1E－7

分散メモリ型並列計算機向けの自動データ分散並列化技術の検討

西谷 康仁†, 太田 寛‡, 布広 永示†, 菊池 純男‡
†(株) 日立製作所ソフトウェア開発本部, ‡新情報処理開発機構

1 はじめに

分散メモリ型の並列計算機において効率の良い並列実行を行なうためには、各プロセッサでのデータ参照がローカルになるようにデータを配置することが必要である。従来の分散メモリ向き並列化コンパイラでは、ループの並列化は自動的に行なえるが、データの分散はユーザーが指定しなければならず、このことが並列計算機向けのプログラムの開発を困難にしていた。この問題を解決するため、自動データ分散技術に関する多くの研究 [1][2] がなされているが、未だ実用化レベルには至っていない。

本稿では、自動データ分散方式の検討について述べる。

2 構成

自動データ分散コンパイラの構成を図1に示す。本処理系は、配列データの分散形状を自動的に決定する自動データ分散決定部と、決定したデータ分散形状に基づいてプログラムを並列化するプログラム並列化部からなる。

データ分散決定部は、プログラムの構造やデータフローを解析して、並列実行に適したデータ分散形状を自動的に決定する。

プログラム並列化部は、以下のような機能を持つ。

- 再分散解析:手続きの入口、出口での再分散や、ユーザー指定により行なわれる再分散について、再分散位置での分散形状を解析し、再分散処理を生成する
- ループ並列化解析:ループの並列化可否判定を行ない、ループのイタレーションを各プロセッサに分散する
- 通信解析:並列化したループに対し、通信の必要性を判定して、必要であれば通信を生成する

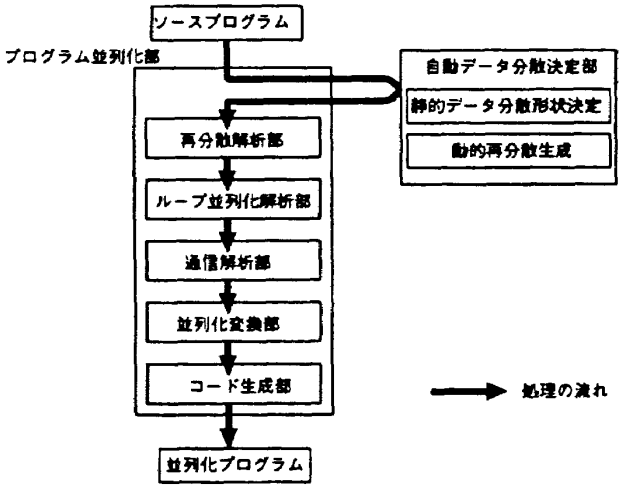

図 1: 自動データ分散コンパイラの構成

- 並列化変換:解析情報を基に、並列化変換を行なう
- コード生成:ソースコードを出力する

プログラム並列化部は、HPF コンパイラ (トランスレータ) [3] と同じ構成、機能を持つ。自動データ分散決定部が加わることによって、HPF 指示を含まない単なる逐次プログラムから、自動的に並列化されたプログラムを生成することが出来るようになる。

3 処理方式

自動データ分散決定部の処理について説明する。自動データ分散決定部は、次の2つのフェーズより構成される。

3.1 静的データ分散形状決定

配列毎に、複数のデータ分散パターンの候補の中から、あるプログラム単位にわたり最適となるデータ分散パターンを選択する。選択の方針は以下の通りである。

(1) 配列毎に、候補となる分散形状の集合を定める。候補集合の爆発的増加を防ぐため、分散形状は1次元のブロック分散のみとする。

(2) ループの分散は OCR(Owner Computes Rule) に基づいて行なわれるので、その配列の定義が現わ

Design of automatic data distribution and parallelization technique for distributed-memory multicomputers.

Yasunori NISHITANI†, Hiroshi OHTA‡, Eiji NUNOHIRO†, Sumio KIKUCHI‡
†Software Development Center, Hitachi Ltd.,‡Real World Computing Partnership

れるループについて、各候補を仮定した時の並列化ループレベル及び通信コストを見積もる。
並列化ループレベルとは、データ分散形状の候補を仮定した時に並列化されるループレベルであり、並列化効率の尺度として用いる。効率の良い並列化とは、より外側ループで並列化されることとする。通信コストは、通信量及び通信回数を元に決定する。

(3) 各ループに対して求めた並列化ループレベル及び通信コストに対して、対象プラットフォームの特性を考慮した重み付けを行なう。

(4) 各ループに対して求めた並列化ループレベル及び通信コストを、分散形状の候補毎に集計して、プログラム単位で基準となる静的データ分散形状 を決定する。

3.2 動的再分散生成

実プログラムでは、プログラムの特性に応じて、データ分散形状を動的に変更することにより実行効率が向上する場合がある。このため、動的再分散処理を自動的に挿入する。具体的には、

(a) 3.1で定められた静的分散形状でループを実行した場合の実行時間

(b) 3.1で選択されなかった分散形状の候補に対する実行時間+再分散2回分 (分散形状の変更の再分散と、元に戻すための再分散) の実行時間

を比較する。実行時間は、3.1で求めた通信コストや、ループ中の演算量から見積もる。その結果、(b) の方がループを効率良く実行できると判断した場合に、配列を動的に再分散する命令を挿入する。

4 例

図2のようなプログラムを例に、上記処理方式がどのように適用されるかを説明する。

静的データ分散形状決定フェーズでは、まず分散形状の候補集合を定める。HPF の記法を用いると、配列 x の分散形状について、(BLOCK,*,*), (*,BLOCK,*), (*,*,BLOCK) の3つの候補が定まる。それぞれについて各ループでの並列化効率を見積もる。すると、DO 10,DO 20 のループについては (*,*,BLOCK) の分散が最も効率が良く (外側ループで並列化した方が良い)、DO 30 のループについては (*,BLOCK,*) の分散が良いと判定する。その結果、静的データ分散形状として (*,*,BLOCK) の分散を採用する。

次に、動的再分散生成フェーズでは、DO 10,DO 20

```
      REAL x(N,N,N),c(N,N,N)
      DO itr=1,MAXITR
        DO 10 k=1,N
          DO 10 j=1,N
            DO 10 i=2,N
              x(i,j,k)=x(i,j,k)-x(i-1,j,k)*c(i,j,k)
10      CONTINUE
        DO 20 k=1,N
          DO 20 j=2,N
            DO 20 i=1,N
              x(i,j,k)=x(i,j,k)-x(i,j-1,k)*c(i,j,k)
20      CONTINUE
        DO 30 k=2,N
          DO 30 j=1,N
            DO 30 i=1,N
              x(i,j,k)=x(i,j,k)-x(i,j,k-1)*c(i,j,k)
30      CONTINUE
      ENDDO
```

図2: サンプルプログラム

のループについては何もしない。DO 30 のループは、3次元目を分散すると、通信を伴う DO ACROSS 型の並列実行となるが、再分散を行なって2次元目を分散すると、通信が無くなって完全な並列実行が行なえる。どちらが速いかは、DO ループ中の計算量や対象とするプラットフォームの特性に大きく依存する。コンパイラは、与えられたパラメータに従ってどちらが速いかを判定し、必要であれば再分散処理を DO 30 ループの前後に挿入する。

5 まとめ

最適なデータ分散を自動的に行なう並列化コンパイラの処理方式について検討した。

今後は、本方式を広範囲な実プログラムに適用し、その妥当性について検証を行なう。また、手続き間での本方式の適用についても検討していく予定である。

参考文献

[1] K.Kennedy and U.Kremer. Automatic Data Layout for High Performance Fortran. In *Proceedings of Supercomputing'95*, San Diego, CA, December 1995.

[2] J.Garcia, E.Ayguadé, and J.Labarta. A Framework for Automatic Dynamic Data Mapping. In *Proceedings of the 8th IEEE Symposium on Parallel and Distributed Processing*, October 1996.

[3] 佐藤真琴, 太田寛, 布広永示. HPF トランスレータ "Parallel FORTRAN" の開発と評価. 情報処理, Vol. 38, No. 2, pp. 105–108, 1997.