ハイパフォーマンス
コンピューティング 46−10
（１９９３．４．２６）

高並列計算機におけるデータ並べ替えの最適化

塚田 晶宇　津田 孝夫
京都大学工学部情報工学教室

疎結合高並列計算機のデータ並べ替えに、二階層記憶の並べ替えモデルである Floyd モデルを応用する。このモデルを用いて、並べ替えに要する通信フェーズ数の下界を導く。通信フェーズ数の下界は、Floyd モデルを適用した場合の入出力の下界に比べ PE 台数分の１である。特に配列データの転置の場合について、通信フェーズ数の下界を満たすアルゴリズムを示し、そのアルゴリズムの転送時間を求める。富士通 AP1000(64PE) で転送時間の測定を行い、モデルから導かれる結果とほぼ一致することが確認された。

Optimizing Data Permutation on Highly Parallel Computers

Akihiro Tsukada　Takao Tsuda
Department of Information Science, Faculty of Engineering, Kyoto University

A permuting model of loosely coupled parallel computer is proposed to get the lower bound of communication phases. Introducing parallelism in Floyd's model, the lower bound on the proposed model is one p'-th of that on the original model, where p' is the number of PEs. An algorithm of matrix transposition which satisfies the least communication phases is also proposed. Its communication time is computed on two kinds of network topology—hyper-cube and AP1000-like torus, and is measured on the Fujitsu AP1000(64PE). The result gives roughly good agreement with the result.

1 はじめに

疎結合の並列計算機では、プロセッサ・エレメント (以下 PE) 間でデータの授受を行う必要があり、応用プログラムに用いられる通信アルゴリズムが実行に大きな影響を与える。とくに頻繁にデータの交換が行なわれる典型として、行列の転置等のデータ並べ替えを挙げることができる。データ並べ替えは多くの科学技術計算に必要な処理であり、データ並べ替えの最適化は、計算の実行時間に大きな影響を与える重要な課題の一つである [1]。

そこで以下では疎結合並列計算機のデータ並べ替えのモデルを提案し、並べ替えに必要な通信フェーズ数の下界を導く。モデルの基礎となるのは、二階層記憶でのデータ並べ換えのモデルである Floyd のモデル [2][3] である。また通信フェーズ数の下界を満たす 2 次元行列転置のアルゴリズム paraFLOYD を提案し、それによる paraFLOYD の転送時間を求める。

通信フェーズの最適化は通信時間の最適化と密接に関連するため、paraFLOYD アルゴリズムは行列転置の高性能なアルゴリズムとして利用できる。

2 Floyd のモデル

Floyd のモデル [2][3] は、二階層記憶において二次記憶上のデータを並べ換えるモデルである。このモデルでは、二次記憶と一次記憶との間のデータ転送は一定数のデータからなるページを単位として行なわれる。並べ替えは 'ページ更新' を繰り返すことによって行われる。'ページ更新' とは、二次記憶から任意の 2 ページを一次記憶に読み込み、一次記憶内でデータを並べ替え、もとの 2 ページへ書き戻すという操作である (図 1 参照)。

Floyd は、データの分布状態に対し目標とする分布に近いほど大きな値をとる関数 V を定義し、1 回の 'ページ更新' による V 関数の増加の最大値を導くことで、必要な 'ページ更新' の数の下界を求めている。

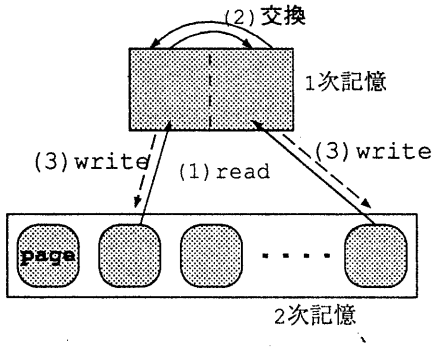

図 1: FLOYD モデルにおける 'ページ更新'

3 疎結合並列計算機のデータ並べ替えモデル

以下のような疎結合並列計算機のモデルを提案する。モデルは p′個の PE の集合と、任意の 2PE 間を結合する結合網からなる。

各 PE に分散して格納されているデータを並べ替える。各データには最終的な行先の情報が付加されているか、PE 内での位置などから計算によって求めることが可能であるとする。また PE 内での並べ換えのコストは無視できるものとする。

データは図 2 に示すような 'PE ペアによるデータ交換' によって並べ替えられる。任意の 2PE でペアを組むことができ、複数の PE ペアによって並列に交換を行うことが可能である。ただし一つの PE は同時に複数の PE ペアに属することはできない。これらより、並列にデータ交換ができる PE ペアの数は高々$\lfloor \frac{p'}{2} \rfloor$である (図 2)。この並列に行われるデータの交換を 1 フェーズとする。

ここで、 PE ペアによるデータ交換が 1 フェーズにつき 1 つのペアしか許されていない制限されたモデルを考えると、Floyd モデルでの情報交換との間で表 1 のように対応させることができる。提案したモデルが並べ替えに要するフェーズ数の下界は、制限されたモデルが同じ並び換えに要するフェーズ数の下界の少なくとも $\frac{1}{\lfloor \frac{p'}{2} \rfloor}$ 倍である。以上をもとに、並べ替えに

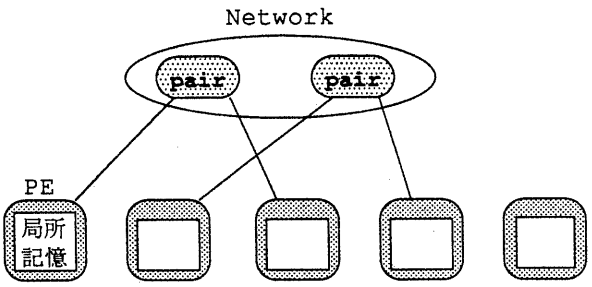

図 2: データ並べ換えのモデル

表 1: FLOYD モデルとの対応

FLOYD のモデル	並列計算機モデル
主記憶	PE 間のリンク
2 次記憶のページ	PE(の local memory)
'ページ更新'	PE のペアによる交換

必要な通信フェーズ数の下界を求める。

まず Floyd モデルと同様にして V関数を定義する [2]。e 関数は V関数の定義のための補助関数である。V関数は、データの分布状態 S が最終的に目標とするデータ分布 Bに近いほど大きな値をとる。

最終状態 Bと初期状態 A での V関数の値が決まっているので、$V(B)-V(A)$ は定数となる。よって 1 フェーズにおけるデータの交換方法として、V関数の増加が最大となるような方法を採用すれば、必要なフェーズ数は最も少なくなる。1 フェーズによる V関数の増加は、ペアの選び方と各ペアでのデータ交換の方法に依存するが、各ペアでのデータの交換方法は Floyd モデルの結果からペアを組んでいる 2 つの PE にそれぞれ存在する同一目的先の (2 つの) データを一つの PE にまとめるように交換する方法が最適である。この時の V関数の増加の最大値は高々2s であり、ペアの選び方に依存しない。ここで s は各 PE の持つデータの数である。

以上より 1 フェーズにおいてできるだけ多くのペアを選び、各ペアでは 2 つの PE にそれぞれ存在する同一目的先のデータを一つの PE にまとめるように交換する時、1 フェーズによる V関数の増加が最大となることがわかる。

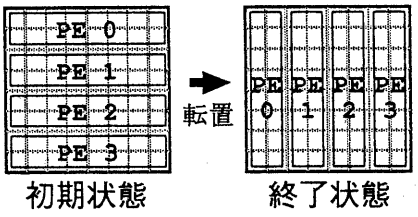

図 3: 正方行列の転置

定理 1
1 フェーズによる V関数の増加の最大値は $p's$ である。ここで、p'は P システムの PE の数、s は 1 PE 当りのデータ数である。

上の定理より、以下の 2 つの定理が成り立つ。

定理 2
任意のデータ並べ替えに要するフェーズ数の下界は $\{V(B)-V(A)\}/(p's)$ である。

定理 3
各 PE の持つデータが、異なる s 個の PE を目的先とするなら、その並べ替えに要するフェーズ数の下界は $\lceil \log_2 s \rceil$ である。

以下簡単のため、p'、s は 2 の冪乗であるとする。一般の場合には、以下に現れる $\log_2 \cdots$ を $\lceil \log_2 \cdots \rceil$ で置き換えれば良い。

4 行列転置の場合

定理 3 を用いて、図 3 のように行単位で PE に保持された正方行列の転置に必要な通信フェーズ数の下界を求める。なお並べ替えの際に各 PE の PE 番号を付け換えても良いとする。$p' \times p'$ 行列の各行を各 PE に分配した場合、定理 3 より ($s = p'$として)、その転置に要するフェーズ数の下界は $\log_2 p'$となる。行列が $N \times N$で $N = kp'$ $(k \in \mathcal{N})$ のときは、$k \times k$ の小行列を 1 つのデータとして扱えば、$p' \times p'$

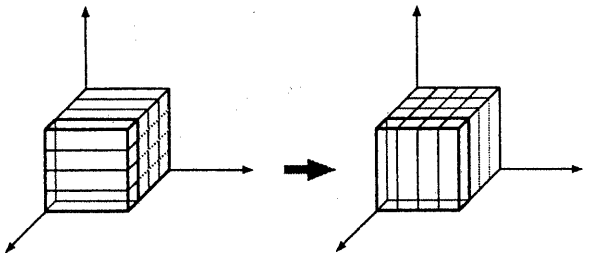

図 4: 3 次元行列の配列転置の様子 ($p' = 16$)

の行列と同様に扱える。

多次元配列の転置の場合でも、データ分配の仕方によっては 2 次元の転置を並列に行うように扱うことができる (図 4参照)。

5 フェーズ数の下界をみたすアルゴリズム

2 次元の正方行列転置の場合のフェーズ数の下界をみたすアルゴリズムの例を示す。基本的には、二階層記憶のデータ並べ替えアルゴリズムである $d\log_w d$ 法 [4] と同様の、radix sort の応用である。このアルゴリズムの存在により、行単位で保持された正方行列の転置については、モデルより導かれるフェーズ数の下界は厳密 (tight) であることがわかる。

アルゴリズム *paraFLOYD*
  cid := 自分の PE番号; /* $0 \ldots p' - 1$ */
  for $i := 1$ **to** $\log_2 p'$
    do
      $data\_to\_send$:= { データ | 行先 PE番号の下 i 桁目 (2^{i-1}の位) が自分と異なる };
      send $data\_to\_send$ **to** $cid \oplus 2^{i-1}$;
      receive from $cid \oplus 2^{i-1}$;
    od

第 i フェーズの終了直後には、PE 番号 cid のローカルメモリ内にあるすべてのデータは、目的行先 PE 番号の下 i 桁までが cid の下 i 桁までと同じである (図 5 参照)。

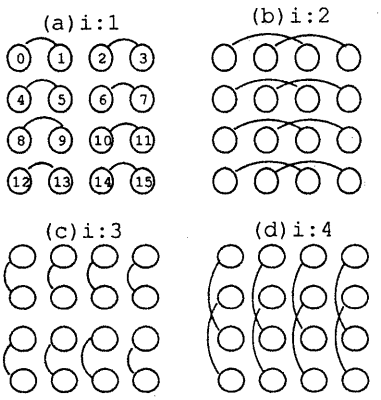

図 5: paraFLOYD の交換の様子

6 行列転置にかかる転送時間

以下では、paraFLOYD アルゴリズムを実際の疎結合並列計算機で実行した場合の転送時間について考察する。

リンクは単方向 (半二重) とし、各 PE では同時には 1 つのメッセージの送信と 1 つのメッセージの受信しかできない 1 ポート通信とする。

ネットワークトポロジとしてハイパーキューブ [5] と富士通 AP1000 型のトーラス [6] の 2 つの形態を考える。

AP1000 型トーラスは 2 次元の正方トーラスで、ハードウェアによりルーティングが行われる。メッセージの転送時間は転送する PE 間の距離 (通過するリンクの数) とは独立である。ルーティングはまずトーラスの東西方向に行われ、次に南北方向に行われる。あるリンクが d 重に競合したら、リンクの見かけのバンド幅が $\frac{1}{d}$に下がったとみなすことができる。PE 番号の付け方は Raster scan labeling とよばれる方法で、図 5(a) のように行う [5]。

この 2 つのモデルそれぞれについて paraFLOYD の転送時間を求める。以下では次の記号を使用する。

- 1 data : { 行列要素 | 同一 PE 内にあり、最終の行先 PE が同じ }
- B: リンクの転送速度、バンド幅 (data/s)

- α: 1 メッセージの転送にかかるオーバーヘッド時間
- 1 cycle：競合がない時に、単位となる大きさの 1 つのメッセージがリンクを通るのに必要な時間
  単位メッセージ長が m data ならば、1 cycle= $\frac{m}{B}$(s/m data)

6.1 ハイパーキューブの場合

ネットワークトポロジがハイパーキューブの場合、paraFLOYD は contention free で実行できる。また行列転置では、その対称性から PE ペアで交換する data の数は$\frac{p'}{2}$で一定である。よって、その転送時間は

$$\log_2 p' \left(2\frac{\frac{p'}{2}}{B}\right)+(\log_2 p')\alpha = \frac{p'\log_2 p'}{B}+\alpha\log_2 p'$$

となる。

6.2 AP1000 型のトーラスの場合

AP1000 風のトーラスのモデル [5] では paraFLOYD はそのまま実行できるが、競合が生じるため、ハイパーキューブの場合よりも通信時間は大きい。paraFLOYD は normal hypercube algorithms [5] とよばれるクラスに属するので、ハイパーキューブを crossover network でシミュレートし、さらにトーラスでシミュレートすることで、おおよそ $(3\sqrt{p'}-2)$ cycle で実行できる [5]。よって、paraFLOYD をトーラス上で実行する場合、その転送時間は

$$(3\sqrt{p'}-2)\frac{\frac{1}{2}p'}{B}+(\log_2 p')\alpha = \frac{\frac{3}{2}p'^{\frac{3}{2}}-p'}{B}+(\log_2 p')\alpha$$

となる。

行単位で保持された (2 次元) 行列転置について得られた結果を表 2 にまとめる。なお多次元配列の場合であっても 2 次元の場合と同様に扱うことができる。

表 2: 行列転置における結果

トポロジ	フェーズ数	転送時間
Hypercube	$\log_2 p'$	$\frac{p'\log_2 p'}{B}+\alpha\log_2 p'$
ap-Torus	$\log_2 p'$	$\frac{\frac{3}{2}p'^{\frac{3}{2}}-p'}{B}+\alpha\log_2 p'$

7 実測

$N\times N$の行列 (データのサイズは double) を転置する paraFLOYD アルゴリズムの AP1000 型トーラスでの転送時間を、富士通 AP1000(64PE) 上で実測した。測定の結果を図 6 に示す。転送時間 Tは先に $T=\frac{\frac{3}{2}p'^{\frac{3}{2}}-p'}{B}+\alpha\log_2 p'$を導いたが、$B$の単位が (data/s) なのでこれを (byte/s) に換算して

$$B=\frac{B_c}{sizeof(double)\left(\frac{N}{p'}\right)^2}$$

を代入する。ここでB_cはリンクのバンド幅 (byte/s) である。代入の結果

$$T=\frac{sizeof(double)\left(\frac{N}{p}\right)^2\left(\frac{3}{2}p'^{\frac{3}{2}}-p'\right)}{B_c}+\alpha\log_2 p'$$

となる。

PE 数 64 のデータから最小二乗法により B_c、αを計算すると下のようになる。

$$B_c=21.636(\text{MB/s}),\quad \alpha=371.18(\mu\text{-sec})$$

実際の B_cは (25MB/s) であり、モデルから導いた値 $B_c=21.6\cdots$と多少異なる。これは送信に line-send() [6] を用いていないことから、割り込みによるタスクスイッチの時間等 OS の介入によるオーバーヘッドが通信時間に含まれてしまうからであると考えられる。line-send() を使用しなかったのは、行列のサイズが大きい場合にメッセージの受信バッファを確保できないためである。

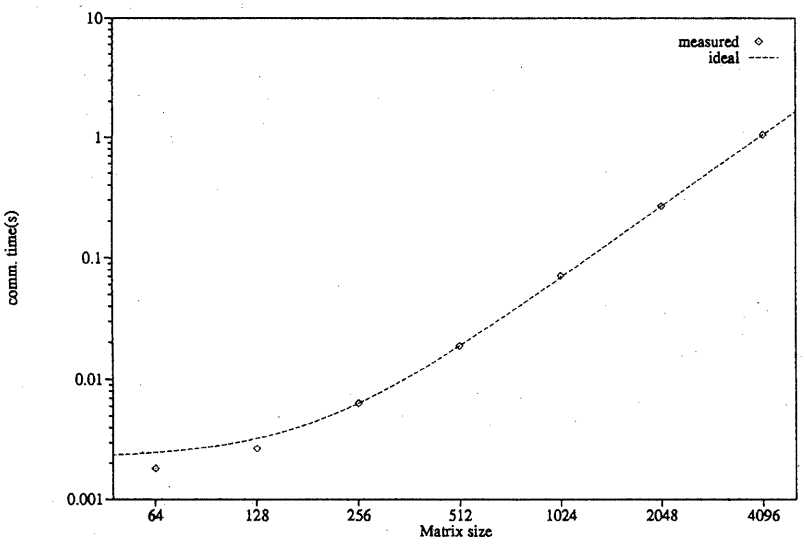

図 6: 測定結果

8 おわりに

本稿では、疎結合並列計算機におけるデータ並べ替え、特に行列転置の最適化を論じた。疎結合並列計算機の行列転置に、FLOYD の情報交換のモデルを応用すると、通信に必要なフェーズ数の下界は $\log_2 p$ である。この下界は厳密 (tight) であり、フェーズ数の下界を満たすアルゴリズム paraFLOYD が存在する。このアルゴリズムは、実際の計算機上でも行列転置のアルゴリズムとして満足のゆく性能を得ることができることを実測により確認した。ネットワーク・トポロジがハイパーキューブ、あるいはトーラスの計算機上では、paraFLOYD の転送時間を各 PE 間リンクのバンド幅 B、PE 数 p'、行列のサイズ Nをパラメータとして計算することができる。

今後の課題として以下のような問題に対し、今回提案したモデルを応用、拡張することが考えられる。

- 行優先で保持された $N \times M$ 行列の転置の最適化
- ソーティング、FFT の PE 間通信の最適化

謝辞

本研究の動機付けとご指導をいただいた津田教授、日頃ご指導ご助言いただく、国枝助教授、岡部助手、ならびに研究室の諸氏に感謝致します。また、適切なコメントをいただいた立命館大学大久保教授、大阪教育大学佐藤隆士助教授に深謝致します。なお実測には、富士通研究所 (株) の AP1000 を利用させて頂きました。

参考文献

[1] David H. Bailey: Experience with Parallel Computers at NASA Ames, RNR Technical Report RNR-91-007, (1991).

[2] R. W. Floyd: Permuting information in idealized two-level storage, in:R. Miller and J. Thatcher, eds., Complexity of Computer Computations, Plenum Press, New York, (1972), 105-109.

[3] Takao Tsuda, Takashi Sato and Takaaki Tatsumi: Generalization of Floyd's model on permuting information in idealized two-level storage, Information Processing Letters, Vol.16, No.4(May 1983), 183–188.

[4] 佐藤 隆士, 津田 孝夫: 2 階層記憶における効率のよいデータ並びかえアルゴリズム, 情報処理学会論文誌, 27-9, (1986-9), 845–852.

[5] I. Chuang, 堀江健志: 高並列計算機 AP1000 のノーマルハイパーキューブアルゴリズムの性能, SWoPP'92–Computer Architecture 95-13, (1992).

[6] T. Shimizu, T. Horie, and H. Ishihata: Low-Latency Message Communication Support for the AP1000, the 19th International Symposium on Computer Architecture, (1992)