数理解析研究所講究録
第215巻 1974年 12-25

12

配列の計算を速くするサブルーチンパッケージ "HISARP" について

日本IBM　加藤 隼一

§1. 主旨

配列処理の高速化のためにアセンブラー言語で多数のサブルーチンを作成し，これらに対応するFORTRANサブルーチンと比較して，実行速度につき最高約20倍の能率が得られた。

§2. 配列処理の高速化

電子計算機のプログラムの中には，所要時間の大部分が配列の計算のために費やされ，しかもこれらの計算のほとんどが配列のごく基本的な処理の組み合わせになっているような種類のものは多い。筆者が数年来携わってきた構造解析関係の多くのプログラムはこのような傾向が顕著にみられる例である。数値計算に広く用いられている連立一次方程式の数値解法のプログラムなども，主要な部分は配列の基本的な処理によって占められている。

従って，計算を能率化するための方法として，モデル化の

段階での工夫や全体的なアルゴリズムの改良の他に，配列処理の高速化ということが重要になってくる。

配列の処理を高速化するための *approach* を次の3つに大別してみよう。

① 配列処理アルゴリズムの工夫（例：行列積の計算における *Winograd* の方法。FFT。）

② ソフトウェアによる高速化（例：コンパイラーによる *optimization*）

③ ハードウェアによる高速化（例：配列処理のための専用演算回路やマルチプロセッサー）

このうち，②に属する *approach* について考えてみる。コンパイラーによる *optimization* の例として，IBM の FORTRAN H コンパイラーの場合は，使用者の選択（OPT=0,1,2 のいずれか）により，コンパイルの速さ，目的プログラムの小ささ，目的プログラムの実行の速さのうちのいずれかを重視してコンパイルを行うようになっている。

FORTRAN プログラムの

```
      SPL=0.0
      DO 100 I=1,N
  100 SPL=SPL+X(I,I)
```

という部分をコンパイルする場合，DO ループの実行の途中

で，和を毎回記憶域SPLに蓄える必要はなくレジスターに置いたままにすることができることや，配列要素X(I,I)のアドレスを求めるのに毎回直接Iの値にまでさかのぼる必要がないことなどを活かしてコンパイルを行えば，より効率のよい目的プログラムを生成することができる。

しかしふつう，コンパイラーによるoptimizationの結果にも，まだ高速化の余地がかなりある。そこで筆者は，FORTRANプログラムの配列処理の部分をCALLステートメントに置きかえ，予めアセンブラー言語で書いておいた高速サブルーチンに実行させるという方法をとった。これは②に属するもう一つの方法である。初めはコンパイラーのoptimizationの機能を単に補完するだけのものであったが，その後いくつかの面でそれ以上のものとなりつつある。

§3. HISARPの概要

HISARP (High Speed Array Processor) は，筆者が構造解析のアプリケーションプログラムの開発に際して配列の高速処理のために作成し使用していた種々のサブルーチンを改良し，新たに系統的に作成して，一つのパッケージにまとめたものである。以前にまとめたもの(version 1)と，最近これに更に改良を加えて新たに整備しつつあるもの(version 2)とがある。

1) HISARPは，配列に関する種々の，ごく基本的な処理，演算を，高速で実行するためのサブルーチンの集まり（パッケージ）である。

2) FORTRAN プログラムからCALLできる。

3) アセンブラー言語で書かれた（マクロ命令で生成された）ものである。即ち，IBM OS/360 アセンブラー言語（Hレベル）を使って定義したマクロ命令により，すべてのサブルーチンが系統的に生成された。（その副次的な効果として，サブルーチンの個数の多さ(version 1では現在315個）にもかかわらず，出取りが容易であった。）

4) 各サブルーチンは

i) それぞれ単一の機能を有する。

ii) 実行時間に関して（特にシステム360モデル195向きに）optimizeしてある。

iii) 一部の例外を除いてself-relocatableかつre-entrantである。

5) optimizeに際しては

i) まず第一に，ループの内部を最優先した。

ただしversion 1とversion 2とではループの構造が異なる。(後述)

ii) ループの外部もできるだけoptimizeした。その結

果，処理される配列が小さくても効率のよさがあまり減殺されない。

6) 扱える配列のタイプは，整数（1バイト長，半語長，1語長）と，実数および複素数（いずれも1倍精度，2倍精度，4倍精度）とで，合計9種類である。

7) システム360のモデル20以外およびシステム370のすべてのモデルにおいて使用可能。ただし4倍精度の演算を含むサブルーチンは4倍精度命令セットを備えたモデルでのみ使用可能。

§4. サブルーチン名

HISARPの機能の説明の一助に，HISARPのサブルーチンの名づけ法を記す。サブルーチンの名前を5文字または6文字とし，次のように3つの部分から構成する。

サブルーチン名＝type·mode·operation

typeは，処理される配列のタイプを表わすためのもので，1文字または2文字からなる。

type＝B/H/I/S/D/Q/CS/CD/CQ

B: バイト整数

H: 半語長整数

I: 1語長整数

S: 1倍精度実数

D: 2倍精度実数

Q: 4倍精度実数

CS: 1倍精度複素数

CD: 2倍精度複素数

CQ: 4倍精度複素数

*mode*は，配列中の要素のうち，どのような並び方のものを処理対象として扱うかを表わすための1文字である。

mode = M/V/T

M: *matrix mode*（マトリックスの全体または一部を扱う。ただし配列中のマトリックス要素の排列はFORTRAN流に*columnwise*であるものとする。）

V: *vector mode*（連続したいくつかの要素を扱う。）

T: *step mode*（一定個毎の要素を扱う。）

なお，マトリックスの場合，V *mode*で列を，T*mode*で行をそれぞれ扱うことができる。

*operation*は，処理や演算の種類を表わすためのもので，3文字からなる。

operation = INI/SCL/ADD/SUB/MOV/XCH/SNA/ACC/ASQ/IPR/PIV/……

INI: 初期値化　　$x_\alpha \leftarrow C$

SCL: スカラー倍　　$x_\alpha \leftarrow Cx_\alpha$

ADD: 加算　　$x_\alpha \leftarrow x_\alpha + y_\alpha$

SUB: 減算　　$x_\alpha \leftarrow x_\alpha - y_\alpha$

MOV: 移動　　$x_\alpha \leftarrow y_\alpha$

XCH: 交換　　$x_\alpha \leftrightarrow y_\alpha$

SNA: スカラー倍を加える　　$x_\alpha \leftarrow x_\alpha + Cy_\alpha$

ACC: 合計　　$a \leftarrow \sum_\alpha x_\alpha$

ASQ: 平方和　　$a \leftarrow \sum_\alpha x_\alpha^2$

IPR: 内積　　$a \leftarrow \sum_\alpha x_\alpha y_\alpha$

PIV: 絶対値最大要素を探す。

(ただし, 1または2次元の添字αの集合はmodeによる。

modeとoperationとの特別な組み合わせ(または融合したもの)として; 現在次の2つがある。

MPRD: マトリックス×マトリックス　($P \leftarrow {}^tXY$)

MVPR: マトリックス×ベクトル　($\vec{p} \leftarrow {}^tX\vec{y}$)

version 1のサブルーチンの現在の個数315は, 以上のすべての組み合わせの数 $9\times(3\times11+2)$ である。

§5. version 1 / version 2

version 1とversion 2との相異は, サブルーチン内のループ(2重以上のループになっている場合には一番内側のも

の。以下でも同様。）の構造の相異に基づいている。

version 1 のループの構造は，FORTRAN の DO ループをそのまま optimize した形になっている。すなわちループは，データそのものの演算のための1組の命令群（演算部）と，配列要素のアドレスの更新やループコントロールのための1組の命令群（コントロール部）とで構成されている。

version 2 のループは，m組の演算部と1組のコントロール部とで構成されており，ループを1回まわる毎にm回分の演算が実行される。mを詰め込み係数と呼ぼう。このループでの演算の必要実行回数をNとすれば，m組の演算部のうち後から mod(N-1,m)+1 番目の組からループの実行を開始すればよい。ループを抜け出すのは必ずループの最後からである。

従って，HISARP のループの構造を FORTRAN 風に表現すれば次のようになる。

version 1：

```
      I=1
1     第 I 組の演算          } 演算部
      I=I+1                  }
      IF(I.LE.N) GO TO 1     } コントロール部
```

version 2：

```
      L=m-MOD(N-1,m)
      I=2-L
      GO TO (1,2,…,m),L
1     第I組の演算                    ┐
2     第I+1組の演算                  ├ 演算部(m組)
      ⋮                              │
m     第I+m-1組の演算                ┘
      I=I+m                          ┐
      IF(I.LE.N)GOTO1                ┘ コントロール部
```

HISARPを受け容れるシステム360,370の各モデルの中でも，特に適当なモデル195は，いわゆるパイプラインコンピューターの1つであり，本文の内容に関連して次のような特徴を持っている。

1) 命令を先取りして一時蓄えておくための，容量8ダブルワードの棚がCPU内にあり，1つのループ全体がこの棚に納まると処理効率が上る。(ループモード)

2) 主記憶域へのアクセスタイムを縮めるために，容量が32Kバイト＝4Kダブルワード(ただしK=2^{10}=1024)

の高速バッファーを有する。

3) CPU内での機能が分化し，演算の並行処理能力が高いので，個々の演算が速いだけでなく，全体としての実効速度が極めて大きくなり得る。

4) ブランチ命令は，他の命令に比較して割合遅い。

version 1 のほとんどのループはループモードで実行される。version 2 のループも原則としてループモードで実行されるようにし，その範囲内で詰め込み係数mを最大にした。

一般に，version 2 は version 1 に比較して，次のような特徴を持ち，より高速になっている。

1) ループ実行中の演算の並行処理の程度が高い。

2) ブランチ命令の回数が少い。

§6. 実行時間の比較

HISARPに含まれる2種類のサブルーチンDVACC(配列の要素の合計), DMPRD(行列積)（いずれも2倍精度実数演算ルーチン。§4参照.）につき，対応するFORTRANプログラムを作成し,(プログラムリスト参照)

Ⓐ FORTRAN G コンパイラーでコンパイルしたもの。

Ⓑ FORTRAN H コンパイラーでコンパイルしたもの。(OPT=2を指定してoptimize。§2参照)

Ⓒ HISARP version 1

@ HISARP version 2

の能率テストをモデル195で行い，次の要領で単位演算当りの平均所要サイクル数Cを算出して比較した。(グラフ参照)

$$C = T_{EX} / I / J / T_C$$

$$J = \begin{cases} N & (\text{DVACCの場合。Nは要素の個数。}) \\ N^3 & (\text{DMPRDの場合。Nは行列の次数。}) \end{cases}$$

T_{EX}: 実行時間(CALLステートメントの時間も含む)の測定値

I: 繰り返しCALL回数

J: CALL1回当りの単位演算の実行回数

T_C: モデル195のマシンサイクルタイム(54n sec.)

能率テストは，同一の引数でサブルーチンを繰り返しCALLし，実行時間が1秒のオーダーになるようにした。このようなテストを，各サブルーチンにつき，Nのいくつかの値について行った。なお，「単位演算」は，DVACCの場合は要素1個分の加算を，またDMPRDの場合は1対の要素の乗算および加算を意味し，Cには一切のオーバーヘッドの負担分が含まれている。

version 2のDVACCは，詰め込み係数m=14で，4個の浮動小数点レジスターに交互に加算を行っているので，Cの値が，オーバーヘッドを含んでいてなおかつ，加算単独の所要サイクル数2より小さくなった場合がある。(グラフ参照)

これは，モデル195の浮動小数点加算器が，互いに独立な加減算なら1サイクル毎に処理を開始することができる，という理由による。ただし複数個のレジスターを使うので，加算の結合の組み合わせが異なってくるために，DVACCに関しては，結果の値に含まれる丸め誤差について，2つのversionの間で差が生じる。

version 2のDMPRDは，詰め込み係数m=5であり，Cの値が，乗算(3サイクル)と加算(2サイクル)の所要サイクル数の単純和5よりも小さくなった場合がある。これは加算と次の乗算とが並行処理された結果である。

Cの値に含まれているオーバーヘッドの負担分はNの値の増加とともに減少し，従ってCは単調減少の傾向を示すはずであるが，テスト結果では，DVACCの場合N=3000からN=10000への途中で，またDMPRDの場合N=60からN=80への途中で，Cの値が増加している(グラフ参照)。これは，繰り返し取り出されるデータが高速バッファーの容量(4096ダブルワード)を越すと，高速バッファーの効力が低下することに起因している(Note参照)。DMPRD(行列積。P←${}^tXY$)の場合，X全体とYの1列分とが高速バッファーに納まらなくなる(すなわちN^2+Nが4096を越える)時に，Cの値が増加するはずであり，テスト結果と一致する。　X,Y全体が高速バッファーに納まらなくなる時

も，Cが増加するが，わずかに過ぎない。

Cの値から，@,Ⓑ,Ⓒ,@の実行時間の比を求めると，ほぼ次の如くである。

DVACCの場合　20.4 : 3.7 : 3.7 : 1　(N=3000の時)

DMPRDの場合　17.2 : 2.9 : 1.6 : 1　(N=40,60の時の平均)

☆プログラムリスト

(比較のために使われたFORTRANプログラム)

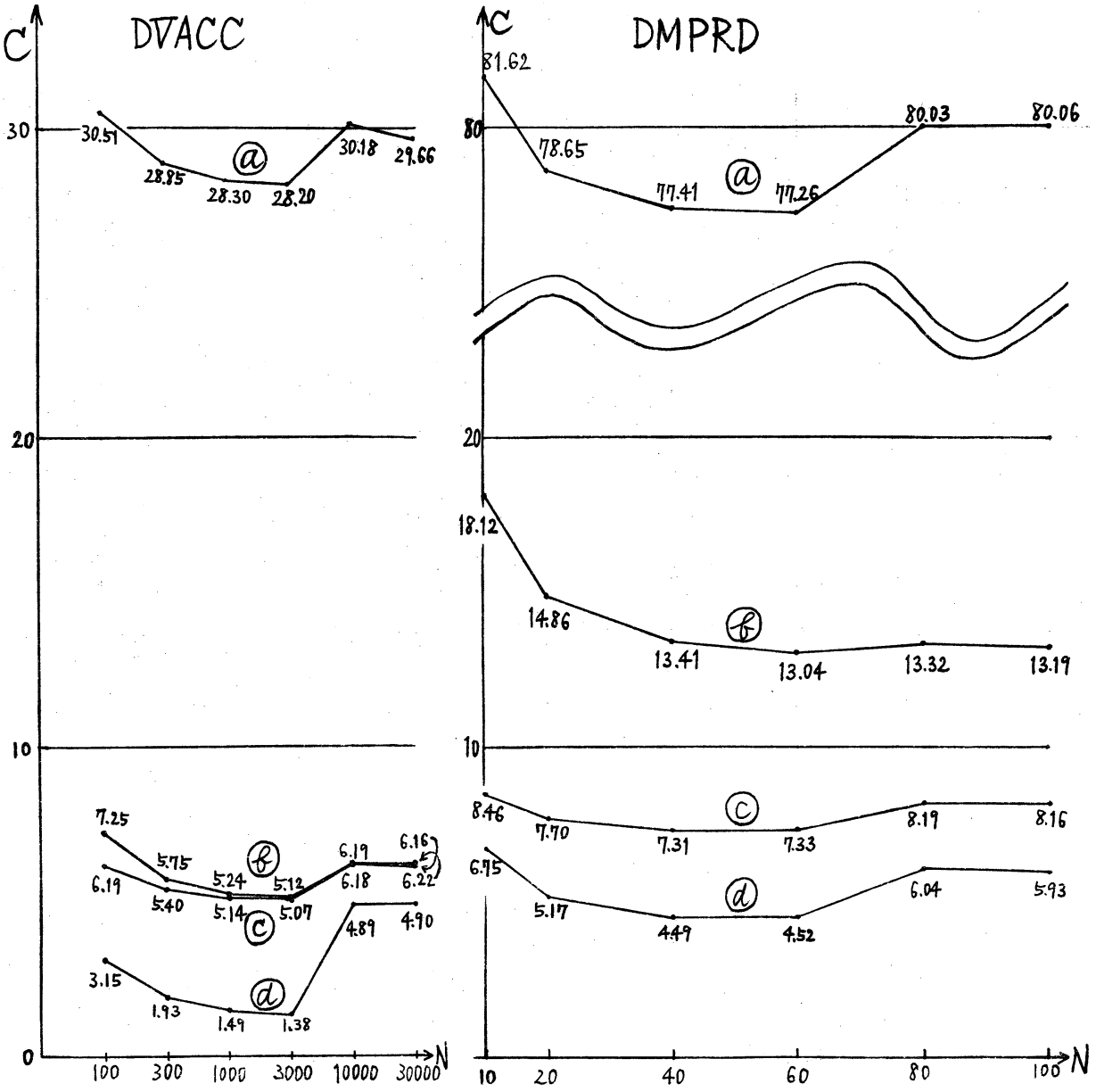
```
      SUBROUTINE DVACC(ACC,X,ICT)
      REAL*8 ACC,X(1)
      ACC=0.0D0
      DO 100 I=1,ICT
  100 ACC=X(I)+ACC
      RETURN
      END

      SUBROUTINE DMPRD(P,X,Y,IDIMP,IDIMX,IDIMY,ICT,JCT,KCT)
      REAL*8 P(IDIMP,KCT),X(IDIMX,JCT),Y(IDIMY,KCT),W
      DO 200 K=1,KCT
      DO 200 J=1,JCT
      W=0.0D0
      DO 100 I=1,ICT
  100 W=X(I,J)*Y(I,K)+W
  200 P(J,K)=W
      RETURN
      END
```

☆Note: 80次の行列の積を求める場合，次のように2度に分けて処理すれば，高速バッファーの機能を活かせる。

```
DIMENSION P(80,80),X(80,80),Y(80,80)
CALL DMPRD(P     ,X     ,Y,80,80,80,80,40,80)
CALL DMPRD(P(41,1),X(1,41),Y,80,80,80,80,40,80)
```

☆ 単位演算当りの平均所要サイクル数C

[ⓐ: FORTRAN Gコンパイラーでコンパイルしたもの。
ⓑ: FORTRAN Hコンパイラーでコンパイルしたもの。
ⓒ: HISARP version 1
ⓓ: HISARP version 2]



以上