長方形メッシュによるデータ分割のアプローチと評価*

1J－4

李 暁傑　　原田 賢一†
慶應義塾大学 計算機科学専攻 横浜市港北区日吉 3-14-1

概要

分散共有メモリ型並列計算機におけるプログラムの実行効率を向上させるための課題の１つに、データの分割と配置の問題（**分割配置**）がある。その結果によって、実行対象となるプログラムの並列性、通信オーバヘッド、および負荷分散が影響される。本稿では、データのアクセス パターンを記述するためのステンシル構造 (stencil structure) をもとに、階層メモリをもつ分散メモリ型マシンを対象として、メモリ アクセスに要するプロセッサ間の通信量を定式化し、通信量を最小にするデータの分割配置法を提案し、この方法の有効性を評価する。

1　まえがき

ハードウェア技術の進歩により、多様なメモリ システムをもつ中規模並列マシンが商用化され、さらに大規模超並列マシンの実用化へ向けての模索が行われている。しかし、中規模並列マシンの場合でも、階層メモリをもつ並列マシンに対して、効率の良いプログラムを作成しようとすると、共有メモリ マシンに比べて、解決しなければならない問題がある。その１つにデータの分割配置の問題がある。すなわち、１つの配列を分割仕方と、各プロセッサへの割り当て方が問題となる。配列の**分割** (partitioning) の仕方によって、プロセッサ間の通信量が増減し、さらに、メモリ階層によってアクセス速度が異なるために、分割された配列の領域をプロセッサの階層メモリにどのように**配置** (distribution) するかによって、プログラムの実行効率を左右する。

一般に、プロセッサ間でのデータ転送には、プロセッサ間通信が必要となるため、CPU 時間に比べて大幅な時間がかかる。そこで、個々のプロセッサによって頻繁にアクセスされるデータ (配列の一部に対する領域) を特定できれば、その部分をローカル メモリ、あるいはキャッシュ メモリに配置することによって、プロセッサ間での通信量を減少させることができる。本稿では、これまでの種々の技法に基づいて、階層メモリをもつ並列マシンに適用可能なデータの最適分割配置法を提案する。この方法は、従来の研究に比べて、次の特徴をもつ。

1. 最適なデータ分割配置は、繰返し空間に基づくデータの分割と、各プロセッサの階層メモリに対するデータ配置の２フェーズからなる。
2. ３レベルのメモリ、すなわち、キャッシュ＋ローカル メモリ＋リモート メモリをもつマシンに対して、最適な分割配置を与える。このほかに、ローカル メモリ＋リモート メモリ、キャッシュ＋共有メモリに対しても適用が可能である。
3. 通信量を求めるために、**アクセス ベクトル**を採用する。これはデータアクセスパターンを記述ものである。具体的なアーキテクチャに依存しないため、多くの分散メモリマシンにこの技法を応用することができる。

2　最適分割配置

本研究では、MIMD 型の階層メモリをもつ並列マシンを対象とし、各プロセッサのメモリは、キャシュ、ローカル メモリ、およびリモート メモリから構成されるものと仮定する。

分散配置の対象を２次元配列 A と B とし、それらは、次に示す形のループによってアクセスされるものと仮定する。ここで、A はデータ依存をもつ配列を、また B はデータ依存のない配列を意味する。分割配置の目標は、並列ループの実行にあたって、各メモリ階層に配置されるデータへのアクセス時間を最小化することである。

```
DO K = 1 , M
DO I = 1 , N1
DO J = 1 , N2
A(I, J) = F(A(I+a1, J+b1), A(I+a2, J+b2), ... ,
          A(I+af, J+bf)) + G(B(I+c1, J+d1),
          B(I+c2, J+d2), ... , B(I+ch, J+dh))
ENDDO
ENDDO
ENDDO
```

上に示した形式のループを**拡張された逐次反復並列ループ** (ESIL: extended sequentially iterated parallel loop) と呼ぶ。

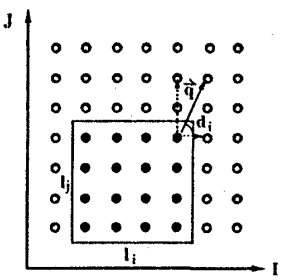

図 2.1 単一アクセス ベクトルと分割領域

分割領域の各境界を**セグメント** (segment) と呼ぶ。長方形分割における水平および垂直セグメントを、それぞれ l_i と l_j で表す。各セグメントについての通信量は、セグメントが１単位の長さだけ延びると、通信量がどの程度増えるのかに

*An Approach to the Rectangular Partitioning and its Evaluation

†Xiaojie LI and Ken'ichi HARADA, *Department of Computer Science, Keio University*

着目して定式化する。分割領域を P としたとき、P の通信量 (communication weight) は、セグメント l_i (または l_j) を越えた領域外要素への参照回数として捉える。

アクセス ベクトルを $q = (a, b)$、分割領域 P のセグメントを l_i と l_j とする。l_i についての各方向の単位通信量を、それぞれ u_i, u_j とすると、$u_i = q \times \sin d_i = b$、$u_j = q \times \sin d_j = a$ である。このとき、P 中の要素への代入のために、単一アクセ スベクトル q によって引き起こされる通信量 C_P は、(1) 式 によって与えられる。

分割領域全体での通信量は、2 つのセグメントについての通信量の和として表されるが、隅の部分にある要素を重複して数えるために、その部分の通信量 $u_i u_j$ を除く必要がある。

$$C_P = l_i u_i + l_j u_j - u_i u_j. \tag{1}$$

ここで、式中のセグメントはその長さを表すものとする。ベクトル集合の場合、集合 $\{q_1, q_2, \ldots, q_k\}$ の各ベクトル q_h を (a_h, b_h) とする。このとき、ベクトル分解によって、水平分解量の集合 $\{a_1, a_2, \ldots, a_k\}$ と、垂直分解量の集合 $\{b_1, b_2, \ldots, b_k\}$ とに分ける。この場合、単位通信量 u_i および u_j は、それぞれ次の式で与えられる。

$$u_i = \sum_{h=1}^{k} |b_h| \qquad u_j = \sum_{h=1}^{k} |a_h| \tag{2}$$

定理 1. 与えられたアクセスベクトル集合 S についての単位通信量を u_i、u_j とすると、セグメント l_i と l_j が次の条件を満たすとき、通信量 C_P は最小となる。

$$\frac{l_i}{l_j} = \frac{u_j}{u_i} \tag{3}$$

キャッシュを含む階層メモリ システムにおいては、常に使われるデータをキャッシュにコピーすることによって、データ アクセスの効率をさらに向上させることができる。キャッシュの導入には、記憶域におけるデータの一貫性、すなわち同一の配列要素について、キャッシュに置かれたデータと主記憶域のデータとが一致する保証を与えるようにしなければならない。

ベクトル集合を $S = \{q_1, q_2, \ldots, q_k\}$、その水平分解量を $\{a_1, a_2, \ldots, a_k\}$、垂直分解量を $\{b_1, b_2, \ldots, b_k\}$ とする。このとき、a^+、b^+、a^-、および b^- を a_i、b_i の最大値と最小値と定義する。

定理 2. プロセッサ p に対する分割領域を $\rho(p) = [\, i_1 : i_2, j_1 : j_2 \,]$ とし、その分割境界を l_i と l_j とする。このとき、上で定義した各集合は、次の関係から求められる。

1. ERW(Exclusive Read and Write) 集合は、$[\, i_1 + a^+ : i_2 - a^-, j_1 + b^+ : j_2 - b^- \,]$ の要素からなる。
2. SREW(Shared Read and Exclusive Write) 集合は、$\rho(p)$ − ERW $= [\, i_1 : i_2, j_1 : j_2 \,] - [\, i_1 + a^+ : i_2 - a^-, j_1 + b^+ : j_2 - b^- \,]$ の要素からなる。
3. SRNW(Shared Read No Write) 集合は、$[\, i_1 - a^- : i_2 + a^+, j_1 - b^- : j_2 + b^+ \,] - [\, i_1 : i_2, j_1 : j_2 \,]$ の要素からなる。
4. NRW(No Read No Write) 集合は、N1 × N2 − (ERW ⋃ SREW ⋃ SRNW) からなる。

簡単なキャッシング技法としては、ERW をキャッシュに配置すればよい。ERW は、他のプロセッサから参照されることがないので、キャッシュとローカル メモリとの間で一貫性を保証するためのデータの入換えは必要がない。並列実行が始まる前に、適当なローカル配列を確保して、それをキャッシング可能と宣言し、その中に ERW を格納すればよい。SREW は、ローカル メモリに置き、SRNW は、リモート メモリに置く。

3 分割配置法の評価

並列マシン ASPIRE を用いて、ここで提案した方法の有効性を実測評価した。

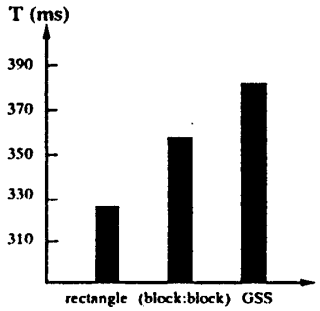

図 3.1: 典型的な分割法との比較

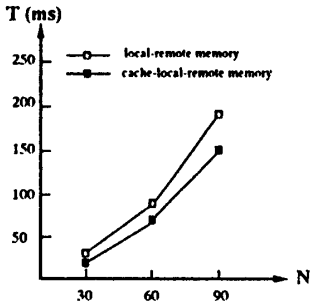

図 3.2: 異なるメモリ システムを用いた実行時間の比較

4 むすび

効率の良い並列プログラムを開発するための手段として、並列コンパイラにおけるデータの自動分割配置が望まれている。本研究では、プロセッサ間の通信量を最小化し、さらに階層メモリの特徴を活かしたデータの最適分割配置法を提案した。

References

[1] M. Gupta and P. Banerjee, *"Demonstration of Automatic Data Partitioning Techniques for Parallelizing Compilers on Multicomputers,"* IEEE trans. Parallel and Distributed System, Vol.3, No.2(March 1992), pp.179-193.

[2] X. Li and K. Harada, *"An Optimal Mapping of a Global Array to Hierarchical Memory Systems with Three Levels,"* in Proc. of International Symposium ISPAN'94, pp.67-74, December, 1994.