プロトコルにおける並列動作を含むイベントシーケンスの生成と表示

4T-2

中原 彰子，相田 仁，斉藤 忠夫
東京大学　工学部

1．はじめに

通信プロトコルの適合性試験では，試験系列や期待される出力をシーケンス図の形で表す場合が多い．そこで，本稿では，各プロセスの動作仕様がツリーの形で定義される場合に，プロセス間の通信を対話的に実行して一連のシーケンスを求め，シーケンス図の形で表示するソフトウェア・ツールについて報告する．

以下，まず2節で，それぞれのプロセスおよびプロセス構造の記述形式および記述例を示す．次に3節で，通信の対話的な実行によるシーケンスの抽出について検討する．さらに4節で，シーケンス図の表示について説明する．

2．システムの記述

2．1　プロセスの記述

本ツールでは，字句／構文解析プログラム（yacc/lex）を用いて LOTOS 記述[1]を読込み，これをツリー構造に変換したものを，プロセスの動作仕様として用いる．ツリーはループによる繰り返しを含んでもよい．ループはツリー上の等価な状態を表す識別子を用いて表現する（図1）．

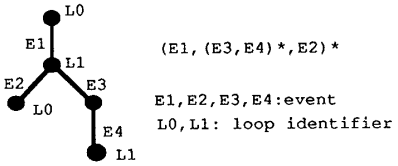


図1．ツリー構造とループの表現

ツリー上の各枝は1つの入出力動作（イベント）に対応し，全体として，プロセスの動作として許されるイベントの順序を規定している．イベントによって受渡しされるデータは，値（"!値"）または片付けされた変数（"?変数名:型"）である．ここで用いるデータ型は，LOTOS のデータ定義部で定義される．本ツールでは，処理の簡略化のため，データ型として値が有限個の要素で表されるものだけを扱う．

2．2　論理プロセス構造の記述

本ツールでは，記述対象システムを静的なプロセスの組合せで表現する．各々のプロセスは，2.1 のツリー構造を用いて記述されている．このような静的な論理プロセス構造を仮定することにより，各層のプロトコル・エンティティなど，システム内に静的に存在するプロセス間で生じる入出力動作を明示的に示すことができる（図2）．

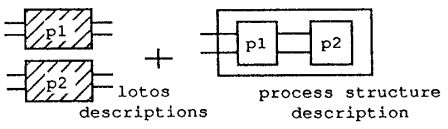


図2．記述対象システムの表現

プロセス構造の記述では，システムに共通な大域的イベント名およびゲート名を，イベントごとに宣言して用いる．また，LOTOS ではプロセス間の通信にランデブー方式を用いているため，通信の向きが特定されない．そこで，これらの宣言とともに，入出力の向きの宣言も行う．

2．3　記述例

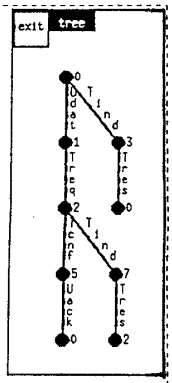
データ転送におけるトランスポート・ユーザ（テスタ）の動作記述例を例1に示す．例1の記述では，プロセスの formal_list は省略して，状態の識別子として扱っている．

例1．論理プロセスの記述

(a) テスタの記述

```
where process Idle :=
      t_in ?d1:data_type
        ; tsap_req ?d1:data_type
          ; Wait
  [] tsap_ind ?d2:data_type
        ; tsap_res! ; Idle
endproc

where process Wait :=
      tsap_cnf!; t_out!; Idle
  [] tsap_ind ?d2:data_type
        ; tsap_res! ; Wait
endproc
```



(b) イベントの宣言

```
Udat[INev, cnt1] := (t_in ?data:data_type)
Uack[OUTev,cnt1] := (t_out!)
Treq[OUTev,tsap] := (tsap_req ?data:data_type)
Tind[INev, tsap] := (tsap_ind ?data:data_type)
Tres[OUTev,tsap] := (tsap_res!)
Tcnf[INev, tsap] := (tsap_cnf!)
```

3．シーケンスの生成

3．1　シーケンスの定義

システム全体の状態変化は，システムを構成する論理プロセス間での通信により表現される．各論理プロセスが実行した一連のイベント（経路と呼ぶ）の集合をシーケンスと呼ぶ．pi をシステムを構成する論理プロセスとして，

シーケンス　＝　{プロセスpi の動作経路}

経路上のイベントは，pi 上での実行順序，および通信による相手プロセスとの同期，の2つの時間的制約を持つ．

3．2　シーケンスの選択条件の記述

一般にシステム全体の動作として可能なシーケンスは複数存在する．しかし，すべての可能な選択枝を利用者に順に提示する方式では，選択に手間がかかり，システム全体の動作も理解しにくい．このため本ツールでは，外部と直接やりとりを行うプロセスに注目し，外部から観察される動作をあらかじめ条件として記述する方式をとる．

A Method for Generation and Desplay of Concurrent Event-Sequence in Protocols
Akiko NAKAHARA, Hitoshi AIDA, Tadao SAITO
University of TOKYO

3．3　シーケンスの選択と実行

次に，3.2 で記述した条件を満たすシステム全体の動作を求める．通信の方式は，双方のプロセスが待ち合わせをするランデブー方式である．実行では，あるプロセスが通信要求を出して待ち状態にはいると，相手側のプロセスを呼出し，通信の成功を確認する．相手側のプロセスについて複数の動作候補がある場合には，利用者への提示／選択を行う．なお本ツールでは，有限領域上での単一化の機構によりランデブーを実現している．

3．4　実行例

トランスポート・テスタ（上位）とトランスポート・エンティティについて，経路の選択と実行を行った結果を示す（例2）．この例では，"テスタにデータ送信を要求(Udat)してから応答(Uack)を受け取るまで"という条件で，全体のシーケンスを求めている．実行にあたっては，まず，プロセス"tester"の動作経路を提示／選択し，次に通信相手のプロセス"s_trans"を呼び出している．

例2．経路の実行と選択

```
$> (pathenv 'tester '((Udat)) '((Uack)))   …条件の記述
1: (Udat)(Treq)(Tcnf)(Uack)                …経路の提示
2: (Udat)(Treq)(Tind)(Tres)
   (Tcnf)(Uack)

   select num: 1                           …経路の選択
 ...move[tester@module0]

call_master[tester]: (t_in @data_type0)    …実行
call_master[tester]: (tsap_req @data_type0)
tester:o_s_d:(@data_type0):suspends        …待ちの発生

call_slave[s_trans]o_s_d                   …相手モジュ
 ...move[trans@module1]                      ール呼出し
1: (Treq)
........

 ... path succeed                          …経路の成功
tester:                                    …得られたシ
   (t_in @data_type0)                        ーケンス
   (tsap_req @data_type0)
   (tsap_cnf)(t_out)
s_trans:
   (tsap_req @data_type2)(tsap_cnf)
```

4．シーケンスの表示

4．1　表示に用いるシーケンス図の構造

図3にシーケンス図の構造を示す．縦棒(①)は論理プロセスに，縦棒と縦棒の間(②)はプロセス間を結ぶゲートに，横線(③)は論理プロセス間での通信に，それぞれ対応する．また，横線と横線の間は時間的な前後関係を表し，横線の矢印はプロセス間のデータの流れの向きを表す．

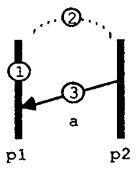


図3．シーケンス図

本ツールでは，シーケンス図の見やすさを考慮して，すべてのプロセスを表示するのではなく，一列に並んだ一部のプロセスだけを表示する．表示プロセスは，ウィンドウを介して，プロセス（ゲート）の V*isual_flag*を ON/OFF することにより，対話的に選択する．

4．2　シーケンスツリーの定義

異なる2組のプロセス間でローカルに生じる通信は順不同であるとみなす．また，1つの入力に続いて複数の出力動作が生じる場合にも，異なる相手プロセスに対する出力は順不同であるとみなす．本稿では，3.1 で定義したシーケンスから，このような並列動作を取り出したものを，シーケンスツリーと呼ぶ．シーケンスツリーの分岐は並列動作を表している．また，ツリー上の等価な状態を参照するラベルによって，並列動作の同期点を表現する．

4．3　シーケンスツリーの生成と表示

4.3 で得られたシーケンスから，以下の手順にしたがって，シーケンスツリーを生成する．

(1) 観察点のプロセスに注目する．
(2) 注目しているプロセスが実行した経路上の先頭の入力／{出力列} ペアを取り出す．
(3) 入力イベントはシーケンスツリーに加える．
(4) 出力イベントならば，相手プロセスの経路に注目して，(2)-(4) を繰り返す．異なるプロセスへの出力はシーケンスツリー上の分岐点になる．
(5) すべてのシーケンスを加えたら，シーケンスツリー上の終端状態における待ち合わせを記述する．

シーケンス図の表示では，シーケンスツリー上での深からイベントの表示位置を求める．シーケンス図表示の例を図4に示す．

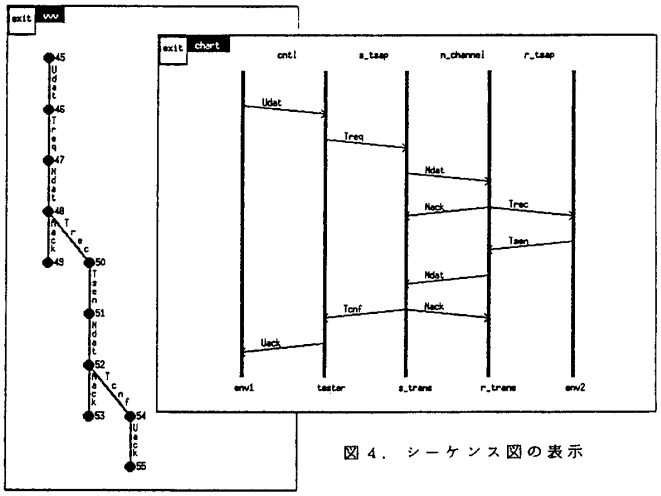


図4．シーケンス図の表示

5．おわりに

本ツールでは，ツリー構造で記述されたプロセスの組合せで対象システムを表現する．そして，並列動作を含むシステム全体の動作を，シーケンスツリーと呼ぶ形式で表し，これをもとにシーケンス図の表示を行う．シーケンスツリーに時間記述を加えることによって，時間的なタイミングを考慮した誤りの検出と簡単な性能の見積りが可能である[2]．シーケンス生成のためのインタフェースの検討，誤りやタイムアウトの扱いの検討などが今後の課題である．

参考文献

(1) ISO: "LOTOS - Formal Description Technique Based on the Temporal Ordering of Observational Behaviour," ISO/DIS8807, 1987.
(2)中原,相田,斉藤: "プロトコル仕様の時間検証および性能評価のための視覚的ツール", 信学全大, 1989（発表予定）．