3G－8

行列演算用言語 LAMAX－Sの開発
－実現方法－

内田智史[1] 井上美明[2] 田中和彦[2]
八巻直一[2] 本郷茂[3] 間野浩太郎[1]

1) 青山学院大学理工学部 2) システム計画研究所 3) 専修大学経営学部

1 はじめに

LAMAX-S の目標の1つは、各社のスーパコンピュータ上でソースプログラムの何の修正も無しにある程度の実行効率を保つことである。FORTRAN でプログラムを作成した場合、FORTRAN コンパイラの自動ベクトル化の機能を利用することになるので、当然、個々のスーパコンピュータに対するチューニングが必要となって来る。LAMAX-S では、そのチューニングを行列演算式から LAMAX-S 処理系が自動的に行って、各社スーパコンピュータ向きに最適化された FORTRAN ソースコードを生成する。これは、LAMAX-S がプリプロセッサであるがゆえに比較的容易に行える。各社スーパコンピュータ向けに最適化されたコードを生成するために、LAMAX-S は、マシン固有のハードウェアとソフトウェアのデータベースを持つ。プリプロセッサはこれらのデータベースを参照しながら、より高速なコードを生成する。また、LAMAX-S は、行列演算（連立方程式解法、ＬＵ分解、スパース行列の扱いなど）に対して積極的にメーカ提供のライブラリを使用する。メーカ提供のライブラリはそのスーパコンピュータ上で最も効率よく動作するようにチューニングされていると思われるからである。図1 に本処理系の概要を示す。

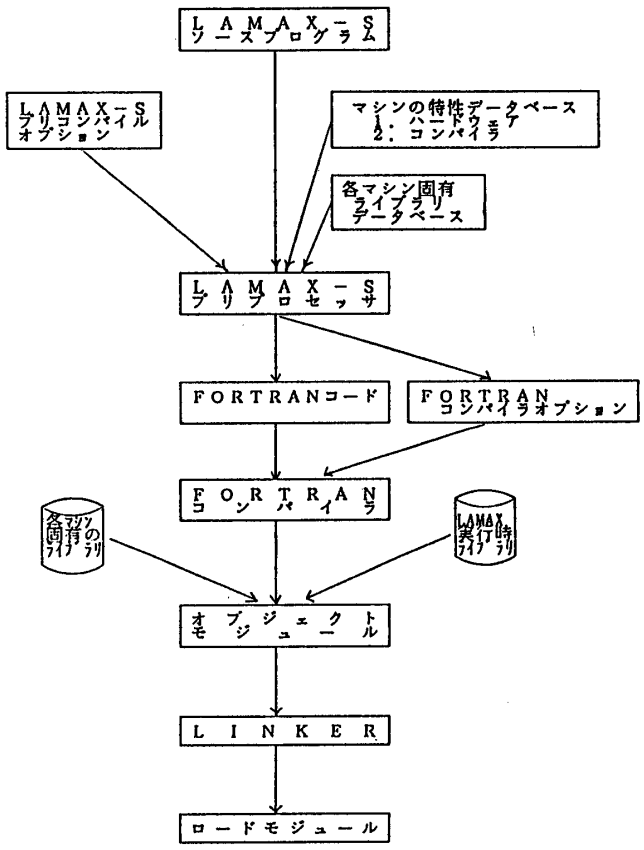

図1 LAMAX-S システムの流れ

2 LAMAX-S とスーパコンピュータの適合性

各社のスーパコンピュータは、それぞれ異なるハードウェア、異なる FORTRAN コンパイラ、チューニングツールを持っている。それぞれのコンピュータごとに異なるチューニング作業を施さなければならない。チューニング作業の1つの例として、行列の乗算のアンローリングがある（これは、スーパコンピュータ登場以前にもさかんに用いられた方法である）。普通に乗算のプログラムを書けば、次のように記述されるであろう。

```
      do 10 k=1,L
       do 10 j=1,N
         do 10 i=1,M
           A(i,j)=A(i,j)+B(i,k)*C(k,j)
10    continue
```

アンローリングを用いると、

```
      L1=(L/4)*4
      do 10 k=1,L1,4
       do 10 j=1,N
         do 10 i=1,M
           A(i,j)=A(i,j)+B(i,k)*C(k,j)
     -                  +B(i,k+1)*C(k+1,j)
     -                  +B(i,k+2)*C(k+2,j)
     -                  +B(i,k+3)*C(k+3,j)
10    continue
      ..... 余った部分は別に処理 .....
```

The Implementation of LAMAX-S(LAnguage for MAtriX calculation on Supercomputer) - the organization of LAMAX-S preprocessor -

Yoshiaki Inoue[1], Kazuhiko Tanaka[1], Satoshi Uchida[2]
Naokazu Yamaki[1], Shigeru Hongo[3], Koutarou Mano[1]
1)Research Institute of Systems Planning 2)Aoyama Gakuin Univ.
3)Senshu Univ.

という記述になる。この技法がわりと効果をもたらすが、何分割すれば良いか等はスーパコンピュータによって異なっている（この例では4分割）。ある種のスーパコンピュータの場合には、30分割程度が最も高速になるという実験結果も得られている。しかしながら、30分割というコーディングは人間の能力の範囲をはるかに越えている。しかし、LAMAX－Sであるならば、30分割をしたFORTRANプログラムを生成することは容易なことである。ある種のチューニングを施すためには、数多くの非人間的なコーディングを迫られる場合があるが、LAMAX－Sのような処理系であれば、比較的容易にそのようなFORTRANプログラムを生成することが出来る。

3 行列演算を主体とした最適化

3.1 抽象度を配列から行列レベルに高めることによって実現可能となる最適化

行列演算式は、FORTRANのDO文、配列による表現に比べて、コンパイラにとってはかなり大量の情報を所有している。たとえば、つぎの代入文があったとする。

A = 1/(X'*X)*Y

この式から、次のようなことが分かる。

1. X'*X の計算結果は対称行列となる。これを T とする。対称行列は、メモリを半分しか必要としない。T の計算量も少なくて済む。
2. $A = T^{-1} * Y$ は、 T A = Y という連立方程式を解くことに他ならない。したがって、T を LU 分解して解いた方が速い。そこで、対称行列 T を係数行列とする連立方程式を解く。

LAMAX-Sでは、このような行列演算に対するいくつかの知識を持ち、これらを基にしてFORTRANソースコードを生成する。

3.2 行列の構造による最適化

行列の構造をプログラムに反映させると記憶効率の観点から利点が多い。対称行列は通常の半分のメモリの半分で済む。対角行列になるとその対角要素だけを持てば良いのでさらに少なくなる。スパース行列の場合には、さらに効率が良くなる。一方、行列演算自体の実行効率も良くなる。たとえば、バンド行列とベクトルの乗算などでは、要素のある場所だけ計算すれば良い。また、ある行列が正値や対称である場合もその特徴を活かした処理ができる。通常のFORTRANでは、これらの情報は分断されてしまい、コンパイラの処理のヒントとならない。LAMAX-Sでは、これらの情報を極力活かし、各構造が効率よくプログラム化できるような機構を持つ。

4 ルール主導型展開法によるコード生成

コンパイラはハードウェアの特性情報、ソフトウェア（コンパイラ）の特性情報などをベースにして、さらにはそのコンピュータ上のライブラリの情報、および、そのライブラリで用いられる各種構造のデータ構造（たとえば、バンド行列 B(i,j) は実際にはどの場所へ格納されるかなど）を参照しながら、コードを生成する。実際には、各種の情報は、あるルールパターンとそれに対応するアクションの集合として扱われる。中間形式に変換されたLAMAX-Sのプログラムの切口にそのパターンが合致するかどうか調べ、最もふさわしいアクションを選択する。この概要図を図2に示す。我々は、これをルール主導型展開法と呼んでいる。

5 クラス階層に基づく行列オブジェクトの管理と実行時ルーチン

各社提供のライブラリがすべての構造を満たすわけではない。そこで、LAMAX-S では、その構造に最も近い（別の言い方をすれば、最も少ないコストで変換できる他の）構造に変換して処理する方式を採用している。たとえば、バンド行列の連立一次方程式を解きたいが、それに対応するライブラリが無い場合には、LAMAX-S は、そのバンド行列を密行列に変換する。これを我々は構造変換と呼んでいる。この変換は、図3に示すクラス階層をベースにして行っている。この関係は、異なる構造間の混合演算の際にも利用する。しかし、現在の方式では、構造変換が頻繁に発生すると著しく実行効率に影響が出る。この点については現在考察中である。

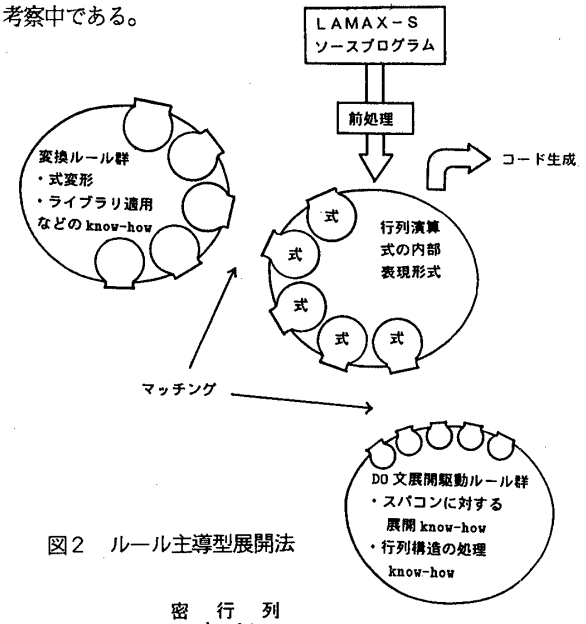

図2 ルール主導型展開法

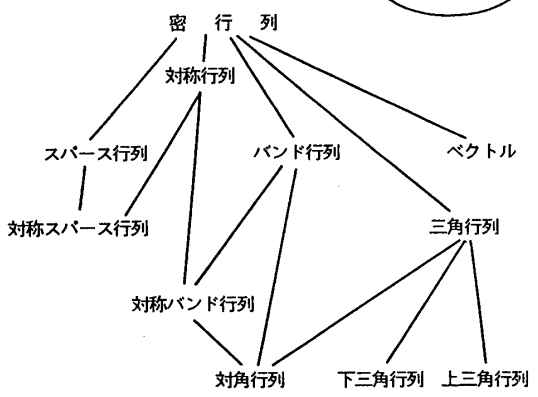

図3 LAMAX-S における行列構造のクラス階層

参考文献 1) 井上, 田中, 内田, 八巻, 本郷, 間野；
行列演算用言語ＬＡＭＡＸ－Ｓの開発 -言語の特徴とその応用
情報処理学会 第４０回全国大会一般講演論文集 (1990.3)