2D－4　分散メモリＭＩＭＤコンピュータにおけるデバッグ環境に必要な機能について

折居茂夫
富士通株式会社スーパーコンピュータシステム部

１．はじめに

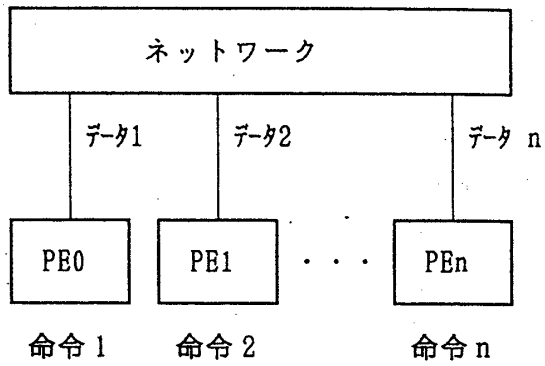
ＣＰＵのゲート遅延時間短縮の限界と，計算に必要なデータ供給を確保するため，分散メモリタイプのＭＩＭＤ方式の計算機（以後，並列計算機と呼ぶ）が今後のスーパーコンピュータの主流になろうとしている〔１〕。そこで本論文では，この方式の並列計算機のデバッグについて議論する。

並列計算機は高性能が達成でき，かつ多量のデータを扱うことができる。反面，非同期で複数の命令が複数データストリームを扱う（図－１）。このため，正常に動作しないプログラムのデバッグをする場合，同期を取って再現性を確保し，全てのデータストリームを調べる必要が生じる。このことは，ベクトル計算機に比べてデバッグ作業量が増加することを意味する。

デバッグの一般的方法の一つは，デバッグ用のツールを用いて計算途中の数値を表示することである。表示した数値を解析してデバッグを行う。

この並列計算のデバッグは，データ並列系言語で記述したプログラムに限ると，幾つかの機能を有した場合，簡単に行うことができる。本論文では，そのようなデバッグ環境を作るために必要な機能と，その環境下におけるデバッグ方法について述べる。

On the Functionalities Required for Debug Environment on Distributed Memory MIMD Computers
Shigeo ORII
Fujitsu Limited, Supercomuter Systems Dept.
1-9-3, Nakase, Mihama-ku, Chiba 261, Japan



図－１　分散メモリＭＩＭＤ計算機
ここに，PE: Processing Element。

２．データ並列系言語で書かれたプログラム

データ並列とは「大規模なデータの処理において，データを複数のプロセッサに分散配置し，これらを並列に処理する並列処理モデル〔２〕」で，これを記述する言語は次の３つの機能を持つ。

・データ分割（データを複数のプロセッサに分散配置）
・手続き分割（DO文等の計算の手続きを各プロセッサに分配）
・データ転送（分散配置したデータの再配置）

この言語の特徴は，データ分割の記述が容易にできることである。

データ並列系言語富士通FORTRAN77 EX/VPPで書かれた並列化プログラムを図－２に示す。この言語は，!XOCL 行で並列化の指示を行う。この例ではPROCESSOR 文でPE台数を指定し，GLOBAL文の中で配列要素の分配を各PEに指示し，SPREAD DO 文及びEND SPREAD文で手続き分割をしている。データ転送はGLOBAL変数によって行う。

```
      !XOCL PROCESSOR P(10)          デ
      !XOCL GLOBAL b(/(P))      ←   | 分
            real a(1000), b(1000)    タ割
            ・・・                    部
   ┌─ !XOCL PARALLEL REGION
冗  │  !XOCL SPREAD DO           ─┐ 手▼
長  │        do 1 i=2,1000         │ 続並
実  │           a(i)=b(i-1) + a(i) │ き列
行  │     1 continue               │ 分実
部  │  !XOCL END SPREAD          ─┘ 割行
   └─ !XOCL END PARALLEL             部▲
```

図－２　データ並列系言語で書かれた
並列化プログラム例
（富士通FORTRAN77 EX/VPPで記述）

ここに，PROCESSOR:PEの台数指定,GLOBAL:全てのPEから定義参照可能な変数の指定，b (/(P)): 10台のPEでデータ分割，PARALLEL REGION: 同じ実行を10台のPEで行う領域,SPREAD DOとEND SPREAD: 10台のPEで並列計算する範囲指定。

3．デバッグ

プログラムが実行時にエラーを出した時は勿論のこと，正常終了したとしても計算結果からプログラムが正常に動いていないと判断された時，デバッグが行われる。原因としては次のものが考えられる。並列計算機を使う時，項目(4) が新たに加わる。

(1) コーディング上のケアレスミス
(2) コンパイラによる最適化の副作用
(3) 計算精度不足
(4) 並列計算の設計ミス（データ依存がある計算を並列化した等々。）

4．デバッグ環境に必要な機能と方法

デバッグを行うためには，バグの箇所を同定する必要がある。ここで，通常のデバッグツールを使用する代わりに，デバッグ箇所の並列処理を逐次処理に変更して同定する方法を提案する。

FORTRAN77 EX/VPPで例を示すと，図－２の!XOCL SPREAD DO 文と，!XOCL END SPREAD文を削除することにより，逐次化が可能となる。この方法は，並列処理と逐次処理の計算結果の一致のみを確認して行うため，デバッグの時間を大幅に短縮できる。

これを適用できる条件は，プログラムが逐次実行可能に作られていることである。この条件の下で次の機能があると，このようなデバッグ環境を実現できる。

a.プログラムの一部に対して，並列・逐次実行を指定できる機能
b.任意の位置で同期をとる機能
c.高速に逐次実行を実現する機能

VPP500を例にとれば，図－２において，項目a.は!XOCL SPREAD DO 文と，!XOCL END SPREAD 文が対応する。項目b.はこれらの文の要素として記述することができる。項目c.に対して，全てのPEが同じ計算を行う「冗長実行」によって高速に達成できる。この冗長実行では全てのPEが同じ処理を行う。

5．まとめと議論

分散メモリ型MIMD計算機において，簡単に並列計算のデバッグを行う方法と必要な機能について述べた。並列計算のデバッグは，一般には困難であるが，DOループを並列化した場合このような機能によって比較的簡単に行うことができる。現在のスーパーコンピュータで扱われる数値計算は主にDOループで時間を費やすため，このようなデバッグは有意義と考える。
尚，逐次実行を現実的に行うためには，1PE の性能が高いことも必要である。

〔文献〕

〔１〕例えば，馬場：超並列マシンへの道，情報処理Vol.32,No.4, pp.348-364 (1991)
奥田：並列計算機と流体解析，機械の研究vol.45, No.1, pp.116-121 (1993)
〔２〕Hillis, Steele: Data Parallel Algorithms, Comm, ACM, Vol.29, No.12, pp.1170 -1183 (Dec. 1986)