Examination in Backus' FP System for Non- von-Neumann Computer

Jun ITO , Tetsuya ITO * and Naoki TAKESHIMA**

* Instructor of Akita Joho Bijinesu Senmongakko, Nakadori 4-chome, Akita 010

** Student of Akita National College of Technology, Iijima Bunkyo-cho, Akita 011

(Received on Oct. 31, 1985)

1. Introduction

Recently studies of functional language are so actively made, of which the most noticeable one may be the functional programing system (usually called Backus' FP system or only FP system) proposed by Backus [1, 2, 3]. In FP system a program is made (or "defined" rather than "made") by combinating or superposing several functions. In this report many programs are made in FP system, and from these ripe experiences FP system is commented on. FP system has some merits as already suggested by Ida and et al [4] who applied FP system to office work concerning purchase of books. Merits of FP system are pointed out from the view point of programming some engineering and scientific calculation. And also a minimum FP system is proposed for purpose of making the system small and powerful whithin the framework of being applicable to practical problems.

2. Example of program

About forty programs are made in FP system. In order to express some merits and features of FP system, two typical examples of these programs are illustrated in the fllowing sections;

2.1 Least Square Method

The n pairs of data x, y should be approximated to linear function y=a+bx. Input and output data are $<x_1, y_1, x_2, y_2, \ldots\ldots\ldots, x_n, y_n>$ and $<a, b>$. Algorithm to get a, b and name to define function are as follows:

$a=\bar{y}-b\bar{x}$······A, $b=Sxy/Sxx$······B, $\bar{x}=\Sigma x_i/n$······XBAR

$\bar{y}=\Sigma y_i/n$······YBAR, $Sxx=\Sigma x_i^2-\frac{1}{n}(\Sigma x_i)^2$······SXX

Σx_i^2······SUMXX

$Sxy=\Sigma x_i y_i-\Sigma x_i \Sigma y_i/n$······SXY

$\Sigma x_i y_i$······SUMXY, Σx_i······SUMX, Σy_i······SUMY

Program for Least Square Method consists of the under-mentioned definitions.

```
Def     LSQ≡[A, B]
        A≡-o[YBAR, ×o [B, XBAR]]
        B≡÷o[SXY, SXX]
        SXY≡-o[SUMXY,÷o[×o[SUMX, SUMY],÷o[LENGTH, 2̄]]]
        SXX≡-o[SUMXX,÷o[×o[SUMX, SUMX],÷o[LENGTH, 2̄]]]
        SUMXY≡NULLoTLoTL→×o[1, 2];+o[×o[1, 2], SUMXYoTLoTL]
```

SUMXX≡NULLoTLoTL→×o[1, 1];+o[×o[1, 1], SUMXXoTLoTL]
XBAR≡(/÷)o[SUMX, LENGTH, $\bar{2}$]
YBAR≡(/÷)o[SUMY, LENGTH, $\bar{2}$]
SUMX≡NULLoTLoTL→1;+o[1, SUMXoTLoTL]
SUMY≡NULLoTLoTL→2;+o[2, SUMYoTLoTL]

2.2 Numerical Integration by Simpson's Rule

Integral S from limits a to b of function f(x) with respect to variable x can be expressed as follows by using Simpson's rule.

$$S=\frac{n}{3}[f(a)+f(b)+4\{f(a+h)+f(a+3h)+\cdots\cdots+f(a+(2n-1)h)\}$$
$$+2\{f(a+2h)+f(a+4h)+\cdots\cdots+f(a+(2n-2)h)\}$$

where h is subinterval and given by (b−a)/2n. Input and output data are <a, b, h> and S.

Def SMP≡×o[(/+)o[FXo1, FXo2, ×o[$\bar{4}$, SO], ×o[$\bar{2}$, SE]],÷o[3, $\bar{3}$]]
SO≡SUMo[$\bar{0}$,+o[1, 3],+o[2, 3],+o[3, 3]]
SE≡SUMo[$\bar{0}$,+o[1,+o[3, 3]], 2,+o[3, 3]]
SUM≡EQo[2,3]→1; SUMo[+o[1, FXo2],+o[2, 4], 3, 4]

If integrand is, for example, $f(x)=x^2+1$, FX is defined by
FX≡+o[×o[ID, ID], $\bar{1}$].

3. Consideration to Programs

In the program of least square method, definition functions except LSQ, A and B so frequently appear and are quite general in conventional engineering calculation. Therefore these functions can be used as library which has already been made in high quality. In case of being prepared to supply these library functions, program of least square method is completed by defining the only three functions LSQ, A and B. Namely by only consideration of general algorithm, program of high quality can be made so effectively. Example of numerical integration by using Simpson's rule is difficult one to quote functions from library. However that this program consists of only four definition functuins means that FP is superior to conventional language in power of expression. Furthermore, an FP program corresponds to mathematical expression, which we are used to from childhood, so well that the algorithm can be built easily unlike conventional programming language.

4. Increase in Power of FP System

FP system should naturally be powerful like the other conventional programming language. That a system is poweful does not mean that it has so many primitive functions and so many program forming operaters (called PFO) and any programs are easily made. Every primitive function or every PFO should be strong itself and should have high frequency of use. Primitive function or PFO of extremely low frequency of use and of overlapping easily expressed by the other should be omitted from the system. At present the auther and et al are making the processor of FP system by using FORTRAN, BASIC, and ASSEMBLY language and personal computer (8 bit). In that case, it is very useful that the system is small and powerful.

From programing engineering calculation and fundamental processor of data, frequencies in use of primitive function and PFO are investigated. Concerning the primitive function, "SELECTER" functions are used most of all and are more than fifty per cent except for arithmetical operators. The next most used primitive functions are "IDENTITY" and "TAIL" which are used about ten per cent respectively. "EQUAL", "LENGTH" and "NULL" follow the above-mentioned primitive functions. It is the most important thing that "LESS THAN" should be introduced into the FP system and is especially useful and effective in data processing program in common with "APPEND RIGHT" and "APPEND LEFT". Allthe above-mendtioned primitive functions and logical operators seem to be able to almost cover every program concerning ordinary numerical calculation appearing engineering or scientific problems. Namely, speaking of primitive functions, if "LESS THAN" and "INTEGER" are added to the FP system proposed by Backus [1], the modified system doubly increases in power, that is, can program twice as many in spite of elimination of several overlapping primitive functions.

In relation to PFO, "COMPOSITION" and "CONSTRUCTION" have the highest frequency in use, and "CONSTANT" and "CONDITION" follow them. "APPLY TO ALL" and "INSERT" are used a little. "BINARY TO UNARY" and "WHILE" are not necessary and can be easily defined by using other functions as follows;

$$(\mathrm{BUFX}) \equiv f \circ [\bar{x}, \mathrm{ID}]$$
$$\mathrm{WHILE} \equiv p \rightarrow (\mathrm{WHILE}\ pf) \circ f;\ \mathrm{ID}$$

5. Conclusion

From programming for engineering and scientific calculation, merits of Backus' FP system were pointed out. And also within the framework of being applicable to practical problems, a minimum FP system was proposed.

References

[1] J. Backus, CACM, Vol. 21, No. 8 (1978), pp. 613–641.
[2] J. Backus, Proc. International Colloquium on the Formalization of Programming Concepts, Lecture Notes in Computer Science, No. 107, Spring-Verlag, Heidelberg.
[3] J. Backus, Proc. Functional Programming Languages and Computer Architecture, Oct. 1980, pp. 1–10.
[4] Ida, Yusa and Kawakubo, Japan Society of Software Science, Proceeding of First Conference, 2F–2 (Special Session), pp. 107–116.