Reduceの阪大センターにおける利用について

大阪大学理学部　山　本　　昇

§1　序

此の度、汎用数式処理システムの一つであるReduce[1]を、大阪大学大型計算機センターにおいて利用出来る事になった。このReduceの利用法について簡単な例をもとに紹介する。

数式処理については既に種々の場所で紹介されているので[2]、御存知の方も多いと思うが、その特徴について簡単に述べておく。我々が日頃目にしているBasic，Fortran，PL／I　等のプログラム言語においては、その対象とするデータは通常、整数、実数、複素数等の数値である。これに対してReduce　等の数式処理システムは、その名が示しているように数式そのものをデータとして、これに演算を施し、数式の形で出力する。例えばFortranにおいては、プログラム中の一文

　　X＝（A＋B）＊＊2

を実行した結果として、XにはA及びBに格納されていた数値の和の2乗を表す数値が格納される［A＝3.0，B＝4.0　なら　X＝25.0　という様に。］。Reduceにおいては、

　　X：＝（A＋B）＊＊2；

を実行した結果、Xの持つ値は数値ではなく、

　　$A^2+2*A*B+B^2$

という数式である。数式処理システムではこの様に数式に対して、加減乗除、記号微分、不定積分等の演算が定義されている。

Reduceは汎用数式処理システムの一つとして、A．C．Hearnによって開発され、現在では世界中で広く利用されている。このReduceシステム全体はLISPで記述されており、LISP 環境の下で動作する。現在Reduceは東北大学大型計算機センターで開発されたSLIP[3]の上で動いている。Reduceでは以下のような操作を実行することができる。

(1)　多項式及び有理式の加減乗除及びそれに伴う式の展開と整理。

(2)　記号微分。

(3)　置き換えとパターン　マッチング。

(4)　行列演算。

(5)　多項式のGCDの計算。

(6)　テンソル計算

(7) γ－行列代数の計算

（残念ながら、現在阪大センターで利用できるバージョンは、不定積分の機能[4]を持っていない。）

Reduceは先に述べた様にＬＩＳＰ で記述されているインタープリンタ方式のシステムである。プログラムはProcedure の定義とその評価というＬＩＳＰ式の実行形態をとるがその記述はAlgol風のものであって、ＬＩＳＰになじまない人でも（ひょっとしたらＬＩＳＰを良く知っている人にも）理解し易く、又習得も容易である。§２において、簡単な例を基にその使用法を説明していくが、文法についての完全な記述が必要な方は、参考文献 1), 5) を参照していただきたい。

§２ 使 用 例

具体例を示す前に、Reduce で使用できる文字セットと名前について一言述べておく。

Reduce で使用できる文字は、

- 英大文字　Ａ～Ｚ，
- 数字　０～９，
- 空白
- 特殊文字　＋｜－｜＊｜／｜＝｜：｜；｜．｜（｜）｜"｜％｜！

に限られる。変数名、Procedure 名等の名前としては英大文字で始まる任意の文字例を用いる事が出来る。その長さについての制限はない。ただし、空白及び特殊文字を名前の文字列が含む場合には、入力時にエスケープ文字（！）をその空白又は特殊文字の前に付け加えなければならない。［例えば ＊ＡＮＳ という変数をプログラム中で引用する際には、！＊ＡＮＳ の様に書く。］またReduce には幾つかの予約語があって、これらの予約語をユーザーが名前として自由に使う事はできない。これらの予約語の一覧を付録Ａに示した。それでは具体例を見ながらReduceを説明して行こう。例によって以下の例において、下線部はユーザの入力部分、その他はシステムの応答である。

例１

① コメント行　Reduce は％記号で始まる文をコメント行と解釈し何もしない。Reduce は自由形式を採用しているために、一つの文はいくつかの行にまたがっていても良いが、文末を示すターミネーター（＄又は；）が必要である。

② 代入文　Reduce においては、代入演算子は：＝と書かれる。

③ ②の値　入力された文の文末が記号；であると、Reduceはその値を自動

例　1

```
% EXAMPLE NO. 1;                                          ①
A:=(X**3+Y**3+Z**3-3*X*Y*Z)/(X+Y+Z);                      ②

                    2          2          2
A := Z  - Z*Y - Z*X + Y  - Y*X + X                        ③

ORDER X,Y,Z;                                              ④

WRITE A;

    2          2          2
X  - X*Y - X*Z + Y  - Y*Z + Z

FACTOR X;                                                 ⑤

WRITE A;

    2                2          2
X  - X*(Y + Z) + Y  - Y*Z + Z

% EXAMPLE FOR MATRIX ALGEBRA;
MATRIX U,V,W;                                             ⑥

U:=MAT((Z+T,X+I*Y),(X-I*Y,-Z+T))$                         ⑦

V:=MAT((COS(Q),SIN(Q)),(-SIN(Q),COS(Q)))$

ON NERO;

W:=V*U*V**(-1);                                           ⑧
W(1,1) := COS(2*Q)*Z + SIN(2*Q)*X + T
W(1,2) := COS(2*Q)*X - SIN(2*Q)*Z + Y*I
W(2,1) := COS(2*Q)*X - SIN(2*Q)*Z - Y*I
W(2,2) :=  - COS(2*Q)*Z - SIN(2*Q)*X + T

WRITE DET U;                                              ⑨

     2    2    2    2
 - X  - Y  - Z  + T

WRITE DET W;

     2    2    2    2
 - X  - Y  - Z  + T

END;
```

的に出力する。文末が記号＄であれば、値は出力されない。

④　ORDER文。Order文を用いることによって出力される項の順序を制御する事ができる。

⑤　FACTOR文。Order 文と同様に出力制御のための文。

⑥　MATRIX文。宣言文の一つ。変数の形は、MATRIXの他にINTEGER，SCALAR，VECTOR，INDEX，OPERATORがある。

⑦　行列Uの成分をセットする。MATは行列を表す関数で、その引数は行列の行ベクトルを表すリストを必要なだけ並べたものである。

⑧　Reduceにおいては行列の演算もこのように簡単に書くことができる。V＊＊(−1)及び１／Vは、Vの逆行列を意味する。

⑨　DETは行列を引数とする関数で、その行列の行列式を値として返す。この他に転置行列を値とする関数TP、行列のトレースを値とするTRACEが用意されている。

例2　パターンマッチングと置き換え。

① LET文、FORALL～LET文。

LETに続く等式の左辺と同じ形をもつ式をその右辺で置き換えることを定義する。ここではさらにFORALLを用いる事によって任意のXに対するパターンマッチングと置換のルールを定義している。

② 上のLET文を用いてこの式のフーリエ変換を求める。

③ フーリエ変換の結果。

④ LET文で定義した置き換えの機能をキャンセルする。以下のプログラム中での式の評価では上の置き換えは行われない。

```
FOR ALL X,Y LET COS(X)*COS(Y)=(COS(X+Y)+COS(X-Y))/2,COS(X)*SIN(Y)=(SIN(X+Y)-
SIN(X-Y))/2,SIN(X)*SIN(Y)=(COS(X-Y)-COS(X+Y))/2,COS(X)**2=(1+COS(2*X))/2,SIN(
X)**2=(1-COS(2*X))/2;                                                    ——①

FACTOR COS,SIN;

(A1*COS(WT)+A3*COS(3*WT)+B1*SIN(WT)+B3*SIN(3*WT))**2;                    ——②

(COS(6*WT)*( - B3**2 + A3**2 ) + 2*COS(4*WT)*( - B3*B1 + A3*A1) + COS(2*WT)*(2*B3*   ——③
    B1 + 2*A3*A1 - B1**2 + A1**2 ) + 2*SIN(6*WT)*B3*A3 + 2*SIN(4*WT)*(B3*A1 + A3*
    B1) + 2*SIN(2*WT)*(B3*A1 - A3*B1 + B1*A1) + B3**2 + A3**2 + B1**2 + A1**2 )/2

FOR ALL X,Y CLEAR COS(X)*COS(Y),COS(X)*SIN(Y),SIN(X)*SIN(Y),COS(X)**2,SIN(X)*   ——④
**2;

END;
```

例３－１　入力ファイル

```
SYSTEM ?FORT N
*GET /YAMAMOTON/EX3"35"                    ①
*CREATE 22                                 ②
CREATED - 22

*FLIST 35                                  ③
 % EXAMPLE NO.3;
 OPERATOR FAC;                             ④
 INTEGER PROCEDURE FAC(N);                 ⑤
       BEGIN INTEGER M;                    ⑥
        M:=1;
      A:IF N=0 THEN RETURN M;              ⑦
        M:=N*M;
        N:=N-1;
        GO TO A;                           ⑧
   END OF FAC;                             ⑨
 OPERATOR FAC1;
 FOR ALL N LET FAC1(N)=IF N=0 THEN 1 ELSE N*FAC1(N-1); ⑩
 ON TEST;                                  ⑪
 FAC (10);
 FAC1(10);
 OFF TEST;                                 ⑫
 ALGEBRAIC PROCEDURE TAYLOR(EX,X,PT,N);
 BEGIN INTEGER K,N1;
       SCALAR Y,S,Z,X1;
       IF N=0 THEN RETURN SUB(X=PT,EX);    ⑬
       Y:=EX;S:=SUB(X=PT,Y);
       X1:=X-PT;Z:=X1;N1:=1;
       FOR K:=1 STEP 1 UNTIL N             ⑭
           DO BEGIN;  Y:=DF(Y,X,1)$        ⑮
           S:=S+Z*SUB(X=PT,Y)/N1$
           Z:=X1*Z$
           N1:=K*N1$
        END;
        RETURN S;
 END OF TAYLOR;                            ⑯
 FACTOR S,Z;
 GEN:=E**(2*S*Z-S**2);
 TAYLOR(GEN,S,0,5);
 COEFF(!*ANS,S,N);                         ⑰
 WRITE H3*FAC(3);
 ON LIST,FORT;                             ⑱
 OUT 22;                                   ⑲
 WRITE "HERMIT(5)=",H5*FAC(5);
 WRITE "HERMIT(4)=",H4*FAC(4);
 SHUT 22;                                  ⑳
 END;                                      ㉑
*CRUN 6091150343/CMD/SREDUCE,R
WELCOME TO SLISP-REDUCE SYSTEM .
```

例３－２

```
 REDUCE 2 (APR-1-82) ....
a IN 35;                                   ㉒
    % OPENED FILE IS...   35               ㉓

 % EXAMPLE NO.3;                           ㉔
 OPERATOR FAC;

 INTEGER PROCEDURE FAC(N);
 BEGIN INTEGER M;
 M:=1;
 A:IF N=0 THEN RETURN M;
 M:=N*M;
 N:=N-1;
 GO TO A;
 END OF FAC;

 OPERATOR FAC1;

 FOR ALL N LET FAC1(N)=IF N=0 THEN 1
 ELSE N*FAC1(N-1);

  ON TEST;
  TIME: 968                                ㉕
  FAC(10);
  3628800
  TIME: 1095                               ㉖
  FAC1(10);
  3628800
  TIME: 1395                               ㉗
  OFF TEST;

  ALGEBRAIC PROCEDURE TAYLOR(EX,X,PT,N);
  BEGIN INTEGER K,N1;
  SCALAR Y,S,Z,X1;
  IF N=0 THEN RETURN SUB(X=PT,EX);
  Y:=EX;
  S:=SUB(X=PT,Y);
  X1:=X-PT;
  Z:=X1;
  N1:=1;
  FOR K:=1 STEP 1 UNTIL N DO BEGIN;
  Y:=DF(Y,X,1)$
  S:=S+Z*SUB(X=PT,Y)/N1$
  Z:=X1*Z$
  N1:=K*N1$
  END;
  RETURN S;
  END OF TAYLOR;

  FACTOR S,Z;

  GEN:=E**(2*S*Z-S**2);

                          2
          (2*S*Z)   (S )
  GEN := E        /E
```

例３－２（続き）

```
TAYLOR(GEN,S,0,5);

     5  5        5  3        5        4  4        4  2        4        3  3        3
(4*S *Z  - 20*S *Z  + 15*S *Z + 8*S *Z  - 24*S *Z  + 6*S  + 12*S *Z  - 18*S *
                                                                                    ㉘

          2  2       2
 Z + 12*S *Z  - 6*S  + 6*S*Z + 3)/3

COEFF(*ANS,S,N);

*** N5 N4 N3 N2 N1 N0 ARE NON ZERO                                                   ㉙

5                                                                                   ㉚

WRITE N3*FAC(3);                                                                    ㉛

     3
24*Z  - 36*Z

ON LIST,FORT;
   % OPENED FILE IS...   22
  % CLOSED FILE IS...  35
a END;                                                                              ㉜

# EVALUATED !

  ENTERING LISP ...
a $*SFILE                                                                           ㉝

          #####  END OF READING FILE CODE 05.  IN TLICS RED(CHR)  #####
- - - - - - - - - - - - - -  F I N  ( S L I S P )  - - - - - - - - - - - - -        ㉞
*
*FLIST 22                                                                           ㉟
HERMIT(5)=160*Z**5-800*Z**3+600*Z
HERMIT(4)=64*Z**4-192*Z**2+48
  % CLOSED FILE IS...  22
*BYE
```

例３　ファイル操作，Procedure

Reduceでは端末からReduceの文を直接入力する他に、ファイルにReduceのプログラムを作っておいて、Reduceシステムからそのファイルを呼び出して実行することもできる。また結果も適当な形でファイルに出力することができる。これによって後日の計算に結果を利用したり、Fortranプログラムでこの結果を利用する事が容易にできる。

① 入力プログラムを作成したファイルをファイルコードfc＝35と定義する。Reduceからはこのファイルコードによってファイルを呼び出す。

② 出力用のテンポラリーファイルを fc＝22として作る。

③ プログラムのリストを取る。

④ OPERATOR宣言。Procedure文によってoperatorを定義する場合にはこの宣言は無くとも良い。

⑤ Procedure文。ProcedureにはFortranのFUNCTIONのように型を明示すること

もできる。Procedureの本体は、Procedure文の次の文と解釈される。

⑥ 複文。Reduceでは他のAlgol系の言語と同様に、BEGIN～ENDによって複数の文をブロック化して一つの文とすることができる。

⑦ ラベル。複文の中では、GOTO, RETURN等で流れを制御する。また文にラベルを付けることができる。ここではAが文のラベルである。

⑧ GOTO文。GOTO文は複文の中でのみ有効である。

⑨ ⑥のBEGINで始まった複文の終わり。ENDとターミネーターの間の文字列は意味を持たない。

⑩ OperatorはPROCEDURE文を用いずとも、LET文によって定義する事ができる。また再帰的な定義も許される（PROCEDURE文においても。）。

⑪ TESTフラグを立てる。TESTフラグを立てると以下の各文の実行毎に、そのTSSセッションで使われたCPU時間を表示する。単位はmsecである。

⑫ TESTフラグを下げる。

⑬ SUB関数。Reduceにおいては、代入又は置き換えのためには代入文（：＝），LET文の他にSUB関数が用意されている。他の二つの方法との違いはこの置き換えはその関数の中だけでのみ有効である事である。ここでは、EXの中に現われたXをPTで置きかえたものを値として返す。

⑭ FOR～：＝～STEP～UNTIL DO文。STEPが1の場合には、FOR I：＝1：N DO～ という様に省略した書き方もできる。

⑮ DF関数。DF（Y, X, 2）＝d^2y/dx^2 の意味。

⑯ TAYLORの定義終わり。

⑰ COEFF関数。！＊ANSをSについて整理し、Sについてn次の項の係数をHnという変数にしまう。！＊ANSは直前の文の値を持つシステムの変数名である。エスケープ文字（！）の使われ方に注意。

⑱ LIST, FORTのフラグを立てる。フラグの効果については付録を見よ。

⑲ OUTコマンド。以後の出力を、ファイルコードで指定したファイルに出力する。

⑳ SHUTコマンド。ファイルへの出力をやめて、以後の出力は標準出力機器に出力する。

㉑ 入力ファイルの終りを示すEND。これがないとシステムエラーとなる。

㉒ INコマンド。ファイルコードで指定されるファイルからReduceプログラムを読み込む。

㉓ ファイルをOPENした事を示すメッセージ。

㉔ ファイルから読み込んだプログラムのエコーバック。

㉕ TSSセッション開始からこれまでのCPU時間（msec）。

㉖㉗ 10の階乗の二つの方法による計算では再帰を用いない方が実行速度の点で有利である事がわかる。

㉘ TAYLOR(GEN, S, 0, 5)の値。

㉙ システムからのエコーバックではエスケープ文字(!)は出力されない。

㉚ COEFFを実行した結果、Sについて5次までの係数があった事を示すメッセージ。

㉛ COEFFの値は、主変数についての最高次数である。

㉜ Reduceシステムを終了するためのEND。ファイルの終りを示すENDはエコーバックされない。

㉝ コマンドファイル処理中の$*$TALKを終了させる。

㉞ SLISPのエンディングメッセージ。

㉟ OUTコマンドによって出力された結果を見る。

§3 Reduceシステムの起動法

§2の例で、Reduceがどんなものであるかという感じは掴んでいただけたと思う。このReduceに興味を持たれた方には、早速に、TSSターミナルに向って御自分で例の幾つかを実行されてみる事をお勧めする。

Reduceを起動するためには、TSSのSYSTEM? モード又はビルドモードにおいて、コマンドファイル処理、

```
CRUN 6091150343/CMD/SREDUCE, R
```

又は

```
CRUN 6091150343/CMD/SREDUCE, R;;tt
```

を実行する。ここでttはコマンドファイル処理のCPU時間を指定するパラメーターで単位は秒である。CRUNコマンド投入後しばらくすると、

```
Please Input BEGIN Command.
```

というメッセージに続いて、入力促進記号として@が出力される。これに対して、

```
BEGIN(nn)
```

を入力する。これでReduceシステムのイニシャライズが行われてReduceコマンド待ちの状態になる。BEGINコマンド中のnnはReduceが結果を出力する際の出力幅を指定するパラメーターで、NIL又は正の整数を指定する(CRTターミナルでは77を指定する事をお勧めする)。

Reduceプログラムを実行した後、Reduceシステムを終了させるには、

```
END;
```

を入力した後

$ * $ FILE

を入力する。

Reduceシステムは基本的にインタープリター方式のシステムであるから、TSSでの利用に適しているが長時間のCPU時間を必要とするJOB等ではバッチによる処理も可能である。この時必要なJCLは、

```
1
$    JOB        16課題番号$パスワード, B
$    PRGRAM     RLHS, NAME/LISPGO
$    PRMFL      H*, R, R, 6091150343/BATCH/LH
$    PRMFL      02, R, S, 6091150343/BATCH/RDMP
$    FILE       01, , 100R
BEGIN
   NIL
    |
    | Reduceプログラム
    |
$       END
```

である。

§4 最後に

以上で阪大センターでのReduceの利用法についての説明を終わるが、最後に2, 3の注意点について記しておく。

Reduceは複雑な式の簡単化等に威力を発揮するが、最終結果は非常に簡単な式であっても途中では膨大な量の式を取り扱わねばならない事がしばしばある。このような状況が生じると、記憶領域の操作等に必要な時間や式の変形のために要する時間は指数的に増大する。これを避けるためには、なるべく長い式の出力は避ける、途中での結果が長くならないようにしながら計算を何段階かに分ける、等の工夫が必要である。さもなければReduceは貴方の予算をアッという間に食べつくしかねない(何しろReduceは通常でも一つのコマンドの実行にmsecオーダーのCPU時間を要するので。)。

Reduceは§1で述べた様にLISPで記述されている。従ってLISPレベルで適当な関数を定義する事によってより広範囲な可能性を実現できる。

Reduceの阪大センターへの移殖には、当センターの研究開発の時間を利用させていただいた。また、SLIP-Reduceのソースを提供していただいた東北大学大型計算機センター、移殖作業について全面的な御支援をして下さった山形大学工業短期大学部　高橋良雄先生、東北大学大型計算機センター　一関京子先生に感謝をささげます。なお、当センターでのReduce使用上の問い合わせは、大型計算機センター　多喜(tel.2837)、理学部　山本(tel.4111)又は、基礎工学部　宮村(tel.4864)まで御連絡下さい。

〔参考文献〕

1) A. Hearn "Reduce-2 User's manual." UCP-19, Univ. of Utah (March 1973.).

[金田康正　「REDUCE　利用の手引」　東大計算機センターニュース, 9 No.2～9 No.10 (1977). はこの邦訳である。]

2) 金田康正　「数式処理言語」　情報処理, 22 No.16, 565(1981).

佐々木建昭, 後藤英一　「計算機による数式処理と物理学への応用」　日本物理学会誌　35 No.2, 99(1980).

R.Pavelle, N.Rothstein and J.Fitch "Computer algebra"　サイエンス　12 No.2, 82(1982).

など。

3) 藤木, 高橋, 一ノ関, 桂　「TSSによるREDUCEの利用」　SENAC 14 No.4, 57(東北大学計算機センター, 1981).

4) A. Hearn「数式処理による積分」情報処理　22 No.7, 639(1981).

5) 東大物性研計算機室編　「REDUCE-2　利用の手引き」(1981).

付録A．予約された名前

予約語	BEGIN, DO, ELSE, END, FOR, FUNCTION, GO, GOTO, LAMBDA, NIL, PRODUCT, RETURN, SUM, STEP, TO
定数	E （自然対数の底）
	I （$\sqrt{-1}$）
Prefix 演算子	ARB, COEFF, COS, DEN, DET, DF, EPS, G, LOG, MAT, NUM, SIN, SUB, TRACE
Infix 演算子	:=, =, +, −, *, **, . , AND, NOT, OR, MEMBER, EQ, GEQ, GREATERP, LEQ, LESSP
コマンド	ALGEBRAIC, ARRAY, CLEAR, COMMENT, END, FACTOR, GO, GOTO, IF, IN, LET, LISP, MASS, MATCH, MATRIX, MSHELL, NOSPUR, OFF, ON, OPERATOR, ORDER, OUT, PROCEDURE, RETURN, SAVEAS, SCALAR, SHUT, SPUR, SYMBOLIC, VECTOR, WEIGHT, WRITE, WTLEVEL

付録B．モードフラグ

コマンド ON, OFF の引数として使われるフラグを以下に示し、ONされた時の機能を説明する。カッコの中はデフォールト値

ALLFAC	(on)	：共通因子をくくり出す。
DEFN	(off)	：REDUCEの入力を、評価せずにLISP形式にして出力する。
DIV	(off)	：共通分母をとらずに、分子の各項を分母で割って出力する。
ECHO	(on)	：入力をエコーバックする。
EXP	(on)	：式を全て展開してから評価する。
FORT	(off)	：FORTRAN形式の出力をする。
GCD	(off)	：分母分子を最大公約数で割る。
INT	(on)	：TSS環境下に適した実行を行う。
LIST	(off)	：各項を1行づつに出力する。
MCD	(on)	：有理式の和を取る時に、共通分母を取る。
MSG	(off)	：診断メッセージが出力される。
NAT	(on)	：出力を自然な形式にする。
NERO	(off)	：零を代入されているものの出力を抑制する。
PRI	(on)	：出力形式を整える。

RAT　　（off）：分子の各項を分母で割った形式で出力する。

RESUBS　（on）：一度置き換えをしたものを再評価して、更に置き換えをしようとする。