Fortranプログラム　テスト/デバッグ支援のためのシステム：Forprexの使用について

牛島, 和夫
九州大学工学部情報工学科

河村, 豊実
九州大学工学部情報工学科

https://doi.org/10.15017/1472548

出版情報：九州大学大型計算機センター広報. 11 (4), pp.270-283, 1978-09-01. 九州大学大型計算機センター
バージョン：
権利関係：

Fortran プログラム テスト／デバッグ支援のためのシステム
ー Forprex の使用について ー

牛島和夫*, 河村豊実*

1. はじめに

プログラムを作る作業は，その順に(1)要求のまとめ，(2)仕様の決定，(3)設計，(4)コーディング，(5)テスト／デバッグと大きく分類される．そして完成したプログラムに対して(6)維持および管理が待っている．

大学の計算センターで行われる計算の大部分は数値計算で占められ，規模も比較的小さく，一人で全ての仕事が行える程度のものが多い．プログラムをする人も，計算機の専門家ではなく，問題を持った当人であり，解くべき問題は明確に認識されている（はずである）ので，プログラミングとは，主として上記(4)と(5)と認識されるのが特徴ではないだろうか．その中で，テスト／デバッグ相はどのように計画され遂行されているのであろうか．プログラムが仕様の通り動くことを確認する作業がテストである．不首尾が発見されれば，その原因を究明して除去しなければならない．これがデバッグである．

計算機を専門としないユーザにとって，このテスト／デバッグ相を遂行する一般的な手掛りは，言語マニュアルやセンターの発行する利用の手引きなどが主なもので，極めて少ないのが現状である．Fortran処理系には，トレースバック機能や配列の添字の限界をチェックする機能などを有しているものも多く，これらを適切に使用することによってエラーの発生点を確定することができる．またデバッグ文やダンプルーチンが備えられているが，これらはどのように利用されているのか利用統計もないので実態は不明である．プログラムの要所にあらかじめ出力文を挿入しておき途中結果を監視するという方法が，比較的よく用いられているようである．

テスト／デバッグ相における問題点を列挙してみよう：

(1) デバッグやテストのための手当ては，コーディングが一応完了した後，あるいは事故が発生した後など，後追いになりやすい．

(2) デバッグ用の出力は過大になりやすく，その中から必要な情報を摘出するのはかなり面倒である．例えば，デバッグ用の出力文が多重ループの最も内側に仕掛けてある場合や，ダンプルーチンの出力を考えてみよ．問題の点に到達する前に出力過多でジョブ打切りにされることすらある．

(3) デバッグ用出力とソーステキストとの対応づけが不便である．ソーステキスト中のどの出力文によって出力され，その経過がどうかを知るための解析作業は容易ではない．

(4) テスト／デバッグ終了後不要になったテスト／テバッグ用の文を除去する際にトラブルが発生することがしばしばある．

(5) 一般ユーザーの使用に都合よく設計されたテスト指向型の道具が極めて少ない．

これらの点を配慮し，文献〔1，2，3〕などを参考にして，九州大学情報工学科のFACOM

＊九州大学工学部情報工学科

230−45S OSIIのもとにテスト／デバッグ支援用の簡単なシステムForprex(Fortran program execution monitor)を，昭和50年度，51年度の卒業研究で試作してもらい，[4,5] 51年4月から学科内で使用してきた．[6] 使用しながら，しばしば改訂を行って現在に至っている．このシステムを，FACOM M−190 OSIV／F4のもとに移し換える作業(大型計算機センターの開発課題)が一段落したので，ここに報告する次第である．

2. システムの機能

Forprexは次のような機能を持っている．

(1) プログラム中の任意の点で指定された条件(拡張された論理式で示す)を判定する．そして，それが満たされない場合(違反＝Violation と称する)には，指定された処理(違反時処理)を行う．

(2) プログラム中の任意の点で指定された変数または配列要素に入っている値を表示する．

(3) プログラム中の任意の点を，指定された回数通過したところで実行を停止する．

(4) プログラム中の任意の点で指定された変数または配列要素がとる値の範囲を計測する．

(5) 各実行文の実行回数，さらには論理IF文の論理式の値が真になる回数を計数する．

(6) 上記の結果を集計し，ソースプログラムテキストと編集して出力する．

(7) 異常終了時にもそれまでに得られた結果を(6)にならって出力する．

上記(1)〜(4)の各機能を使うには，特別な注釈行(C＆文と称する)をプログラム中の要所に置くことにより，プログラム中の計測点を指定し，さらに，条件，計測の対象となる変数などを指定する．

C＆文は，第1桁をC，第2桁を＆とし，第7桁から第72桁に上記(1)〜(4)の指定をする文を書いたものである．一つの文が1行中に入りきらない場合にはC＆継続行を用いる．C＆継続行は，空白でも0でもない文字をC＆行の第6桁に記入した行をいう．

C＆文で上記(1)〜(4)を，次の4種類の文によって指定する．

(1) ASSERT文

(2) DISPLAY文

(3) HALT文

(4) RANGE文

まず，例題を通して，これら各文の使用法を概観しよう．

3. 例 題

図1は，2階の線形常微分方程式の境界値問題を差分近似により三項方程式として解くプログラムを，Forprexにかけたものである．ここで解いている問題は，

$$\frac{d^2x}{dt^2} + x = 0$$

```
FACOM OSIV/F4    FORPREX          V-03 L-01      DATE 78-10-20  TIME 14:43

ISN          FORTRAN  STATEMENT                                    EXECUTIONS        TRUE

      C
  1           DIMENSION A(1024,3),UL(1024,3),B(1024),X(1024)
  2           DIMENSION R(1024),DX(1024)
  3           M=1024
  4           READ(5,501)NFIRST,NLAST                                       1
  5       501 FORMAT(2I5)
  6           READ(5,502) XO,XN,YO,YN                                       1
  7       502 FORMAT(4F10.0)
  8           DO 10 J=NFIRST,NLAST                                          1
  9             N=2**J-1                                                    1
 10             N1=2**(J-2)                                                 1
      C
 11           CALL PROBLM(N,M,A,B,XO,XN,YO,YN)                              1
 12           CALL DECOMP(N,M,A,UL)                                         1
 13           CALL SOLVE(N,M,UL,B,X)                                        1
 14           CALL IMPRUV(N,M,A,UL,B,X,R,DX)                                1
 15           CALL OUTPUT(N+1,X,N1,N,N1)                                    1
      C
 16        10 CONTINUE                                                      1
 17           STOP                                                          1
 18           END


ISN          FORTRAN  STATEMENT                                    EXECUTIONS        TRUE

 19           SUBROUTINE DECOMP(NN,M,A,UL)                                  1
 20           DIMENSION A(M,3),UL(M,3)
 21           N=NN                                                          1
 22           UL(1,1)=0.                                                    1   ( 1 )
 23           UL(1,2)=A(1,2)                                                1
 24           DO 1 I=1,N                                                    1
 25         1   UL(I,3)=A(I,3)                                           1023
 26           DO 2 I=2,N                                                    1
 27             EM=-A(I,1)/UL(I-1,2)                                     1022
 28             UL(I,1)=-EM                                              1022
 29             UL(I,2)=A(I,2)+EM*A(I-1,3)                               1022
      C#                          RANGE ( EM,UL(I,2) )                    1022

                                          <EM    > FIRST=  0.5000002
                                                   LAST =  0.9995500
                                                   MAX  =  0.9995500
                    ( 2 )                                          (PASS=     1022)
                                                   MIN  =  0.5000002
                                                                   (PASS=        1)

                                          <UL    > FIRST= -1.499998
                                                   LAST = -1.000448
                                                   MAX  = -1.000448
                                                                   (PASS=     1022)
                                                   MIN  = -1.499998
                                                                   (PASS=        1)

 30         2 CONTINUE                                                   1022
 31           RETURN                                                        1
 32           END


ISN          FORTRAN  STATEMENT                                    EXECUTIONS        TRUE

 33           SUBROUTINE SOLVE(NN,M,UL,B,X)                                 6
 34           DIMENSION UL(M,3),B(M),X(M)
 35           N=NN                                                          6
 36           X(1)=B(1)                                                     6
 37           DO 1 I=2,N                                                    6
 38             X(I)=B(I)-UL(I,1)*X(I-1)                                 6132
 39         1 CONTINUE                                                   6132
 40           X(N)=X(N)/UL(N,2)                                             6
 41           DO 2 IBACK=2,N                                                6
 42             I=N+1-IBACK                                              6132
 43             X(I)=(X(I)-UL(I,3)*X(I+1))/UL(I,2)                       6132
 44         2 CONTINUE                                                   6132
 45           RETURN                                                        6
 46           END


ISN          FORTRAN  STATEMENT                                    EXECUTIONS        TRUE

 47           SUBROUTINE IMPRUV(NN,M,A,UL,B,X,R,DX)                         1
 48           DIMENSION A(M,3),UL(M,3),B(M),X(M)
 49           DIMENSION R(M),DX(M)
 50           DATA EPS,ITMAX/1.E-6,20/
      C
 51           N=NN                                                          1
 52           XNORM=0.                                                      1
 53           DO 1 I=1,N                                                    1
 54             XNORM=AMAX1(XNORM,ABS(X(I)))                             1023
 55         1 CONTINUE                                                   1023
 56           ITER=0                                                        1
 57           IF(XNORM .LE. 0.0) GO TO 10                                   1          0 (  0.0%)
 58           DO 9 ITER=1,ITMAX                                             1
 59             DO 5 I=1,N                                                  5
 60               SUM=0.D0                                               5115
 61               DO 4 J=1,3                                             5115
 62                 JJ=I+J-2                                            15345
 63                 IF(JJ .EQ. 0)  GO TO 4                              15345          5 (  0.0%)
 64                 IF(JJ .EQ. N+1)GO TO 4                              15340          5 (  0.0%)
      C#                              ASSERT (1.LE.JJ .AND. JJ.LE.N)     15335
      C#  *                            ELSE DISPLAY (I,J,JJ)  HALT 1

                    ( 3 )                  *    VIOLATION COUNT          0 (  0.0%)

 65                 SUM=SUM+DBLE(A(I,J))*X(JJ)                          15335
 66           4     CONTINUE                                            15345
 67                 R(I)=B(I)-SUM                                        5115
 68           5   CONTINUE                                               5115
      C
```

```
69              CALL SOLVE(N,M,UL,R,DX)                                          5
      C
70              DXNORM=0.0                                                       5
71              DO 6 I=1,N                                                       5
72                T=X(I)                                                      5115
73                X(I)=X(I)+DX(I)                                             5115
      C#                                ASSERT(X(I).LT.T)                       5115
      C#    *                             ELSE DISPLAY 1,100,40 (X(I),T)
```

```
*    VIOLATION COUNT    =    1667.  ( 32.6%)

* VIOLATION(          1)   (PASS=       1024)
     X     =  0.1176212E-02;        T     =  0.1171712E-02;

* VIOLATION(         41)   (PASS=       1064)
     X     =  0.4821051E-01;        T     =  0.4802636E-01;

* VIOLATION(         81)   (PASS=       1104)
     X     =  0.9515107E-01;        T     =  0.9478450E-01;
```

(4)

```
74              DXNORM=AMAX1(DXNORM,ABS(X(I)-T))                             5115
75          6   CONTINUE                                                     5115
      C#                              DISPLAY 10 (DXNORM)                      5
```

```
* PASS     (          1)
     DXNORM=  0.3153539E-01;

* PASS     (          2)
     DXNORM=  0.1546085E-02;

* PASS     (          3)
     DXNORM=  0.8249283E-04;

* PASS     (          4)
     DXNORM=  0.4768372E-05;

* PASS     (          5)
     DXNORM=  0.7748004E-06;
```

(5)

```
76              IF(DXNORM .LE. EPS*XNORM) GO TO 10                           5          1 ( 20.0%)
77          9 CONTINUE                                                       4
78              ITER=ITMAX+1                                                 0
79         10 CONTINUE                                                       1
80              RETURN                                                       1
81              END
```

```
ISN             FORTRAN  STATEMENT                                    EXECUTIONS      TRUE

 82             SUBROUTINE OUTPUT(KIZAMI,A,M1,M2,M3)                          1
 83             DIMENSION A(2)
 84             WRITE(6,666)KIZAMI,(A(I),I=M1,M2,M3)                          1
 85         666 FORMAT( 5X,2HN=,I4/(8E16.7))
 86             RETURN                                                        1
 87             END
```

```
ISN             FORTRAN  STATEMENT                                    EXECUTIONS      TRUE

 88             SUBROUTINE PROBLM(NN,M,A,B,XO,XN,YO,YN)                       1
 89             DIMENSION A(M,3),B(M)
 90             N=NN                                                          1
 91             FN=N+1                                                        1
 92             H=(XN-XO)/FN                                                  1
 93             DO 1 I=1,N                                                    1
 94               A(I,1)=1.                                                1023
 95               A(I,2)=H*H-2.                                            1023
 96               A(I,3)=1.                                                1023
 97               B(I)=0.                                                  1022
 98           1 CONTINUE                                                   1023
 99             B(1)=B(1)-YO*A(1,1)                                           1
100             B(N)=B(N)-YN*A(N,3)                                           1
101             RETURN                                                        1
102             END
```

```
ROUTINE         EXECUTIONS      PERCENT

MAIN                    12          0.0
DECOMP                4096          3.4
SOLVE                18428         15.4
IMPROV               93100         77.8
OUTPUT                   3          0.0
PROBLM                4100          3.4

** TOTAL **         119771
```

(6)

図 1.　Forprex出力例

$$\begin{cases} x(0)=0 \\ x(1)=1 \end{cases}$$

である．このプログラムは，文献〔7〕に示したプログラムと同じものである．主プログラムは，

PROBLM ……問題の設定
DECOMP ……三角分解
SOLVE ……第1近似解
IMPRUV ……反復改良
OUTPUT ……結果の出力

を呼び出しており，IMPRUVがさらにSOLVEを呼び出している．

図1は，いわゆるFortranのソースプログラムリストとほとんど同じ形式をしているが，ところどころ違った形式をしている．それを順に説明しよう．以下，番号は図中の番号と同じ．

(1) 各プログラム単位の右端に，EXECUTIONS欄があり，これは各実行文の実行回数を示す．さらにその右側にあるTRUE欄は，論理IF文が真になったときの回数を示し，これはFordap[3,11]にかけて得られるものと同じである．

(2) SUBROUTINE DECOMP中の特別な注釈行．

```
C&          RANGE (EM,UL(I,2))
```

により，この点を通過する直前に，変数EMおよびUL(I,2)に得られている値の初期値，最終値，最大値，最小値をサンプルして表示することを指示する．EMの最大値は0.9995500で，1022回目の通過の際に実現したことがわかる．EXECUTIONS欄の値から，このDOループの本体は1022回繰返されており，最大値は最終値でもある．

一方UL(I,2)は添字Iの値が2からNまで変化するので，一つの配列要素の最大値や最小値を示しているのではないことに要注意．すなわち，初期値はUL(2,2)に代入された値であり，最大値はUL(1023,2)に代入された値ということになる(最大値の実現する1022回目の通過時にI=1023となっているから)．

(3)

```
C&      ASSERT(1. LE. JJ. AND. JJ. LE. N)
C&     +   ELSE DISPLAY(I,J,JJ) HALT 1
```

この点の直前で，$1 \leqq JJ \leqq N$が成立していることを主張(Assert)している．ELSE以下は，この主張に違反したときの処置を指示するもの．

```
DISPLAY (I,J,JJ)
```

により，違反発生時にI,J,JJの値を表示することを指定し，

```
HALT 1
```

により，違反が1回生じたら実行を停止することを指定している．

ここでは15335回ASSERT文が検査されたが違反は1回もなかったことを，

```
*    VIOLATION COUNT      0
```

によって示している．

(4)

```
C&        ASSERT(X(I). LT. T)
C&     +   ELSE DISPLAY 1,100,40 (X(I),T)
```

この文は，ASSERT文の機能を示すために挿入したもので，必ずしも模範的な使用法ではない．プログラマは，この点でX(I)<Tが必ず成り立つと誤認した．従って，5115回の通過のうち1667回違反が発生している．DISPLAY指定の1，100，40は，1667回の違反のうち，1回目と41回目と81回目に限りX(I)とTの値を表示するように指定している．第1回目の違反は，1024回目，第41回の違反は1064回目の通過時に生じていることがわかる．このように選択的な出力を指定できるのが，このシステムの特徴である．

(5)

```
C&        DISPLAY 10 (DXNORM)
```

前記(3)ASSERT文のELSE以下を独立させたもの．この点を通過するときに，DXNORMの値を表示することを指示している．10は最初の10回通過するときだけ出力し，11回目以降は出力を抑制する．

```
DISPLAY 1, 10, 1(DXNORM)
```

と書いたものと同じである．なお，10を省略すると30と書いたものと見なすことにしている．

このプログラムは，DXNORMの収束を意図しており，ここに出力された結果から，確かに0.31×10^{-1}，0.15×10^{-2}，0.82×10^{-4}，…と収束していく様子を確認できる．

(6) これは，ここで使用されている6個のプログラム単位のEXECUTIONS欄とTRUE欄との値の総和をそれぞれ加算したものとその百分率を示す．ただし，CONTINUE文の値は加算からはずしている．この値から，プログラムの実行が集中しているプログラム単位の見当をつけることができる．実際問題として，μsec単位で実行できる．

```
I = I + 1
```

のような文も，msec単位の時間のかかる入出力文も1回の実行を一様に1と数えているので，ここで示す値は，実行時間の比ではない．あくまでも，実行の集中を知る一つの手掛りに過ぎないことに注意．

図2に，SUBROUTINE IMPRUVのソーステキストを，Fortflow[8]により流れ図付きで示す．図1と図2を比較し，ソーステキストとテスト／デバッグ出力との対応づけが容易なことを見ていただきたい．

```
                          1        SUBROUTINE IMPRUV(NN,M,A,UL,B,X,R,DX)
                          2        DIMENSION A(M,3),UL(M,3),B(M),X(M)
                          3        DIMENSION R(M),DX(M)
                          4        DATA EPS,ITMAX/1.E-6,20/
                            C
                   |      5        N=NN
                   |      6        XNORM=0.
                   |      7        DO 1 I=1,N
           0)----->|
           :       |      8          XNORM=AMAX1(XNORM,ABS(X(I)))
           0<------|      9      1 CONTINUE
                   |     10        ITER=0
 4<----------(*-(->)     11        IF(XNORM .LE. 0.0) GO TO 10
 |                 |     12        DO 9 ITER=1,ITMAX
 | 3)------------->|
 | :               |     13          DO 5 I=1,N
 | : 2)----------->|
 | : :             |     14            SUM=0.D0
 | : :             |     15            DO 4 J=1,3
 | : : 1)------->  |
 | : : :           |     16              JJ=I+J-2
 | : : : 0<-(*-(->)    17              IF(JJ .EQ. 0)  GO TO 4
 | : : : 0<-(*-(->)    18              IF(JJ .EQ. N+1)GO TO 4
 | : : : : |       |        C&                                ASSERT (1.LE.JJ .AND. JJ.LE.N)
 | : : : : |       |        C&    +                            ELSE DISPLAY (I,J,JJ)  HALT 1
 | : : : : |       |     19              SUM=SUM+DBLE(A(I,J))*X(JJ)
 | : : : 0)----->|   4
 | : : 1<--------|     20        4     CONTINUE
 | : :           |     21              R(I)=B(I)-SUM
 | : 2<----------|     22        5   CONTINUE
 | :             |        C
 | :          (-)     23          CALL SOLVE(N,M,UL,R,DX)
 | :             |        C
 | :             |     24          DXNORM=0.0
 | :             |     25          DO 6 I=1,N
 | :       0)----->|
 | :       :     |     26            T=X(I)
 | :       :     |     27            X(I)=X(I)+DX(I)
 | :       :     |        C&                                ASSERT(X(I).LT.T)
 | :       :     |        C&    +                            ELSE DISPLAY 1,100,40 (X(I),T)
 | :       :     |     28            DXNORM=AMAX1(DXNORM,ABS(X(I)-T))
 | :       0<----|     29        6   CONTINUE
 | :             |        C&                                 DISPLAY 10 (DXNORM)
 4<+---------(*-(->)     30          IF(DXNORM .LE. EPS*XNORM) GO TO 10
 | 3<----------|     31        9 CONTINUE
 |             |     32          ITER=ITMAX+1
 4)------------>|  10
               |     33       10 CONTINUE
              *-R    34          RETURN
                     35          END
```

図2. SUBROUTINE IMPRUV

4. C&文の仕様

C&文の仕様を以下にまとめる．以下で〔 〕内は省略可，｛ ｝内は選択を示し，｛ ｝内の下線は，標準値を示す．

4.1 ASSERT文

C& ASSERT ＜拡張された論理式＞

〔ELSE〕〔＜DISPLAY指定＞〕〔＜HALT指定＞〕

＜拡張された論理式＞は｛$\frac{+}{*}$｝(＜添数付論理式＞)とする．ただし＜添数付論理式＞は，＜Fortran論理式＞か，(＜添数付論理式＞，＜DO型仕様＞)とする．なお＜DO型仕様＞

とは，$i = m_1, m_2, m_3$ か，$i = m_1, m_2$ で i は制御変数で3重まで付加えることを許す．式の前に付けた＋（プラス）は，各DO型仕様の制御変数に対するFortran論理式の論理和，＊（スター）は論理積を表す．

例　＊((X(I).GE.X(I+1),I=1,N-1))により，X(1)≦X(2)≦…≦X(N)を表す．

プログラムの実行の制御がC&文の直前まで到達すると，その点での＜拡張された論理式＞を評価し，その値が真であればそのまま次の文に制御が移る．偽ならば違反が検出されて，ELSE以下にある＜DISPLAY指定＞および（または）＜HALT指定＞の処理を行って，次の文に実行の制御を移す．

＜DISPLAY指定＞は次の通り．

DISPLAY 〔n_1〔, n_2〔, n_3〕〕〕　（v_1, v_2, …, vm）

ここで，n_1, n_2, n_3 は正整定数（$0 < n_1 \leqq n_2 \leqq 2^{31}-1$，$0 < n_3 \leqq 2^{31}-1$）である．$v_1$, v_2, …, vmは，変数名または配列要素名の並びである（$m \leqq 20$）．

この指定により，n_1 回目の違反から n_2 回目の違反まで，n_3 回ごとに，v_1 … vmの値を表示することを指示する．n_3 を省略すると $n_3 = 1$ とみなす．n_2 を省略すると，1，n_1，1と解釈する．全部省略した場合は，1，30，1と解釈する．

変数または配列要素の値は各型に対応した標準の書式で出力される．なお，▼変数名と書くと指定された変数の値を文字とみなして，文字出力を行う．また，￥変数名と書くと，データを16進形式で出力する．なお，▼変数名（￥変数名）は▼配列要素名（￥配列要素名）としてもよい．

＜HALT指定＞は次の通り．

HALT　〔n〕

ここで，nは正整定数（$0 < n \leqq 2^{31}-1$）とする．省略されるとn＝1と解釈する．

この指定により，違反がn回起きたら，プログラムの実行を終了することを指示する．

＜DISPLAY指定＞と＜HALT指定＞の記述の順序，有無は任意である．

4.2　DISPLAY文

DISPLAY文は次の書式とする．

C&　　＜DISPLAY指定＞

これは，ASSERT文のDISPLAY処理と同等である．ただし，変数または配列要素の値の出力制御は，違反回数でではなく通過回数で行う．

4.3　HALT文

HALT文は次の書式とする．

C&　　HALT　〔n〕

これは，ASSERT文のHALT処理と同等である．ただし，違反回数ではなく，通過回数がn回

になったら，プログラムの実行を終了することを指示する．なお，ASSERT文のELSE以下と同様に，DISPLAY文とHALT文をあわせて1行に書くこともできる．

4.4 RANGE文

C& RANGE (v_1, v_2, ……, vm)

ここで，v_1, …, vmは，20個以下の変数名または配列要素名の並びである．(m ≦20)

この文の直前の文の実行が終了した時点における，v_1, …, vmの初期値，最大値，最小値，最終値を，その値を実現した通過回数とともに表示する．ただし，複素数型，論理型のデータに対しては初期値と最終値のみ表示する．

5. デバッグ文との比較

Forprexの特徴を理解するには，Forprexの機能を別の方法で実現させることを考えてみるとよい．FACOM OSIV/F4には二つのFortranコンパイラがあり，Fortran GEでは，デバッグ文が使用できる．副プログラムの引用をトレースするSUBTRACE指定や，範囲外の添字式をチェックするSUBCHK指定は，＊PROCESSによっても指定できるので，ここでは，TRACE指定とDISPLAY指定に限って議論しよう．

TRACE ON文が実行されてから，TRACE OFF文が実行されるまでの間，DEBUG文でTRACEを指定されたプログラム中の文番号に行き当たるたびに，

TRACE 文の番号

を出力する．従って，うかつに指定すると出力がたちまち過大になり，結果の解析は大変である．それよりも，Forprex(あるいはFordap)によって得られるEXECUTIONS欄の実行回数の方がずっと理解が容易である．

図1(4)のASSERT文をDEBUG文を用いて書き換えると，例えば次のようになるだろう．

選択的出力を指定しているため，かなり面倒である．

```
      DEBUG
      AT 50
      IF(X(I).LT.T) GO TO 55
      ICOUNT=ICOUNT+1
      IF(ICOUNT.GT.100) GO TO 55
      IF(NCOUNT.NE.ICOUNT) GO TO 55
      XI=X(I)
      DISPLAY XI,T,ICOUNT
      NCOUNT=NCOUNT+40
   55 CONTINUE
```

ただし，ASSERT文のあった位置に，

```
   50 CONTINUE
```

を挿入しておかなければならない．また，ICOUNTと NCOUNTは，プログラムの実行に先立っ

て，0および1とに初期設定しておかねばならぬ．サブルーチンから出て再び入ってくる可能性を考えて，これらの変数は，共通領域に置いておく必要がある．DISPLAY文で，ICOUNTの値も出力するよう指定しているのは，違反発生回数を知るためである．この点の通過回数や違反発生の回数とその比率などを知るためには，プログラムの終了（STOP文の実行）直前にそのような処理をするように，プログラムしておかねばならない．しかもそれらは，全く離れ離れに出力される．これによってForprexが，過大になりがちなデバッグ出力を抑制し選択的に出力させ，しかもその結果を見やすく編集することに意を配っていることを，理解していただけるであろう．図1(5) DISPLAY文のデバッグ文による書換えは演習問題として試みていただきたい．

デバッグ文には，場合によって，変数の値を強制的に変更することが例えばつぎのようにして実現できる：

```
DEBUG
AT 10
IF(I.LE.0) I=1
END
```

Iは本来正でなければならないにもかゝわらず，何らかの原因で負になってしまう．その虫のために，プログラムが先に進まない．とにかく先に進めたいといったような状況を想定するとよいだろう．Forprexは execution monitor が主目的であって，計算の流れを左右するような指定はできないようになっている．上のようなデバッグ文を使っていると，デバッグ終了後にそれを取はずした時に，もとのプログラムがやはりうまく動かないといったことが起りうる．

デバッグ文を使ってRANGE文を書換えると，かなり長いプログラムができあがる．Forprexの処理手順を説明することとほとんど変らないことになりかねないので省略する．

以上のように，ASSERT文，DISPLAY文，RANGE文等それぞれと同等の機能を自分でプログラムするのはかなり面倒であり，まして計算終了後の編集は簡単には実現できない．

6. おわりに

このシステムの効用や問題点の主なものをまとめる．

(1) C&文による出力結果がC&文の直後に編集されて表示されるので，ソーステキストとの対応が大変よい．DISPLAY指定による選択的出力は便利である．

(2) テストのための手当てはどうしても後追いになりやすい．プログラムの設計時に，ASSERT文を用意させる動機づけになり得よう．後追いとなった場合でも，拡張された論理式によって，本来の実行文に別の見方をすることにより，誤りの発見につながることがある．

(3) C&文は用済後もそのまま注釈として残しておいて構わない．実行文を変更したときは，C&文の内容も変更しておかないと，Forprexシステムにかけたとき違反が検出されることがあるので実行文とC&文の対応をチェックできる．

(4) 拡張された論理式の表現がFortran文法の制約を受ける．この論理式の中に関数呼出しを含んでいると副作用のおそれがある．

(5) 特にASSERT文の使用には，それを効果的に使用する技術が要求される．

(6) C&文を含まないソーステキストをForprexにかけると，実行回数（図1(1)）とそのまとめ（図1(7)）が出力される．これはFordapによる実行解析から時間の計測を除いたものと同等である．従って，効果的なテストを行うために，まずC&文なしでForprexにかけて実行回数を知った上で，要所を見極めて，C&文を用意するといった方法をとることもできるだろう．FordapはC& 文をそのまま注釈行と見なすから，C&文を含んでいる場合は，Fordapとの交互使用も有効である．

(7) Fortran GEでは，TSSによるインタラクティブデバッグが可能である．十分な計画なしにインタラクティブデバッグを実行しても期待した効果をあげにくい．バッチ（またはリモートバッチ）によるForprex等を利用し，ソースプログラムとの対応付けの容易さ，選択的出力の効果を発揮させ，全体を十分把握した上で，インタラクティブデバッグ機能と併用することを計画したらよい．

冒頭に述べたように，このシステムは，九大情報工学科で約2年間内部的に使用してきたものである．大型計算機センターの利用者に使用していただくことによって，いろいろ問題点が発見されることと思う．御批判をいただいて，さらに改善の糧にしたいと考えている．御協力をお願いしたい．

付録として，Forprexジョブ制御文のそろえ方，システムの制限，エラーメッセージをまとめた．また，このシステムの処理の概要を知ることは，このシステムの利用をより円滑にすると考えられるので処理の流れ図を付加えておく（図3参照）．

付　録

1. システムの制限

(1) 入力する原始プログラムは，Fortran HEコンパイラにかけて，文法違反のないことが確認されているものとする．

(2) SYSXXX（XXXは3桁の数字）の形の英字名は使用できない．

(3) 一つのプログラム単位内の変数名，配列名の総数は250個以下でなければならない．

(4) 65000から65535までの文番号は，使用できない．

(5) データセット識別番号，96，97，98，99は使用できない．

(6) C&文の総数はプログラム全体で50個以下とする．

(7) DISPLAY文，RANGE文で一度に指定できる変数名，配列要素名の数は，20個以下とする．

(8) RANGE文で指定する変数名，配列要素名の総数は，プログラム全体で100個以下とする．

2. エラーメッセージ

C& 文の解析中に誤りを検出した場合，その誤りに関するメッセージを出力する．各誤りに対する処置は，本文で起きた場合はその文を注釈行とみなす．また，ASSERT文の二つの指定の中で

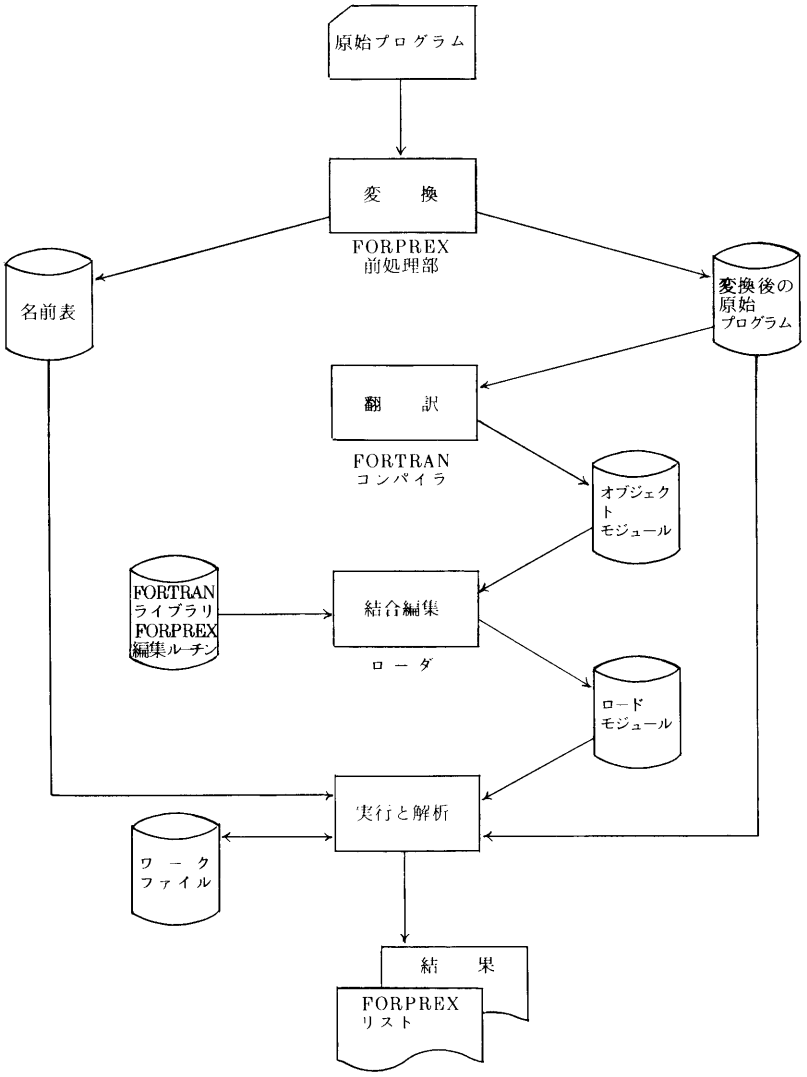

図 3.　Forprexの処理の流れ

起きた場合は，その指定および以降の指定を無視する．

メッセージの出力形式

ERROR=code：＜メッセージ本文＞

code：　エラーの内容

100：　英字名がない．または，未定義の変数名，配列名である．

200：　定数がない．

300：　区切りがない．または，未定義の区切り（語）である．

302：　'='（等号）がない．

307 : '(' （左かっこ）がない.

308 : ')' （右かっこ）がない.

309 : ',' （コンマ）がない.

エラーが検出された場合，次のような情報を，エラーのあった行の次に出力する.

----------->

ここで，矢印の先端の直前で誤りを検出し，先端は次に読む文字の位置を示す.

3. カタログドプロシジャ

形式：

プロシジャ名	記号パラメータ	プロシジャ・ステップ名
FORPREX	[, SYSOUT={A / K / R}] [, PRVLIB＝データセット名]	PREX FORT GO

機能： Forprex 前処理部で処理した原始プログラムを Fortran HEコンパイラで翻訳し，ローダによる結合・編集後，実行・解析を行う.

記号パラメータ：

SYSOUT：出力クラスを指定する.

A（LP），K（カナ付LP），R（端末出力）

PRVLIB：組み込みたい私用ライブラリのデータセット名を指定する.

関連するDD名：

PREX.SYSIN（必須） 原始プログラム用

GO.SYSIN 実行時の入力データ用

Fortran HEコンパイラオプションの標準値：

NAME(MAIN),OPTIMIZE(2),OBJECT,NODECK,GOSTMT,NORENT,
NONAME,NODPROF,LINECOUNT(60),FLAG(I),NOSOURCE,NOXREF,
NOLIST,NOMAP,NOFORMAT,NODUMP,AUTODBL(NONE),NOPR,EBCDIC,
NOJISF,NOALC,NOASTER,NOBYNAME,NOSEQ,SIZE(MAX),NOINLOG2,
NOTERM,NONUM,NOLIL,READ(5),PRINT(6)

ローダオプションの標準値：

NOMAP,NORES,CALL,NOLET,NAME=**GO,PRINT,NOALIAS,NOTERM,
LINECOUNT=60

＊コンパイラオプションを指定する時は，必ずNOSOURCEも指定すること.

4. 使用例

(1) 原始プログラム，データともカード入力の例

```
//     EXEC   FORPREX
//PREX.SYSIN DD *
```

原始プログラムカード

```
//GO.SYSIN       DD *
```

データカード

```
/*
```

(2) データセットから原始プログラムを入力する例.

```
//     EXEC     FORPREX
//PREX.SYSIN    DD DSN=F1978.TEST.FORT,DISP=SHR
//GO.SYSIN      DD *
```

データカード

```
/*
```

参考文献

1. Stucki, L. G. and Foshee, C. L. *New assertion Concepts for Self-Metric Software Validation*, 2nd Intnl. Conf. on Reliable Software, 1975.
2. Satterthwaite, E. *Debugging Tools for High Level Languages*, Software-Pract. & Exper., Vol. 2 pp. 197-217, 1972.
3. 藤村，牛島　プログラムの実行解析システムの作成と使用について，九大工学集報 Vol. 48, pp. 95-101, 1975.
4. 牛島，河村，見戸　デバッグ／テスト支援システムの作成，S51電気四学会九州支部連大，1976.
5. 牛島，河村，武富　デバッグ／テスト支援システムForprexの機能拡張について，S52電気四学会九州支部連大，1977.
6. FORPREX使用説明書　九大情報工学科ソフトウエア研究室，1976.
7. 牛島　数値計算プログラムの動作解析とその段階的高速化，数値計算における誤差（一松・戸川編）pp. 173-183，共立出版，1975.
8. 牛島，高比良　流れ図付きソースプログラム作表システム，本号 pp. 284-288，1978.
9. 富士通　OSIV FORTRAN文法書，64SP-3030-3.
10. 富士通　OSIV/F4 FORTRANインタラクティブデバッグ使用手引書，64SP-3200-1.
11. 牛島，藤村　M-190 OSIV/F4 FORDAPシステム，九大大型計算機センター広報，Vol. 11. No. 1, pp. 29-33, 1978.