

会話形式による通信プロトコル検証システム

覚 埜 高 音 石 坂 充 弘

(三菱電機株式会社 情報電子研究所)

1. まえがき

近年のデータ通信の発展に伴い多くの端末や計算機間の相互通信が、データ通信網を利用して促進される方向にある。特に、オフィスオートメーションシステムにおけるデータ通信網として各種のローカルエリアネットワークが発表されている。

このような各種データ通信網では、端末一網間、端末一端末間で通信回線を介してデータなどを送受する信号形式や手順に関する規約（通信プロトコル）を定めている。

この通信プロトコルが一意に解釈できなかつたり、論理的に矛盾があると、送信データが正しく相手に到達しなかつたり、網が動作不能となつたりする。このため通信プロトコルの設計において一意に解釈可能な記述手段と、規定した通信プロトコルに論理的矛盾がないかを調べる検証手段が必要であり、種々の手法が発表されている。(1)～(6)

これまでの検証システムの多くはバッチ処理で行われており、設計の初期段階では、同一原因による論理誤り結果が多大なリスティング用紙として出力され、これを検査するだけでも多くの時間と手間を費やした。本稿で発表する通信プロトコル検証システムPVS-IはTSS端末から会話的にシミュレータを利用することにより、検証中にユーザが介入でき、設計誤りを早期に発見・修正できる特徴がある。

PVS-Iは、2者間の通信プロトコルを対象とし、伝送路上での送信信号の紛失を含め両者の通信プロトコルの状態遷移の所定状態への到達可能性を計算機上で網羅的にシミュレートし、そのシミュレートした状態遷移系列をディスプレイ上に出力するとともに、その系列の所定状態への到達可能性(1)正常、(2)デッドロック、(3)ループがある)及びイクセプス(excessive:所定の状態に達したが信号の授受が正しく行われない場合)の判定結果を表示する。また、通信プロトコルの修正時などに、信号の授受が正しく行われる方向に設計者を導くガイダンス機能も有している。

以下、2章では、対象となる通信プロトコルモデル

と検証アルゴリズムについて、3章ではPVS-Iの構成とその機能について紹介し、4章で利用方法と適用例について述べ、最後に5章で性能評価を行う。

2. プロトコルモデルと検証アルゴリズム

2.1 プロトコルモデル

通信しあう2個のプロセスが各々図2.1に示すような状態遷移図で表現されるプロトコルをPVS-Iの検証対象としている。両プロセス間でやりとりされる信号等の遷移要因(以下イベントと呼ぶ)は、アークに付されている信号で表現される。信号名に付された「+」は信号を受理することを、「-」は信号を送信することを、また「*」はプロセス間で授受されないイベント、例えばタイムアウトや上位層からの起動命令などを示す。

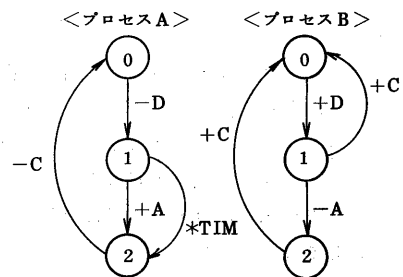


図 2.1 プロトコルモデル

このPVS-Iでは、演算、数値の大小比較などの処理を行わないため、授受する信号のシーケンス番号などが異なると、すべて別イベント及び別状態として定義してモデル化する必要がある。

その他モデル化に対して以下のような定義、制限を設けている。

① 各プロセスは到着した信号を処理する前に一旦待ち行列(以下QUEUEと呼ぶ)に入れ、実際に受理して遷移する時に到着順にこのQUEUEから取出す。このQUEUEはFIFOである。各プロセスの受

信バッファ数の制限はこのQUEUEの長さに制限を設けることによりモデル化する。例えば、受信バッファが1の場合には、 $|QUEUE| \leq 1$ とする。

② 伝送路遅延はない。

③ タイムアウトは、タイムアウト時間を計測するのではなく、上位層からの送信起動と同様にプロセスのある状態における入力として定義する。

④ 伝送誤りは、相手プロセスにおいてパリティ誤りやブロック誤りなどとして検出されるものであるが、ここでは誤りのある信号の送受信としてユーザが定義する。

⑤ 信号の伝送紛失については、このPVS-Iがユーザの定義なしに自動的にその状態を発生する。

2.2 検証アルゴリズム

このPVS-Iで用いているアルゴリズムは、一般に良く用いられる「到達可能解析」により行っている。直観的には規定されたプロトコルの状態を次々にトレースしていくものであり、遷移系列を逐次生成しながら下記検証項目を調べる。この時、ユーザの指示があれば、すべての送信イベントに対して紛失状態への遷移も自動的に行う。

検証項目

- ① 正常 両プロセスが所定の状態に到達し、各QUEUEが空きの場合の遷移系列。
- ② デッドロック 両プロセスが互いに待ち状態となる、いわゆるすくみの状態に陥った遷移系列。
- ③ ループ 正常な動作に復帰できずに無限回同じ動作を繰り返す遷移系列。
- ④ excessive 両プロセスは所定の状態に到達するがその時、どちらかまたは両方のQUEUEが空きでない遷移系列。
- ⑤ 疑似終了 この項目は上記①～④とは異なり、計算時間の短縮と出力結果の削減を計るために設けた遷移系列判定である。検証している遷移系列において後続の遷移が以前に検証した遷移と同じになると判定した場合に、その時点でこの遷移系列を疑似終了として検証をやめる。

検証アルゴリズムの概要は以下の通りで、これを図2.1に示すプロトコルモデルに適用した例を図2.2に示す。

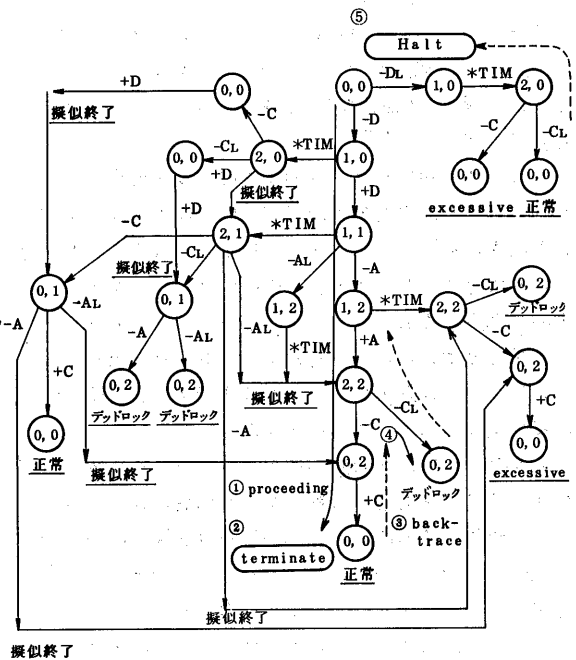
① 両プロセスA、Bは初期状態から開始する。この状態において、両プロセスが遷移可能なイベント中の1つを選び（図2.2では信号Dの送信）、その遷移先に進む（これをproceedingと呼ぶ）。

② 順次proceedingを繰返し、所定の状態に達するか、またはproceedingできない状態になると（これをterminateと呼ぶ）、その状況に応じ前述の検証項目と遷移系列を出力する。

③ 次に今まで進んで来た経路を逆戻りし、最寄りの未選択イベントが存在する合成状態（これを分岐点と呼ぶ）に達するまで行われる（この動作をback-traceと呼ぶ）。

④ 分岐点に達すると、未選択の遷移可能なイベントの1つを選び②に戻る。以上②～④の動作を繰返す。

⑤ 最後に初期状態に戻り検証を終了する（Haltと呼ぶ）。



→ : proceeding

---> : back-trace

XL : 送信信号Xの伝送路紛失

(X,Y) : プロセスAとプロセスBの合成状態

図 2.2 検証アルゴリズム

3. PVS-I

3.1 構成

PVS-Iは図3.1に示すプログラム構成で、MEL COM-COSMO 900ⅡのオペレーティングシステムUTS/VS上でTSSで運用される。各プログラムはファイルを介して情報の受渡しを行う。以下にその機能概略を示す。

(1) 入力処理プログラム (PVS:IN)

入力プロトコルの記述フォーマット等の検査を行い、次の検証処理プログラムで実行可能な形式に変換する。

(2) 検証処理プログラム (PVS:VAL)

入力されたプロトコルの誤り検出、TSS端末からのコマンド、修正データなどの入力制御、検証の中断/再開のためのセーブ/リストア制御などを行う。会話形式において用いられるコマンドを表3.1に示す。受信バッファ数や伝送路上での信号紛失の有無の指定は、このプログラムが起動された時に初期設定としてユーザが入力するようになっている。

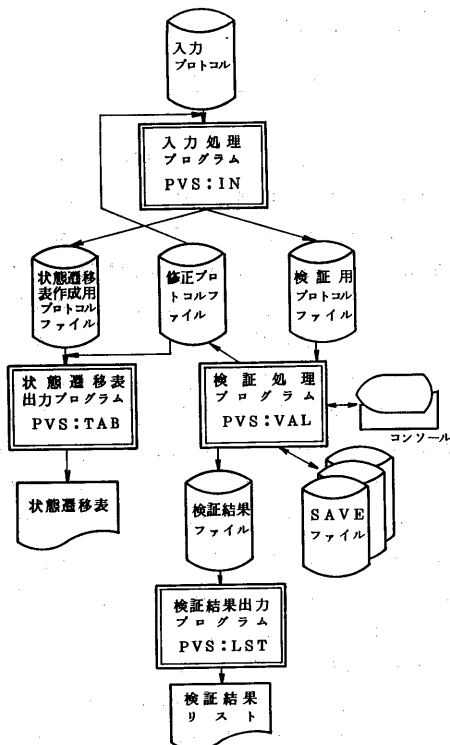


図 3.1 PVS-Iのプログラム構成

(3) 検証結果出力プログラム (PVS:LST)

検証結果を検証項目別にラインプリンタに出力する。

(4) 状態遷移表出力プログラム (PVS:TAB)

入力されたプロトコルデータに従って状態遷移表を出力する。図2.1の状態遷移表を図3.2に示す。

表 3.1 コマンド一覧表

コマンド	DISPLAY モード	UPDATE モード	機 能 内 容
AM	△	○	検証遷移系列数の指定
GO	○	○	検証の実行指示
END	○	○	全実行の終了指示
SE	△	○	各プロセスの初期状態番号及び到達状態番号の指定
EE	△	○	外部イベント(*×××)の選択
RESET		○	シミュレータを初期化する
SAVE		○	検証の中断及び途中結果のセーブ
LM	△	○	プロセスの各状態の通過回数設定
TD		○	遷移系列の画面表示を抑圧する このコマンドに続いて、抑圧する検証項目を入力する
TRACE		○	検証結果を通信シーケンスで表示する
PAGE		○	TRACEのサブコマンド：出力制御
LINE		○	同上
OFF		○	TRACEコマンドをキャンセルする
ENTRY		○	プロトコルの修正実行起動
LP	△	○	検証結果のファイルへの出力を抑圧する
HELP	△	△	コマンドの内容案内
DISP		○	DISPLAYモードに変更する
UPDATE	○		UPDATEモードに変更する

注) ○：値の設定が可能
△：表示のみ

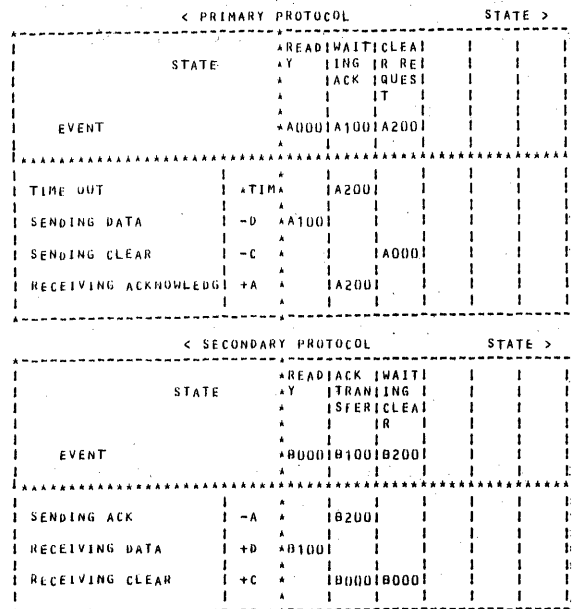


図 3.2 状態遷移表例

検証処理プログラムはFORTRANで記述され、その他はCOBOLで記述されている。各プログラムのステップ数を表3.2に示す。これらのプログラムは、TSS端末から「STARTプログラム名」を入力することにより起動される。入力する被検証プロトコルは、最初にファイルとして用意してもよいし、ENTRYコマンドによりPVS-Iと会話しながら入力してもよい。

表 3.2 プログラムステップ数

PVS:IN	PVS:LST	PVS:TAB	PVS:VAL
675	370	647	3300

3.2 機 能

(1) 画面表示

会話型シミュレーションにおける検証結果の表示は、PVS:VALがUPDATEモード時に「TRACE」コマンドによって起動され、一つの実行列毎に通信シーケンスと判定結果をディスプレイ上に出力する。図4.3にJISベーシック手順の検証における画面表示例を示す。

両端の数字(1, 2...)は遷移順序を示している。この内側に両プロセスの状態番号が表示され、初期状態番号と最終到達状態番号には二重カッコが付されている。画面の中央に送出信号が表示され、これには相手プロセスに到達する状況により次の4通りの表示がなされる。

- (a) 受理された信号
 $\rightarrow - (- CR) \rightarrow$
- (b) 受理規定がなかった信号
 $\rightarrow - (- CR) \rightarrow ?$
- (c) 伝送中に紛失した信号
 $\rightarrow - (- CR) \rightarrow //$
- (d) 受信バッファがオーバーフローとなって棄られた信号
 $- CR | - (BUF \ OVER) \rightarrow$

検証結果の画面表示が不要な場合には、検証項目別に画面表示の抑圧が可能である。

(2) 修正機能

検証した結果、プロトコルの一部に修正の必要が生じた場合にはこの機能を用いることができ、「ENTRY」コマンドにより起動される(図4.4(a))。

修正機能として、追加(ADD)と削除(DELETE)があり、修正の対象として次のいずれかを選択できる。

- (a) 入力したプロトコルに修正を加えプロトコルデータファイルを作成する。
- (b) (a)に加え、検証中のプロトコルテーブルも修正し、以後ユーザの指定により修正したプロトコルで検証を続行することも可能である。

また、ある信号の送信を追加する場合、両プロセスの修正する状態番号を指定し、一方のプロセスに追加する信号の送出を定義すると、相手プロセスの指定状態に自動的に上記送信信号の受信が定義され、ユーザにこの受信による遷移先状態番号の入力を指示するように、信号の授受が正しく行われる方向にユーザを導く機能を有している。

(3) SAVE機能

会話型のシミュレーションを行うと時間がかかるため、計算機の利用時間制限で中断しなければならないことがおこる。このため、その時点までの実行状態をセーブしたり、再開するリストア機能を有している。

4. PVS-Iの利用

4.1 PVS-Iによる検証法

通信プロトコルの設計における検証ツールとしては、設計の効率化を考えた場合、いかに早く設計の誤りを発見できるかが重要なポイントの1つである。特に設計の初期段階のように多くの誤りを含む場合には、いかに長大な検証を行ったとしても、同一の論理矛盾により発生する誤りが多大なリストとして出力されるだけで、ほとんど無意味となることもありうる。これに対して会話型の検証システムでは一度に多くの検証を行うことはできないが、検証結果毎に発見された誤りに対して修正を行いながらシミュレーションの実行を制御できる。このように会話型シミュレータは設計の初期段階における検証に有効である。また検証した遷移系列は、判定項目を付して通信シーケンスの形でディスプレイ上に表示されるため、ユーザにとってプロセスの状態遷移や信号の送受信動作が把握しやすい。

上述したように、ここで示す検証法はPVS-Iの支援を受けて、実際の検討は設計者等のユーザにまかされているため、部分的に自動化された検証法といえる。

＜検証手順＞

step 2 以後に発生するプロトコル修正などの作業で参照するため、入力したプロトコルの状態遷移表を出力する(②)。

step 3 プロトコルの到達状態番号や信号の紛失を含むか否かなど、検証の実行開始にともなう初期条件を設定する(③)。

step 4 PVS-1 に検証の実行コマンド (GO) を与えると、実行した判定結果を付して遷移系列を通信シーケンスでディスプレイに表示する (④, ⑤, ⑥)

step 5 誤り表示があると、ユーザは**step 2**で出力した状態遷移表とディスプレイ上の通信シーケンスを検討して修正を行う。検証が未終了の場合には**step 4**に戻る(⑦, ⑨)。

step 6 検証が終了すると，最終の状態遷移表を出力する。

以上の手順により誤りがない場合には、PVS-I で定義している範囲の論理構成について正当であるとす
る。

データの欠落や重複については検証項目に含まれていないため、所定の状態に到達すると正常と判断されるが、この時に表示される通信シーケンスをユーザが検討することによって、欠落や重複の検査が可能である。

4.2 適用例

以下に図 4.2 の状態遷移図で規定されるベーシック手順（絶対応答方式）に PVS-I を適用した例を示す。

ベーシック手順では、データリンク確立後通信状態となり、相手からの肯定応答 (ack) 受信後に次のデータの送信が可能となる。応答が返って来ないとタイムアウトとなり、応答督促信号 (eng²) を送る。これに対して、受信側は最前の応答を返す。

ここでは、このプロトコルにおいて、データリンク設定と終結における回復手順及び伝送誤りを省略した状態遷移に変更して、PVS-Iに適用した。

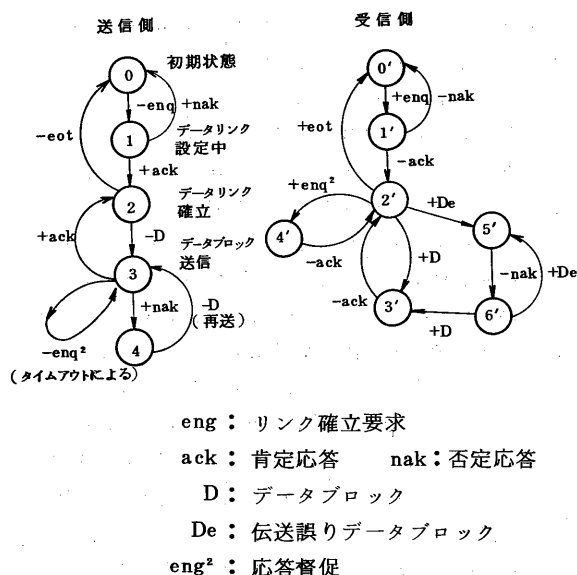
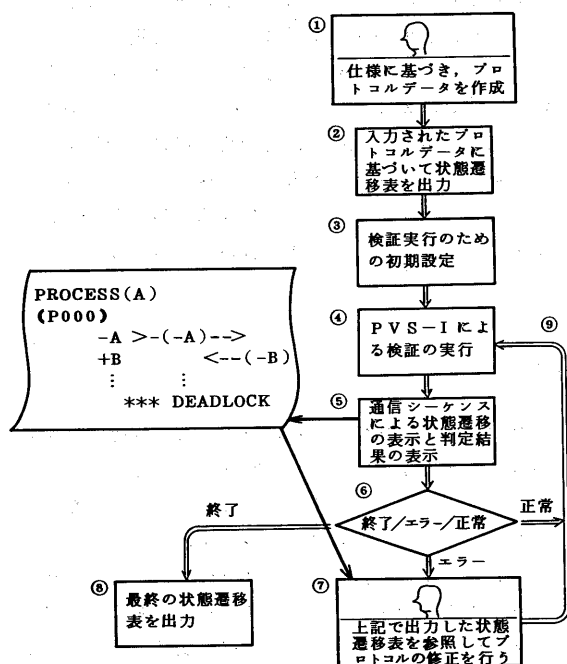


図 4.2 ベーシック手順（絶対応答方式）

```

*** PROTOCOL SIMULATION MODEL (*BASIC ZETTAI OTO MODE)*
PROCES (A)                                     PROCES (B)
(( A000 ))                                     (( B000 ))
1 (( A100 )) -ENQ      >--(-ENQ      )-->      +ENQ      (( B100 )) 2
4 (( A200 )) +ACK      <--(-ACK      )-<      -ACK      (( B200 )) 3
5 (( A300 )) -D        >--(-D        )-->      +D        (( B300 )) 4
6 (( A200 )) +ACK      <--(-ACK      )-<      -ACK      (( B200 )) 7
7 (( A000 )) -EOT      >--(-EOT      )-->      +EOT      (( B000 )) 10
(a) *** NORMAL END SEQUENCE ***

*** PROTOCOL SIMULATION MODEL (*BASIC ZETTAI OTO MODE)*
PROCES (A)                                     PROCES (B)
(( A000 ))                                     (( B000 ))
1 (( A100 )) -ENQ      >--(-ENQ      )-->      +ENQ      (( B100 )) 2
(( A100 ))                                     //--(-ACK      )-<      -ACK      (( B200 )) 3
(b) *** DEADLOCK SEQUENCE ***

*** PROTOCOL SIMULATION MODEL (*BASIC ZETTAI OTO MODE)*
PROCES (A)                                     PROCES (B)
(( A000 ))                                     (( B000 ))
1 (( A100 )) -ENQ      >--(-ENQ      )-->      +ENQ      (( B100 )) 2
4 (( A200 )) +ACK      <--(-ACK      )-<      -ACK      (( B200 )) 3
5 (( A300 )) -D        >--(-D        )--//
6 (( A300 )) -EQ2      >--(-EQ2      )-->      +EQ2      (( B400 )) 7
7 (( A300 )) +ACK      <--(-ACK      )-<      -ACK      (( B200 )) 8
9 (( A200 )) -EOT      >--(-EOT      )--//
10 (( A000 ))                                     (( B200 ))
(c) *** DEADLOCK SEQUENCE ***

```

図 4.3 ベーシック手順(絶対応答方式)の検証例

このプロトコルを検証した画面表示の一部を図 4.3 に示す。図 4.3(a)は正常な時の通信シーケンスである。図 4.3(b)はデータリンク設定中に ack 応答が紛失した場合、これに対する回復動作がないためデッドロックになっている。図 4.3(c)は通信終了信号 (eot) が紛失した場合にも回復動作がなくデッドロックとなることを示している。またデータブロックを送信後、応答がなくタイムアウトにより応答督促をした場合にデータの欠落が発生しうることがあることも示している。

次に通信終了信号が紛失した場合の回復手順として受信側からも終了信号 (DET) を送信できるように修正を加えた例とその結果を図 4.4(a), (b) に示す。

まず修正を行う両プロセスの状態番号を設定し (図 4.4(a) に示す例では A000 と B200), プロセス B の B200 状態から信号 DET を送信し B000 状態に遷移するように入力する。すると, PVS-I の機能により自動的に, プロセス A の A000 状態に信号 DET の受信が設定される。ユーザはこの信号受信による遷移先をして A000 を入力する。上記修正後の検証結果 (図 4.4(b)) を見ると, 通信終了信号 (eot) が紛失しても両プロセスは初期状態 (A000 と B000) に戻る。

なお, 実際の J I S ベーシック手順では, データリンク設定時や通信終了時における各信号の紛失などに対する回復手順が R 3 及び R 4 として規定している。また, データの欠落を防ぐ方式としては, J I S あるいは I S O 準拠のベーシック手順の一つに肯定応答として二つの信号 (ack と nak) を交互に送る交互応答方式があり, この交互応答方式では図 4.3(c) に示すような原因によるデータブロックの欠落が生じないことが, 本検証システムにより確認できる (図 4.5)。

5. 評価

PVS-I は会話形式で行われるため, 実際の検証時間はユーザのオペレーション時間に依存する。よってここでは, 図 4.2 に示す適用例に対する実行結果から PVS-I の能力を評価する。表 5.1 にこの検証に要した CPU 時間, 実行した遷移系列数, 検証結果の出力ファイル量を示す。この実行では検証結果はすべてディスクに出力し, ディスプレイには表示していない。また, 擬似的な到達終了処理を行わなかった場合の CPU 時間等を並記している。

```

?ENTRY
*** OLD PROTOCOL FILE UPDATE MODE START ***
*** WHAT DO YOU DO (MEMORY TABLE UPDATE (ENTER Y OR N) ***

?Y
* SELECT MODE OK OR NO (ENTER Y OR N) *

?Y
* WHAT DO YOU DO ( ADD OR DELETE ) (ENTER A OR D ) *

?A
* ENTER NAME OF ENTRY STATE IN PROCESS (A) *

?A000
* ENTER NAME OF ENTRY STATE IN PROCESS (B) *

?B200
* CONFIRM THE NAME OF ENTRY STATE (ENTER Y OR N) *

?Y
* NAME OF DEPARTURE STATE : ( A000 ) < A > *****
* ENTER NAME OF ARC

?N0
* NAME OF DEPARTURE STATE : ( A000 ) < A > *****
-ENQ(A100)
* CONTINUE INPUT OR NOT (ENTER ARC OR E) *

?R
* NAME OF DEPARTURE STATE : ( B200 ) < B > *****
+E02(B400) +D (B300) +EOT(B000) +DE (B500)
* CONTINUE INPUT OR NOT (ENTER ARC OR E) *

?-DET
* ENTER NAME OF ENTRY STATE

?B000
* THE FOLLOWING TRANSITION ORDER HAS BEEN ENTERED INTO PROTOCOL *
* NAME OF DEPARTURE STATE : ( A000 ) < A > *****
* ENTER NAME OF ARC : (+DET) *
* ENTER NAME OF ENTRY STATE

?A000
* CONTINUE INPUT OR NOT (ENTER ARC OR E) *

?END

```

(a) 修正例

```

*** PROTOCOL SIMULATION MODEL (*BASIC(ZETTAI OTQ MODE)*
PROCES (A)                                     PROCES (B)
(( A000 ))                                     (( B000 ))
1  -ENQ      >-( -ENQ      )-->                +ENQ      2
4  +ACK      <--(-ACK      )-<    -ACK      3
5  -D        >-( -D        )--//              (( B200 ))
6  -E02      >-( -E02      )-->                +E02      7
9  +ACK      <--(-ACK      )-<    -ACK      8
10 -EOT      >-( -EOT      )--//              (( B200 ))
12 +DET      <--(-DET      )-<    -DET      11
(( A000 ))                                     (( B000 ))
*** NORMAL END SEQUENCE ***

```

(b) その結果

図 4.4 修正例とその検証結果例

```

*** PROTOCOL SIMULATION MODEL (*BASIC(KOGO OTQ MODE)*
PROCES (A)                                     PROCES (B)
(( A000 ))                                     (( B000 ))
1  -ENQ      >-( -ENQ      )-->                +ENQ      2
4  +ACK      <--(-ACK      )-<    -ACK      3
5  -D        >-( -D        )--//              (( B200 ))
6  -E02      >-( -E02      )-->                +E02      7
9  +ACK      <--(-ACK      )-<    -ACK      8
10 -D        >-( -D        )-->                +D        11
13 +NAK      <--(-NAK      )-<    -NAK      12
14 -EOT      >-( -EOT      )-->                +EOT      15
(( A000 ))                                     (( B000 ))
*** NORMAL END SEQUENCE ***

```

図 4.5 ページック手順(交互応答方式)の検証例

表 5.1 実行時間と出力ファイル量

	本システム	疑似到達終了なし
CPU時間	19秒	3.75秒
実行した遷移系列数	55	129
出力ファイル量	104Kバイト	208Kバイト

注) 使用計算機 MELCOM COSMO 900Ⅱ
ファイルアクセス時間を含む
TSSで実行

擬似的な到達終了処理を設けることにより、遷移系列中の重複する遷移列の実行が省略でき、CPU時間、実行する遷移系列数及び検証結果の出力量が大幅に軽減できている。表 5.1 に示す例ではCPU時間、実行した遷移系列数及び出力ファイル量が約2分の1に減少している。

次に入力可能なプロトコルの状態数であるが、これは使用できる計算機のメモリ量に依存するが、本システムにおける入力可能なプロトコルの状態数の例を表 5.2 に示す。入力可能な最大状態数は、そのプロトコルの一つの状態において発生する最大イベント数とユーザが指定する受信バッファ数によって決まる。PVS-Iにより検証した既存プロトコルの規模を表 5.3 に示す。

表 5.2 PVS-I の入力可能なプロトコル規模

	状態当りの最大イベント数	受信バッファ数	入力可能最大状態数
例1	10	4	90
例2	30	4	52

表 5.3 既存プロトコルの規模 (一例)

プロトコル名	状態数	状態当りの最大イベント数
ベーシック手順	12	4
勧告X.25 CALL	10	7
勧告S.70 準拠トランスポート	17	8

問題点

使用した結果より、以下の問題点を指摘できる。

① PVS-I は判定、演算等の処理を行っていないため HDLC 等で規定されている処理動作 (シーケンス番号検査など) の検証を行う場合にモデル化しづらい。

② また同様の理由により、データ送受信の繰返し動作やリトライオーバー動作のモデル化が困難である。

③ 会話形式でプロトコルの修正を行う時には画面に状態遷移図が表示されると便利である。

④ 信号紛失発生は、現在すべての信号に対して有無を一括して指定する形になっているが、個々の信号に対して紛失の有無が指定できる方が便利である。

6. おわりに

本稿では会話型プロトコル検証システム PVS-I の概要とその利用について述べ、簡単な評価を行った。ユーザは PVS-I の会話機能を利用することにより、シミュレーションの実行に介入でき、設計の誤りを早期に発見及び修正ができる。また、検証実行した遷移系列が通信シーケンスの形で画面に出力されるため、ユーザにとってプロトコルのとりうる状態遷移が把握しやすい。

なお、プロトコルの処理動作の検証が可能な汎用プロトコル検証シミュレータを開発中である。

謝 辞

本稿を作成するにあたり有益なご討論を頂いた九州大学工学部情報工学科牛島教授と荒木助手に感謝いたします。

参考文献

- 河岡、友永、高橋、"プロトコル記述方法と検証方法について", ソフトウェア工学 8-5, 1978
- P. Zafiropulo, "Protocol Validation by Duologue Matrix Analysis", IEEE Tran. Comm. Vol. COM-26, No. 8, 1978
- 伊藤、"多者間プロトコル検証アルゴリズムの提案", 分散処理システム研資 3-8, 1979
- 蒲池他、"タイムベトリネットを用いた通信プロトコルの記述、検証法に関する考察", 信学技報 IN82-33
- 郷原、白鳥、野口、"通信プロトコルの効果的設計法", 信学技報 EC81-70
- 吉田、野村、田中、"データ欠落・重複を検査する通信プロトコル検証法", 信学論(B) 63-B, 1, 1980