mathcal{E}}$ の順序 $\ll_R$ とともに後で説明する．また，($\sharp_5$) の関数 $check : \{\mathtt{tt}, \mathtt{ff}\} \times Ary^0_{pn(\mathcal{E})} \times Trn_{pn(\mathcal{E})} \to \{\mathtt{tt}, \mathtt{ff}\}$ は，アルゴリズムの出力 (b, ρ_0, t) から真偽値 b を取り出す関数である．

$\mathbf{Rec}^{\mathcal{E}}$ は再帰プロセスを生成するために，配列 $(\rho_0 \cup \sigma_0)$ を畳み込むことを試みる．再

帰要求があるとき，この畳み込みなしにアルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ は決して停止しない．関数 $g : PN \times \mathcal{I} \to \mathcal{I}$ はリナンバリング関数であり，プロセス ψ の状態 i を j にリナンバリングする関数を $(\psi; i \to j)$ と書く．この関数は次のように，ポインタ，配列，配列遷移上に拡張される:

$$
\begin{aligned}
g(u) &= \{(\psi; g(\psi, i)) : (\psi; i) \in u\}, \\
g(\rho) &= \{(g(u), S) : (u, S) \in \rho\}, \\
g(t) &= \{(g(\psi, i), \alpha, \psi, g(\psi, i')) : (i, \alpha, \psi, i') \in t\}.
\end{aligned}
$$

集合 G は，$i \neq g'(\psi, i)$ ならば「**プロセス ψ の状態 i と状態 $g'(\psi, i)$ の両方が同じ仕様を満たす** $(*)$」ようなリナンバリング関数 g' の集合である $(\sharp_6)$．しかし，離接によって多くの異なるプロセスが同じ仕様を満たすことができるので，条件 $(*)$ はプロセス ψ の二つの状態が同じであるための必要条件であるが十分条件ではない．すなわち，畳み込みに失敗したときのために，リナンバリング関数の集合 G は必ず不変関数 id(全ての (ψ, i) で $id(\psi, i) = i$) を含んでいる．集合 G から一つの g を選択するときは，$\mathcal{E}$-充足可能になる (順序 $\ll_R$ に関して) 最小のリナンバリング関数 g を選択する $(\sharp_{7,8})$．もし G に含まれる全ての g' が $\mathcal{E}$-充足不能ならば，最大の g' が選択される $(\sharp_7)$.

$\mathbf{Pos}^{\mathcal{E}}$ は，まず環境 $\mathcal{E}$ によって禁止されているアクションが要求されていないかをチェックする $(\sharp_9)$．もしそのようなアクションが要求されている場合は `ff` を出力する．要求されていない場合は，$\mathbf{Pos}^{\mathcal{E}}$ は σ_0 に含まれている仕様の可能遷移にしたがって新しく遷移を遷移集合 t に追加する $(\sharp_{10})$．そして，定義 7.5.2の条件 (i) を満たすように，その可能遷移の後に要求される仕様を σ にセットする $(\sharp_{11})$．集合 new は σ_0 によって追加される遷移情報 $(u, \alpha\Psi, v)$ とその後に要求される仕様 S' の組の集合である $(\sharp_{12})$．ここで，関数 $newid$ は $\rho_0 \cup \sigma_0$ で使われていない新しい固有の整数を各 $(u, \alpha\Psi, S')$ に対して割り当てる関数である.

$\mathbf{Nec}^{\mathcal{E}}$ は t に含まれる遷移によって導かれる全ての必然遷移を探索し，定義 7.5.2の条件 (ii) を満たすように，その必然遷移の後に要求される仕様を追加する $(\sharp_{13})$．ただし，すでに処理されている要求は σ から削除される $(\sharp_{14})$.

$\mathbf{Dis}^{\mathcal{E}}$ と $\mathbf{Rec}^{\mathcal{E}}$ の順序 $\ll_D$ と $\ll_R$ は，各々集合 Γ_0 と G から要素を取り出すために使われる $(\sharp_{5,8})$．順序 $\ll_D$ と $\ll_R$ による要素の選択方法は後に示す定理 7.5.3と定理 7.5.4に無関係であるので，それらの順序を注意深く定める必要はない．ただし，次のことが考慮されるべきである.

- 順序 $\ll_D$ は低いコストの仕様が選ばれるように定義されるべきである．例えば，各仕様 $S_0 \in Sp_0$ のコスト $Cost(S_0)$ は次のように定義される．

$$Cost(\mathtt{tt}) = Cost(\langle\alpha\Psi\rangle S) = Cost([\alpha\Psi]S) = 0$$
$$Cost(S_0 \wedge T_0) = Cost(S_0) + Cost(T_0)$$
$$Cost(r::S_0) = r + Cost(S_0)$$

- 順序 $\ll_R$ は，再帰プロセスを合成してアルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ を停止させるために，なるべく多くの整数を畳み込めるようなリナンバリング関数が優先して選ばれるように定義されるべきである．

アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ の適用例

アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ を 7.5.2小節で紹介した仕様の例 S_1 (図 7.8) に適用した結果を図 7.11に示す．この出力 $(\mathtt{tt}, \rho'_{05}, t'_5)$ は，仕様 S_1 が $\mathcal{E}$-充足可能であることを示しており，(ρ'_{05}, t'_5) は 7.5.2小節で与えられた (net)-閉遷移付配列 (ary, trn) と同じである．

図 7.11においてステップ 10, 11, 16が重要である．ステップ 10では，要素 $((0,2), S_3)$ が $S_{01} \xrightarrow{b\{q\}}_{[]} S_3$ と $S_{02} \xrightarrow{b\{q\}}_{\langle\rangle} S_{05}$ により導かれる．これは，S_{01} の必然要求 $[b\{q\}]S_3$ を S_{02} の可能要求 $\langle b\{q\}\rangle S_{05}$ に分配することに相当する．ここで，S_{01} と S_{02} の間にはプロセス p に対する要求 $\langle a\{p\}\rangle$ があるが，プロセス q に対する要求は S_{01} と S_{02} の間で保存されているため，このような分配が必要である．

ステップ 11では，S_{03} のコスト 0 よりも $S_{04} \equiv 1::\langle c\{p,q\}\rangle S_{03}$ のコスト 1 の方が高いが，S_3 からは S_{04} が選択されている．これは，S_{03} が選択されると $[d\{p\}]\mathtt{ff}$ と矛盾するためにである．充足可能であるためには，コストが高くなっても同期 $c\{p,q\}$ をもつ S_{04} を選択する必要がある．

ステップ 16では，g_1 によってプロセス q の状態 ID が 4 から 0 に変更されている．これは，$((1,4), S_{02}) \in \sigma_{05}$ かつ $((1,0), S_{02}) \in \rho_{04}$ が示すように，ポイント $(1,4)$ と $(1,0)$ で同じ仕様 S_{02} を満たすことが要求されているためである．このリナンバリングにより，再帰プロセスが生成され，$((1,4), S_{02})$ は σ_{05} から削除される．

アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ の実装

アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ の有効性を調べるために，Java 言語を用いて充足可能性判定ツールを試験的に実装した．定義したクラスは 27 個，総行数は約 2400 行である．

1	$\mathbf{Sat}^{net}(S_1)$	
2	$= \mathbf{Dis}^{net}(\emptyset, \emptyset, \sigma_1)$	$\sigma_1 := \{((0,0), S_1)\}$
3	$= \mathbf{Rec}^{net}(\emptyset, \emptyset, \sigma_{01})$	$\sigma_{01} := \{((0,0), S_{01})\}$
4	$= \mathbf{Pos}^{net}(\emptyset, \emptyset, \sigma_{01})$	
5	$= \mathbf{Nec}^{net}(\sigma_{01}, t_1, \sigma_2)$	$t_1 = \{(0, a, p, 1)\}, \sigma_2 := \{((1,0), S_2)\}$
6	$= \mathbf{Dis}^{net}(\sigma_{01}, t_1, \sigma_2)$	
7	$= \mathbf{Rec}^{net}(\sigma_{01}, t_1, \sigma_{02})$	$\sigma_{02} := \{((1,0), S_{02})\}$
8	$= \mathbf{Pos}^{net}(\sigma_{01}, t_1, \sigma_{02})$	
9	$= \mathbf{Nec}^{net}(\rho_{02}, t_2, \sigma_3)$	$\rho_{02} := \sigma_{01} \cup \sigma_{02}, t_2 := t_1 \cup \{(0, b, q, 2)\}, \sigma_3 := \{((1,2), S_{05}\}$
10	$= \mathbf{Dis}^{net}(\rho_{02}, t_2, \sigma_4)$	$\sigma_4 := \sigma_3 \cup \{((0,2), S_3)\}$
11	$= \mathbf{Rec}^{net}(\rho_{02}, t_2, \sigma_{04})$	$\sigma_{04} := \sigma_3 \cup \{((0,2), S_{04})\}$
12	$= \mathbf{Pos}^{net}(\rho_{02}, t_2, \sigma_{04})$	
13	$= \mathbf{Nec}^{net}(\rho_{04}, t_4, \sigma_5)$	$\rho_{04} := \rho_{02} \cup \sigma_{04}, t_4 := t_2 \cup \{(0, c, p, 3), (2, c, q, 3), (2, e, q, 4)\}$
14	$= \mathbf{Dis}^{net}(\rho_{04}, t_4, \sigma_5)$	$\sigma_5 := \{((3,3), S_{03}), ((1,4), S_2)\}$
15	$= \mathbf{Rec}^{net}(\rho_{04}, t_4, \sigma_{05})$	$\sigma_{05} := \{((3,3), S_{03}), ((1,4), S_{02})\}, g_1 := (q; 4 \to 0)$
16	$= \mathbf{Pos}^{net}(\rho_{04}, t'_4, \sigma'_{05})$	$t'_4 := g_1(t_4), \sigma'_{05} := g_1(\sigma_{05}) - g_1(\rho_{04}) = \{((3,3), S_{03})\}$
17	$= \mathbf{Nec}^{net}(\rho_{05}, t_5, \sigma_6)$	$\rho_{05} := \rho_{04} \cup \sigma'_{05}, t_5 := t'_4 \cup \{(3, d, p, 5)\}, \sigma_6 := \{((5,3), S_1)\}$
18	$= \mathbf{Dis}^{net}(\rho_{05}, t_5, \sigma_6)$	
19	$= \mathbf{Rec}^{net}(\rho_{05}, t_5, \sigma_{06})$	$\sigma_{06} := \{((5,3), S_{01})\}, g_2 := (p; 5 \to 0)(q; 3 \to 0)$
20	$= \mathbf{Pos}^{net}(\rho'_{05}, t'_5, \emptyset)$	$\rho'_{05} := g_2(\rho_{05}), t'_5 := g_2(t_5), g_2(\sigma_{06}) - \rho'_{05} = \emptyset$
21	$= \mathbf{Nec}^{net}(\rho'_{05}, t'_5, \emptyset)$	
21	$= \mathbf{Dis}^{net}(\rho'_{05}, t'_5, \emptyset)$	
22	$= (\mathtt{tt}, \rho'_{05}, t'_5)$	

図 7.11: 仕様 S_1 へのアルゴリズム $\mathbf{Sat}^{net}$ の適用例

図 7.12にそのツールの実行画面を示す．横長のウィンドウが 6 つのボタンをもつコントローラで，もう一方が情報を表示するウィンドウである．機能は仕様の読み込み (仕様はテキストファイルとして入力)，充足可能性の判定，分散システムの合成，ログの保存，ログのクリアである．図 7.12の情報表示ウィンドウには，7.5.2小節で用いた仕様 S_1(図 7.8) の充足可能性を判定し，分散システムを合成した結果が表示されている．充足可能性の判定結果には，閉遷移付配列 (Ary, Trn) の他，バックトラックした回数 (1 回) と計算時間 (0.22 秒) が表示されている．1 回バックトラックした理由は，S_3 から S_{03} か S_{04} を選択するときに，通信をしない S_{03} を優先して選択したためである．実際には仕様 S_1 を満たすためには通信は必要なため，S_{04} に選択しなおすためにバックトラックが起こっている．尚，図 7.12で得られている閉遷移付配列 (Ary, Trn)

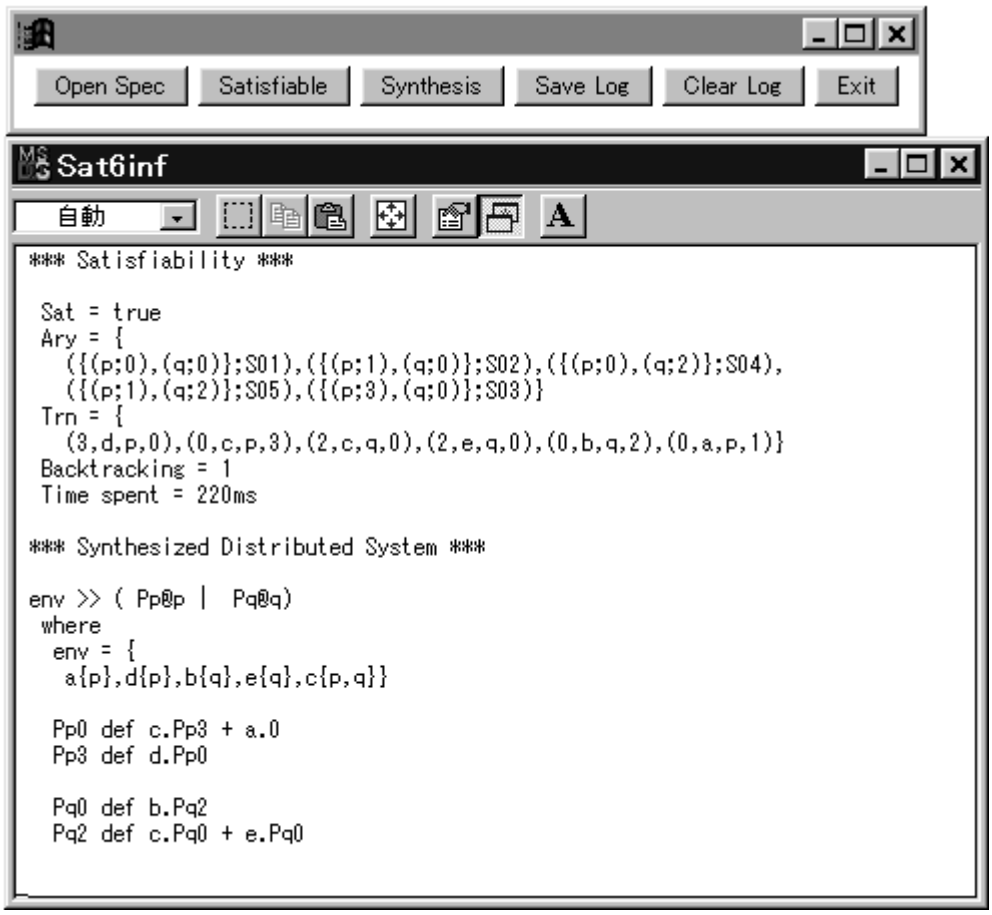

図 7.12: 充足可能性判定ツール実行画面

は 7.5.2小節で示した図 7.9の閉遷移付配列 (ary, trn) に同じである．

アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ の完全性と健全性

アルゴリズム $\mathbf{Sat}^{\mathcal{E}}(S)$ の出力が真 $\mathtt{tt}$ であるならば，そのときに限り，仕様 S は $\mathcal{E}$-充足可能であることを示す定理 7.5.3と定理 7.5.4を与える．

定理 7.5.3 仕様 S は $\mathcal{E}$-充足可能であり，$\mathbf{Sat}^{\mathcal{E}}(S)$ が停止するならば，$check(\mathbf{Sat}^{\mathcal{E}}(S)) = \mathtt{tt}$ である．

証明 $\mathcal{E} \subseteq Act$，$\rho_0, \rho_0', \sigma, \sigma_0 \in Ary_{pn(\mathcal{E})}$，$t, t' \in Trn_{\Psi}$，$h$ はリナンバリング関数とする．まず，$h(\rho_0) \subseteq \rho_0'$，$h(\sigma_0) \subseteq \rho_0'$，$h(t) \subseteq t'$，$int(t) \subseteq int(\rho_0 \cup \sigma_0)$，かつ (ρ_0', t') は $\mathcal{E}$-閉遷移付配列と仮定したとき，次の 4 つの補題が成り立つことを証明する．ここで，$int(t)$ と $int(\rho)$ は各々遷移 t と配列 ρ に含まれている全ての整数 (状態 ID) の集合である．

(1) もし $\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma)$ が停止し，$\sigma \longmapsto\!\!\!\!\rightarrow \sigma_0$ ならば，$check(\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma)) = \mathtt{tt}$ である．

(2) もし $\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0)$ が停止するならば，$check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0)) = \mathtt{tt}$ である．

(3) もし $\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0)$ が停止するならば，$check(\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0)) = \mathtt{tt}$ である．

(4) もし $\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma)$ が停止し，$\sigma \longmapsto\!\!\!\!\rightarrow \sigma_0$ ならば，$check(\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma)) = \mathtt{tt}$ である．

以下，適用されたサブアルゴリズムの数に関する帰納法を用いて，$(1), \cdots, (4)$ を証明する．

(1) $\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma)$ が停止し，$\sigma \longmapsto\!\!\!\!\rightarrow \sigma_0$ とする．

- $\sigma = \emptyset$ の場合：$(\sharp_1)$ より，$\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma) = (\mathtt{tt}, \rho_0, t)$ である．すなわち，$check(\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma)) = \mathtt{tt}$ である．
- その他の場合：仮定 $\sigma \longmapsto\!\!\!\!\rightarrow \sigma_0$ より $(\sharp_2)$ の場合はない．すなわち $(\sharp_{3,4,5})$ より，ある σ_0' について，$\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma) = \mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0' - \rho_0)$ である．ここで，σ_0' は次の条件を満たす配列である．

  $(*_1)$ $\sigma_0' \in \Gamma_0 = \{\sigma_0'' : \sigma \longmapsto\!\!\!\!\rightarrow \sigma_0''\}$

  $(*_2)$ $check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0' - \rho_0)) = \mathtt{tt}$，または全ての $\sigma_0'' \in \Gamma_0$ で $\sigma_0'' \ll_D \sigma_0'$

  $(*_3)$ $\sigma_0'' \ll_D \sigma_0'$ のような全ての $\sigma_0'' \in \Gamma_0$ で $check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0'' - \rho_0)) = \mathtt{ff}$

  ここで，$\sigma \longmapsto\!\!\!\!\rightarrow \sigma_0$ であるので，$(*_1)$ より $\sigma_0 \in \Gamma_0$ である．σ_0 と σ_0' の大小関係について次の 3 つの場合に分けて示す．

  (a) $\sigma_0 \ll_D \sigma_0'$ の場合：$(*_3)$ より，$check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0 - \rho_0)) = \mathtt{ff}$ である．しかし，$h(\rho_0) \subseteq \rho_0'$，$h(\sigma_0 - \rho_0) \subseteq \rho_0'$，$h(t) \subseteq t'$，$int(t) \subseteq int(\rho_0 \cup (\sigma_0 - \rho_0))$ であるので，帰納法の仮定 (2) より，$check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0 - \rho_0)) = \mathtt{tt}$ を得る．すなわち，矛盾するため $\sigma_0 \ll_D \sigma_0'$ の場合はない．

  (b) $\sigma_0 = \sigma_0'$ の場合：$h(\rho_0) \subseteq \rho_0'$，$h(\sigma_0 - \rho_0) \subseteq \rho_0'$，$h(t) \subseteq t'$，$int(t) \subseteq int(\rho_0 \cup (\sigma_0 - \rho_0))$ であるので，帰納法の仮定 (2) より，$check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0' - \rho_0)) = check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0 - \rho_0)) = \mathtt{tt}$ を得る．

  (c) $\sigma_0' \ll_D \sigma_0$ の場合：σ_0' よりも大きい σ_0 が存在するので，$(*_2)$ より，$check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0' - \rho_0)) = \mathtt{tt}$ を得る．

  よって，$check(\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma)) = check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0' - \rho_0)) = \mathtt{tt}$ である．

(2) $\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0)$ が停止するとする．$\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0)$ の定義より，$\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0) = \mathbf{Pos}^{\mathcal{E}}(g(\rho_0), g(t), g(\sigma_0) - g(\rho_0))$ である．ここで，g は次の条件を満たすリナンバリング関数である．

($*_1$) $g \in G = \{g' : \forall(\psi, i).\ (g'(\psi, i) = i$ または
$\exists(u, S_0) \in \sigma_0.\ \exists u'.\ (u', S_0) \in \rho_0 \cup \sigma_0, u(\psi) = i, u'(\psi) = g'(\psi, i)\}$

($*_2$) $check(\mathbf{Pos}^{\mathcal{E}}(g(\rho_0), g(t), g(\sigma_0) - g(\rho_0))) = \mathtt{tt}$，または
全ての $g' \in G$ で $g' \ll_R g$.

($*_3$) $g' \ll_R g$ のような全ての $g' \in G$ で，
$check(\mathbf{Pos}^{\mathcal{E}}(g'(\rho_0), g'(t), g'(\sigma_0) - g'(\rho_0))) = \mathtt{ff}$.

ここで，各 (ψ, i) について $id(\psi, i) = i$ となるような不変関数 id は必ず G に含まれている．id と g の大小関係について次の 3 つの場合に分けて示す．

(a) $id \ll_R g$ の場合：($*_3$) より $check(\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0 - \rho_0)) = \mathtt{ff}$ である．しかし，$h(\rho_0) \subseteq \rho_0'$, $h(\sigma_0 - \rho_0) \subseteq \rho_0'$, $h(t) \subseteq t'$, $int(t) \subseteq int(\rho_0 \cup (\sigma_0 - \rho_0))$ であるので，帰納法の仮定 (3) より，$check(\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0 - \rho_0)) = \mathtt{tt}$ である．すなわち，矛盾するため $id \ll_R g$ の場合はない．

(b) $id = g$ の場合：$h(\rho_0) \subseteq \rho_0'$, $h(\sigma_0 - \rho_0) \subseteq \rho_0'$, $h(t) \subseteq t'$, $int(t) \subseteq int(\rho_0 \cup (\sigma_0 - \rho_0))$ であるので，帰納法の仮定 (3) より，$check(\mathbf{Pos}^{\mathcal{E}}(g(\rho_0), g(t), g(\sigma_0) - g(\rho_0))) = check(\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0 - \rho_0)) = \mathtt{tt}$ を得る．

(c) $g \ll_R id$ の場合：($*_2$) より $check(\mathbf{Pos}^{\mathcal{E}}(g(\rho_0), g(t), g(\sigma_0) - g(\rho_0))) = \mathtt{tt}$ を得る．

よって，$check(\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0)) = check(\mathbf{Pos}^{\mathcal{E}}(g(\rho_0), g(t), g(\sigma_0) - g(\rho_0))) = \mathtt{tt}$ である．

(3) $\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0)$ が停止するとする．$(u, S_0) \in \sigma_0$ とすると，$h(\sigma_0) \subseteq \rho_0'$ より，$(h(u), S_0) \in \rho_0'$ である．また，(ρ_0', t') は $\mathcal{E}$-閉遷移付配列であるので，もし $S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ ならば，$\alpha\Psi \in \mathcal{E}$ ある．すなわち，$\{\alpha\Psi : \exists(u, S_0) \in \sigma_0.\ \exists S'.\ S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'\} \subseteq \mathcal{E}$ である．よって，($\sharp_9$) の場合はないので，$\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0) = \mathbf{Nec}^{\mathcal{E}}(\rho_0 \cup \sigma_0, t \cup t'', \sigma')$

である．ここで，t'' と σ' は次のように与えられる．

$$(*_1) \quad t'' = \{(u(\psi), \alpha, \psi, v(\psi)) : \exists S'.\, \exists \Psi.\, (u, \alpha\Psi, v, S') \in new, \psi \in \Psi\}$$
$$(*_2) \quad \sigma' = \{(u[v], S') : \exists \alpha\Psi.\, (u, \alpha\Psi, v, S') \in new\},$$
$$(*_3) \quad new = \{(u, \alpha\Psi, v, S') : \exists S_0.\, (u, S_0) \in \sigma_0, S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S',$$
$$v = \{(\psi; newid(\rho_0 \cup \sigma_0, u, \alpha\Psi, S')) : \psi \in \Psi\}\ \}$$

まず，$\{(u_i, \alpha_i\Psi_i, v_i, S_i') : i \in I\} = new$ となるように，I, u_i, $\alpha_i\Psi_i$, v_i, S_i' をおき，さらに各 $i \in I$ について，$n_i = newid(\rho_0 \cup \sigma_0, u_i, \alpha_i\Psi_i, S_i')$ とする．このとき，各 $i \in I$ について，$(*_3)$ より，$v_i = \{(\psi, n_i) : \psi \in \Psi_i\}$ かつ $(u_i, S_{0i}) \in \sigma_0$ かつ $S_{0i} \xrightarrow{\alpha_i\Psi_i}_{\Diamond} S_i'$ となる S_{0i} が存在する．ここで，各 $i \in I$ について，$u_i' = h(u_i)$ とおくと，$h(\sigma_0) \subseteq \rho_0'$ より，$(u_i', S_{0i}) = (h(u_i), S_{0i}) \in \rho_0'$ である．すなわち，(ρ_0', t') は $\mathcal{E}$-閉遷移付配列であるので，各 $i \in I$ について，ある $v_i' \in Pnt_{\Psi_i}$ と S_{0i}' について，$\alpha_i\Psi_i \in \mathcal{E}$，$S_i' \mapsto S_{0i}'$，$(u_i'[v_i'], S_{0i}') \in \rho_0'$，かつ各 $\psi' \in \Psi_i$ について，$(u_i'(\psi'), \alpha_i, \psi', v_i'(\psi')) \in t'$ を得る．

次に，リナンバリング関数 h' を $h' = h(\psi; n_i \to v_i'(\psi))_{\psi \in \Psi_i, i \in I}$ とおき，配列 σ_0' を $\sigma_0' = \{(u_i[v_i], S_{0i}') : i \in I\}$ とおく．このとき，各 $i \in I$ について，$h'(u_i) = u_i'$, $h'(v_i) = v_i'$, $\sigma' \mapsto\!\!\!\to \sigma_0'$, $h'(\rho_0 \cup \sigma_0) \subseteq \rho_0'$, $h'(\sigma_0') \subseteq \rho_0'$, $h'(t \cup t'') \subseteq t'$, $int(t \cup t'') \subseteq int((\rho_0 \cup \sigma_0) \cup \sigma_0')$ が成り立つことを順に示す．

(a) 各 $i \in I$ について，$(u_i, S_{0i}) \in \sigma_0$ であり，関数 $newid$ は $\rho_0 \cup \sigma_0$ で使われていない新しい整数 n_i $(i \in I)$ を割り当てるので，$h'(u_i) = h(u_i) = u_i'$ である．また，$h'(v_i) = h'(\{(\psi, n_i) : \psi \in \Psi_i\}) = \{(\psi, v_i'(\psi)) : \psi \in \Psi_i\} = v_i'$ である．

(b) 各 $i \in I$ について，$S_i' \mapsto S_{0i}'$ であるので，$(*_2)$ より，$\sigma' = \{(u_i[v_i], S_i') : i \in I\} \mapsto\!\!\!\to \{(u_i[v_i], S_{0i}') : i \in I\} = \sigma_0'$ を導ける．

(c) 関数 $newid$ は $\rho_0 \cup \sigma_0$ で使われていない新しい整数を割り当てるので，$h'(\rho_0) = h(\rho_0) \subseteq \rho_0'$，$h'(\sigma_0) = h(\sigma_0) \subseteq \rho_0'$ である．すなわち，$h'(\rho_0 \cup \sigma_0) \subseteq \rho_0'$ である．

(d) $(u, S_0') \in h'(\sigma_0')$ とすると，$\sigma_0' = \{(u_i[v_i], S_{0i}') : i \in I\}$ であるので，ある $j \in I$ で，$u = h'(u_j[v_j])$ かつ $S_0' \equiv S_{0j}'$ である．すなわち，$(u, S_0') = (h'(u_j[v_j]), S_{0j}') = (h'(u_j)[h'(v_j)], S_{0j}') = (u_j'[v_j'], S_{0j}') \in \rho_0'$ を得る．

(e) 関数 $newid$ は $\rho_0 \cup \sigma_0$ で使われていない新しい整数を割り当て，$int(t) \subseteq int(\rho_0 \cup \sigma_0)$ であるので，$h'(t) = h(t) \subseteq t'$ である．また，$(j, \alpha, \psi, j') \in h'(t'')$

とすると，$(*_1)$ より，ある $i \in I$ で，$j = h'(u_i(\psi))$，$j' = h'(v_i(\psi))$，$\alpha_i = \alpha$，$\psi \in \Psi_i$ である．すなわち，$(j, \alpha, \psi, j') = (h'(u_i(\psi)), \alpha_i, \psi, h'(v_i(\psi))) = (u'_i(\psi), \alpha_i, \psi, v'_i(\psi)) \in t'$ である．よって，$h'(t \cup t'') \subseteq t'$ である．

(f) $j \in int(t'')$ とする．$(*_1)$ より，ある ψ と $i \in I$ で，$j = u_i(\psi)$ または $j = v_i(\psi)$ である．もし $j = u_i(\psi)$ ならば，$(u_i, S_{0i}) \in \sigma_0$ であるので，$j \in int(\sigma_0)$ である．一方，もし $j = v_i(\psi)$ ならば，$(u_i[v_i], S'_{0i}) \in \sigma'_0$ であるので，$j \in int(\sigma'_0)$ である．すなわち，$int(t'') \subseteq int(\sigma_0 \cup \sigma'_0)$ である．また，仮定より $int(t) \subseteq int(\rho_0 \cup \sigma_0)$ であるので，$int(t \cup t'') \subseteq int((\rho_0 \cup \sigma_0) \cup \sigma'_0)$ を得る．

以上，$\sigma' \longmapsto \sigma'_0$, $h'(\rho_0 \cup \sigma_0) \subseteq \rho'_0$, $h'(\sigma'_0) \subseteq \rho'_0$, $h'(t \cup t'') \subseteq t'$, $int(t \cup t'') \subseteq int((\rho_0 \cup \sigma_0) \cup \sigma'_0)$ であるので，帰納法の仮定 (4) より，$check(\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0)) = check(\mathbf{Nec}^{\mathcal{E}}(\rho_0 \cup \sigma_0, t \cup t'', \sigma')) = \mathtt{tt}$ を得る．

(4) $\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma)$ が停止し，$\sigma \longmapsto \sigma_0$ であるとする．$\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma)$ の定義より，$\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma) = \mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma \cup \sigma')$ である．ここで，σ' は次のような配列である．

$$\sigma' = \{(u[v], S') : \exists S_0.\, \exists \alpha\Psi \in \mathcal{E}.\, (u, S_0) \in \rho_0, S_0 \xrightarrow{\alpha\Psi}_{[\!]} S', v \in Pnt_\Psi,$$
$$(\forall \psi \in \Psi.\, (u(\psi), \alpha, \psi, v(\psi)) \in t), (\forall S'_0 \in \{S''_0 : S' \mapsto S''_0\}.\, (u[v], S'_0) \notin \rho_0)\}$$

このとき，ある σ'_0 について，$\sigma' \longmapsto \sigma'_0$ かつ $h(\sigma'_0) \subseteq \rho'_0$ を示す．このためには，「各 $(u[v], S') \in \sigma'$ について，$S' \mapsto S'_0$ かつ $(h(u[v]), S'_0) \in \rho'_0$ となる S'_0 がある」$(*)$ ことを示せば十分である．

$(u[v], S') \in \sigma'$ とする．すなわち，ある S_0 と $\alpha\Psi \in \mathcal{E}$ について，$(u, S_0) \in \rho_0$，$S_0 \xrightarrow{\alpha\Psi}_{[\!]} S'$，$v \in Pnt_\Psi$，かつ各 $\psi \in \Psi$ で，$(u(\psi), \alpha, \psi, v(\psi)) \in t$ であり，$S' \mapsto S'_0$ のような各 S'_0 について，$(u[v], S'_0) \notin \rho_0$ である．ここで，$u' = h(u) \in Pnt_{pn(\mathcal{E})}$ とおくと $(u', S_0) = (h(u), S_0) \in h(\rho_0) \subseteq \rho'_0$ である．さらに，$v' = h(v) \in Pnt_\Psi$ とおくと，各 $\psi \in \Psi$ について，$(u(\psi), \alpha, \psi, v(\psi)) \in t$ であるので，$(u'(\psi), \alpha, \psi, v'(\psi)) = (h(u)(\psi), \alpha, \psi, h(v)(\psi)) = (h(u(\psi)), \alpha, \psi, h(v(\psi))) \in h(t) \subseteq t'$ を得る．すなわち，$(u', S_0) \in \rho'_0$，$S_0 \xrightarrow{\alpha\Psi}_{[\!]} S'$，$\alpha\Psi \in \mathcal{E}$，$v' \in Pnt_\Psi$，かつ各 $\psi \in \Psi$ について $(u'(\psi), \alpha, \psi, v'(\psi)) \in t')$ であり，(ρ'_0, t') は $\mathcal{E}$-閉遷移付配列であるので，ある S'_0 について，$S' \mapsto S'_0$ かつ $(u'[v'], S'_0) \in \rho_0$ を得る．すなわち，$(h(u[v]), S'_0) = (h(u)[h(v)], S'_0) = (u'[v'], S'_0) \in \rho'_0$ である．$(*)$ の証明完了．

よって，ある σ'_0 について，$\sigma' \longmapsto \sigma'_0$ かつ $h(\sigma'_0) \subseteq \rho'_0$ が得られる．さらに，仮定より $h(\sigma_0) \subseteq \rho'_0$ であるので，$\sigma \cup \sigma' \longmapsto \sigma_0 \cup \sigma'_0$ かつ $h(\sigma_0 \cup \sigma'_0) \subseteq \rho'_0$ で

ある．すなわち，$\sigma \cup \sigma' \longmapsto\!\!\!\!\rightarrow \sigma_0 \cup \sigma'_0$，$h(\rho_0) \subseteq \rho'_0$，$h(\sigma_0 \cup \sigma'_0) \subseteq \rho'_0$，$h(t) \subseteq t'$，$int(t) \subseteq int(\rho_0 \cup (\sigma_0 \cup \sigma'_0))$ であるので，帰納法の仮定 (1) より $\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma) = check(\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma \cup \sigma')) = \mathtt{tt}$ を得る．

以上，4 つの補題を証明した．最後に本定理を証明する．S は $\mathcal{E}$-充足可能であり，$\mathbf{Sat}^{\mathcal{E}}(S)$ は停止するとする．すなわち，$D \models S$ かつ $env(D) = \mathcal{E}$ のような D が存在する．よって，命題 7.5.1より，ある $(u, S_0) \in \rho_0$ で $S \mapsto S_0$ のような $\mathcal{E}$-閉遷移付配列 (ρ_0, t) が存在する．ここで，$u_{init} = \{(\psi; 0) : \psi \in pn(\mathcal{E})\}$，$\sigma_{init} = \{(u_{init}, S)\}$，$\sigma_{0init} = \{(u_{init}, S_0)\}$ とおくと，$S \mapsto S_0$ であるので，$\sigma_{init} \longmapsto\!\!\!\!\rightarrow \sigma_{0init}$ を得る．また，$\mathbf{Sat}^{\mathcal{E}}(S)$ の定義より，$\mathbf{Sat}^{\mathcal{E}}(S) = \mathbf{Dis}^{\mathcal{E}}(\emptyset, \emptyset, \sigma_{init})$ である．ここで，$h(u_{init}) = u$ となるようにリナンバリング関数 h をおくと，$h(\sigma_{0init}) = \{(h(u_{init}), S_0)\} = \{(u, S_0)\} \subseteq \rho_0$ である．すなわち，$\sigma_{init} \longmapsto\!\!\!\!\rightarrow \sigma_{0init}$, $h(\emptyset) \subseteq \rho_0$, $h(\sigma_{0init}) \subseteq \rho_0$, $h(\emptyset) \subseteq t$, $int(\emptyset) \subseteq int(\sigma_{0init})$ であるので，上記の補題 (1) より，$check(\mathbf{Dis}^{\mathcal{E}}(\emptyset, \emptyset, \sigma_{init})) = \mathtt{tt}$ を得る． ∎

定理 7.5.4 もし $\mathbf{Sat}^{\mathcal{E}}(S)$ が停止し，かつ $\mathbf{Sat}^{\mathcal{E}}(S) = (\mathtt{tt}, \rho_0, t)$ ならば，(ρ_0, t) は $\mathcal{E}$-閉遷移付配列であり，$S \mapsto S_0$ となる $(u, S_0) \in \rho_0$ が存在する．

証明 $\mathcal{E} \subseteq Act$，$\rho_0, \sigma, \sigma_0 \in Ary_{pn(\mathcal{E})}$，$t \in Trn_{\Psi}$ とする．まず，$\mathcal{E}$-閉遷移付配列の定義を次のように拡張する：遷移付配列 (ρ_0, t) が $(\mathcal{E}, \sigma, \sigma_0)$-閉遷移付配列であるとは，各 $(u, S_0) \in \rho_0$ について，次の 2 条件 $(i), (ii)$ が成り立つことである．

(i') もし $S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ ならば，ある $v \in Pnt_{\Psi}$ について，$\alpha\Psi \in \mathcal{E}$，
$(\forall \psi \in \Psi.\, (u(\psi), \alpha, \psi, v(\psi)) \in t)$ であり，下記の (c_1) または (c_2) が成り立つ．

(ii') もし $S_0 \xrightarrow{\alpha\Psi}_{[]} S'$，$v \in Pnt_{\Psi}$，$\alpha\Psi \in \mathcal{E}$，
$(\forall \psi \in \Psi.\, (u(\psi), \alpha, \psi, v(\psi)) \in t)$ ならば，下記の (c_1) または (c_2) が成り立つ．

(c_1) $(u[v], S') \in \sigma$ である．

(c_2) ある S'_0 について，$S' \mapsto S'_0$ かつ $(u[v], S'_0) \in \rho_0 \cup \sigma_0$ である．

また，(ρ_0, t) が $(\mathcal{E}, \sigma, \sigma_0)$-半閉遷移付配列であるとは，各 $(u, S_0) \in \rho_0$ について，上記の条件 (i) が成り立つことである．この定義より，$(\mathcal{E}, \emptyset, \emptyset)$-閉遷移付配列と $\mathcal{E}$-閉遷移付配列は同じである．このとき，次の 4 つの補題を証明する．

(1) もし $\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma)$ が停止して $\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma) = (\mathtt{tt}, \rho'_0, t')$ であり，かつ (ρ_0, t) が $(\mathcal{E}, \sigma, \emptyset)$-閉遷移付配列ならば，$(\rho'_0, t')$ は $\mathcal{E}$-閉遷移付配列であり，$\rho_0 \subseteq_{sp} \rho'_0$ かつある σ_0 について，$\sigma \longmapsto\!\!\!\!\rightarrow \sigma_0 \subseteq_{sp} \rho'_0$ である．

(2) もし $\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0)$ が停止して $\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0) = (\mathtt{tt}, \rho_0', t')$ であり，かつ (ρ_0, t) が $(\mathcal{E}, \emptyset, \sigma_0)$-半閉遷移付配列ならば，$(\rho_0', t')$ は $\mathcal{E}$-閉遷移付配列であり，$\rho_0 \cup \sigma_0 \subseteq_{sp} \rho_0'$ である．

(3) もし $\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0)$ が停止して $\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0) = (\mathtt{tt}, \rho_0', t')$ であり，かつ (ρ_0, t) が $(\mathcal{E}, \emptyset, \sigma_0)$-半閉遷移付配列ならば，$(\rho_0', t')$ は $\mathcal{E}$-閉遷移付配列であり，$\rho_0 \cup \sigma_0 \subseteq_{sp} \rho_0'$ である．

(4) もし $\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma)$ が停止して $\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma) = (\mathtt{tt}, \rho_0', t')$ であり，かつ (ρ_0, t) が $(\mathcal{E}, \sigma, \emptyset)$-半閉遷移付配列ならば，$(\rho_0', t')$ は $\mathcal{E}$-閉遷移付配列であり，$\rho_0 \subseteq_{sp} \rho_0'$ である．

ここで，$\subseteq_{sp}$ は配列に含まれる仕様の包含関係であり，次のように定義される．

$$\rho \subseteq_{sp} \sigma \iff \{S : \exists u.\ (u, S) \in \rho\} \subseteq \{S : \exists u.\ (u, S) \in \sigma\}$$

以下，適用されたサブアルゴリズムの数に関する帰納法により，$(1), \cdots, (4)$ を証明する．

(1) $\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma)$ は停止して $\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma) = (\mathtt{tt}, \rho_0', t')$ であり，かつ (ρ_0, t) が $(\mathcal{E}, \sigma, \emptyset)$-閉遷移付配列であるとする．

- $\sigma = \emptyset$ の場合：$(\sharp_1)$ より $(\mathtt{tt}, \rho_0', t') = (\mathtt{tt}, \rho_0, t)$ である．このとき，(ρ_0, t) は $\mathcal{E}$-閉遷移付配列であり，$\rho_0 = \rho_0' \subseteq_{sp} \rho_0'$ かつ $\sigma = \emptyset \mapsto\!\!\!\!\!\rightarrow \emptyset \subseteq_{sp} \rho_0'$ である．
- $\sigma \not\mapsto\!\!\!\!\!\rightarrow$ の場合：$(\sharp_2)$ より $(\mathtt{tt}, \rho_0', t') = (\mathtt{ff}, \rho_0, t)$ であるが，これは矛盾している．すなわち，$\sigma \mapsto\!\!\!\!\!\rightarrow$ である．
- その他の場合 $(\sigma \neq \emptyset$ かつ $\sigma \mapsto\!\!\!\!\!\rightarrow)$：$(\sharp_3)$ より，ある σ_0 について，$(\mathtt{tt}, \rho_0', t') = \mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma) = \mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0 - \rho_0)$ かつ $\sigma \mapsto\!\!\!\!\!\rightarrow \sigma_0$ である．以下，(ρ_0, t) が $(\mathcal{E}, \emptyset, \sigma_0 - \rho_0)$-半閉遷移付配列であることを示すため，$(u, S_0) \in \rho_0$ とする．

  (i') $S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ とする．(ρ_0, t) は $(\mathcal{E}, \sigma, \emptyset)$-閉遷移付配列であるので，ある $v \in Pnt_\Psi$ について，$\alpha\Psi \in \mathcal{E}$，かつ全ての $\psi \in \Psi$ について $(u(\psi), \alpha, \psi, v(\psi)) \in t$ であり，$(u[v], S') \in \sigma$ または，ある S_0' について，$S' \mapsto S_0'$ かつ $(u[v], S_0') \in \rho_0$ である．

  - $(u[v], S') \in \sigma$ の場合：$\sigma \mapsto\!\!\!\!\!\rightarrow \sigma_0$ であるので，ある S_0' について，$S' \mapsto S_0'$ かつ $(u[v], S_0') \in \sigma_0$ である．すなわち，$(u[v], S_0') \in \sigma_0 \subseteq \rho_0 \cup \sigma_0 = \rho_0 \cup (\sigma_0 - \rho_0)$ を得る．

– 上記以外の場合：ある S_0' について，$S' \mapsto S_0'$ かつ $(u[v], S_0') \in \rho_0 \subseteq \rho_0 \cup \sigma_0 = \rho_0 \cup (\sigma_0 - \rho_0)$ である．

よって，(ρ_0, t) は $(\mathcal{E}, \emptyset, \sigma_0 - \rho_0)$-半閉遷移付配列である．ここで，$(\mathtt{tt}, \rho_0', t') = \mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0 - \rho_0)$ であるので，帰納法の仮定 (2) より，(ρ_0', t') は $\mathcal{E}$-閉遷移付配列であり，$\rho_0 \cup \sigma_0 = \rho_0 \cup (\sigma_0 - \rho_0) \subseteq_{sp} \rho_0'$ を得る．さらに，$\rho_0 \subseteq_{sp} \rho_0 \cup \sigma_0 \subseteq_{sp} \rho_0'$ である．

(2) $\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0)$ が停止して $\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0) = (\mathtt{tt}, \rho_0', t')$ であり，(ρ_0, t) は $(\mathcal{E}, \emptyset, \sigma_0)$-半閉遷移付配列であるとする．$\mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0)$ の定義より，あるリナンバリング関数 g で，$(\mathtt{tt}, \rho_0', t') = \mathbf{Rec}^{\mathcal{E}}(\rho_0, t, \sigma_0) = \mathbf{Pos}^{\mathcal{E}}(g(\rho_0), g(t), g(\sigma_0) - g(\rho_0))$ である．以下，$(g(\rho_0), g(t))$ が $(\mathcal{E}, \emptyset, g(\sigma_0) - g(\rho_0))$-半閉遷移付配列であることを示すため，$(u', S_0) \in g(\rho_0)$ とする．すなわち，ある u で，$g(u) = u'$ かつ $(u, S_0) \in \rho_0$ である．

(i') $S_0 \xrightarrow{\alpha\Psi}_{\emptyset} S'$ とする．このとき，(ρ_0, t) は $(\mathcal{E}, \emptyset, \sigma_0)$-半閉遷移付配列であるので，ある $v \in Pnt_\Psi$ について，$\alpha\Psi \in \mathcal{E}$，かつ全ての $\psi \in \Psi$ について，$(u(\psi), \alpha, \psi, v(\psi)) \in t$ であり，ある S_0' について，$S' \mapsto S_0'$ かつ $(u[v], S_0') \in \rho_0 \cup \sigma_0$ である．すなわち，$v' = g(v)$ とすると，$(u'[v'], S_0') = (g(u[v], S_0') \in g(\rho_0 \cup \sigma_0) = g(\rho_0) \cup g(\sigma_0) = g(\rho_0) \cup (g(\sigma_0) - g(\rho_0))$ を得る．また，各 $\psi \in \Psi$ についても，$(u'(\psi), \alpha, \psi, v'(\psi)) = (g(u(\psi)), \alpha, \psi, g(v(\psi))) \in g(t)$ である．

よって，$(g(\rho_0), g(t))$ は $(\mathcal{E}, \emptyset, g(\sigma_0) - g(\rho_0))$-半閉遷移付配列である．すなわち，$(\mathtt{tt}, \rho_0', t') = \mathbf{Pos}^{\mathcal{E}}(g(\rho_0), g(t), g(\sigma_0) - g(\rho_0))$ であるので，帰納法の仮定 (3) より，(ρ_0', t') は $\mathcal{E}$-閉遷移付配列であり，$g(\rho_0) \cup (g(\sigma_0) - g(\rho_0)) \subseteq_{sp} \rho_0'$ を得る．さらに，$\rho_0 \cup \sigma_0 \subseteq_{sp} g(\rho_0) \cup (g(\sigma_0) - g(\rho_0)) \subseteq_{sp} \rho_0'$ である．

(3) $\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0)$ が停止して $\mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0) = (\mathtt{tt}, \rho_0', t')$ であり，かつ (ρ_0, t) が $(\mathcal{E}, \emptyset, \sigma_0)$-半閉遷移付配列であるとする．もし $(\sharp_9)$ の条件が成り立てば，$(\mathtt{tt}, \rho_0', t') = (\mathtt{ff}, \rho_0, t)$ となり矛盾する．すなわち，$(\mathtt{tt}, \rho_0', t') = \mathbf{Pos}^{\mathcal{E}}(\rho_0, t, \sigma_0) = \mathbf{Nec}^{\mathcal{E}}(\rho_0 \cup \sigma_0, t \cup t'', \sigma')$ である．ここで，t'' と σ' は次のように与えられる．

$$
\begin{aligned}
t'' &= \{(u(\psi), \alpha, \psi, v(\psi)) : \exists S'.\, \exists \Psi.\, (u, \alpha\Psi, v, S') \in new, \psi \in \Psi\} \\
\sigma' &= \{(u[v], S') : \exists \alpha\Psi.\, (u, \alpha\Psi, v, S') \in new\}, \\
new &= \{(u, \alpha\Psi, v, S') : \exists S_0.\, (u, S_0) \in \sigma_0, S_0 \xrightarrow{\alpha\Psi}_{\emptyset} S', \\
&\qquad\qquad v = \{(\psi; newid(\rho_0 \cup \sigma_0, u, \alpha\Psi, S')) : \psi \in \Psi\}\ \}
\end{aligned}
$$

以下，$(\rho_0 \cup \sigma_0, t \cup t'')$ が $(\mathcal{E}, \sigma', \emptyset)$-半閉遷移付配列であることを示すため，$(u, S_0) \in \rho_0 \cup \sigma_0$ とする．

(i') $S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ とする．

- $(u, S_0) \in \rho_0$ の場合：(ρ_0, t) が $(\mathcal{E}, \emptyset, \sigma_0)$-半閉遷移付配列であるので，ある $v \in Pnt_\Psi$ で，$\alpha\Psi \in \mathcal{E}$ かつ全ての $\psi \in \Psi$ について，$(u(\psi), \alpha, \psi, v(\psi)) \in t$ であり，ある S_0' で，$S' \mapsto S_0'$ かつ $(u[v], S_0') \in \rho_0 \cup \sigma_0$ を得る．
- $(u, S_0) \in \sigma_0$ の場合：$(\sharp_9)$ より，$S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ であるので，$\alpha\Psi \in \mathcal{E}$ である．$v = \{(\psi; newid(\rho_0 \cup \sigma_0, u, \alpha\Psi, S')) : \psi \in \Psi\} \in Pnt_\Psi$ とおくと，$(u, \alpha\Psi, v, S') \in new$ である．すなわち，$(u[v], S') \in \sigma'$ である．また，各 $\psi \in \Psi$ について，$(u(\psi), \alpha, \psi, v(\psi)) \in t''$ を得る．

以上，$\alpha\Psi \in \mathcal{E}$ かつ $v \in Pnt_\Psi$ かつ各 $\psi \in \Psi$ で $(u(\psi), \alpha, \psi, v(\psi)) \in t \cup t''$ であり，$(u[v], S') \in \sigma'$，または，ある S_0' について $S' \mapsto S_0'$ かつ $(u[v], S_0') \in \rho_0 \cup \sigma_0$ である．

よって，$(\rho_0 \cup \sigma_0, t \cup t'')$ は $(\mathcal{E}, \sigma', \emptyset)$-半閉遷移付配列である．すなわち，$\mathbf{Nec}^{\mathcal{E}}(\rho_0 \cup \sigma_0, t \cup t'', \sigma') = (\mathtt{tt}, \rho_0', t')$ であるので，帰納法の仮定 (4) より，(ρ_0', t') は $\mathcal{E}$-閉遷移付配列であり，$\rho_0 \cup \sigma_0 \subseteq_{sp} \rho_0'$ を得る．

(4) $\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma)$ が停止して $\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma) = (\mathtt{tt}, \rho_0', t')$ であり，(ρ_0, t) が $(\mathcal{E}, \sigma, \emptyset)$-半閉遷移付配列であるとする．$\mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma)$ の定義より，ある σ' で，$(\mathtt{tt}, \rho_0', t') = \mathbf{Nec}^{\mathcal{E}}(\rho_0, t, \sigma) = \mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma \cup \sigma')$ である．ここで，σ' は次のような配列である．

$$\sigma' = \{(u[v], S') : \exists S_0.\, \exists \alpha\Psi \in \mathcal{E}.\, (u, S_0) \in \rho_0, S_0 \xrightarrow{\alpha\Psi}_{[\!]} S', v \in Pnt_\Psi,$$
$$(\forall \psi \in \Psi.\, (u(\psi), \alpha, \psi, v(\psi)) \in t), (\forall S_0' \in \{S_0'' : S' \mapsto S_0''\}.\, (u[v], S_0') \notin \rho_0)\}$$

以下，(ρ_0, t) が $(\mathcal{E}, \sigma \cup \sigma', \emptyset)$- 閉遷移付配列であることを示すため，$(u, S_0) \in \rho_0$ とする．

(i') $S_0 \xrightarrow{\alpha\Psi}_{\Diamond} S'$ とする．(ρ_0, t) は $(\mathcal{E}, \sigma, \emptyset)$-半閉遷移付配列であるので，ある $v \in Pnt_\Psi$ で，$\alpha\Psi \in \mathcal{E}$ かつ各 $\psi \in \Psi$ で，$(u(\psi), \alpha, \psi, v(\psi)) \in t$ であり，$(u[v], S') \in \sigma$，またはある S_0' で，$S' \mapsto S_0'$ かつ $(u[v], S_0') \in \rho_0$ である．

(ii') $S_0 \xrightarrow{\alpha\Psi}_{[\!]} S'$，$\alpha\Psi \in \mathcal{E}$，$v \in Pnt_\Psi$，全ての $\psi \in \Psi$ について $(u(\psi), \alpha, \psi, v(\psi)) \in t$ であるとする．このとき，$S' \mapsto S_0'$ のような全ての S_0' について $(u[v], S_0') \notin$

ρ_0 ならば，$(u[v], S') \in \sigma'$ である．すなわち，$(u[v], S') \in \sigma'$，または，ある S'_0 について，$S' \mapsto S'_0$ かつ $(u[v], S'_0) \in \rho_0$ である．

よって，(ρ_0, t) は $(\mathcal{E}, \sigma \cup \sigma', \emptyset)$- 閉遷移付配列である．すなわち，$\mathbf{Dis}^{\mathcal{E}}(\rho_0, t, \sigma \cup \sigma') = (\mathtt{tt}, \rho'_0, t')$ であるので，帰納法の仮定 (1) より，(ρ'_0, t') は $\mathcal{E}$-閉遷移付配列であり，$\rho_0 \subseteq_{sp} \rho'_0$ を得る．

以上，4つの補題を証明した．最後に本定理を証明する．$\mathbf{Sat}^{\mathcal{E}}(S)$ は停止して $\mathbf{Sat}^{\mathcal{E}}(S) = (\mathtt{tt}, \rho_0, t)$ とする．$\mathbf{Sat}^{\mathcal{E}}(S)$ の定義より，$(\mathtt{tt}, \rho_0, t) = \mathbf{Sat}^{\mathcal{E}}(S) = \mathbf{Dis}^{\mathcal{E}}(\emptyset, \emptyset, \sigma_{init})$ である．ここで，$\sigma_{init} = \{(u_{init}, S)\}$ かつ $u_{init} = \{(\psi; 0) : \psi \in pn(\mathcal{E})\}$ である．明らかに，$(\emptyset, \emptyset)$ は $(\mathcal{E}, \sigma_{init}, \emptyset)$-閉遷移付配列である．ここで，上記の補題 (1) より，(ρ_0, t) は $\mathcal{E}$-閉遷移付配列であり，ある σ_{01} について，$\sigma_{init} \longmapsto \sigma_{01} \subseteq_{sp} \rho_0$ を得る．また，$\sigma_{init} = \{(u_{init}, S)\} \longmapsto \sigma_{01}$ であるので，ある S_0 について，$S \mapsto S_0$ かつ $\sigma_{01} = \{(u_{init}, S_0)\}$ である．すなわち，$(u_{init}, S_0) \in \sigma_{01} \subseteq_{sp} \rho_0$ である． ∎

命題 7.5.2と定理 7.5.4より，アルゴリズム $\mathbf{Sat}^{\mathcal{E}}(S)$ の出力が真 $\mathtt{tt}$ であれば，$\mathbf{Ds}^t_{\mathcal{E}}(u) \models S$ を得ることができるので，S は $\mathcal{E}$-充足可能である．すなわち，定理 7.5.3と定理 7.5.4 は $\mathcal{E}$-充足可能に対する，アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ の完全性と健全性を示している．ただし，状態数が有限な仕様 S に対してさえ，このアルゴリズム $\mathbf{Sat}^{\mathcal{E}}(S)$ の停止性は保証されていない．このアルゴリズムの停止性については 7.6節で詳細に述べる．

7.6 充足可能性の決定不能性

7.5節で，仕様の充足可能性を判定するアルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ を与えたが，その停止性は保証されていなかった．本節では，まずそのアルゴリズムが停止しない例を示す．

例 7.6.1 有限状態仕様 INF を次のように定義する．

$$
\begin{aligned}
INF &\equiv SQ \wedge EQ \wedge \langle a\{p\}\rangle(\langle b\{p\}\rangle \mathtt{tt} \wedge [a\{p\}]\mathtt{ff}),\\
SQ &\stackrel{\text{def}}{\equiv} [a\{p\}]\langle a\{q\}\rangle\langle a\{q\}\rangle SQ \wedge [b\{p\}](\langle b\{q\}\rangle \mathtt{tt} \wedge [a\{q\}]\mathtt{ff}),\\
EQ &\stackrel{\text{def}}{\equiv} [a\{q\}]\langle a\{p\}\rangle EQ \wedge [b\{q\}](\langle b\{p\}\rangle \mathtt{tt} \wedge [a\{p\}]\mathtt{ff}).
\end{aligned}
$$

SQ の $[a\{p\}]\langle a\{q\}\rangle\langle a\{q\}\rangle$ の部分はプロセス p が a を実行するならば，プロセス q は a を 2 回実行できることを要求している．また，EQ の $[a\{q\}]\langle a\{p\}\rangle$ はプロセス q が a を実行するならば，プロセス p は a を実行できることを要求している．

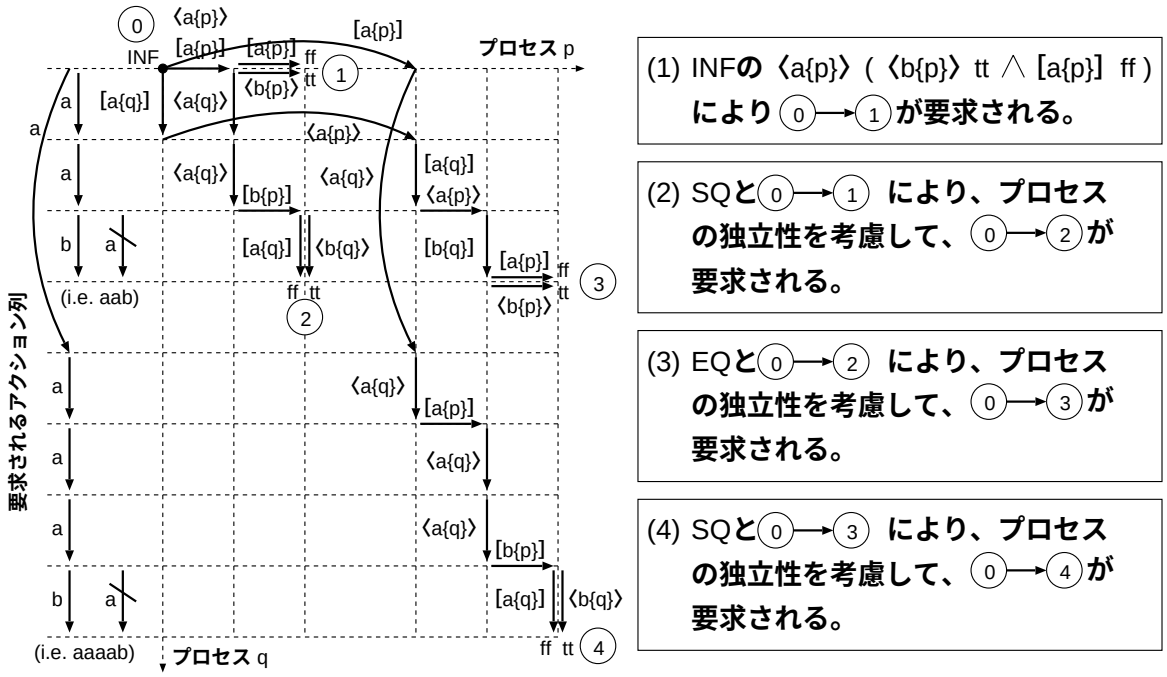

図 7.13: 仕様 INF の要求の一部

ここで重要なことは，プロセス p がアクション a を実行してからプロセス q がアクション aa を実行できるならば，p がアクション a を実行する前にも q は aa を実行できることである (プロセスの独立性)．すなわち，a の数は繰り返し 2 倍化され，アクション列 $a \cdots ab$ の長さは指数関数的に増加する．すなわち，INF は任意の $n \geq 0$ について，2^n 回アクション a を実行した後に b を実行できなければならない．図 7.13は INF がどのように a を増加させるかを示している．例えば，次の分散システムは INF を満たす．

$$SYS_{AB} \equiv AB@p|AB@q$$

$$AB \equiv \sum\{\underbrace{a.\cdots.a}_{2^n \text{ 個}}.b.\mathbf{0} : n \geq 0\}$$

このように，INF は充足可能であるが，それを満たす分散システムの状態数は無限になる．このような仕様をアルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ に入力した場合，$\mathbf{Sat}^{\mathcal{E}}$ は停止しない． ∎

以下，7.6.1小節で，無制限文法 (0 型文法，または成区構造文法とも呼ばれる) [34] を紹介する．この文法のメンバーシップ問題は決定不能であることが知られている．続いて，7.6.2小節と 7.6.3小節で，決定不能を証明するために必要な，G 仕様と G-分散

システムを定義する．そして，7.6.4小節で，無制限文法のメンバーシップ問題は $SP@$ の充足可能性判定問題に変換可能であることを示す．これは，$SP@$ の充足可能性が決定不能であることを意味している．

7.6.1 アクション列文法

まず，初期アクション γ から書換え規則にしたがって (無限に) アクション列を生成する文法を定義する．

定義 7.6.1 組 $G = \langle \mathcal{A}, \gamma, \rho \rangle$ は**アクション列文法**である．ここで，$\mathcal{A} \subseteq AN$ はアクションの有限集合，$\gamma \in AN$ は**初期アクション**と呼ばれるアクション，ρ は次のような**書換え規則**の有限集合である：

$$\rho \subseteq (\mathcal{A}^* - \{\varepsilon\}) \times \mathcal{A}^*$$

ここで，$\mathcal{A}^*$ は $\mathcal{A}$ に含まれる 0 個以上のアクションを結合して得られる有限アクション列の集合 (その要素を $\tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}, \cdots$ により表す) である．特殊な記号 ε は空列を表し，$\tilde{\alpha}\varepsilon = \varepsilon\tilde{\alpha} = \tilde{\alpha}$ である． ∎

書換え規則 $(\tilde{\alpha}, \tilde{\beta})$ は，$\tilde{\alpha}_1\tilde{\alpha}\tilde{\alpha}_2$ の形のアクション列に適用され，新しいアクション列 $\tilde{\alpha}_1\tilde{\beta}\tilde{\alpha}_2$ を生成する．アクション列文法 G において，書換え規則によって繰り返し生成される全てのアクション列の集合は G-言語と呼ばれ，$L(G)$ と書かれる．これは次のように定義される．

定義 7.6.2 $G = \langle \mathcal{A}, \gamma, \rho \rangle$ をアクション列文法とする．このとき，G-**言語** $L(G)$ は次のように与えられる：

$$\begin{aligned} L^{(0)}(G) &= \{\gamma\} \\ L^{(n+1)}(G) &= \{\tilde{\alpha}_1\tilde{\beta}\tilde{\alpha}_2 : \exists\tilde{\alpha}.\ \tilde{\alpha}_1\tilde{\alpha}\tilde{\alpha}_2 \in L^{(n)}(G), (\tilde{\alpha}, \tilde{\beta}) \in \rho\} \\ L(G) &= \bigcup_{n\geq 0} L^{(n)}(G) \end{aligned}$$

∎

例 7.6.2 $G = \langle \{a, b\}, a, \{(a, bab), (bba, \varepsilon)\} \rangle$ をアクション列文法とする．第 1, 2 ステップでは，書換え規則 (a, bab) だけが適用でき，

$$L^{(0)}(G) = \{a\},\ L^{(1)}(G) = \{bab\},\ L^{(2)}(G) = \{bbabb\},$$

となる．第 3 ステップでは，両方の書換え規則 $(a, bab), (bba, \varepsilon)$ が適用でき，次のようになる：$L^{(3)}(G) = \{bbbabbb, bb\}$. ∎

定義 7.6.2にみられるように，書換え規則に対して制限がないので，アクション列文法は無制限文法 [34] である．また，無制限文法のメンバーシップ問題は決定不能であることが知られている．すなわち，アクション列文法 G とアクション列 $\tilde{\alpha}$ が与えられたとき，$\tilde{\alpha} \in L(G)$ か $\tilde{\alpha} \notin L(G)$ を判定する停止性が保証されたアルゴリズムはない．無制限文法では，ある書換え規則 (ex. (a, bab)) はアクション列を長くし，ある書換え規則 (ex. (bba, ε)) はアクション列を短くすることができるため，例えば長さが 10 のアクション列を生成するためにさえ，何回書換え規則を適用すればよいか決められず，メンバーシップ問題は決定不能になる．

7.6.2 G-仕様

本小節では，アクション列文法 G によって生成される全てのアクション列を実行することを分散システムに要求する仕様を定義する．この仕様を G-**仕様**と呼ぶ．

定義 7.6.3 $p, q \in PN$，$G = \langle \mathcal{A}, \gamma, \rho \rangle$ はアクション列文法とする．有限状態をもつ仕様 $GR_{(G)}$ は G から次のように定義される．

$$
\begin{aligned}
GR_{(G)} &\equiv RW_{(G)} \wedge EQ_{(G)}^{(q,p)} \wedge \langle \gamma\{p\}\rangle\langle \mathtt{end}\{p\}\rangle \mathtt{tt} \\
RW_{(G)} &\stackrel{\text{def}}{=} \bigwedge\{[\tilde{\alpha}\{p\}]^*\langle \tilde{\beta}\{q\}\rangle^* EQ_{(G)}^{(p,q)} : (\tilde{\alpha}, \tilde{\beta}) \in \rho\} \\
&\qquad \wedge \bigwedge\{[\alpha\{p\}]\langle \alpha\{q\}\rangle RW_{(G)} : \alpha \in \mathcal{A}_{\mathtt{end}}\} \\
EQ_{(G)}^{(\psi,\varphi)} &\stackrel{\text{def}}{=} \bigwedge\{[\alpha\{\psi\}]\langle \alpha\{\varphi\}\rangle EQ_{(G)}^{(\psi,\varphi)} : \alpha \in \mathcal{A}_{\mathtt{end}}\}
\end{aligned}
$$

ここで，$\mathcal{A}_{\mathtt{end}} = \mathcal{A} \cup \{\mathtt{end}\}$，$\mathtt{end}$ は $\mathtt{end} \notin \mathcal{A}$ のような特殊なアクションであり，仕様 $\langle \tilde{\alpha}\{\psi\}\rangle^* S$ と $[\tilde{\alpha}\{\psi\}]^* S$ は次のように定義される略記法である：

$$
\langle \tilde{\alpha}\{\psi\}\rangle^* S \equiv \begin{cases} S & (\tilde{\alpha} = \varepsilon) \\ \langle \beta\{\psi\}\rangle\langle \tilde{\alpha}'\{\psi\}\rangle^* S & (\tilde{\alpha} = \beta\tilde{\alpha}') \end{cases}
$$

$$
[\tilde{\alpha}\{\psi\}]^* S \equiv \begin{cases} S & (\tilde{\alpha} = \varepsilon) \\ [\beta\{\psi\}][\tilde{\alpha}'\{\psi\}]^* S & (\tilde{\alpha} = \beta\tilde{\alpha}') \end{cases}
$$

∎

まず，$GR_{(G)}$ を理解するために，$GR_{(G)}$ の形が例 7.6.1の仕様 INF に似ていることに注目することは有効である．仕様 $RW_{(G)}$ の $[\tilde{\alpha}\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^*$ の部分は，書換え規則 $(\tilde{\alpha},\tilde{\beta})\in\rho$ にしたがってプロセス p がアクション列 $\tilde{\alpha}$ を実行するならば，プロセス q はアクション列 $\tilde{\beta}$ を実行できることを要求し，$[\alpha\{p\}]\langle\alpha\{q\}\rangle RW_{(G)}$ と $EQ^{(p,q)}_{(G)}$ の部分は，その書換えの前後では q は p と同じアクション列を実行できることを要求している．また，$EQ^{(q,p)}_{(G)}$ により，p は q と同じアクション列を実行できることも要求されている．

以下，しばしば次の連続したラベル付遷移の記述を利用する．

$$P\xrightarrow{\tilde{\alpha}\cdot}P' \iff \exists P_i.\ P\xrightarrow{\alpha_1\cdot}P_1\xrightarrow{\alpha_2\cdot}\cdots\xrightarrow{\alpha_n\cdot}P'$$
$$D\xrightarrow{\tilde{\alpha}\{\psi\}}D \iff \exists D_i.\ D\xrightarrow{\alpha_1\{\psi\}}D_1\xrightarrow{\alpha_2\{\psi\}\cdot}\cdots\xrightarrow{\alpha_n\{\psi\}}D'$$

ここで，$\tilde{\alpha}=\alpha_1\cdots\alpha_n$ である．次の命題は $GR_{(G)}$ が文法 G によって生成される全てのアクション列を実行することを要求することを示している．

命題 7.6.1 G をアクション列文法とする．このとき，もし $(P@p|Q@q)\models GR_{(G)}$ かつ $\tilde{\alpha}\in L(G)$ ならば，ある P' について，$P\xrightarrow{\tilde{\alpha}\cdot}\xrightarrow{\mathtt{end}\cdot}P'$ である．

証明 $G=\langle\mathcal{A},\gamma,\rho\rangle$ かつ $(P@p|Q@q)\models GR_{(G)}$ かつ $\tilde{\alpha}\in L^{(n')}(G)$ とし，n' に関する帰納法を用いて証明する．$n'=0$ の場合は，$\tilde{\alpha}=\gamma$ である．また，$GR_{(G)}\xrightarrow{\gamma\{p\}}_{\emptyset}\langle\mathtt{end}\{p\}\rangle\mathtt{tt}$ かつ $(P@p|Q@q)\models GR_{(G)}$ であるので，ある P' について，$P\xrightarrow{\gamma\cdot}\xrightarrow{\mathtt{end}\cdot}P'$ を得る．

$n'=n+1$ $(n\geq 0)$ とする．このとき，$\tilde{\alpha}\in L^{(n+1)}(G)$ であるので，ある $\tilde{\alpha}_1\tilde{\beta}_2\tilde{\alpha}_3$ と $\tilde{\alpha}_2$ について，$\tilde{\alpha}=\tilde{\alpha}_1\tilde{\beta}_2\tilde{\alpha}_3$ かつ $\tilde{\alpha}_1\tilde{\alpha}_2\tilde{\alpha}_3\in L^{(n)}(G)$ かつ $(\tilde{\alpha}_2,\tilde{\beta}_2)\in\rho$ である．すなわち，帰納法の仮定より，ある P_1,P_2,P' について，$P\xrightarrow{\tilde{\alpha}_1\cdot}P_1\xrightarrow{\tilde{\alpha}_2\cdot}P_2\xrightarrow{\tilde{\alpha}_3\cdot}\xrightarrow{\mathtt{end}\cdot}P'$ を得る．図 7.14に本証明で使われるアクション列の関係を示す．

まず，$P\xrightarrow{\tilde{\alpha}_1\cdot}P_1$ の遷移の長さに関する帰納法を用いて，ある Q_1 について，$Q\xrightarrow{\tilde{\alpha}_1\cdot}Q_1$ かつ $(P_1@p|Q_1@q)\models RW_{(G)}$ を示す．長さ 0 の場合 $P\xrightarrow{\varepsilon\cdot}P_1$ は，$P\equiv P_1$ より明らかである．$\tilde{\alpha}_1=\tilde{\alpha}'_1\alpha'$ とする．すなわち，ある P'_1 について，$P\xrightarrow{\tilde{\alpha}'_1\cdot}P'_1\xrightarrow{\alpha'\cdot}P_1$ である．長さに関する帰納法の仮定から，ある Q'_1 について $Q\xrightarrow{\tilde{\alpha}'_1\cdot}Q'_1$ かつ $(P'_1@p|Q'_1@q)\models RW_{(G)}$ を得る．また，**Name** と $\mathbf{Com}_1$ より，$(P'_1@p|Q'_1@q)\xrightarrow{\alpha'\{p\}}(P_1@p|Q'_1@q)$ を導く．ここで，$\alpha'\in\mathcal{A}$ より，$RW_{(G)}\xrightarrow{\alpha'\{p\}}_{[\![}\langle\alpha'\{q\}\rangle RW_{(G)}$ であるので，$(P_1@p|Q'_1@q)\models\langle\alpha'\{q\}\rangle RW_{(G)}$ を得る．すなわち，ある D_1 について，$(P_1@p|Q'_1@q)\xrightarrow{\alpha'\{q\}}D_1$ かつ $D_1\models RW_{(G)}$ を得る．この遷移は **Name** と $\mathbf{Com}_2$ によって導かれるので，ある Q_1 について，$Q'_1\xrightarrow{\alpha'\cdot}Q_1$ かつ $D_1\equiv P_1@p|Q_1@q$ を得る．すなわち，$Q\xrightarrow{\tilde{\alpha}'_1\cdot}Q'_1\xrightarrow{\alpha'\cdot}Q_1$ かつ $(P_1@p|Q_1@q)\equiv D_1\models RW_{(G)}$ が得られた．

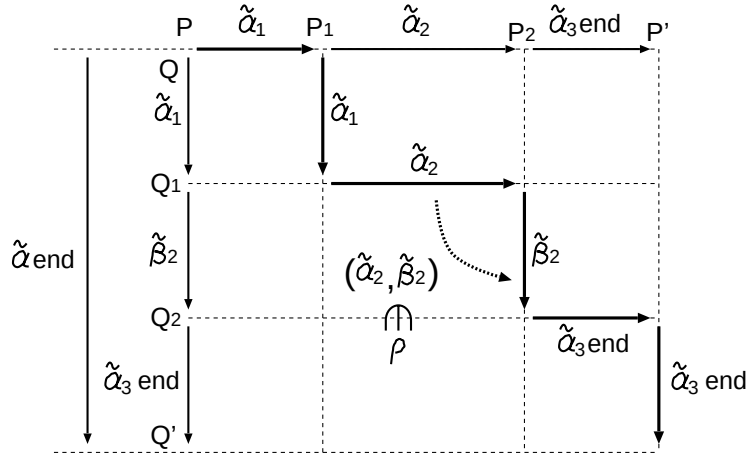

図 7.14: 命題 7.6.1 の証明で使われるアクション列

第 2 の遷移 $P_1 \xrightarrow{\tilde{\alpha}_2 \cdot} P_2$ に対しては，$(P_1@p|Q_1@q) \models RW_{(G)}$ かつ $(\tilde{\alpha}_2, \tilde{\beta}_2) \in \rho$ であるので，$[\tilde{\alpha}_2\{p\}]^*\langle\tilde{\beta}_2\{q\}\rangle^* EQ^{(p,q)}_{(G)}$ が要求される．すなわち，必然要求 $[\tilde{\alpha}_2\{p\}]^*$ により，必ず $(P_2@p|Q_1@q) \models \langle\tilde{\beta}_2\{q\}\rangle^* EQ^{(p,q)}_{(G)}$ である．さらに，可能要求 $\langle\tilde{\beta}_2\{q\}\rangle^* EQ^{(p,q)}_{(G)}$ は，プロセス q がアクション列 $\tilde{\beta}_2$ を実行でき，その後に $EQ^{(p,q)}_{(G)}$ を満たすことができることを要求しているので，ある Q_2 について，$Q_1 \xrightarrow{\tilde{\beta}_2 \cdot} Q_2$ かつ $(P_2@p|Q_2@q) \models EQ^{(p,q)}_{(G)}$ を得る．

第 3 の遷移 $P_2 \xrightarrow{\tilde{\alpha}_3 \cdot} \xrightarrow{\mathtt{end} \cdot} P'$ については，第 1 の遷移 $P \xrightarrow{\tilde{\alpha}_1 \cdot} P_1$ のときと同様に，遷移の長さに関する帰納法を用いて，ある Q' について，$Q_2 \xrightarrow{\tilde{\alpha}_3 \cdot} \xrightarrow{\mathtt{end} \cdot} Q'$ かつ $(P'@p|Q'@q) \models EQ^{(p,q)}_{(G)}$ を得る．

最後に，$(P@p|Q@q) \models EQ^{(q,p)}_{(G)}$ であるので，$Q \xrightarrow{\tilde{\alpha} \cdot} \xrightarrow{\mathtt{end} \cdot} Q'$ の長さに関する帰納法を適用して，ある P'' について，$P \xrightarrow{\tilde{\alpha} \cdot} \xrightarrow{\mathtt{end} \cdot} P''$ を得る． ∎

7.6.3 G-分散システム

本小節では，実行するアクションの列がアクション列文法 G によって生成されるアクション列であるような分散システムを定義する．これを G-分散システムと呼ぶ．

定義 7.6.4 各アクション列文法 G について，G-**分散システム** $\mathbf{Ds}_{(G)}$ を次のように定義する．

$$\mathbf{Ds}_{(G)} \equiv \mathbf{Pr}_{(G)}@p \mid \mathbf{Pr}_{(G)}@q, \qquad \mathbf{Pr}_{(G,0)} \equiv \gamma.\mathtt{end}.\mathbf{0},$$
$$\mathbf{Pr}_{(G)} \equiv \textstyle\sum_{n\geq 0} \mathbf{Pr}_{(G,n)}, \qquad \mathbf{Pr}_{(G,n+1)} \equiv \mathbf{Rw}_{(G)}(\mathbf{Pr}_{(G,n)}),$$

$$\mathbf{Rw}_{(G)}(P) \equiv \textstyle\sum\{\alpha.\mathbf{Rw}_{(G)}(P') : P \xrightarrow{\alpha\cdot} P'\} + \sum\{\tilde{\beta}.P' : P \xrightarrow{\tilde{\alpha}\cdot} P', (\tilde{\alpha},\tilde{\beta}) \in \rho\},$$

ここで，$\tilde{\alpha}.P$ は次のように与えられる略記法である．

$$\tilde{\alpha}.P \equiv \begin{cases} P & (\tilde{\alpha} = \varepsilon) \\ \beta.\tilde{\alpha}'.P & (\tilde{\alpha} = \beta\tilde{\alpha}') \end{cases}$$

∎

以下，$\mathbf{Ds}_{(G)}$ の性質を示す 2 つの命題を与える．まず，次の命題 7.6.2に示すように，$\mathbf{Ds}_{(G)}$ が実行できるアクション列の集合は $L(G)$ である．

命題 7.6.2 G をアクション列文法とする．このとき，次の関係が成り立つ．

$$\tilde{\alpha} \in L(G) \iff \exists D'.\, \mathbf{Ds}_{(G)} \xrightarrow{\tilde{\alpha}\{p\}} \xrightarrow{\mathtt{end}\{p\}} D'$$

証明 $(\Rightarrow)$ の場合：$\tilde{\alpha} \in L^{(n')}(G)$ ならば $\mathbf{Pr}_{(G,n')} \xrightarrow{\tilde{\alpha}\cdot} \xrightarrow{\mathtt{end}\cdot} \mathbf{0}$ を，n' に関する帰納法を用いて証明する．$n' = 0$ ならば，$L^{(0)}(G) = \{\gamma\}$ かつ $\mathbf{Pr}_{(G,0)} \equiv \gamma.\mathtt{end}.\mathbf{0}$ より明らかである．

$n' = n+1$ $(n \geq 0)$ とする．$L^{(n+1)}(G)$ の定義より，ある $\tilde{\alpha}_1\tilde{\beta}_2\tilde{\alpha}_3$ と $\tilde{\alpha}_2$ について，$\tilde{\alpha} = \tilde{\alpha}_1\tilde{\beta}_2\tilde{\alpha}_3$ かつ $(\tilde{\alpha}_2,\tilde{\beta}_2) \in \rho$ かつ $\tilde{\alpha}_1\tilde{\alpha}_2\tilde{\alpha}_3 \in L^{(n)}(G)$ である．ここで，帰納法の仮定より，$\mathbf{Pr}_{(G,n)} \xrightarrow{\tilde{\alpha}_1} \xrightarrow{\tilde{\alpha}_2} \xrightarrow{\tilde{\alpha}_3} \xrightarrow{\mathtt{end}} \mathbf{0}$ を得る．すなわち，$\mathbf{Pr}_{(G,n+1)} \equiv \mathbf{Rw}_{(G)}(\mathbf{Pr}_{(G,n)})$ の定義と $(\tilde{\alpha}_2,\tilde{\beta}_2) \in \rho$ より，$\mathbf{Pr}_{(G,n+1)} \xrightarrow{\tilde{\alpha}_1} \xrightarrow{\tilde{\beta}_2} \xrightarrow{\tilde{\alpha}_3} \xrightarrow{\mathtt{end}} \mathbf{0}$ のような遷移が可能である．

$(\Leftarrow)$ の場合：$\mathbf{Pr}_{(G,n')} \xrightarrow{\tilde{\alpha}\cdot} \xrightarrow{\mathtt{end}\cdot} \mathbf{0}$ ならば $\tilde{\alpha} \in \bigcup_{i\leq n'} L^{(i)}(G)$ を，n' に関する帰納法を用いて証明する．$n' = 0$ ならば，上記の $(\Rightarrow)$ の場合と同様に明らかである．

$n' = n+1$ $(n \geq 0)$ とする．$\mathbf{Pr}_{(G,n+1)} \equiv \mathbf{Rw}_{(G)}(\mathbf{Pr}_{(G,n)}) \xrightarrow{\tilde{\alpha}} \xrightarrow{\mathtt{end}} \mathbf{0}$ であるので，$\mathbf{Pr}_{(G,n)} \xrightarrow{\tilde{\alpha}} \xrightarrow{\mathtt{end}} \mathbf{0}$ か，またはある $\tilde{\alpha}_1\tilde{\beta}_2\tilde{\alpha}_3$ と $\tilde{\alpha}_2$ について，$\tilde{\alpha} = \tilde{\alpha}_1\tilde{\beta}_2\tilde{\alpha}_3$ かつ $(\tilde{\alpha}_2,\tilde{\beta}_2) \in \rho$ かつ $\mathbf{Pr}_{(G,n)} \xrightarrow{\tilde{\alpha}_1} \xrightarrow{\tilde{\alpha}_2} \xrightarrow{\tilde{\alpha}_3} \xrightarrow{\mathtt{end}} \mathbf{0}$ を得る．$\mathbf{Pr}_{(G,n)} \xrightarrow{\tilde{\alpha}} \xrightarrow{\mathtt{end}} \mathbf{0}$ の場合は，帰納法の仮定より $\tilde{\alpha} \in \bigcup_{i\leq n} L^{(i)}(G) \subseteq \bigcup_{i\leq n+1} L^{(i)}(G)$ である．一方，$\tilde{\alpha} = \tilde{\alpha}_1\tilde{\beta}_2\tilde{\alpha}_3$ かつ $(\tilde{\alpha}_2,\tilde{\beta}_2) \in \rho$ かつ $\mathbf{Pr}_{(G,n)} \xrightarrow{\tilde{\alpha}_1} \xrightarrow{\tilde{\alpha}_2} \xrightarrow{\tilde{\alpha}_3} \xrightarrow{\mathtt{end}} \mathbf{0}$ の場合は，帰納法の仮定より，$\tilde{\alpha}_1\tilde{\alpha}_2\tilde{\alpha}_3 \in \bigcup_{i\leq n} L^{(i)}(G)$ を得る．すなわち，$(\tilde{\alpha}_2,\tilde{\beta}_2) \in \rho$ であるので，$\tilde{\alpha}_1\tilde{\beta}_2\tilde{\alpha}_3 \in \bigcup_{i\leq n+1} L^{(i)}(G)$ が得られる． ∎

また，G-分散システムと G-仕様の間に充足関係が成り立つ．

命題 7.6.3 G をアクション列文法とする．このとき，次の関係が成り立つ．

$$\mathbf{Ds}_{(G)} \models GR_{(G)}$$

証明 $G = \langle \mathcal{A}, \gamma, \rho \rangle$ とする．次の集合 $\mathcal{R}$ が充足部分集合であることを証明する．

$$\begin{aligned}
\mathcal{R} = \ & \{(\mathbf{Ds}_{(G)}, GR_{(G)})\} && (1)\\
& \cup\{(\mathtt{end}.\mathbf{0}@p|Q@q, \langle \mathtt{end}\{p\}\rangle \mathtt{tt}), (P@p|Q@q, \mathtt{tt}) : P, Q \in Pr\} && (2)\\
& \cup\{(P'@p|Q@q, [\tilde{\alpha}'\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)}) :\\
& \qquad \exists P.\, \mathbf{Rw}_{(G)}(P) \subseteq_+ Q, P \xrightarrow{\tilde{\alpha}\cdot} P', (\tilde{\alpha}\tilde{\alpha}', \tilde{\beta}) \in \rho\} && (3)\\
& \cup\{(P'@p|Q@q, \langle\tilde{\alpha}\{q\}\rangle^* RW_{(G)}) : \exists P.\, \mathbf{Rw}_{(G)}(P) \subseteq_+ Q, P \xrightarrow{\tilde{\alpha}\cdot} P'\} && (4)\\
& \cup\{(P'@p|Q@q, \langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)}) : \exists P.\, P \subseteq_+ Q, P \xrightarrow{\tilde{\beta}\cdot} P'\} && (5)\\
& \cup\{(P@p|P'@q, \langle\tilde{\alpha}\{p\}\rangle^* EQ^{(q,p)}_{(G)}) : P \xrightarrow{\tilde{\alpha}\cdot} P'\} && (6)
\end{aligned}$$

ここで，$P \subseteq_+ Q$ はプロセス Q が選択プロセスとして P をもつことを表す．すなわち，$Q \equiv \cdots + P + \cdots$ である．$(D, S) \in \mathcal{R}$ とする．このとき，上記の各場合 $(1), \cdots, (6)$ について，定義 7.4.4の条件 (i), (ii) が成り立つことを以下に示す．

(1) $(D, S) = (\mathbf{Ds}_{(G)}, GR_{(G)})$ の場合：まず，$GR_{(G)} \mapsto GR_{(G)}$ である．

(i) $GR_{(G)} \xrightarrow{\alpha\Psi}_{\Diamond} S'$ とする．$GR_{(G)}$ の定義より，$\alpha\Psi = \gamma\{p\}$ かつ $S' \equiv \langle \mathtt{end}\{p\}\rangle\mathtt{tt}$ である．ここで，$\mathbf{Pr}_{(G,0)} \xrightarrow{\gamma\cdot} \mathtt{end}.\mathbf{0}$ より，$\mathbf{Ds}_{(G)} \xrightarrow{\gamma\{p\}} D' \equiv \mathtt{end}.\mathbf{0}@p|\mathbf{Pr}_{(G)}@q$ を導ける．すなわち，$(D', S') \equiv (\mathtt{end}.\mathbf{0}@p|\mathbf{Pr}_{(G)}@q, \langle \mathtt{end}\{p\}\rangle\mathtt{tt}) \in \mathcal{R}_{(2)}$ である．

(ii) $GR_{(G)} \xrightarrow{\alpha\Psi}_{[\!]} S'$ かつ $\mathbf{Ds}_{(G)} \xrightarrow{\alpha\Psi} D'$ とする．$GR_{(G)}$ の定義より，次の 3 つの場合が考えられる．

- ある $\tilde{\alpha}', \tilde{\beta}$ について，$\Psi = \{p\}$, $S' \equiv [\tilde{\alpha}'\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)}$, $(\alpha\tilde{\alpha}', \tilde{\beta}) \in \rho$ の場合：すなわち，$\mathbf{Ds}_{(G)} \xrightarrow{\alpha\{p\}} D'$ であるので，ある P' について，$\mathbf{Pr}_{(G)} \xrightarrow{\alpha\cdot} P'$ かつ $D' \equiv P'@p|\mathbf{Pr}_{(G)}@q$ である．さらに，$\mathbf{Pr}_{(G)}$ の定義より，ある n について，$\mathbf{Pr}_{(G,n)} \xrightarrow{\alpha\cdot} P'$ かつ $\mathbf{Rw}_{(G)}(\mathbf{Pr}_{(G,n)}) \equiv \mathbf{Pr}_{(G,n+1)} \subseteq_+ \mathbf{Pr}_{(G)}$ である．よって，$(D', S') \equiv (P'@p|\mathbf{Pr}_{(G)}@q, [\tilde{\alpha}'\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)}) \in \mathcal{R}_{(3)}$ である．
- $\Psi = \{p\}$, $S' \equiv \langle\alpha\{q\}\rangle RW_{(G)}$, $\alpha \in \mathcal{A}_{\mathtt{end}}$ の場合：上記の場合と同様に，ある P' について，$\mathbf{Pr}_{(G)} \xrightarrow{\alpha\cdot} P'$ かつ $D' \equiv P'@p|\mathbf{Pr}_{(G)}@q$ であり，さらに，ある n について，$\mathbf{Pr}_{(G,n)} \xrightarrow{\alpha\cdot} P'$ かつ $\mathbf{Rw}_{(G)}(\mathbf{Pr}_{(G,n)}) \subseteq_+ \mathbf{Pr}_{(G)}$ を得る．すわなち，$(D', S') \equiv (P'@p|\mathbf{Pr}_{(G)}@q, \langle\alpha\{q\}\rangle RW_{(G)}) \in \mathcal{R}_{(4)}$ である．

- $\Psi = \{q\}$, $S' \equiv \langle\alpha\{p\}\rangle EQ^{(q,p)}_{(G)}$, $\alpha \in \mathcal{A}_{\mathtt{end}}$ の場合：上記の場合と同様に，ある P' について，$\mathbf{Pr}_{(G)} \xrightarrow{\alpha\cdot} P'$ かつ $D' \equiv \mathbf{Pr}_{(G)}@p|P'@q$ である．すなわち，$(D', S') \equiv (\mathbf{Pr}_{(G)}@p|P'@q, \langle\alpha\{p\}\rangle EQ^{(q,p)}_{(G)}) \in \mathcal{R}_{(6)}$ である．

(2) ある P, Q について，$(D, S) = (\mathtt{end}.\mathbf{0}@p|Q@q, \langle\mathtt{end}\{p\}\rangle\mathtt{tt})$ または $(P@p|Q@q, \mathtt{tt})$ の場合：$S \equiv \mathtt{tt}$ の場合は明らかである．$(D, S) = (\mathtt{end}.\mathbf{0}@p|Q@q, \langle\mathtt{end}\{p\}\rangle\mathtt{tt})$ とする．$\langle\mathtt{end}\{p\}\rangle\mathtt{tt} \mapsto \langle\mathtt{end}\{p\}\rangle\mathtt{tt}$ であり，$\langle\mathtt{end}\{p\}\rangle\mathtt{tt}$ の要求遷移は可能遷移 $\langle\mathtt{end}\{p\}\rangle\mathtt{tt} \xrightarrow{\mathtt{end}\{p\}}_{\Diamond} \mathtt{tt}$ のみである．ここで，$\mathtt{end}.\mathbf{0} \xrightarrow{\mathtt{end}\cdot} \mathbf{0}$ より，$\mathtt{end}.\mathbf{0}@p|Q@q \xrightarrow{\mathtt{end}\{p\}} D' \equiv \mathbf{0}@p|Q@q$ かつ $(D', \mathtt{tt}) \equiv (\mathbf{0}@p|Q@q, \mathtt{tt}) \in \mathcal{R}_{(2)}$ を得る．

(3) ある $P, P', Q, \alpha, \tilde{\alpha}', \tilde{\beta}$ について，$(D, S) = (P'@p|Q@q, [\tilde{\alpha}'\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)})$, $\mathbf{Rw}_{(G)}(P) \subseteq_+ Q$, $P \xrightarrow{\tilde{\alpha}\cdot} P'$, $(\tilde{\alpha}\tilde{\alpha}', \tilde{\beta}) \in \rho$ の場合：$\tilde{\alpha}'$ が空列の場合とそうでない場合に分けて示す．

- $\tilde{\alpha}' = \varepsilon$ の場合：$P \xrightarrow{\tilde{\alpha}\cdot} P'$, $(\tilde{\alpha}\tilde{\alpha}', \tilde{\beta}) \in \rho$, $\tilde{\alpha}' = \varepsilon$ であるので，$\mathbf{Rw}_{(G)}(P)$ の定義より，$\tilde{\beta}.P' \subseteq_+ \mathbf{Rw}_{(G)}(P) \subseteq_+ Q$ を得る．さらに，$\tilde{\beta}.P' \xrightarrow{\tilde{\beta}\cdot} P'$ である．すなわち，$(D, S) = (P'@p|Q@q, \langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)})$, $\tilde{\beta}.P' \subseteq_+ Q$, $\tilde{\beta}.P' \xrightarrow{\tilde{\beta}\cdot} P'$ であり，これは (5) の場合に同じである．この場合の証明は後の (5) で示す．
- ある $\alpha_1\tilde{\alpha}''$ について，$\tilde{\alpha}' = \alpha_1\tilde{\alpha}''$ の場合：$S \mapsto [\alpha_1\tilde{\alpha}''\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)}$ である．明らかに $[\alpha_1\tilde{\alpha}''\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)}$ は可能要求をもたない．
  (ii) $[\alpha_1\tilde{\alpha}''\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)} \xrightarrow{\alpha_1'\Psi}_{[\![}\ S'$ かつ $D \xrightarrow{\alpha_1'\Psi} D'$ とする．**Nec** より，$\alpha_1 = \alpha_1'$, $\Psi = \{p\}$, $S' \equiv [\tilde{\alpha}''\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)}$ である．すなわち，$D \xrightarrow{\alpha_1\{p\}} D'$ であるので，ある P'' について，$P' \xrightarrow{\alpha_1\cdot} P''$ かつ $D' \equiv P''@p|Q@q$ を得る．さらに，$P \xrightarrow{\tilde{\alpha}\alpha_1\cdot} P''$, $\mathbf{Rw}_{(G)}(P) \subseteq_+ Q$, $(\tilde{\alpha}\alpha_1\tilde{\alpha}'', \tilde{\beta}') \in \rho$ であるので，$(D', S') \equiv (P''@p|Q@q, [\tilde{\alpha}''\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)}) \in \mathcal{R}_{(3)}$ である．

(4) ある $P, P', Q, \tilde{\alpha}$ について，$(D, S) = (P'@p|Q@q, \langle\tilde{\alpha}\{q\}\rangle^* RW_{(G)})$, $P \xrightarrow{\tilde{\alpha}\cdot} P'$, $\mathbf{Rw}_{(G)}(P) \subseteq_+ Q$ の場合：$\tilde{\alpha}$ が空列の場合とそうでない場合に分けて示す．

- $\tilde{\alpha} = \varepsilon$ の場合：$RW_{(G)}$ の定義より，$S \equiv RW_{(G)} \mapsto S_0$ のような S_0 が唯一存在し，S_0 は可能要求をもたない．
  (ii) $S_0 \xrightarrow{\alpha'\Psi}_{[\![}\ S'$ かつ $D \xrightarrow{\alpha'\Psi} D'$ とする．$RW_{(G)}$ の定義より，この要求遷移 $S_0 \xrightarrow{\alpha'\Psi}_{[\![}\ S'$ について，次の 2 つの場合が考えられる．

– ある $\tilde{\alpha}'', \tilde{\beta}$ について，$\Psi = \{p\}$, $S' \equiv [\tilde{\alpha}''\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)}$, $(\alpha'\tilde{\alpha}'', \tilde{\beta}) \in \rho$ の場合：すなわち，$D \equiv P'@p|Q@q \xrightarrow{\alpha'\{p\}} D'$ であるので，ある P'' について，$P' \xrightarrow{\alpha'\cdot} P''$ かつ $D' \equiv P''@p|Q@q$ である．よって，$P \equiv P' \xrightarrow{\alpha'\cdot} P''$, $\mathbf{Rw}_{(G)}(P) \subseteq_+ Q$, $(\alpha'\tilde{\alpha}'', \tilde{\beta}) \in \rho$ であるので，$(D', S') \equiv (P''@p|Q@q, [\tilde{\alpha}''\{p\}]^*\langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)}) \in \mathcal{R}_{(3)}$ である．

– $\Psi = \{p\}$, $S' \equiv \langle\alpha'\{q\}\rangle RW_{(G)}$, $\alpha' \in \mathcal{A}_{\mathtt{end}}$ の場合：上記の場合と同様に，ある P'' について，$P' \xrightarrow{\alpha'\cdot} P''$ かつ $D' \equiv P''@p|Q@q$ である．よって，$P \equiv P' \xrightarrow{\alpha'\cdot} P''$, $\mathbf{Rw}_{(G)}(P) \subseteq_+ Q$ であるので，$(D', S') \equiv (P''@p|Q@q, \langle\alpha'\{q\}\rangle RW_{(G)}) \in \mathcal{R}_{(4)}$ である．

- ある $\alpha_1\tilde{\alpha}'$ について，$\tilde{\alpha} = \alpha_1\tilde{\alpha}'$ の場合：$S \mapsto \langle\alpha_1\tilde{\alpha}'\{q\}\rangle^* RW_{(G)}$ である．

  (i) $\langle\alpha_1\tilde{\alpha}'\{q\}\rangle^* RW_{(G)} \xrightarrow{\alpha''\Psi}_{\emptyset} S'$ とする．**Pos** より，$\alpha'' = \alpha_1$, $\Psi = \{q\}$, $S' \equiv \langle\tilde{\alpha}'\{q\}\rangle^* RW_{(G)}$ である．ここで，$P \xrightarrow{\tilde{\alpha}\cdot} P'$ より，ある P_1 について，$P \xrightarrow{\alpha_1\cdot} P_1 \xrightarrow{\tilde{\alpha}'\cdot} P'$ である．さらに，$\mathbf{Rw}_{(G)}(P)$ の定義より，$\mathbf{Rw}_{(G)}(P) \xrightarrow{\alpha_1\cdot} \mathbf{Rw}_{(G)}(P_1)$ を導ける．よって，$\mathbf{Rw}_{(G)}(P) \subseteq_+ Q$ であるので，$D \equiv P'@p|Q@q \xrightarrow{\alpha_1\{q\}} D' \equiv P'@p|\mathbf{Rw}_{(G)}(P_1)@q$ を得る．すなわち，$P_1 \xrightarrow{\tilde{\alpha}'\cdot} P'$ であるので，$(D', S') \equiv (P'@p|\mathbf{Rw}_{(G)}(P_1)@q, \langle\tilde{\alpha}'\{q\}\rangle^* RW_{(G)}) \in \mathcal{R}_{(4)}$ である．

(5) ある $P, P', Q, \tilde{\beta}$ について，$(D, S) = (P'@p|Q@q, \langle\tilde{\beta}\{q\}\rangle^* EQ^{(p,q)}_{(G)})$, $P \subseteq_+ Q$, $P \xrightarrow{\tilde{\beta}\cdot} P'$ の場合：$\tilde{\beta}$ が空列の場合とそうでない場合に分けて示す．

- $\tilde{\beta} = \varepsilon$ の場合：$S \equiv EQ^{(p,q)}_{(G)} \mapsto \bigwedge\{[\alpha''\{p\}]\langle\alpha''\{q\}\rangle EQ^{(p,q)}_{(G)} : \alpha'' \in \mathcal{A}_{\mathtt{end}}\}$ である．$S_0 \equiv \bigwedge\{[\alpha''\{p\}]\langle\alpha''\{q\}\rangle EQ^{(p,q)}_{(G)} : \alpha'' \in \mathcal{A}_{\mathtt{end}}\}$ とおく．また，$P \xrightarrow{\varepsilon\cdot} P'$ であるので，$P \equiv P'$ である．

  (ii) $S_0 \xrightarrow{\alpha''\Psi}_{\emptyset} S'$ かつ $D \xrightarrow{\alpha''\Psi} D'$ とする．S_0 の定義より，$\Psi = \{p\}$ かつ $S' \equiv \langle\alpha''\{q\}\rangle EQ^{(p,q)}_{(G)}$ である．すなわち，$D \xrightarrow{\alpha''\{p\}} D'$ であるので，ある P'' について，$P' \xrightarrow{\alpha''\cdot} P''$ かつ $D' \equiv P''@p|Q@q$ を導ける．さらに，$P \equiv P' \xrightarrow{\alpha''\cdot} P''$ かつ $P \subseteq_+ Q$ であるので，$(D', S') \equiv (P''@p|Q@q, \langle\alpha''\{q\}\rangle EQ^{(p,q)}_{(G)}) \in \mathcal{R}_{(5)}$ である．

- ある $\beta_1\tilde{\beta}'$ について，$\tilde{\beta} = \beta_1\tilde{\beta}'$ の場合：$S \mapsto \langle\beta_1\tilde{\beta}'\{q\}\rangle^* EQ^{(p,q)}_{(G)}$ である．

  (i) $\langle\beta_1\tilde{\beta}'\{q\}\rangle^* EQ^{(p,q)}_{(G)} \xrightarrow{\beta''\Psi}_{\emptyset} S'$ とする．**Pos** より，$\beta'' = \beta_1$, $\Psi = \{q\}$, $S' \equiv \langle\tilde{\beta}'\{q\}\rangle^* EQ^{(p,q)}_{(G)}$ である．ここで，$P \xrightarrow{\tilde{\beta}\cdot} P'$ より，ある P_1 について，$P \xrightarrow{\beta_1\cdot}$

$P_1 \xrightarrow{\tilde{\beta}'\cdot} P'$ である．さらに，$P \subseteq_+ Q$ であるので，$Q \xrightarrow{\beta_1\cdot} P_1$ を得る．この遷移から，$D \equiv P'@p|Q@q \xrightarrow{\beta_1\{q\}} D' \equiv P'@p|P_1@q$ を導ける．すなわち，$P_1 \xrightarrow{\tilde{\beta}'\cdot} P'$ であるので，$(D', S') \equiv (P'@p|P_1@q, \langle\tilde{\beta}'\{q\}\rangle^* EQ^{(p,q)}_{(G)}) \in \mathcal{R}_{(5)}$ である．

(6) 上記の (5) の場合に対称に証明できる．

∎

7.6.4 決定不能性

まず，G-仕様と G-分散システムを用いて，アクション列文法のメンバーシップ問題が $SP@$ の有限状態仕様の充足可能性の判定によって解けることを示す．

定理 7.6.4 G をアクション列文法とする．このとき，次の関係が成り立つ．

$$(GR_{(G)} \wedge [\tilde{\alpha}\{p\}]^*[\mathtt{end}\{p\}]\mathtt{ff}) \text{が充足可能} \iff \tilde{\alpha} \notin L(G)$$

証明 ($\Rightarrow$) の場合：仮定より，$D \models (GR_{(G)} \wedge [\tilde{\alpha}\{p\}]^*[\mathtt{end}\{p\}]\mathtt{ff})$ となる D が存在する．この仕様は，必ずプロセス p と q に対して可能要求をもち，それ以外のプロセスに対しては何も要求しないので，一般性を失うこと無く D は $P@p|Q@q$ の形をもつと仮定できる．今，$\tilde{\alpha} \in L(G)$ を仮定する．命題 7.6.1より，ある P' について，$P \xrightarrow{\tilde{\alpha}\cdot} \xrightarrow{\mathtt{end}\cdot} P'$ である．すなわち，**Name** と **Com** により，ある D' ついて，$D \xrightarrow{\tilde{\alpha}\{p\}} \xrightarrow{\mathtt{end}\{p\}\cdot} D'$ を導ける．しかし，これは $D \models [\tilde{\alpha}\{p\}]^*[\mathtt{end}\{p\}]\mathtt{ff}$ であることに矛盾する．よって，$\tilde{\alpha} \notin L(G)$ である．

($\Leftarrow$) の場合：対偶を示すため，$(GR_{(G)} \wedge [\tilde{\alpha}\{p\}]^*[\mathtt{end}\{p\}]\mathtt{ff})$ は充足不能であると仮定する．命題 7.6.3より，$\mathbf{Ds}_{(G)} \models GR_{(G)}$ であるので，もし $\mathbf{Ds}_{(G)} \models [\tilde{\alpha}\{p\}]^*[\mathtt{end}\{p\}]\mathtt{ff}$ ならば，$\mathbf{Ds}_{(G)} \models (GR_{(G)} \wedge [\tilde{\alpha}\{p\}]^*[\mathtt{end}\{p\}]\mathtt{ff})$ となり，この仕様が充足不能であるという仮定に矛盾する．すなわち，$\mathbf{Ds}_{(G)} \not\models [\tilde{\alpha}\{p\}]^*[\mathtt{end}\{p\}]\mathtt{ff}$ である．これは，ある D' について，$\mathbf{Ds}_{(G)} \xrightarrow{\tilde{\alpha}\{p\}} \xrightarrow{\mathtt{end}\{p\}\cdot} D'$ を意味している．すなわち，命題 7.6.2より，$\tilde{\alpha} \in L(G)$ を得る．

∎

ここで，無制限文法のメンバーシップ問題は決定不能であることと，アクション列文法は無制限文法であることから，次の結論を得る．

系 7.6.5 真の並行プロセス代数 *DS*@ のためのプロセス論理 *SP*@ における充足可能性は決定不能である．∎

すなわち，*SP*@ の仕様の充足可能性を判定する停止性の保証されたアルゴリズムは存在しない．この結果は ($(GR_{(G)} \wedge [\tilde{\alpha}\{p\}]^*[\mathtt{end}\{p\}]\mathtt{ff}$) のように) 仕様が有限状態をもち，離接演算子をもたず，通信に対する要求がない場合でも成り立つ．

7.7 関連研究

Langerak はプロセス代数 LOTOS を用いて，仕様 (LOTOS の逐次的記述) を並行システム (LOTOS の並行的記述) に振舞いを変えずに変換する方法を提案した [31]．しかし，仕様とシステムの関係に '等価性' が使われているため，システムの振舞いは仕様に完全に記述されている必要があり，この方法では柔軟な仕様を表現できない．

また，プロセス論理 (μ 計算) や命題時相論理 (PTL) で記述された柔軟な仕様からシステムを合成する方法も提案されている [29, 36]．しかし，μ 計算や PTL では並行性を明記できないため，合成されるシステムは逐次的である．これらの方法 [29, 36] によって合成された逐次システムを上記の Langerak の方法 [31] を用いて並行システムに変換することも考えられるが、[31] の方法では任意のプロセス間で常に通信可能であることが仮定されているため、時間的空間的な制約によって通信できない状態をもつような要求に対しては適用できない。*SP*@ では真の並行性によって並行システムに対する要求を記述でき，かつ通信の許可や禁止を要求することもできる．プロセス論理的な仕様から通信の禁止を考慮して並行システムを合成する場合，本章で議論してきた充足可能性の決定不能性を無視することはできない．

アクションの位置や因果関係を考慮して，真の並行性を表現できる多くのプロセス代数が提案されている．[7, 6, 13, 14, 30, 42]．しかし，そのためのプロセス論理の充足可能性は議論されていなかった．

興味深い関連研究として [35] がある．ここでは，分散システムのための時相論理として Step-TL が提案され，その充足可能性が研究されている．例えば，$\langle a, b\rangle\mathtt{tt}$ はアクション a と b が並行に実行されることを要求しており，逐次的な要求 $\langle a\rangle\langle b\rangle\mathtt{tt} \wedge \langle b\rangle\langle a\rangle\mathtt{tt}$ とは明確に区別されている．この並行性の概念は並行ステップと呼ばれる．*SP*@ では，並行的な要求 $\langle a\{p\}\rangle\langle b\{q\}\rangle\mathtt{tt}$ と逐次的な要求 $\langle a\{p\}\rangle\langle b\{p\}\rangle\mathtt{tt}$ は，プロセス名によって明確に

区別されている．このように，並行ステップと真の並行性は似てはいるが，正確には違うものである．並行ステップでは，並行性は各様相演算子のなかで閉じている．例えば，仕様 $\langle a, b\rangle\langle c\rangle\mathtt{tt} \wedge [c]\mathtt{ff}$ では，必然要求 $[c]\mathtt{ff}$ が可能要求 $\langle c\rangle$ に分配されることはなく，この仕様は充足可能である．一方，真の並行性では，仕様 $\langle a\{p\}\rangle\langle b\{q\}\rangle\langle c\{r\}\rangle\mathtt{tt} \wedge [c\{r\}]\mathtt{ff}$ では，必然要求 $[c\{r\}]\mathtt{ff}$ が $\langle a\{p\}\rangle\langle b\{q\}\rangle$ を越えて可能要求 $\langle c\{r\}\rangle$ に分配されるため，この仕様は充足不能である．実際に，Step-TL の充足可能性は決定可能であることが示されている [35](Theorem 2.8)．文献 [35] にもいくつかの決定不能な特性が示されているが，それらはいくつかの条件 (例：決定的なシステムが存在するか) によって導かれるものであり，*SP*@ の決定不能性とは異なる．

7.8 おわりに

本章では，分散システムを記述するための真の並行プロセス代数 *DS*@ とその仕様を記述するためのプロセス論理 *SP*@ を定義し，仕様の充足可能性を判定するアルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ を与えた．また，充足可能であれば，それを満たす分散システムを合成する方法も与えた．アルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ の停止性は保証されていないが，一般的な多くの仕様に対して停止すると考えている．

7.6節ではアルゴリズム $\mathbf{Sat}^{\mathcal{E}}$ が停止しない例を示し，さらに *SP*@ の仕様の充足可能性は決定不能であることを証明した．すなわち，*SP*@ の仕様の充足可能を判定する停止性が保証されたアルゴリズムは存在しない．この結果は仕様が有限状態をもち，離接演算子をもたず，通信に対する要求がない場合でも成り立つ．*DS*@ は非常に簡単な真の並行プロセス代数であり，ここで得られた結果は他の多くの真の並行プロセス代数に対しても成り立つ．

第 8 章

総括

本論文では，並行システムの設計支援にプロセス代数を利用する方法について述べてきた．まず，並行システムの例としてプロダクションルールのプロセス代数による検証例を示した．次に可算ブロードキャスト通信や減衰通信をもつ並行システムに適したプロセス代数を提案し，振舞いの等価性を定義して，その公理系等，検証に必要な性質を明らかにした．そして，複数の柔軟な仕様から詳細な仕様を合成する方法や，分散システムを合成する方法を与えた．本研究の成果は次のようにまとめられる．

1. 形式的手法によって信頼性の高いシステムを構築することができるが，その記述のしにくさから，広く利用されるには至っていない．3章で提案した CCSPR はプロダクションルールの記述し易さを高めたプロセス代数である．このように各システムに適した枠組を与えることは，システム設計に形式的手法を普及させるための方法の一つである．また，CCSPR には展開規則が用意されており，CCSPR の記述を従来のプロセス代数 CCS の記述に展開することによって，従来の検証系を利用することができる．

2. 拡張性の高いシステムを構築するために，可算ブロードキャストを利用できるソフトウェアバスとして VIABUS が実装されている．4章では，VIABUS 上に構築されたシステムを記述し，その振舞いを検証するために適したプロセス代数として CCB を提案した．そして，従来の等価性では，可算ブロードキャストをもつシステムの振舞いの等価性を適切に表現できないことを指摘し，CCB に最も適した等価性として監視合同を定義した．さらに，監視合同がどのような等価性であるかを明らかにするために，その健全で完全な公理系を与えた．

3. MBone のように送信者がメッセージの受信範囲を指定できる通信方式がある．5章では，このような通信方式を表現できるプロセス代数として CCSG を提案した．CCSG では，各メッセージにその重要度を示す値として**グレイド**が付加されており，このグレイドが距離とともに減衰することによって，そのメッセージの受信範囲を制御することができる．そして，遠くの重要でないメッセージを無視した近似的な解析ができるように，観測レベル r を指定できるレベル $\langle r \rangle$ 等価を定義した．さらに，レベル $\langle r \rangle$ 等価がどのような等価性であるかを明らかにするために，その健全で完全な公理系を与えた．

4. 設計の初期段階で完全な仕様を記述することは容易ではない．そこで，並行動作に対する柔軟 (不完全) な仕様を記述できるように，プロセス代数に論理演算子を導入することが研究されている．6章では特に不動点に着目し，複数の異なる解が存在する振舞いの等式が与えられたとき，その最小の解 (最小不動点) と最大の解 (最大不動点) を得る方法を示した．最小不動点によって，いつかは実行される特性を記述することができる．さらに，仕様の充足可能性の判定方法と，逐次プロセスの合成方法を与えた．一方，並行合成演算子と離接演算子が混在する場合は重要な特性が成り立たなくなる問題も指摘した．

5. 並行動作するプロセスを含む分散システムの振舞いは複雑であり，その設計支援が期待されている．7章では，分散システムを記述するために真の並行プロセス代数 *DS*@，その柔軟な仕様を記述するためにプロセス論理 *SP*@ を定義して，停止性は保証されていないが，仕様の充足可能性を判定するアルゴリズム **Sat** を与えた．さらに，充足可能な仕様から分散システムを合成する方法も示した．*SP*@ では，並行合成演算子の代わりに真の並行性を考慮して並行動作に対する柔軟な仕様を記述するため，6章の最後に述べた問題が生じない．一方，プロセス論理 *SP*@ の仕様の充足可能性は決定不能であることを証明した．すなわち，その充足可能性を判定する停止性の保証されたアルゴリズムは存在しない．この結果は仕様が有限状態をもち，離接演算子をもたず，通信に対する要求がない場合でも成り立つ．*DS*@ は非常に簡単な真の並行プロセス代数であり，この成果は他の多くの真の並行プロセス代数に対しても成り立つ．

本論文の重要な成果として，プロセス論理 *SP*@ の仕様の充足可能性は決定不能であることを証明したことがあげられる．すなわち，*SP*@ で記述された全ての仕様の充足

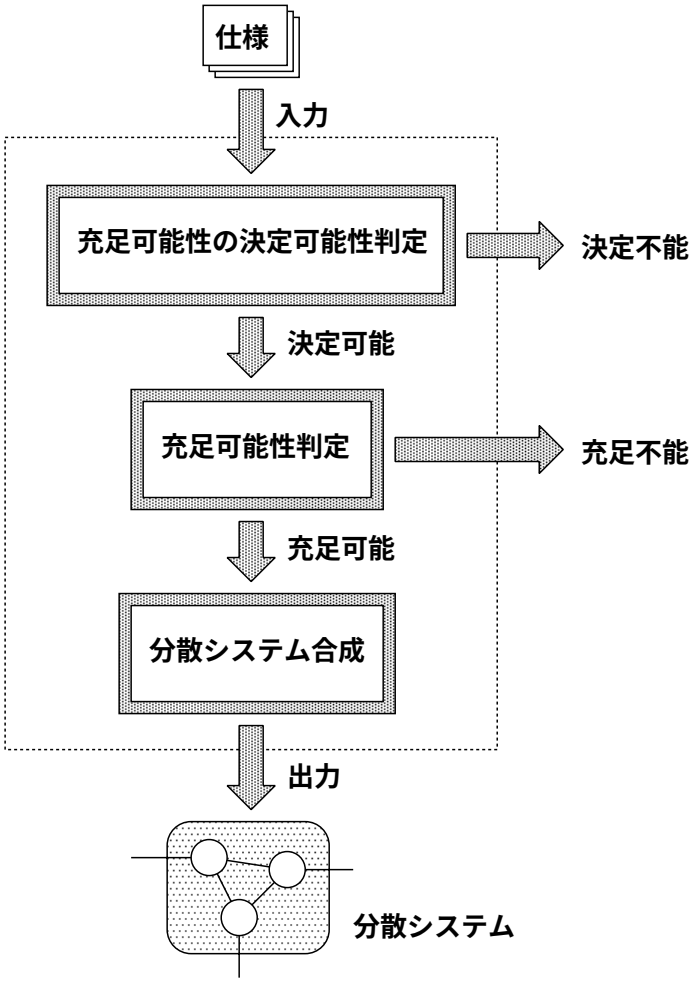

図 8.1: 分散システム合成の手続き

可能性を判定することはできない (判定できる仕様と判定できない仕様がある)．そこで，*SP@* で記述された仕様が与えられたとき，その仕様の充足可能性が決定可能か決定不能かを判定する方法が重要となる．この方法が確立されれば，図 8.1に示すように，事前に充足可能性が決定不能な仕様を排除することによって，仕様から分散システムを確実に合成できるようになる．

今回，仕様の充足可能性の決定不能性の証明に無制限文法のメンバーシップ問題の決定不能性を用いたが，逆に仕様の充足可能性判定の停止問題を項書換え系の停止問題に置き換えることができると考えている．項書換え系の停止問題については様々な性質が明らかにされており，*SP@* の仕様の充足可能性の決定性にそれらを利用できると期待している．

謝辞

本論文をまとめる過程で，種々適切なる御指導と御鞭撻を頂いた，静岡大学情報学部教授 富樫敦博士に深く感謝いたします．また，本論文をまとめる上で，多くの貴重な御意見を頂いた，静岡大学情報学部教授 中谷広正博士，静岡大学情報学部教授 鈴木淳之博士，静岡大学情報学部教授 渡辺尚博士，静岡大学情報学部助教授 佐藤文明博士に深く感謝いたします．

本研究を進める上で，種々適切なる御指導と御鞭撻を頂いた，産業技術総合研究所情報処理研究部門部門長 大蒔和仁博士に深く感謝いたします．また，3章のアクティブデータベースについて貴重な御意見と御討論頂いた情報処理研究部門 小島功氏，4章と 5章について貴重な御意見と御討論頂いた VIABUS の開発者である情報処理研究部門 佐藤豊博士に深く感謝いたします．

7章の充足可能性判定アルゴリズムを実装した Java プログラムは，情報処理研究部門 中田秀基博士に 1996 年に実装して頂いた強等価を判定する Java プログラムの一部を再利用している．中田秀基博士に深く感謝いたします．

本論文をまとめる機会を与えて頂いた，情報処理研究部門グローバル情報技術グループ グループリーダー 戸村哲博士に深く感謝いたします．

参考文献

[1] M.Abadi and A.D.Gordon, "Reasoning about Cryptographic Protocols in the Spi Calculus", CONCUR'97, LNCS 1243, Springer-Verlag, pp.59-73, 1997.

[2] 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.

[3] A.Aiken, J.Widom, and J.M.Hellerstein, "Behavior of Database Production Rules: Termination, Confluence, and Observable Determinism", Proc. of the 1992 ACM SIGMOD Conference, pp.59 - 68, 1992.

[4] J.C.M.Baeten and W.P.Weijland, *Process Algebra*, Cambridge Tracts in Theoretical Computer Science 18, Cambridge University Press, 1990.

[5] J.C.M.Baeten and J.A.Bergstra, "Real Space Process Algebra", CONCUR'91, LNCS 527, Springer-Verlag, pp.96 - 110, 1991.

[6] G.Boudol, I.Castellani, M. Hennessy, and A.Kiehn, "Observing localities", *Theoretical Computer Science*, Vol.114, pp.31-61, 1993.

[7] H.Bowman and J.P.Katoen, "A true Concurrency Semantics for ET-LOTOS", International Conference on Application of Concurrency to System Design (CSD'98), IEEE Computer Society Press. 1998.

[8] E.Brinksma, "Constraint-oriented specification in a constructive formal description technique," LNCS 430, Springer-Verlag, pp.130-152, 1989.

[9] S.Ceri, P.Fraternali, S.Paraboschi, and L.Tanca, "An Architecture for Integrity Constraint Maintenance in Active Databases", Proc. Second International Computer Science Conference '92, pp.238 - 244, 1992.

[10] 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.

[11] Chi-Chang Jou and S.A.Smolka, "Equivalences, Congruences, and Complete Axiomatizations for Probabilistic Processes", CONCUR'90, LNCS 458, Springer-Verlag, pp.367 - 383, 1990

[12] R.Cleaveland, J.Parrow, and B.Steffen, "The Concurrency Workbench : A Semantics-Based Tool for the Verification of Concurrent Systems", ACM Transactions on Programming Languages and Systems, Vol.15, No.1, pp.36-72, 1993.

[13] P.Degano and C.Priami, "Causality for Mobile Processes", ICALP'95, LNCS 944, Springer-Verlag, pp.660 - 671, 1995.

[14] P.Degano and C.Priami, "Non-interleaving semantics for mobile processes", *Theoretical Computer Science*, Vol.216, pp.237-270, 1999.

[15] R.Focardi, R.Forrieri, and V.Panini, "The Secturity Checker: A Semantics-based Tool for the Verification of Security Properties", 8th Computer Security Foundations Workshop, pp.60-69, 1995.

[16] S.Gatziu, A.Geppert, and K.R.Dittrich, "Integrating Active Concepts into an Object-Oriented Database System", Proc. Database Programming Language, The third International Workshop, pp.399 - 415, 1991.

[17] S.Gatziu and K.R.Dittrich, "Detecting Composite Events in Active Database Systems Using Petri Nets", Proc. Fourth International Workshop on Research Issues in Data Engineering Active Database Systems, pp.2 - 9, 1994.

[18] C.A.R.Hoare, *Communicating sequential processes*, Prentice-Hall, 1985.

[19] U.Holmer, “Interpreting Broadcast Communication in SCCS”, CONCUR’93, LNCS 715, Springer-Verlag, pp.188 - 201, 1993.

[20] 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.

[21] H.Ishikawa, “Active Databases”, *Journal of Information Processing Society of Japan*, Vol.35, No.2, pp.120 - 129, 1994.

[22] Y.Isobe, Y.Sato, and K.Ohmaki, “A Calculus of Countable Broadcasting Systems”, AMAST’95, LNCS 936, Springer-Verlag, pp.498 - 504, 1995.

[23] Y.Isobe, Y.Sato, and K.Ohmaki, “Approximative Analysis by Process Algebra with Graded Spatial Actions”, AMAST’96, LNCS, Springer-Verlag, July 1-5, 1995, Germany.

[24] Y.Isobe and K.Ohmaki, “A Process Logic for Distributed System Synthesis”, APSEC2000, IEEE Computer Society Press, pp.62-69, 2000.

[25] Y.Isobe, Y.Sato and K.Ohmaki, “Least Fixpoint and Greatest Fixpoint in a Process Algebra with Conjunction and Disjunction”, *IEICE Trans. on Fundamentals of Electronics, Communications and Computer Science*, Vol.E83-A, No.3, pp.401-411, 2000.

[26] Y.Isobe, Y.Sato, and, K.Ohmaki, “Eventuality in LOTOS with a disjunction operator”, ASIAN’98, Asian Computing Science Conference, LNCS 1538, Springer-Verlag, pp.263–281, 1998.

[27] P.C.Kanellakis and S.A.Smolka,“CCS expressions, finite state processes, and three problems of equivalence”, *Information and Computation*, Vol.86, pp.43–68, 1990.

[28] S.Kimura, A.Togashi and N.Shiratori, “Synthesis algorithm for recursive processes by μ-calculus”, Algorithmic Learning Theory, LNCS 872, Springer-Verlag, pp.379–394, 1994.

[29] S.Kimura, A.Togashi and N.Shiratori, "Inductive Synthesis of Recursive Processes from Logical Properties", *Information and computation*, Vol.163, No.2, pp.257-284, 2000.

[30] P.Krishnan, "Distributed CCS", CONCUR'91, LNCS527, Springer-Verlag,pp.393-407,1991.

[31] R.Langerak, "Decomposition of functionality : a correctness preserving LOTOS transformation", PSTV X, pp.229-242, 1990.

[32] K.G.Larsen, "Modal specifications, automatic verification methods for finite state systems", LNCS 407, Springer-Verlag, pp.232–246, 1989.

[33] K.G.Larsen, B.Steffen, and C.Weise, "A constraint oriented proof methodology based on modal transition systems", Tools and Algorithms for the Construction and Analysis of Systems, LNCS 1019, Springer-Verlag, pp.17–40, 1995.

[34] P.Linz, *An Introduction to Formal Languages and Automata*, Jones and Bartlett Publishers, 1990.

[35] K.Lodaya, R.Parikh, R.Ramanujam, and P.S.Thiagarahan, "A Logical Study of Distributed Transition Systems", *Information and computation*, Vol.119, pp.91-118, 1995.

[36] Z.Manna and P.Wolper, "Synthesis of communicating processes from temporal logic specifications," *ACM Trans. on Programming Languages and Systems*, Vol.6, No.1, pp.67-93, 1984.

[37] D.R.McCarthy and U.Dayal, "The Architecture Of An Active Data Base Management System", Proc. of the 1989 ACM SIGMOD Conference, pp.215 - 224, 1989.

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

[39] R.Milner, J.Parrow, and D.Walker, "A Calculus of Mobile Processes, parts I and II", *Information and Computation*, Vol.100, No.1, pp.1-40 and 41-77, 1992.

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

[41] F.Moller and C.Tofts,“A Temporal Calculus of Communicating Systems”, CONCUR’90, LNCS 458, Springer-Verlag, pp.401 - 415, 1990

[42] U.Montanari and D.Yankelevich, “A Parametric Approach to Localities”, ICALP’92, LNCS 623, Springer-Verlag, pp.617 - 628, 1992.

[43] K.V.S.Prasad, “A Calculus of Broadcasting Systems”, TAPSOFT’91, LNCS 493, Springer-Verlag, pp.338 - 358, 1991

[44] K.V.S.Prasad, “A Calculus of Value Broadcasts”, PARLE’93, LNCS 694, Springer-Verlag, pp.391 - 402, 1993

[45] D.Sangiorgi, “Locality and True-concurrency in calculi for mobile processes”, TACS’94, LNCS 789, Springer-Verlag, pp.405 - 424, 1994.

[46] M.W.A.Steen, H.Bowman, and J.Derrick, “Composition of LOTOS specification”, PSTV XV, pp.73–88, 1995.

[47] M.W.A.Steen, H.Bowman, J.Derrick, and E.A. Boiten, “Disjunction of LOTOS specification”, FORTE X/PSTV XVII, pp.177-192, 1997.

[48] M.W.A.Steen, “Consistency and composition of process specifications (Chap.5)”, Ph.D Thesis, University of Kent at Canterbury, pp.97–136, 1998.

[49] A.W.Roscoe and L.Wulf, “Composing and Decomposing Systems under Security Properties”, 8th Computer Security Foundations Workshop, pp.9-15, 1995.

[50] 佐藤豊, 大蒔和仁, “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.

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

[52] C.Stirling, “An introduction to modal and temporal logics for CCS”, Concurrency: Theory, Language, and Architecture, LNCS 491, Springer-Verlag, pp.2–20, 1989.

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

[54] M.Tsujigao, T.Hikita, T, and J.Ginbayashi, “Process Composition and Interleave Reduction in Parallel Process Specification”, Proceedings of JCSE’93, Japan, p-p.218 - 225, 1993.

[55] G.Winskel, “Synchronization trees”, *Journal of Theoretical Computer Science*, Vol.34, pp.33 - 82, 1984.

[56] ISO 8807, “Information Processing Systems–Open System Interconnection–LOTOS–A formal description technique based on the temporal ordering of observational behavior”, Feb. 1989.

[57] JP MBone 運用グループ, “JP MBone 実験参加の手引 1995 年 02 月 06 日版”, ftp://ftp.kyoto.wide.ad.jp/multicast/jp-mbone/mbone-jp-intro.txt, 1995.

著者発表論文

論文

1. Y.Isobe, I.Kojima, and K.Ohmaki, “Analysis of Database Production Rules by Process Algebra”, IEICE Trans. on Information and Systems, Vol.E78-D, No.8, pp.992-1002, 1995 年 8 月.

2. 磯部, 佐藤, 大蒔, “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”, コンピュータソフトウェア, Vol.13, No.1, pp.55-70, 日本ソフトウェア科学会, 1996 年 1 月.

3. 磯部, 佐藤, 大蒔, “グレイド付き空間プロセス代数による近似解析”, コンピュータソフトウェア, Vol.14, No.2, pp.4-21, 日本ソフトウェア科学会, 1997 年 3 月.

4. Y.Isobe, Y.Sato, and K.Ohmaki, “Least Fixpoint and Greatest Fixpoint in a Process Algebra with Conjunction and Disjunction”, IEICE Trans. on Fundamentals, Vol.E83-A, No.3, pp.401-411, 2000 年 3 月.

国際会議

1. Y.Isobe, Y.Sato, and K.Ohmaki, “A Calculus of Countable Broadcasting Systems”, Fourth International Conference on Algebraic Methodology and Software Technology (AMAST’95), LNCS 936, Springer-Verlag, pp.489-503, 1995 年 7 月.

2. Y.Isobe, Y.Sato, and K.Ohmaki, “Approximative Analysis by Process Algebra with Graded Spetial Actions”, Fifth International Conference on Algebra-

ic Methodology and Software Technology (AMAST'96), LNCS 1101, Springer-Verlag, pp.336-350, 1996 年 7 月.

3. Y.Isobe, Y.Sato, and K.Ohmaki, "Eventuality in LOTOS with a Disjunction Operator", 4th Asian Computing Science Conference (ASIAN'98), LNCS 1538, Springer-Verlag, pp.263-281, 1998 年 12 月.

4. Y.Isobe and K.Ohmaki, "A Process Logic for Distributed System Synthesis", 7th Asia-Pacific Software Engineering Conference (APSEC 2000), IEEE Computer Society Press. pp.62-69, 2000 年 12 月.

口頭発表

1. 磯部, 小島, 大蒔, "プロセス代数によるプロセス生成機能をもつ並行システムの解析", 第 15 回プログラミング-言語・基礎・実践-研究会, 情報処理学会研究報告 94-PRG-15, pp.51-58, 1994 年 1 月.

2. 磯部, 小島, 大蒔, "アクティブデータベースの動作解析のためのプロセス代数の開発", 情報処理学会 第 99 回データベースシステム研究会, 情報処理学会研究報告 94-DBS-99, pp.147-154, 1994 年 7 月.

3. 磯部, 佐藤, 大蒔, "CCB:ブロードキャスト機能をもつプロセス代数の提案", 情報処理学会 第 19 回プログラミング-言語・基礎・実践-研究会, 情報処理学会研究報告 94-PRG-19, pp.19-26, 1994 年 11 月.

4. 磯部, 佐藤, 大蒔, "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 年 12 月.

5. 磯部, 佐藤, 大蒔, "ブロードキャスト用プロセス代数 (CCB) のための観測的合同関係", 電子情報通信学会 コンピュテーション研究会, 情報処理学会研究報告 94-PRG-21, pp.65-72, 1995 年 3 月

6. 磯部, "並行ソフトウェア設計のための記述モデルとその実際-プロセス代数-", 第 8 回 回路とシステム軽井沢ワークショップ論文集, pp.263-268, 1995 年 4 月.

7. Y.Isobe, Y.Sato, and K.Ohmaki, “Approximative Analysis by Spatial Process Algebra”, 第一回 代数、論理、幾何と情報科学研究集会 (ALGI), 1995 年 10 月.

8. 磯部, 佐藤, 大蒔, “空間プロセス代数による近似解析”, ソフトウェア工学の基礎 II(FOSE’95), 近代科学社レクチャーノート 15, pp.71-80, 1995 年 12 月.

9. 磯部, 佐藤, 大蒔, “グレイド付きプロセス代数 CCSG のための公理系とプロセス論理”, 第 9 回 回路とシステム軽井沢ワークショップ論文集, pp.401-406, 1996 年 4 月.

10. 磯部, 中田, 佐藤, 大蒔, “フィルタ強等価を基にした部分的仕様の合成方法”, ソフトウェア工学の基礎 III(FOSE’96), 近代科学社レクチャーノート 17, pp.1-8, 1996 年 12 月.

11. 磯部, 中田, 佐藤, 大蒔, “強フィルタ双模倣を基にした部分仕様の段階的合成”, 第 10 回 回路とシステム軽井沢ワークショップ論文集, pp.327-332, 1997 年 4 月.

12. 磯部, 中田, 佐藤, 大蒔, “静的選択アクションを用いた多重仕様の段階的合成”, ソフトウェア工学の基礎 IV (FOSE’97), 近代科学社レクチャーノート 19, pp.12-19, 1997 年 12 月.

13. 磯部, 佐藤, 大蒔, “要求から柔軟な仕様を合成する方法”, 第 5 回 代数、論理、幾何と情報科学研究集会 (ALGI),1998 年 1 月.

14. 磯部, 佐藤, 大蒔, “準弱双模倣性をもとにした仕様の段階的合成方法”, 情報処理学会 プログラミング研究会, 情報処理学会研究報告 98-PRO-18, Vol.98, No.30, pp.33-40, 1998 年 3 月.

15. 磯部, 佐藤, 大蒔, “離接演算子をもつ LOTOS 仕様における偶発性”, 電子情報通信学会 コンカレント工学研究会, 電子情報通信学会技術研究報告 CAS98-51, Vol.98, No.379, pp.9-16, 1998 年 10 月.

16. 磯部, 佐藤, 大蒔, “部分仕様の段階的統合”, ソフトウェア工学の基礎 V (FOSE’98), 近代科学社レクチャーノート 20, pp.30-39, 1998 年 11 月.

17. 磯部, 佐藤, 大蒔, “プロセス代数における最小不動点”, 第 12 回 回路とシステム軽井沢ワークショップ論文集, pp.349-354, 1999 年 4 月.

18. 磯部, 大蒔, “プロセス論理演算子をもつプロセス代数”, 電子情報通信学会 コンカレント工学研究会, 電子情報通信学会技術研究報告 CST99-7, Vol.99, No.98, pp.49-56, 1999 年 5 月.

19. 磯部, 大蒔, “分散システムを段階的に合成するための形式的仕様記述言語”, 電子情報通信学会 1999 年 基礎境界ソサイエティ大会論文集, pp.144, 1999 年 9 月.

20. 磯部, 大蒔, “分散システム合成のための形式的仕様記述言語”, ソフトウェア工学の基礎 VI (FOSE'99), 近代科学社レクチャーノート 22, pp.36-43, 1999 年 11 月.

21. 磯部, 大蒔, “論理的な仕様から分散システムを合成する方法の検討”, 電子情報通信学会 コンカレント工学研究会, 電子情報通信学会技術研究報告 CST99-59, Vol.99, No.539, pp.31-38, 2000 年 1 月.

22. 磯部, “プロセス計算におけるセキュリティ”, 電子情報通信学会 2000 年 総合大会, 電子情報通信学会 2000 年 総合大会講演論文集 基礎境界, pp.452-453, 2000 年 3 月.

23. 磯部, 大蒔, “論理的な仕様から分散システムを合成する方法”, 第 13 回 回路とシステム軽井沢ワークショップ論文集, pp.281-286, 2000 年 4 月.

24. 磯部, 大蒔, “分散システムのためのプロセス論理の充足可能性”, ソフトウェア工学の基礎 VII (FOSE2000), 近代科学社レクチャーノート 25, pp.37-44, 2000 年 11 月.

25. 磯部, 大蒔, “分散システムのためのプロセス論理の充足可能性判定ツール”, 電子情報通信学会 コンカレント工学研究会, 電子情報通信学会技術研究報告 CST2000-34, Vol.100, No.571, pp.7-14, 2001 年 1 月.