プログラムのページ

担当　中　西　正　和

7401　制限エントリのデシジョンテーブルと表プログラムのプリコンパイラ

守屋慎次・斎藤剛・平松啓二
（東京電機大学工学部電気通信工学科）

1.　はじめに

デシジョンテーブルまたは表プログラム[1]（両者をテーブルと総称する）のワンパス・プリコンパイラ FORTAB/LM を作成した．入力は FORTRAN プログラム内に埋込まれた制限エントリテーブル，出力は FORTRAN プログラム，アルゴリズムはマスク法[1]である．FORTAB 自身も FORTRAN (FACOM 230-60) で書かれている．ここでは FORTAB の目的プログラムを参考にして方式を説明し，次いでテーブルの条件部を処理するプログラムについて述べる．

2.　プリコンパイラの概要

アルゴリズムは文献 1)に詳しく述べられているので，ここでは FORTAB に採られた方式を，Fig. 1 の例にもとづいて説明する．

（1）　原始プログラム

```
      INTEGER A,B,C,D
C
      *DOTAB MAXTAB        ←CALL文に相当
C
      *DOTAB MAXTAB
      STOP                 ←テーブル名
C
C
      *TABLE MAXTAB
       READ(5,100)A,B,C,D
       WRITE(6,200)A,B,C,D
       MAX=A
      *CONDITION/ 1 2 3 4
       MAX.GE.B   Y Y Y N
     2 MAX.GE.C   Y Y N *
     3 MAX.GE.D   Y N * *
      *ACTION
       MAX=D      * X * *
       MAX=C      * * X *
       GO TO 3    * * X *
       MAX=B      * * * X
       GO TO 2    * * * X
      *ENDACT
       WRITE(6,300)MAX
  100  FORMAT(4I10)
  200  FORMAT(1X,4I10)
  300  FORMAT(1X,I10)
      *ENDTAB
C
      END
```

条件部　テーブル部　アクション部　テーブルの本体

Fig. 1　プログラムの例

テーブルは Fig. 1 のように書かれる．テーブルには open 型と closed 型がある．Fig. 1 は closed 型の例で ¥DOTAB 文で呼ばれる．詳細については文献2)を参照されたい．

（2）　目的プログラム

Fig. 1 を FORTAB で変換した FORTRAN プログラムを Fig. 2 に示す．

Fig. 1 の条件部の 3 行は，Fig. 2 のそれぞれ C_1, C_2, C_3 のように変換される．C_1 について考えると，1 行目の

ITABD＝1000

は，Fig. 1 の条件部の

MAX.GE.B　　Y　Y　Y　N

```
      INTEGER A,B,C,D
C
      ITABJ0= 1          } 1回目の¥DOTAB
      GO TO 9000
 8011 CONTINUE
C
      ITABJ0= 2          } 2回目の¥DOTAB
      GO TO 9000
 8012 CONTINUE
      STOP
C                                 すべてのルールから選ぶ用意
C
 9000 CONTINUE
       READ(5,100)A,B,C,D          偽 側   真 側
       WRITE(6,200)A,B,C,D
       MAX=A
       ITABT=1111111111          {Y Y Y N
C1     ITABD= 1000               {0 0 0 1
       IF( MAX.GE.B ) ITABD= 111 ←{1 1 1 0
       ITABT=LAND(ITABT,ITABD)
       GO TO 9100
     2 ITABT= 111  ←ルール 3,2,1 から選ぶ用意.
C2   9100 CONTINUE                {Y Y N *
       ITABD= 1100               {0 0 1 1
       IF( MAX.GE.C ) ITABD= 1011 ←{1 1 0 1
       ITABT=LAND(ITABT,ITABD)
       GO TO 9101
     3 ITABT= 11  ←ルール2,1 から選ぶ用意.
C3   9101 CONTINUE                {Y N * *
       ITABD= 1110               {0 1 1 1
       IF( MAX.GE.D ) ITABD= 1101 ←{1 0 1 1
       ITABT=LAND(ITABT,ITABD)
       ITABJ=LAOR(ITABT, 0)
       GO TO(9201,9200,9201,9201),ITABJ ← * X * *
 9200   MAX=D
 9201 GO TO(9203,9203,9202,9203),ITABJ ← * * X *
 9202   MAX=C
Ac    9203 GO TO(9205,9205,9204,9205),ITABJ ← * * X *
 9204   GO TO 3
 9205 GO TO(9207,9207,9207,9206),ITABJ ← * * * X
 9206   MAX=B
 9207 GO TO(9209,9209,9209,9208),ITABJ ← * * * X
 9208   GO TO 2
 9209 CONTINUE
       WRITE(6,300)MAX
  100  FORMAT(4I10)
  200  FORMAT(1X,4I10)
  300  FORMAT(1X,I10)
       GO TO 9001
C                                 ¥DOTAB 文の
 9001 GO TO(8011,8012),ITABJ0 ←直後に戻る
      END
```

Fig. 2　Fig. 1 の目的プログラム

において「MAX. GE. B」が偽の場合に対する用意である．数 1000 は YYYN を，Y→0，N→1，・→1 とコード化した列 0001 を，処理の都合上逆にしたもの．C_1 の2行目の

IF (MAX. GE. B) ITABD=111

は，条件の判定と，条件が真の場合に対する処理を行う．数 111 は YYYN を，Y→1，N→0，・→1 とコード化した列 1110 を逆にしたもの．C_1 の3行目の

ITABT=LAND (ITABT, ITABD)

の LAND は，ITABT と ITABD の論理積を値として持つ関数副プログラムである．たとえば，A=3，B=4 の場合「MAX. GE. B」は偽であるから，

ITABT=1111111111

ITABD=0000001000

となり LAND の値は，0000001000 となる．下位から4桁目だけに1があるから，これはルール4が選ばれたことを示す．何桁目に1があるかをその値とする関数副プログラムが

ITABJ=LAOR (ITABT, r)

の LAOR である．r は原始プログラムに ELSE ルールが書かれていないとき0，書かれているとき ELSE 以外のルールの総数に1加えた数が与えられる．関数 LAOR は，ITABT の下位から何桁目に1があるかを値として持つと同時に，1の個数がゼロの場合 (ELSE を選ぶ)，2個以上の場合（論理に誤まりがある）に対する処理も行う．従って，ITABJ の値は選ばれたルール番号である．

Fig. 1 の条件部の

2　MAX. GE. C　　Y　Y　N　・

に対する目的プログラムは Fig. 2 の C_2 である．C_2 の4～6行目は C_1 と同様．1～3行目が条件への (GO　TO　2 などによる）飛込み[1] に対する用意である．条件エントリに「・」があるのはルール4であるから，飛込みの際ルール4を選択の範囲から除外する必要がある[1]．これに対する目的プログラムが，Fig. 2 の C_2 の2行目

2　ITABT=111

である．数 111 は YYN・を，Y→1，N→1，・→0 とコード化した列 1110 を逆にしたもの．

Fig. 1 のアクション部は，Fig. 2 の Ac 部のように変換される．Ac 部とその他の部分については，Fig. 2 上の説明で明らかであろう．

（3）プリコンパイラ

```
1         SUBROUTINE CENTRY(SEP,RULEN,COL,IN,ER,BP,NSTUB,LAB2)
C*****************************************************************
C         CONDITION 1-GYO O NYURYOKU SITE SONO OBJECT PROGRAM O
C          TSUKURU ROUTINE
C       ROUTINE INPUT  -- CONDITION 1-GYO BUN
C                         IN(1),....,IN(80) FORMAT WA 80A1
C       ROUTINE OUTPUT -- CONDITION 1-GYO NO OBJECT PROGRAM
C                         IOUT(1),....,IOUT(80)
C*****************************************************************
C       IN     -- CONDITION 1-GYO NO CARD IMAGE
C       SEP    -- SIGN TO ENTRY NO KUGIRI NO COLUMN
C       RULEN  -- RULE NO KAZU
C       COL    -- ENTRY SYMBOL GA KAKARERU RULE NO COLUMN
C                 COL(1),...,COL(RULEN)=*CONDITION RUN DE SITEI SARERU
C       LST    -- ENTRY SYMBOL(CONDITION ENTRY NI KAKAREMU MOZI)
C       ER     -- ERROR CODE
C                 ER=0  ERROR NASI
C                 ER=10 YURUSARENAI ENTRY SYMBOL GA ARU
C                 ER=11 ENTRY SYMBOL NO COLUMN NO AYAMARI
C       NP     -- ERROR GA HAKKEN SARETA COLUMN
C       NSTUB  -- IF O SAISHO NO CONDITION-GYO
C                 NE O ZAISHO ME IGO NO CONDITION-GYO
C       LAB2   -- OBJECT PROGRAM NI TSUKUHARERU STATEMENT NUMBER
C       IOUT   -- OUTPUT BUFFER
C       KK     -- SYUTSURYOKU SAREMU CONSTANT NO KISU(BINARY+DECIMAL ETC.)
C       DUMMY1,DUMMY2 -- COMMON NO NAGASA NO CHOSEI
C*****************************************************************
2         COMMON /AREA/ DUMMY1(51),IOUT(80),DUMMY2(102)
3         DIMENSION IN(80),COL(35),LST(5)
4         INTEGER SEP,RULEN,COL,IN,ER,BP,N,Y,V,NN,YY,VV,YN
5         DATA LST/'*','*','*','*','*'/
6         DATA YY/10/
C         LABEL CHECK
7         DO 10 I=1,5
8         IF(LST(I).NE.' ') GO TO 110
9      10 CONTINUE
C         'YES' 'NO' NI TAISURU BIT PATTERN O TSUKURU
C          Y -- 'YY' NO BIT PATTERN
C          N -- 'N' NO BIT PATTERN
10     20 N=0
11        Y=0
12        V=1
13        K=SEP+1
14        DO 40 I=1,RULEN
15        BP=COL(I)
16        NN=1
17        YY=1
18        IF(BP.EQ.K) GO TO 40
19        ER=11
C         ENTRY SYMBOL NO COLUMN NO CHECK
20        DO 30 MM=1,BP-1
21        IF(IN(MM).NE.' ') RETURN
22     30 CONTINUE
C         ENTRY SYMBOL NO CHECK
23     40 K=BP+1

24        DO 50 L=1,5
25        IF(IN(BP).EQ.LST(L)) GO TO (60,80,70,80,80),L
26     50 CONTINUE
C         ENTRY SYMBOL NO MACHIGAI
27        ER=10
28        RETURN
C         ENTRY SYMBOL GA 'N'
29     60 YY=0
30        GO TO 80
C         ENTRY SYMBOL GA 'Y'
31     70 NN=0
C         ENTRY SYMBOL GA ' ','*'
32     80 N=N+NN*V
33        Y=Y+YY*V
34     90 V=V*VV
C         -- IF(* 'N NO BIT PATTERN' -- 0 OUTPUT
35        ENCODE(80,1,IOUT)N
36        CALL OUT
37        DO 100 I=SEP,72
38    100 IN(I)=' '
C         -- IF( 'CONDITION STUB' ) ITABD= 'Y NO BIT PATTERN' -- 0 OUTPUT
39        ENCODE(80,2,IOUT)(IN(I),I=7,50),Y
40        CALL OUT
C         -- ITABT=LAND(ITABT,ITABD) -- 0 OUTPUT
41        ENCODE(80,3,IOUT)
42        CALL OUT
43        ER=0
44        RETURN
C
C         CONDITION-GYO NI STATEMENT NUMBER GA KAKARETE IRU TOKI NO SHORI
C
C         -- GO TO 'LAB2' -- 0 OUTPUT
45    110 CALL GOTO(LAB2)
C         ' ','*' IGAI NO ENTRY SYMBOL NI TAISURU BIT PATTERN O TSUKURU
C          YN -- ' ','*' IGAI NO ENTRY SYMBOL NO BIT PATTERN
46        YN=0
47        V=1
48        ER=11
49        IA=0
50        DO 120 I=1,RULEN
51        IF( IN( COL(I) ) .NE. ' ') IA=1
52        IF( IN( COL(I) ) .EQ. '*' ) IA=0
53        YN=YN+IA*V
54    120 V=V*VV
C         -- ' STATEMENT NUMBER' ITABT='YN NO BIT PATTERN' -- 0 OUTPUT
55        ENCODE(80,4,IOUT) (IN(I),I=1,6),YN
56        CALL OUT
57        IF(NSTUB.EQ.0) GO TO 140
C         -- 'LAB2' CONTINUE -- 0 OUTPUT
58        CALL CONT(LAB2)
59    130 LAB2=LAB2+1
60        GO TO 20
C         -- 'LAB2' ITABT='YN NO BIT PATTERN' -- 0 OUTPUT
61    140 ENCODE(80,5,IOUT) LAB2,YN
62        CALL OUT

63        GO TO 130
64      1 FORMAT(6X,6HITABD=,I10)
65      2 FORMAT(6X,3HIF(,A4A1,8H) ITABD=,I10)
66      3 FORMAT(6X,23HITABT=LAND(ITABT,ITABD))
67      4 FORMAT(6A1,6HITABT=,I10)
68      5 FORMAT(I5,1X,6HITABT=,I10)
69        END
```

Fig. 3　条件部1行を読みその目的プログラムを出力するサブルーチン

Fig. 1 の例ではこのサブルーンチが3回呼び出される．

プリコンパイラ FORTAB は，原始プログラムの1ステートメントを入力し，即座にその目的プログラ

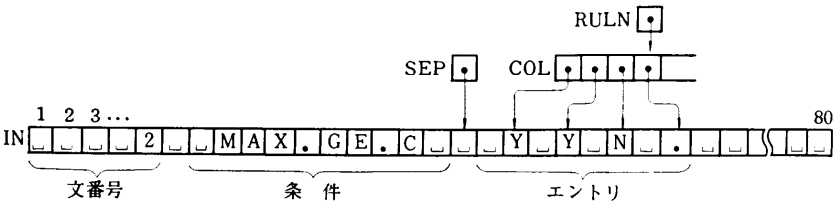

Fig. 4

ムを生成出力するように作られている．FORTRAN 文，¥記号付の文，それにアクション部を入力した場合の処理については，Fig. 1，2 に関するこれまでの説明に止め，ここでは条件部を入力した場合に限って説明する．Fig. 3 は，条件部1行とそれに関する情報を入力し，その1行分の目的プログラムを出力するサブルーチン CENTRY である．¥CONDITION 文を処理した後，¥ACTION 文が現れるまでに書かれた条件行（文番号，条件スタブ，条件エントリ）の処理の際，各条件行で1回ずつ CENTRY が呼出される．CENTRY への入力データの例を，Fig. 4 に示す．Fig. 4 は，Fig. 1 に示すテーブルの第2行目の条件行の処理の際に CENTRY が呼ばれた場合を示している．CENTRY を呼ぶ前の「¥CONDITION/　1　2　3　4　」なる文の処理時に，/の書かれたカラムを変数 SEP に，数1～4の書かれたカラムを COL(1)～COL(4) に，そして，ルール数4を RULEN に格納しておく．IN(1)～IN(80) は，条件行のカードイメージである．CENTRY へのその他の入力として，プリコンパイラが作り出す文番号の LAB2（この場合 9100），何行目の条件行であるかを示す NSTUB（1行目=0，2行目=1，…）がある．Fig. 4のデータを入力した場合，CENTRY は，Fig. 2の C_2 部に示す合計6行の目的プログラムを出力する．「GO TO 9100」を出力するのがサブルーチン GOTO，「9100 CONTINUE」を出力するのがサブルーチン CONT である．残りの4行は出力バッファ IOUT(1)～IOUT(80) から，サブルーチン OUT により出力される．また，変数 VV は，目的プログラム中に生成される定数の基数を指定している．FORTAB では，VV=10 のときルールの最大数は 11，VV=2 とすれば 35 である．

3. おわりに

制限エントリーテーブルのプリコンパイラ FORTAB の特長は，ルールマスクが条件の数に無関係に2語 (ITABT, ITABD) だけであること，ワンパスによりプリコンパイラを短く実現できること(CENTRY を含め FORTAB は約 370 ステートメントで作られている）などである．しかしながら，ルールの選択を実行時に行なうこと，条件の真理値がいかなる場合でもすべての条件判定をしなければならないこと，アクションエントリに×印が書かれていなくてもすべてのアクションの計算型 GOTO 文を実行しなければならないことなど，目的プログラムの効率はあまり良くない．効率に主眼を置いたシステムについては別の機会に報告したい．

おわりに，計算機利用などで種々ご援助いただいた日本情報処理開発センター研究課長甲賀氏に深謝します．

参 考 文 献

1) 守屋・平松：表言語とその処理法について，情報処理，Vol. 13，1972，No. 1，pp. 13-21.
2) 守屋・斎藤・平松：FORTAB/LMマニュアル，東京電機大学情報工学研究室.

（昭和48年11月8日受付）