マルチメディア通信と分散処理　46－7
（１９９０．　７．　１２）

LOTOS仕様からの効率的な試験系列の自動生成

岡崎直宣　　高橋薫　　白鳥則郎　　野口正一

東北大学電気通信研究所

適合性試験の試験系列の自動生成に関して，ISOで開発されたFDT(形式記述技法)であるLOTOSによって表現された仕様から，対応する試験系列を木状のLOTOS表現であるTree-LOTOS表現として生成する方法を提案する．この方法では，LOTOSの形式的意味を与えるLTSにおいて変数を拡張したVTSを導入し，このVTSを基に仕様と等価な試験系列を生成する．VTSを導入することによって，表現がコンパクトになるだけでなく，試験実施時における系列の選択に有効な情報を試験系列に与えることができる．また，VTSの最小形を定義し，VTSを最小形に変換することによって，より効率的な試験系列を生成することができる．

Automatic Generation of the Efficient Test Sequences from LOTOS Specifications

Naonobu OKAZAKI, Kaoru TAKAHASHI, Norio SHIRATORI, Shoichi NOGUCHI

Research Institute of Electrical Communication, TOHOKU UNIVERSITY
2-1-1, Katahira, Aoba-ku, Sendai, Japan 980

In this paper, we propose a new technique for automatically deriving the test sequences from a LOTOS specification. In the proposed technique, test sequences correspond to a finite "tree" which is represented as a Tree-LOTOS expression, which is subset of LOTOS. And a LTS with variables called VTS is used to derive test sequences, which is not only for compactness of expression but to get a effective information for selection of test sequences at the execution of tests. Furthermore the reduced normal form of VTS is defined, and by using the reduced normal form efficient test sequences is obtained. An application example of this technique is given to demonstrate its effectiveness.

1. はじめに

情報通信システムなどの分散処理システムの普及拡大に伴い，そのソフトウェアも大規模化，複雑化，多様化する傾向にある．このような状況において，適合性試験の重要性がますます高まっている．

適合性試験に関する問題の中で主要な部分を占めるものの1つが，試験システムの試験系列の生成に関する問題である．これは，現状では人手で行われており，試験の信頼性やコストの面で問題があった．

試験系列の自動生成に関して，従来幾つかの研究が行われている．それらの大部分は，システムを有限状態機械(FSM)でモデル化する方法[4][5][6][7]であり，系列の生成条件や系列長などの問題によって，現実の適用には限界がある．

一方，プロトコル仕様はFDT(Formal Description Technique)化が進んでおり，FDT仕様からの試験系列の自動生成が望まれる．FDT仕様からの試験系列の自動生成に関しての従来の研究としては，FSMモデルを基礎とするEstelle仕様及びSDL仕様からの自動生成法[9][13]，及びLOTOS仕様からの純形式的な生成法[10]がある．Estelle及びSDLはそのモデルがFSMあるいは変数付きFSM(EFSM)である．これらの仕様からの生成手法は上のFSMモデルからの生成手法を応用したものであり，その限界もまた同様である．また，LOTOS[3]仕様からの生成法は，数学的手法のみを用いて試験系列を生成する手法である．但し，ここではLOTOSのサブセットであるBasic-LOTOSを対象としており，従ってデータ構造に関しては全く扱っていない等，実用的には様々な限界がある．

本稿では，LOTOS仕様からの試験系列の自動生成の新たな手法を提案する．ここでは，データ構造を扱ったFull-LOTOSを対象とし，弱bisimulation等価[3]の概念を用いて木状の試験系列を生成することによって，形式的でかつより実用的で効率の良い試験系列の生成を行う．特に，LOTOSの意味表現であるラベル付き遷移システム(LTS[3])に変数表現を加えた変数付きLTS(VTS)を導入することにより，従来のLOTOSからの生成手法では全く扱えない変数表現を試験系列に取り入れることができ，より実用的な試験系列を得ることができる．さらに，弱bisimulation関係によるLTSの最小化の概念[11]を拡張し，VTSの最小化を行なうことにより，より効率的な試験系列を得ることができる．試験系列は，木状のLOTOS表現であるであるTreeLOTOSによって表される．

以下に，2.では適合性試験の環境とそのモデルの抽象化，試験系列についての定義を述べ，3.ではLOTOSの形式的意味を与えるLTSについて，その等価性と最小形について述べる．4.ではLOTOS仕様からの試験系列の生成法を提案し，その手法を論じる．5.では，4.で提案した手法の適用例を示す．6.ではまとめと今後の課題について述べる．

2. 適合性試験環境の抽象化

2.1 システムのモデル

通信システムにおける通信は同期通信と非同期通信に分けることができる．非同期通信では，通信に関与する複数のシステム間で，互いに独立に送受信が行われる．同期通信では，通信に関与する複数のシステム間で，同期が完全に取れた時点で通信が成立する．これは，1つのシステムがテスタ等の外部環境とやり取りする通信においても同様である．

従って，非同期通信のモデルにおいては，システムの動作は

(1) システムに対する入力イベントの発生
(2) システムの内部状態に従って出力イベントを発生
(3) 同時に，内部状態の変更

によって表される．このようなシステムの動作の仕様は，従来

(i) 状態集合 S
(ii) 入力集合 I
(iii) 出力集合 O
(iv) 遷移関数 $\delta : S \times I \rightarrow S$
(v) 出力関数 $\omega : S \times I \rightarrow O$

によって定義されるFSM(Finite State Model)，あるいはこれに変数と述語を加えたEFSM(Extended FSM)に基づいて記述されるのが一般的である．

この様に，非同期通信システムにおいては，入力，出力の区別が重要である．

一方，同期通信システムにおいては，通信はそれに関与するすべてのシステムで同時に起こるため，入出力の区別は重要でなく，“何が”起こったかだけが重要である．そこで，通信の入出力の区別をせずに，それらをイベントあるいはアクションとして抽象化してシステムの動作を表すことができる．従って，システムの動作は，

(1) システムと環境との間のイベントの発生
(2) 同時に，内部状態の変更

によって表される．この時システムの動作の仕様は，上の(i)と

(vi) イベント集合 E
(vii) 遷移関数 $\delta : S \times E \rightarrow S$

によって定義することができる．

一方，このような動作仕様の定義に対し，仕様記述においてより抽象的な記述を可能とするように，遷移に非決定性を含めることが考えられる．ここで，遷移の非決定性とは，状態とイベントが決まったとき，それによって起こる遷移が1つに定まらないことを言う．この時，システムの動作の仕様は，上の(i)，(vi)，および次の(viii)によって定義される．

(viii) 遷移関係 $T = \{ \text{—}e\rightarrow \mid \text{—}e\rightarrow \subseteq S \times S, e \in E \}$

さらに，外部環境で制御，観測できないような特別なイベントである内部イベントを取り入れることで，システムの非決定性を明確に表した記述が可能になる．この時，システムの動作の仕様は，上の(i)と，次の(ix)，(x)によって定義される．

(ix) アクション集合 $A = E \cup \{i \mid i$は内部イベント$\}$
(x) 遷移関係 $T' = \{ \text{—}a\rightarrow \mid \text{—}a\rightarrow \subseteq S \times S, a \in A \}$

以上の(i)，(ix)，(x)によって定められるシステムをラベル付き遷移システム(LTS)あるいは単に遷移システムと呼ぶ．

このように，通信における入出力を1つのアクションとして抽象化し，さらに非決定的遷移を許すようなシステムのモデルは，LTSとして考えることができる．以下ではLTSに基づいたシステムのモデルについて考察を行う．

LTSは次のように定義される．

【 定義2.1 】: LTS (labelled transition system)
LTS *Sys* は4項組 $<S, A, T, s_0>$ である．ここで，
S : 状態の集合 ;
A : アクションの集合 ;
T : 遷移関係の集合 $\{ \text{—}a\rightarrow \mid \text{—}a\rightarrow \subseteq S \times S, a \in A \}$;
s_0 ($\in S$) : *Sys* の初期状態. □

遷移関係 $\text{—}a\rightarrow$ の要素(s, s')を遷移 (transition) と呼び，$s \text{—}a\rightarrow s'$ のように表す．また，状態と遷移の数が共に有限のものを有限遷移システムと呼ぶ．以下，本稿ではLTSが有限である場合について考え，有限遷移システムをLTSと表す．

2.2 システムの試験と試験系列

従来のFSMをモデルとするシステムの試験は，ある入力系列を与えたときにシステムの出力として現れる反応を見ることで行われてきた．また，このシステムに与える入力の系列を試験系列と呼んだ．

そして，その試験系列を用いて試験を行ったとき，上の2つの関数，遷移関数 δ および出力関数 ω をいかに完全に調べることができるかがその試験系列の能力とされた．また試験系列に出現する入力の回数の総和をその試験系列の長さとした．そして，従来，主にこの試験系列の能力および長さに関して研究が行われてきた[8]．

ただし，従来のこれらの研究では，システムの記述と実現されたシステムとに次のような仮定をおいていた．

- システムの仕様記述 : 完全定義，決定的である．
- 実現されたシステム : 内部状態数が既知である．

これらの仮定は，実際には非常にきつい条件と言えるもので，実現性に乏しい．

本研究では，より実際的な立場を取り，上のような制限をおかない場合を考える．そこで，この立場での試験および試験系列の概念を次のように考える．

LTSをモデルとする同期通信システムの『試験』は，IUT(Implementation Under Test)がその時点でどんなアクションを取り得るかを調べ，テスタがその環境を提供した時にIUTがその通りのアクションを行うかどうかを調べることである．

また，従来の試験系列は，テスタがそれに従って試験を実行する，いわば試験のプログラムであった．本研究では，この試験系列に対応するものとして，『試験スイート』を考える．試験スイートはシステムの初期状態を根とする木構造で表すことができる．そして，根から1つの葉へたどるパスには1回の試験でたどるアクションの時間系列が表されており，これを試験ケースと呼ぶ．

なお，試験スイート中の非決定性は，試験の種類や状況により試験実施者が選択して実行するものとする．

また，従来の試験系列の能力の定義に代わって，ここでの試験スイートの能力について，次のように定める．

【 定義2.2 】: 試験スイートの能力
試験スイートは，それが仕様と等価であるとき，その時に限って，その能力が完全であるという． □

以下では，仕様記述言語として，同期通信でLTSモデルを意味モデルとするLOTOSを対象とし，LOTOSで記述された仕様から試験スイートを生成する手法について考察する．試験スイートは表形式と木構造を用いたTTCN表現を用いて表す．

3. LOTOSの形式的意味と等価性

本章では，LOTOSの形式的意味を与えるLTSとその等価性について文献[3][の内容を簡単に紹介し，その等価性に基づいたLTSの最小化について文献[11]に基づいて述べる．

3.1 ラベル付き遷移システム(LTS) [3]

LOTOS仕様の形式的モデルは，プロセスの動的振る舞いの記述である動作式のモデルのLTSである．
LTSにおいて，状態は動作式でラベル付けされ，遷移はアクションでラベル付けされる．LOTOSのオペレーショナル-セマンティックス(操作的意味)を与える公理と推論規則から定義される遷移関係に基づいて，与えられた動作式からLTSを生成することができる．生成方法については文献を参照されたい．

3.2 等価性[3][12]

LOTOSにおける等価性は弱bisimulation等価(weak bisimulation equivalence)と呼ばれ，これは，
『外部的に観測可能な有限回の動作(アクション)によって区別できないプロセスを同一視する』
ことに基づいている．外部的に観測可能な動作はアクション系列の遷移関係 "$=t\Rightarrow$" によって定式化される．

【 定義3.1 】: アクション系列の遷移関係
Sys=$<S, A, T, s_0>$ をLTSとする．
(1) $s, s' \in S$, $a_1, \cdots, a_n \in A$ とし，t をアクションの系列 $a_1 \cdots a_n$ としたとき，関係 $\xrightarrow{t}$ を次のように定義する．
$s=s_0 \xrightarrow{a_1} s_1, \cdots, s_{n-1} \xrightarrow{a_n} s_n=s'$ であるような状態 $s_0, \cdots, s_n \in S$ が存在するとき，これを $s \xrightarrow{t} s'$ で表す．特に，n=0に対しては $s \xrightarrow{\varepsilon} s$ である．ここで，ε は空系列を表す．
(2) $s, s' \in S$, $a_1, \cdots, a_n \in A-\{i\}$, t を観測可能なアクションの系列 $a_1 \cdots a_n \in (A-\{i\})^*$ とし，i^k をk($k \geq 0$)個の内部アクションの系列としたとき，関係 $=t\Rightarrow$ を次のように定義する．
$s \xrightarrow{i^{k_0} a_1 i^{k_1} \cdots a_n i^{k_n}} s'$ であるようなアクションの系列 $i^{k_0} a_1 i^{k_1} \cdots a_n i^{k_n}$ が存在するとき，これを $s=t\Rightarrow s'$ と表す．特に，$s \xrightarrow{i^k} s'$ を $s=\varepsilon\Rightarrow s'$ で表し，また，すべてのsに対して $s=\varepsilon\Rightarrow s$ とする． □

関係 "$=t\Rightarrow$" を用いて，弱bisimulation関係を以下のように定義する．

【 定義3.2 】: 弱bisimulation関係
Sys1=$<S_1, A, T_1, s_{01}>$, Sys2=$<S_2, A, T_2, s_{02}>$ を任意のLTSとし，$S=S_1 \cup S_2$ とする．次のような，状態の集合S上の二項関係 $R \subseteq S \times S$ を弱bisimulation関係という．
状態の組(s_1, s_2)が$(s_1, s_2) \in R$であるのは，任意の観測可能なアクションの系列 $t \in (A-\{i\})^*$ に対して，次の条件(a)と(b)が成り立つときかつそのときだけである．
(a) $s_1 = t\Rightarrow s_1'$ であるような $s_1' \in S$ が存在するならば，$s_2 = t\Rightarrow s_2'$ で $(s_1', s_2') \in R$ であるような $s_2' \in S$ が存在する．
(b) $s_2 = t\Rightarrow s_2'$ であるような $s_2' \in S$ が存在するならば，$s_1 = t\Rightarrow s_1'$ で $(s_1', s_2') \in R$ であるような $s_1' \in S$ が存在する． □

弱bisimulation関係を用いて，状態，LTSの弱bisimulation等価をそれぞれ以下のように定義する．

【 定義3.3 】: 状態の弱bisimulation等価
二つのLTSにおいて，初期状態の組(s_{01}, s_{02})を含むような弱bisimulation関係 $R \subseteq S \times S$ が存在するとき，関係Rにある状態s_1とs_2は弱bisimulation等価であるといい，$s_1 \sim_S s_2$ と書く． □

二つの状態s_1とs_2が弱bisimulation等価であるとき，s_1とs_2を初期状態とする二つのLTSを考えると，それらは必然的に弱bisimulation等価になる．

【 定義3.4 】: LTSの弱bisimulation等価
Sys1=$<S_1, A, T_1, s_{01}>$, Sys2=$<S_2, A, T_2, s_{02}>$ を任意のLTSとし，$S=S_1 \cup S_2$ とする．初期状態の組(s_{01}, s_{02})を含むような弱bisimulation関係 $R \subseteq S \times S$ が存在するとき，Sys1とSys2は弱bisimulation等価であるといい，Sys1 $\sim_S$ Sys2と書く． □

3.3 LTSの最小化

LOTOS仕様のモデルであるLTSに対して，それと弱bisimulation等価なLTSは一般に無数に存在する．それらのLTSの中で，標準となるものがあれば便利である．本節では，そのような標準的なLTSとして最小形という概念を導入する．以下に，LTSの最小形を定義し，その性質について考察する．与えられたLTSを最小形に変換する方法については，文献[11]を参照されたい．
まず，LTSの最小形を定義する．

【 定義3.5 】: LTSの最小形
LTS Sysに対して，Sysと弱bisimulation等価なLTSの中で，状態と遷移の数が最少のものをSysの最小形と呼ぶ． □

単にあるLTS Sysが最小形であるというときには，Sysと弱bisimulation等価な任意のLTS Sys'に対して，Sysが Sys' の最小形であることを意味するものとする．

【 命題3.1 】
LTS Sysが最小形であれば，Sysの任意の2つの状態は互いに弱bisimulation等価ではない． □

LTSの最小形に関して次の定理3.1が成り立つ．

【 定理3.1 】
2つのLTS Sys1=$<S_1, A, T_1, s_{01}>$ と Sys2=$<S_2, A, T_2, s_{02}>$ がともに最小形で，Sys1とSys2が弱bisimulatuion等価であれば，
(a)任意の$s_1 \in S_1$に対して，$(s_1, s_2) \in R$である$s_2 \in S_2$がただ一つ存在し，また，
(b)任意の$s_2 \in S_2$に対して，$(s_1, s_2) \in R$である$s_1 \in S_1$がただ一つ存在する．
ここでRは$(s_{01}, s_{02}) \in R$である弱bisimulatuion関係である． □

定理3.1により，次の系3.2が得られる．

【 系3.2 】
LTS Sysの最小形は同型を除いてただ一通りに定まる． □

4. LOTOS仕様からの試験スイートの生成法：最小木法

LOTOSはその形式的意味解釈がLTSによって与えられる．ところが，LTS上での試験スイート数は一般に非常に多くなるので，試験実現の立場からは，これらの多くの試験スイートの中から実現可能な数の有効な試験ケースを選び出さなければならない．そのため，試験スイートに仕様中に表現されるデータ変数を含むことが必要になる．そこで，本研究で提案する最小木法では，変数付LTS(VTS)を導入し，このVTSから試験スイートを生成する方法をとる．ここでは，VTSに含まれる変数がそのまま試験スイートに反映されるため，試験実現の立場から，より有効な試験スイートを得ることができる．
また，仕様から得られるVTSは一般に等価なものが多数存在し，必ずしもその中で最小のもの(冗長な状態を含まないもの)が得られるとは限らない．従って，このままこれを試験スイートに変換した場合には，一般には冗長な試験スイートになる．そこで最小木法では，VTSの最小形(最小VTS)を定義し，仕様から得られるVTSを最小VTSに変換することによって試験スイートの効率化を計る．

4.1 最小木法の構成

最小木法は図4.1のように構成される．
まず，LOTOS仕様からのVTSの導出体系によってVTSを得る．さらにLTSへの変換規則である具現化規則によって対応するLTSを得る．次にその具現化されたLTSの最小化を行い，その際の情報を基にVTSの最小化を行い，最小VTSを得る．さらに最小

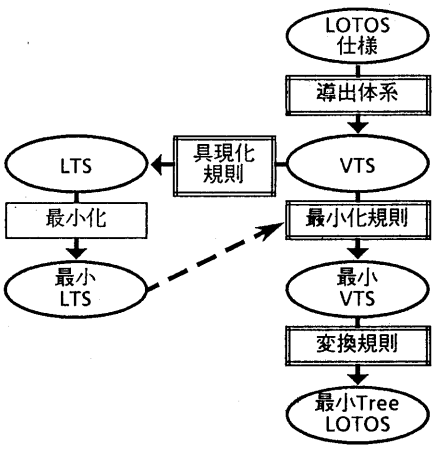


図4.1　最小VTSを用いた試験スイートの導出

TreeLOTOSへの変換規則によって，最小TreeLOTOSとして試験スイートを得る．

以下，4.2ではまずVTSを定義し，LOTOS表現からのVTSの導出体系を与え，VTSとLTSとの対応関係である具現化規則によってVTSの意味を与える．4.3ではVTSの最小化について議論し，4.4では最小VTSから最小TreeLOTOSへの変換について述べる．

4.2　変数付きLTS

4.2.1　VTS

LOTOS動作式 B に対応する変数付き遷移システム(VTS) $Vsys$ は，変数(環境によって値が決まる変数)とそれに関連した条件(選択述語およびガード)が付随したLTSである．VTS $Vsys$ は4.2.2に示す導出体系を用いることによってその動作式に関して得られる．$Vsys$ 中の変数に具体的なデータ値を割当て解釈したものは，LOTOS動作式 B に対応するLTS Sys と等価となる．

VTSは以下のように定義され，変数および述語付きの状態遷移図風に表現することができる．

【 定義4.1 】: 変数付き遷移システム(VTS)

VTS $Vsys$ は次の7項組である．

$Vsys = <N, G, AS, V, CA, Tr, n_0>$

ここで，

N : 状態の集合

G : ゲートの集合

AS : 代数仕様

V : S上の変数の集合

CA : 条件アクションの集合

各条件アクション $ca \in CA$ は

ca = action　または

$ca = [cond_1] \cdots [cond_m]$ action

の形をとる．ここで，

$cond_i\ (1 \le i \le m)$ は <S,OP>，V上の等式．

action は

① i　または

② $g\ d_1 \cdots d_n\ [SP_1] \cdots [SP_h]$　または

③ $i\ (v_1 \cdots v_k\ [SP_1] \cdots [SP_h])$.

但し，

$g \in G$.

$d_j\ (1 \le j \le n)$ は

$!t_j$ または

$?x_j : s_j$ のどちらか．

但し，

t_j は <S, OP>，V上の項，

$x_j : s_j$ はV中のソート s_j の変数 x_j.

$SP_j (1 \le j \le h)$ は <S,OP>，V上の等式．

$v_j\ (1 \le j \le k)$ は $?x_j : s_j$.

Tr : 遷移関係の集合

$Tr = \{ —ca→ \mid —ca→ \subseteq N \times N, ca \in CA \}$

n_0 : $Vsys$ の初期状態 $(n_0 \in N)$　□

例えば，"$(s_1, s_2) \in —ca→$, $ca = [cond_1] \cdots [cond_m]$ action" は，等式 "$cond_1$", …, "$cond_m$" が成立するならばアクション "action" の生起により状態 "s_1" から状態 "s_2" への遷移が可能であることを表す．これを "$s_1 —ca→ s_2$" と表記する．また，定義3.1と同様にVTSにおけるアクション系列の遷移関係 $—ct→$ および $=ct⇒$ を定義する．(ただし，ctは条件アクションの系列．)

4.2.2　VTSの導出体系

VTS $Vsys$ は，それに対応するLOTOS動作式 B から，定義4.2で定めるVTSの導出体系によって得られる．VTSの導出体系は，動作式から対応するVTSを導出するための公理と推論規則から成る．以下，動作式Bにおける変数 $x_1, \cdots, x_n$ のオカレンスへの項 $t_1, \cdots, t_n$ によるネーミングクラッシュ(名前衝突)を避けた同時代入を　$[t_1/x_1, \cdots, t_n/x_n]B$　と表記する．

また，抽象データ型の代数仕様 AS=<S, OP, E> が与えられた時，以下の記法を導入する．

- Ω_{AS}:　ASから生成される導出体系
- $[t]_{AS}$:　<S, OP> 上の基礎項(ground term) t の AS-合同類
  $[t]_{AS} = \{ t' \mid \Omega_{AS} \vdash t = t' \}$

【 定義4.2 】: VTS $Vsys$ の導出体系

ここでは，省略記法的なものを除いたLOTOSのサブセットを扱う．LOTOS仕様 CLS=<AS, BS> から生成される導出体系 Ψ_{CLS} は，以下で示す公理と推論規則によって定義される．(ただし，ここではLTSの導出体系と特に異なる部分のみを示す．)

公理(一部)

(2) action prefix　　(通信アクション)

- $g\ d_1 \cdots d_n\ ;B —g\ v_1 \cdots v_n→ [y_1'/y_1, \cdots, y_m'/y_m]B$

但し，

$\{y_1, \cdots, y_m\} = \{ x_i \mid d_i = ?x_i : s_i,\ 1 \le i \le n \}$

とおき，y_j' は新しいユニークな変数名$(1 \le j \le m)$とする．この時，

$v_i = ?y_j' : s_i$　$(d_i = ?y_j : s_i\ (1 \le i \le n, 1 \le j \le m)$ の時．)

$v_i = !\,t_i'$　$(d_i = !t_i (1 \le i \le n)$，$t_i$ が基礎項の時．

t_i' は $[t_i]_{AS}$ の代表元．)

$v_i = !\,t_i$　$(d_i = !t_i (1 \le i \le n)$，$t_i$ が基礎項でない時．)

とする．

(3) action prefix　　(選択述語付き通信アクション)

- $g\ d_1 \cdots d_n\ [SP_1] \cdots [SP_h]; B$
  $—g\ v_1 \cdots v_n\ [SSP_1] \cdots [SSP_k]→ [y_1'/y_1, \cdots, y_m'/y_m]B$

但し，(2)と同じ条件が成立．更に，次の条件が必要．

$[y_1'/y_1, \cdots, y_m'/y_m]([SP_1] \cdots [SP_h])$ を $[SP'_1] \cdots [SP'_h]$ とおく．これらの選択述語(等式) $SP'_1, \cdots, SP'_h$ の内，基礎等式(ground equation) を $SP''_1, \cdots, SP''_f$ とし，残りを $SSP_1, \cdots, SSP_k$ とする $(f+k=h)$．この時，各 $SP''_i\ (1 \le i \le f)$ について，$\Omega_{AS} \vdash SP''_i$.

推論規則(一部)

推論規則の定義のため，各条件アクションについて，そのゲート名を返す次のような関数 *name* を導入する．

$name([cond_1] \cdots [cond_n]\, g\, v_1 \cdots v_m\, [SP_1] \cdots [SP_h]) = g$,

$name([cond_1] \cdots [cond_n]\, \delta) = \delta$,

$name([cond_1] \cdots [cond_n]\, i) = i$,

$name([cond_1] \cdots [cond_n]\, i\,(v_1 \cdots v_m\, [SP_1] \cdots [SP_h])) = i$

(5) hiding

- $$\frac{B —ca→ B',\quad name(ca) \notin \{g_1, \cdots, g_n\}}{\text{hide } g_1, \cdots, g_n \text{ in } B —ca→ \text{hide } g_1, \cdots, g_n \text{ in } B'}$$

- $$\frac{B —[cond_1] \cdots [cond_k]\, g v_1 \cdots v_m\, [SP_1] \cdots [SP_h]→ B',\quad g \in \{g_1, \cdots, g_n\}}{\text{hide } g_1, \cdots, g_n \text{ in } B \;—[cond_1] \cdots [cond_k]\, i\,(t_1 \cdots t_f\, [SP_1] \cdots [SP_h])→ \text{hide } g_1, \cdots, g_n \text{ in } B'}$$

但し，$v_1, \cdots, v_m$ の内，?x:s の形(すなわち，変数宣言の形)をしているものを $t_1, \cdots, t_f$ とする．

(9) guarding

- $\dfrac{B \xrightarrow{ca} B' \quad , \quad P\text{が基礎等式であり，} \Omega_{AS} \vdash P}{[P]\,B \xrightarrow{ca} B'}$
- $\dfrac{B \xrightarrow{ca} B' \quad , \quad P\text{が基礎等式でない}}{[P]\,B \xrightarrow{[P]ca} B'}$

(12)general parallel

- $\dfrac{B1 \xrightarrow{ca} B1' \quad , \quad name(ca) \notin G \cup \{\delta\}}{B1 \,|[G]|\, B2 \xrightarrow{ca} B1' \,|[G]|\, B2}$

  $\dfrac{B2 \xrightarrow{ca} B2' \quad , \quad name(ca) \notin G \cup \{\delta\}}{B1 \,|[G]|\, B2 \xrightarrow{ca} B1 \,|[G]|\, B2'}$
- $\dfrac{B1 \xrightarrow{[cond_{11}]\cdots[cond_{1r}]\delta} B1' \,,\quad B2 \xrightarrow{[cond_{21}]\cdots[cond_{2s}]\delta} B2'}{B1 \,|[G]|\, B2 \xrightarrow{\substack{[cond_{11}]\cdots[cond_{1r}]\\ [cond_{21}]\cdots[cond_{2s}]\delta}} B1' \,|[G]|\, B2'}$

  $\dfrac{\begin{array}{l}B1 \xrightarrow{[cond_{11}]\cdots[cond_{1r}]\ g\ v_{11}\cdots v_{1n}\,[SP_{11}]\cdots[SP_{1k}]} B1' \,,\\ B2 \xrightarrow{[cond_{21}]\cdots[cond_{2s}]\ g\ v_{21}\cdots v_{2n}\,[SP_{21}]\cdots[SP_{2h}]} B2' \,,\ g \in G\end{array}}{B1 \,|[G]|\, B2 \xrightarrow{\substack{[cond_{11}]\cdots[cond_{1r}][cond_{21}]\cdots[cond_{2s}]\\ g\ v_1\cdots v_n\,[SP_1]\cdots[SP_m]}} [\sigma 1]\,B1' \,|[G]|\, [\sigma 2]\,B2'}$

但し，各 v_{1i}, v_{2i} $(1 \le i \le n)$ は次の(a)~(e)内のいずれかの形．

(a) $v_{1i} = ?x_{1i}:s_i\,,\ v_{2i} = ?x_{2i}:s_i$
(b) $v_{1i} = ?x_{1i}:s_i\,,\ v_{2i} = !$ソート s_i の項 t_{2i}
(c) $v_{1i} = !$ソート s_i の項 $t_{1i}\,,\ v_{2i} = ?x_{2i}:s_i$
(d) $v_{1i} = !$ソート s_i の基礎項 $t_{1i}\,,\ v_{2i} = !$ソート s_i の基礎項 t_{2i}
　　($t_{1i} = t_{2i}$ が基礎等式で，$\Omega_{AS} \vdash t_{1i} = t_{2i}$)
(e) $v_{1i} = !$ソート s_i の項 $t_{1i}\,,\ v_{2i} = !$ソート s_i の項 t_{2i}
　　($t_{1i} = t_{2i}$ が基礎等式でない)

(a)の時，$v_i = ?x_i:s_i$ (x_i は新しいユニークな変数名)，$\sigma_{1i} = x_i/x_{1i}\,,\ \sigma_{2i} = x_i/x_{2i}$.
(b)の時，$v_i = !t_{2i}\,,\ \sigma_{1i} = t_{2i}/x_{1i}\,,\ \sigma_{2i}$ = 恒等代入．
(c)の時，$v_i = !t_{1i}\,,\ \sigma_{1i}$ = 恒等代入，$\sigma_{2i} = t_{1i}/x_{2i}$.
(d)の時，$v_i = !t_{1i}\,,\ \sigma_{1i}$ = 恒等代入，σ_{2i} = 恒等代入．
(e)の時，$v_i = !t_{1i}\,,\ \sigma_{1i}$ = 恒等代入，σ_{2i} = 恒等代入，
$[=(t_{1i}, t_{2i})] \in \{[SP_1], \cdots, [SP_m]\}$.

また，$[\sigma 1] = [\sigma_{11}, \cdots, \sigma_{1n}]$，$[\sigma 2] = [\sigma_{21}, \cdots, \sigma_{2n}]$ とおき，
$[\sigma 1]\,([SP_{1j}]) \in \{[SP_1], \cdots, [SP_m]\}$ (各 SP_{1j} $(1 \le j \le k)$ について)
$[\sigma 2]\,([SP_{2j}]) \in \{[SP_1], \cdots, [SP_m]\}$ (各 SP_{2j} $(1 \le j \le h)$ について)
とする． □

B をLOTOS仕様の初期プロセス定義の動作式とする．B に対応するVTS $Vsys_1$ は，上の導出体系を用いて次の定義4.3，定義4.4によって得られる．

【 定義4.3 】: 動作式の derivatives
LOTOS 仕様 CLS=<AS, BS> が与えられたとき，これに関連した動作式 B の derivative の集合 $DER_{CLS}(B)$ は，以下の条件を満たす最も小さな集合である．
① $B \in DER_{CLS}(B)$
② $B' \in DER_{CLS}(B)$ で，ある条件アクション ca について $\Psi_{CLS} \vdash B' \xrightarrow{ca} B''$ ならば $B'' \in DER_{CLS}(B)$ □

動作式 B の derivatives の導出の上で出現する条件アクションの集合を $CA_{CLS}(B)$，条件アクション中に出現する変数の集合を $V_{CLS}(B)$，同様にゲートの集合を $G_{CLS}(B)$ とする．

【 定義4.4 】: 動作式に対応するVTS
LOTOS 仕様 CLS=<AS, BS> の初期プロセス定義の動作式 B に対応するVTS $Vsys_1 = VTS_{CLS}(B)$ は，次の7項組である．

$VTS_{CLS}(B) = <N, G, AS, V, CA, Tr, B>$

ここで，
$N = DER_{CLS}(B)$，$G = G_{CLS}(B)$，$V = V_{CLS}(B)$，
$CA = CA_{CLS}(B)$，$Tr = T_{CLS}$
ただし，
$T_{CLS} = \{ \xrightarrow{ca} \mid \xrightarrow{ca} \subseteq DER_{CLS}(B) \times DER_{CLS}(B), ca \in CA_{CLS}(B)\}$,
$\xrightarrow{ca} = \{(B1, B2) \mid \Psi_{CLS} \vdash B1 \xrightarrow{ca} B2\}$ □

4.2.3 VTSの具現化

LOTOS仕様 CLS=<AS, BS> の初期プロセス定義の動作式 B に対応するVTS $VTS_{CLS}(B)$ の意味は，$VTS_{CLS}(B)$ に出現する変数に具体値を割当て，関連した条件(ガードと選択述語)をASの下で解釈すること(具現化)によって得られるLTSとして定義される．

抽象データ型の代数仕様 AS=<S, OP, E> が与えられたとする．また，S上の変数の集合 $V = \{x_1, \cdots, x_n\}$ が与えられたとする．この時，以下の記法を導入する．

- $Q_{s,AS}$: ソート $s \in S$ の基礎項の AS-合同類の集合(つまり，s の値領域)
  $Q_{s,AS} = \{[t]_{AS} \mid t$ は $s \in S$ の基礎項$\}$
- $sort(x_i)$: 変数 $x_i \in V$ のソート
- $A_{V,AS}$: V の変数の組から成る集合 $\{(x_1, \cdots, x_n)\}$ から変数の値領域 $Q_{sort(x1),AS} \times \cdots \times Q_{sort(xn),AS}$ へのすべての関数の集合(つまり，変数への値割り当て)
- $\sigma(r)$: 項または等式を r とした時，r 中に現れる変数を $\sigma \in A_{V,AS}$ による値を示す基礎項(合同類の代表元)で置き換えて得られる新たな項または等式
- $x \leftarrow v$: 変数 $x \in V$ への値 $v \in Q_{sort(x),AS}$ の割当て
- $\sigma_{[y1 \leftarrow v1, \cdots, ym \leftarrow vm]}$ $(\sigma_{[y1 \leftarrow v1, \cdots, ym \leftarrow vm]} \in A_{V,AS})$: $\sigma \in A_{V,AS}$ に対して，$y_1 \leftarrow v_1, \cdots, y_m \leftarrow v_m$ だけが σ と異なる値割当て

VTSの意味は，以下の定義4.5~定義4.8によって定められる．

【 定義4.5 】: 具現化規則
具現化規則は，以下の①~③からなる．

① $q \xrightarrow{[cond_1]\cdots[cond_m]\,i} q'$ なる遷移が VTS に存在し，各 j $(1 \le j \le m)$ について，$\Omega_{AS} \vdash \sigma(cond_j)$.
$\Rightarrow ((q, \sigma), (q', \sigma)) \in \xrightarrow{i}$

② $q \xrightarrow{[cond_1]\cdots[cond_m]\,g\,d_1\cdots d_n\,[SP_1]\cdots[SP_h]} q'$ なる遷移が VTS に存在し，各 $j\,(1 \le j \le m)$ について，$\Omega_{AS} \vdash \sigma(cond_j)$．また，$\{y_1, \cdots, y_k\} = \{x_i \mid d_i = ?x_i:s_i\,,\ 1 \le i \le n\}$ とおき，各 w $(1 \le w \le h)$ について，$\Omega_{AS} \vdash \sigma_{[y_1 \leftarrow u_1, \cdots, y_k \leftarrow u_k]}(SP_w)$. ここで，$u_z \in Q_{sort(y_z),AS}\,(1 \le z \le k)$.
$\Rightarrow ((q, \sigma), (q', \sigma_{[y_1 \leftarrow u_1, \cdots, y_k \leftarrow u_k]})) \in \xrightarrow{g<v_1 \cdots v_n>}$
ここで，
$v_i = [\sigma(t_i)]_{AS}$ ($d_i = !t_i$ $(1 \le i \le n)$ の時)
$v_i = u_z$ ($d_i = ?x_i:s_i\,,\ x_i = y_z$ $(1 \le i \le n\,,\ 1 \le z \le k)$ の時)

③ $q \xrightarrow{[cond_1]\cdots[cond_m]\,i\,(?y_1:s_1\cdots?y_k:s_k\,[SP_1]\cdots[SP_h])} q'$ なる遷移が VTS に存在し，各 $j\,(1 \le j \le m)$ について，$\Omega_{AS} \vdash \sigma(cond_j)$.
また，各 w $(1 \le w \le h)$ について，$\Omega_{AS} \vdash \sigma_{[y_1 \leftarrow u_1, \cdots, y_k \leftarrow u_k]}(SP_w)$. ここで，$u_z \in Q_{s_z,AS}\,(1 \le z \le k)$.
$\Rightarrow ((q, \sigma), (q', \sigma_{[y_1 \leftarrow u_1, \cdots, y_k \leftarrow u_k]})) \in \xrightarrow{i}$ □

【 定義4.6 】: VTSの再構成遷移集合
VTS $Vsys_1 = <N, G, AS, V, CA, Tr, n_0>$ の再構成遷移集合 RT $(Vsys)$ は，具現化規則によって定義される各 $a \in Act$ についての P 上の関係の集合

$\{ \xrightarrow{a} \mid \xrightarrow{a} \subseteq P \times P\,,\ a \in Act \}$

である．ここで，
$AS = <S, OP, E>$，$P = N \times A_{V,AS}$,
$Act = \{i\} \cup \{ g<v> \mid g \in G, v \in (\cup \{Q_{s,AS} \mid s \in S\})^* \}$ □

【 定義4.7 】: 初期値割当てによるVTSの再構成遷移集合
初期値割当て $\iota \in A_{V,AS}$ によるVTS $Vsys_1 = <N, G, AS, V, CA, Tr, n_0>$ の再構成遷移集合 $RT_\iota(Vsys_1)$ は，$Vsys_1$ の再構成遷

移集合 $RT_\iota(Vsys_1)$ から，状態 (n_0, ι) から到達不能な遷移を取り除いて得られる集合である．□

$RT_\iota(Vsys_1)$ に含まれる状態の集合を $States(RT_\iota(Vsys_1))$
(ただし $States(RT_\iota(Vsys_1)) = \{q \mid q \in P$ は $RT_\iota(Vsys_1)$ において $(n_0, \iota) \in P$ から到達可能$\}$),
アクションの集合を $Actions(RT_\iota(Vsys_1))$
(ただし $Actions(RT_\iota(Vsys_1)) = \{a \mid a \in Act.\ RT_\iota(Vsys_1)$ において $\xrightarrow{a} \neq \emptyset\}$)
と表す．

【 定義4.8 】: VTSの意味
初期値割当て $\iota \in A_{V,AS}$ による VTS $Vsys_1 = <N,G,AS,V,CA,Tr,n_0>$ の意味は，以下で定義される LTS $Sys_2 = TS_\iota(Vsys_1)$ である．

$TS_\iota(Vsys_1) = <States(RT_\iota(Vsys_1)),$
$Actions(RT_\iota(Vsys_1)), RT_\iota(Vsys_1), (n_0, \iota)>$ □

次に，LOTOS仕様 CLS=<AS, BS> の初期プロセス定義の動作式 B に対応して得られる VTS $Vsys_1 = VTS_{CLS}(B)$ を具現化することによって得られる LTS $Sys_2 = TS_\iota(Vsys_1)$ と，B の意味を与える LTS Sys_1 との関係について調べる．まず，2つのLTSの bisimulation 等価を定義4.9，定義4.10によって定める．

【 定義4.9 】: bisimulation 関係
$Sys_1 = <S_1, Act, T_1, s_{01}>$，$Sys_2 = <S_2, Act, T_2, s_{02}>$ を LTS とし，$S = S_1 \cup S_2$ とする．状態対 $(s_1, s_2) \in R$ と，任意のアクション $a \in Act$ に対して，次の条件①と②が成り立つ時，関係 $R \subseteq S \times S$ を bisimulation 関係という．
① $s_1 \xrightarrow{a} s_1'$ である $s_1' \in S$ が存在するならば，
$s_2 \xrightarrow{a} s_2'$ で $(s_1', s_2') \in R$ である $s_2' \in S$ が存在する．
② $s_2 \xrightarrow{a} s_2'$ である $s_2' \in S$ が存在するならば，
$s_1 \xrightarrow{a} s_1'$ で $(s_1', s_2') \in R$ である $s_1' \in S$ が存在する． □

【 定義4.10 】: LTS の bisimulation 等価
$Sys_1 = <S_1, Act, T_1, s_{01}>$，$Sys_2 = <S_2, Act, T_2, s_{02}>$ を LTS とし，$S = S_1 \cup S_2$ とする．$(s_{01}, s_{02}) \in R$ であるような bisimulation 関係 $R \subseteq S \times S$ が存在する時，Sys_1 と Sys_2 は bisimulation 等価であるという． □

LOTOS 勧告書(ISO 8807)で示される LTS の導出体系(以下，体系 A と呼ぶ)と上で示した VTS の導出体系(以下，体系 B と呼ぶ)の関係について，次の定理が成り立つ．

【 定理4.1 】LTS の導出体系と VTS の導出体系の関係
LOTOS 仕様 CLS=<AS,BS> に関連した閉じた動作式を B とする．この時 Sys_1 と Sys_2 は bisimulation 等価である．ここで，
$Sys_1 = TS_{CLS}(B)$，$Sys_2 = TS_\iota(VTS_{CLS}(B))$
ただし，
$TS_{CLS}(B)$: B に対応する LTS
$VTS_{CLS}(B)$: B に対応する VTS
ι : $VTS_{CLS}(B)$ の変数への任意の初期値割当て
$TS_\iota(VTS_{CLS}(B))$: ι による $VTS_{CLS}(B)$ の意味としての LTS. □

次の例4.1によって，定理4.1が成り立つことを示す．

【 例 4.1 】bisimulation 等価の例
次のような動作式 B を考える(関連したデータ型については省略．ガードおよび選択述語の書き方は省略形を採用)．

```
B  =  hide g in g ? x : nat [0≦x≦1]; f ? y : nat;
          (  [0≦y≦1]  -> (h ! x; stop | [h] | h ! y; stop)
          [] [y≧2]    -> e ! err; stop    )
```

B に対応する LTS $TS_{CLS}(B)$ は図4.2の通りである．
また，B に対応する VTS $VTS_{CLS}(B)$ は図4.3の通りである．
さらに，この VTS の意味は図4.4の LTS $TS_\iota(VTS_{CLS}(B))$ である．
次のような状態間の関係を $TS_{CLS}(B)$ と $TS_\iota(VTS_{CLS}(B))$ の間に与えることによって，両者は bisimulation 等価となる．

s0 ⇔ (s0', ι)　　s1 ⇔ (s1', σ1)　　s2 ⇔ (s1', σ5)
s3 ⇔ (s2', σ2)　　s4 ⇔ (s2', σ3)　　s5 ⇔ (s2', σ4)
s6 ⇔ (s2', σ6)　　s7 ⇔ (s2', σ7)　　s8 ⇔ (s2', σ8)
s9 ⇔ (s3', σ2)　　s9 ⇔ (s3', σ7)　　s10 ⇔ (s4', σ4)
s10 ⇔ (s4', σ8)　　…　　□

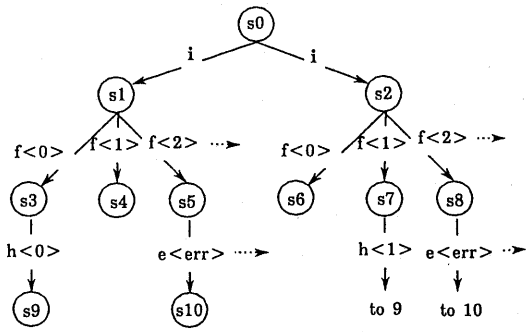


図4.2　B に対応する LTS $TS_{CLS}(B)$

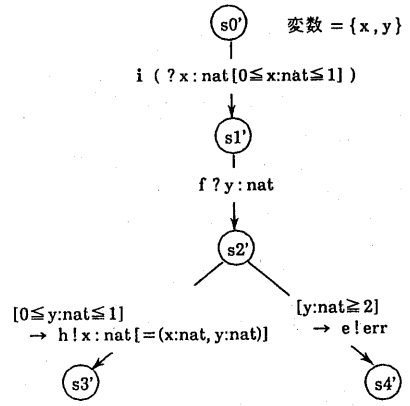
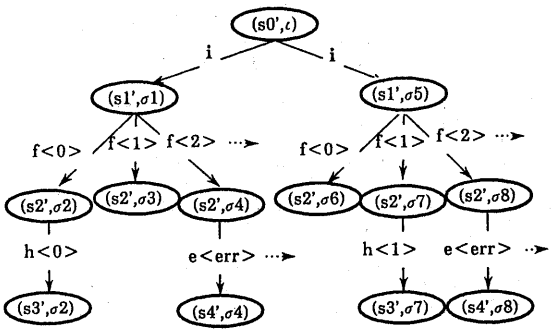


図4.3　B に対応する VTS $VTS_{CLS}(B)$



ここで，
ι = [x と y へのランダムに選んだ値割り当て]
$\sigma 1 = \iota_{[x \leftarrow 0]}$　　$\sigma 2 = \sigma 1_{[y \leftarrow 0]}$　　$\sigma 3 = \sigma 1_{[y \leftarrow 1]}$
$\sigma 4 = \sigma 1_{[y \leftarrow 2]}$　　$\sigma 5 = \iota_{[x \leftarrow 1]}$　　$\sigma 6 = \sigma 5_{[y \leftarrow 0]}$
$\sigma 7 = \sigma 5_{[y \leftarrow 1]}$　　$\sigma 8 = \sigma 5_{[y \leftarrow 2]}$

図4.4　$TS_\iota(VTS_{CLS}(B))$

以上のように，具現化によって VTS の意味を与えることができる．これに基づいて，VTS の等価性は次のように定められる．

【 定義4.11 】: VTS の等価性
初期値割当て ι が与えられたとき，2つの VTS $Vsys_1$, $Vsys_2$ は，それぞれの意味を与える LTS $Sys_1 = TS_\iota(Vsys_1)$, $Sys_2 = TS_\iota(Vsys_2)$ が等価である時，その時に限って ι のもとで等価であるという． □

次に，具現化による VTS と LTS のそれぞれの状態の対応関係 P を定める．

【 定義4.12 】: 具現化による状態の対応関係P
VTS $Vsys_1$=<N,G,AS,V,CA,Tr,n_0>の意味として得られるLTS Sys_2を
$Sys_2 \rightleftharpoons TS_t(Vsys_1) = <S, A, T, s_0>$
ただし, $S=States(RT_t(Vsys_1))$, $A=Actions(RT_t(Vsys_1))$,
$T = RT_t(Vsys_1)$, $s_0 = (n_0, \iota)$
とする. この時, 関係 $P \subseteq N \times 2^S$ を次のように定める.
$n_i \in N, S_i \subseteq S$に対して,
$(n_i, S_i) \in P$
iff $S_i = \{(n_i, \sigma) \mid (n_i, \sigma) \in S\}$ □

4.3 最小VTSの導出

前節で定めたVTSとLTSの状態間の関係PとLTS上での状態の等価性を表す弱bisimulation関係"$\sim_S$"を用いて, VTSの状態間の関係"$\sim_N$"を定める.

【 定義4.13 】: VTSの状態間の関係"$\sim_N$"
VTS $Vsys_1$=<N,G,AS,V,CA,Tr,n_0>の意味として得られるLTS Sys_2を
$Sys_2 = TS_t(Vsys_1) = <S, A, T, s_0>$
ただし, $S=States(RT_t(Vsys_1))$, $A=Actions(RT_t(Vsys_1))$,
$T = RT_t(Vsys_1)$, $s_0 = (n_0, \iota)$
とする. 状態集合Sの部分集合B_i, B_jに対して,
$s_i \in B_i \Rightarrow \exists s_j \in B_j, s_i \sim_S s_j$,
$s_j \in B_j \Rightarrow \exists s_i \in B_i, s_j \sim_S s_i$
(ただし, $(n_i, B_i), (n_j, B_j) \in P$)
が成立する時, その時に限って状態集合Nの要素n_iとn_jが"等価である"といい, "$n_i \sim_N n_j$"と表す. □

上のようにして定められた関係"$\sim_N$"について, 次のような定理が成り立つ.

【 定理4.2 】
関係"$\sim_N$"は, VTSの状態集合N上の同値関係である. □

同値関係"$\sim_N$"によって, VTS $Vsys_1$の状態集合Nを同値類に分割すると, 各同値類に属する状態は互いに等価である. 従って, 得られた同値類分割の代表元を状態とする新しいVTS $Vsys'$を構成すれば, $Vsys_0'$はもとのVTS $Vsys_1$と等価で, 状態数が最少のものになる. すなわち, $Vsys_0'$は冗長な状態を含まない. この$Vsys_0'$を$Vsys_1$の"準最小VTS"と呼ぶ.

【 定義4.14 】: 準最小VTS
VTS $Vsys_1$ = <N,G,AS,V,CA,Tr,n_0>の状態集合Nを同値関係"$\sim_N$"による同値類分割を行い, 得られた分割πを基に次のように構成される $Vsys_0'$ = <N',G,AS,V,CA,Tr',n_0'> を$Vsys_1$の準最小VTSと定義する.
(1)N'={[n]|n∈N, [n]は分割πのnが属する同値類の代表元},
(2) Tr'={[n] —ca→ [n'] | n —ca→ n', ca∈CA},
(3) n_0'=[n_0]. □

準最小VTS $Vsys_0'$は, 冗長な状態を含まないが, 冗長な遷移を含む場合がある. すなわち, $Vsys_0'$の2つの状態nとn'において, $n \xrightarrow{ca} n'$と$n \xRightarrow{ca} n'$ ($ca \in CA$)という異なる遷移が存在する可能性がある. そこで, $Vsys_0'$から等価性を保存したまま冗長な遷移の削除を行って構成される$Vsys_0$=<N',G,AS,V,CA,Tr",n_0'>を最小VTSと定める.

【 定義4.15 】: 最小VTS
準最小VTS $Vsys_0'$ = <N',G,AS,V,CA,Tr',n_0'> から次のようにして構成される$Vsys_0$ = <N',G,AS,V,CA,Tr",n_0'> を $Vsys_1$の最小VTSと定義する.
Tr" = Tr' − { $n \xrightarrow{ca} n'$ | $n \xrightarrow{ca} n'$と異なる遷移
$n \xRightarrow{ca} n'$が存在する. ($ca \in CA$) }. □

以上のようにして定義される最小VTS $Vsys_0$=<N',G,AS,V,CA,Tr",n_0'>とVTS $Vsys_1$=<N,G,AS,V,CA,Tr,n_0>との関係について, 定義4.11に基づいて次の定理が成り立つ.

【 定理4.3 】
VTS $Vsys_1$ = <N,G,AS,V,CA,Tr,n_0> と, 最小VTS $Vsys_0$ = <N',G,AS,V,CA,Tr",n_0'> とは等価である. □

4.4 最小TreeLOTOS

この節では, 前節で得られた最小VTSから, 最小TreeLOTOSを得る方法について述べる.
TreeLOTOSは, LOTOSのサブセットとして次のように定める.

【 定義4.16 】: TreeLOTOS
LOTOSのシンタクスを **choice-expression**[3] のみに制限したLOTOSのサブセットをTreeLOTOSという. □

最小VTSから, 最小TreeLOTOSを得るために, ここでは次の手順を考える.
まず, 最小VTSをTree状に変換(Tree化)し, Tree化最小VTSを得る. このTree化最小VTSはTreeLOTOSと構造がよく一致するため, これから試験木生成規則によって直接的に最小TreeLOTOSに変換することができる.
最小VTSのTree化を定義するために, VTSにおけるパスPath(n)を定義4.17のように定める. これは, 初期状態n_0から始まり状態nに至る全ての系列中の状態の集合である.

【 定義4.17 】: パス
VTS $Vsys$=<N,G,AS,V,CA,Tr,n_0>において,
(1) $n \in Path(n) (n \in N)$
(2) $n_0 \xrightarrow{ca_1} n_1 \xrightarrow{ca_2} \cdots \xrightarrow{ca_{k-1}} n_{k-1} \xrightarrow{ca_k} n_k$
(ただし, $n_1, \cdots, n_k \in N, n_i \neq n_j (i \neq j, i, j = 0, \cdots, k)$,
$ca_1, \cdots, ca_k \in CA$)
なる遷移系列が存在する.
$\Rightarrow n_i \in Path(n_k) (i = 0, \cdots, k)$ □

Tree化最小VTSを次のように定める. これは, 最小VTSの中の合流を"開いた"ものである.

【 定義4.18 】: Tree化最小VTS
最小VTS $Vsys_0$ = <N',G,AS,V,CA,Tr",n_0'>が与えられたとき, 以下のようにして得られるVTS $Vsys_t$ = <N",G,AS,V,CA,Tr"',n_0'>をTree化最小VTSと定義する.
step0 Tr"' ← Tr"; N" ← N'
step1 if $n_{11} \xrightarrow{ca_{11}} n_{20}, \cdots, n_{1n} \xrightarrow{ca_{1n}} n_{20}$
(ただし, $ca_{1i} \in CA$, $n_{20} \in N'$,
$n_{20} \notin Path(n_{1i}) (i=1,\cdots,n) (n \geq 2)$)
なる遷移が存在.
then Tr"' ← Tr"' ∪ { $n_{11} \xrightarrow{ca_{11}} n_{21}, \cdots,$
$n_{1n} \xrightarrow{ca_{1n}} n_{2n}$}
(但し, $n_{21}, \cdots n_{2n} \notin N'$);
Tr"' ← Tr"' − { $n_{11} \xrightarrow{ca_{11}} n_{20}, \cdots,$
$n_{1n} \xrightarrow{ca_{1n}} n_{20}$};
N" ← N" ∪ { $n_{21}, \cdots, n_{2n}$};
N" ← N" − { n_{20}}; goto step2
else then stop.
step2 if $n_{20} \xrightarrow{ca'} n_3$ ($ca' \in CA$, $n_3 \in N'$)
なる遷移が存在.
then Tr"' ← Tr"' ∪ { $n_{21} \xrightarrow{ca'} n_3, \cdots,$
$n_{2n} \xrightarrow{ca'} n_3$};
Tr"' ← Tr"' − { $n_{20} \xrightarrow{ca'} n_3$};
goto step2
else then goto step1 □

Tree化最小VTSにおいて, 表4.1のような対応をつけることによってこれを直接的にTreeLOTOSに変換することができる.

Tree化最小VTS	TreeLOTOS
状態	Behaviour式
条件 action	条件 action
actionの系列	action prefix expression
Tree の分岐	choice expression

表4.1 Tree化最小VTSとTreeLOTOSの対応

この対応関係に基づいて, 最小TreeLOTOSを次のように定める.

【 定義4.19 】: 最小TreeLOTOS
Tree化最小VTS $Vsys_t$ = <N",G,AS,V,CA,Tr"', n_0'> より表4.1

の対応関係に基づいて変換して得られるTreeLOTOSを最小TreeLOTOSと定義する．□

5. 適用例

以上のことを実際の仕様例に適用した例を示す．次のような仕様を考える．(データ型については省略)

```
specification EXAMPLE [U,L] : noexit
behaviour
V?m:int[0≦m≦2] ; P0[L,U](m)
where
   process P0[L,U](m:int) : noexit :=
      U?x:mes?v:int ; U!nak ; P0[L,U](m)
   [] U?x:mes?v:int ;
         (  i ; U!nak ; P0[L,U](m)
         [] L!x[v=m] ; U?x:mes?v:int ;
            (  L!x[v=m] ; P0[L,U](m)
            [] i ; U!nak ; P0[L,U](m)
         ) )
   endproc
endspec
```

この仕様例より得られるVTSは図5.1のようになる．

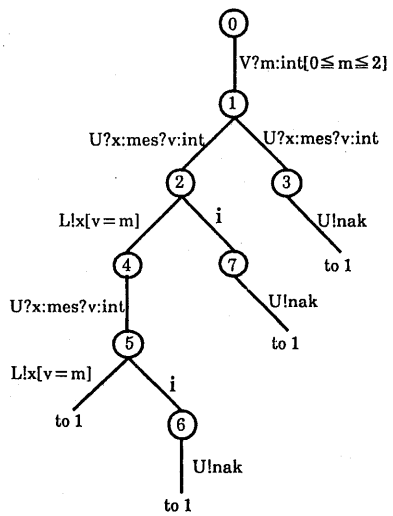


図5.1　仕様例より得られるVTS

さらにこのVTSより得られる最小VTSは図5.2のようになる．

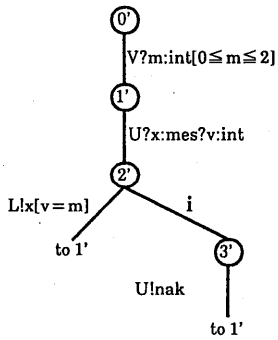


図5.2　図5.1のVTSより得られる最小VTS

最後にこの最小VTSより得られる最小TreeLOTOSは次のようになる．

```
specification EXAMPLE-Tree [U,L] : noexit
behaviour
V?m:int[0≦m≦2] ; P0[L,U](m)
where
   process P0[L,U](m:int) : noexit :=
      U?x:mes?v:int ;
         (  i ; U!nak ; P0[L,U](m)
         [] L!x[v=m] ; P0[L,U](m)
         )
   endproc
endspec
```

6. まとめ

本稿では，データ構造を扱ったLOTOS仕様からの試験系列の自動生成の手法を提案した．ここでは，試験系列を試験スイートとして定め，より実用的な試験スイートを得るために，変数付きLTSであるVTSを導入した．そして，VTSの導出体系とVTSの意味を与える具現化規則を定め，この導出体系がLOTOSで定められているLTSの導出体系と矛盾がないことを示した．さらに，より効率的な試験スイートを得るために，弱bisimulation等価の概念によるVTSの最小化を行った．ここでは，VTSの等価性と最小形を定め，最小形が等価性を保存することを示した．

本稿では，試験スイートはLOTOSの構文を制限したサブセットであるTreeLOTOSによって表された．このTreeLOTOSは，木状構造であるため，ISOで規格化が進められている試験系列記述用言語TTCN[2]とその構造がよく一致する．そこで，このTreeLOTOSをTTCN表現に変換して表すことが考えられる．

現在，本方法を用いた試験スイートの自動生成の支援システムを構築中である．

今後の課題としては，この支援システムの構築の他，大規模なシステムの記述に対する本方法の有効性の確認，本方法で得られる試験スイートを用いた試験実施の支援方法の研究等がある．

参考文献

[1] ISO : "OSI Conformance Testing Methodology and Framework Part 2 : Abstract Test Suite Specification" ISO 9646.

[2] ISO : "OSI Conformance Testing Methodology and Framework Part 3 : The Tree and Tabular Combined Notation (TTCN)", ISO/DP 9646.

[3] ISO : "LOTOS - A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour",ISO 8807, 1989.

[4] S. Naito and M. Tsunoyama : "Fault detection for sequential machines by transition tours," in *Proc. IEEE Fault Tolerant Comput. Conf.*, 1981.

[5] T. Chow : "Testing software design modeled by finite-state machines," *IEEE Trans. Software Eng.*, vol. SE-4, pp. 178-187, 1978.

[6] G. Gonenc : "A method for the design of fault detection experiment," *IEEE Trans. Comput.*, vol. C-19, pp. 551-558, 1970.

[7] 佐藤 他 : "有限オートマトンに基づくシステムの試験系列自動生成手法の提案 —単一遷移検査系列法—", 信学会論文誌, vol. J 72-B-I No. 3, 1989.

[8] D. P. Sidhu and T, Leung : "Formal Method for Protocol Testing : A Detailed Study," *IEEE Trans. Software Eng.*, vol. 15-4, pp. 413-426, April 1989.

[9] B. Sarikaya, G. Boshmann, E. Cerny : "A test design methodology for protocol testing," *IEEE Trans. Software Eng.*, vol. SE-13, No. 5, 1987.

[10] E. Brinksma : "A theory for the derivation of tests," *Eighth IFIP WG6.1 Protocol Specification, Testing, and Verification* , 1988.

[11] 神長, 高橋, 白鳥, 野口 : "LOTOS仕様の効率的な等価性判定法", 電子情報通信学会論文誌, Vol. J-73-D-1 No. 2 pp. 214-224, 1990.

[12] 高橋, 神長, 白鳥 : "LOTOS言語の特質と処理系の現状と動向", 情報処理学会誌, Vol. 31 No. 1 pp. 35-46, 1990.

[13] 佐藤 他 : "SDL記法からの実行型試験シーケンスの生成 ", 情報処理学会研究報告, MDP 39-6, 1988.

[14] 岡崎, 高橋, 白鳥, 野口 : "LOTOS仕様からのTTCN表現によるテストシーケンスの自動生成", 電子情報通信学会技術報告, IN 89-26, 1989.