バイトニックソートの並列計算機へのマッピング

1S−9

野口泰生、萩原つね子、赤星直輝、武理一郎
富士通研究所

1 はじめに

Batcher[1]により開発されたバイトニックソートは最も高速な並列ソート法の一つである。バイトニックソートは本来はソートネットワークのためのアルゴリズムである。しかし実際には一般の並列計算機を用いて複数の比較器を少数のPEにマッピングし、比較器間の結合をPE間の通信でシミュレートすることで実現されている[2]。特に超立方体結合の並列計算機では比較器間の結合を効率よくシミュレートできるのでバイトニックソートはよく用いられる。バイトニックソートを超立方体結合の並列計算機に移植する場合、従来のマッピングでは1つのPEに1つのソートエレメントを置く(1PE-1エレメントマッピング)[3]。このマッピングではPE間通信を超立方体結合上で閉塞なしに処理できる。またどの通信も超立方体結合上の隣接間のみに限定できる。しかし1つの比較を2つのPEで重複して行なう無駄が生じる。これに対して著者等は1つのPEに2つのソートエレメントを置く新マッピングを考案した(1PE-1比較マッピング)。新マッピングでは比較の重複がない。また1PE-1エレメントマッピングと同様にPE間通信を超立方体結合上で閉塞なしに処理し、どの通信も超立方体結合上の隣接間のみに限定できる。本報告では、バイトニックソート、1PE-1エレメントマッピングについて簡単にレビューした後、1PE-1比較マッピングを紹介しその性能評価を行なう。

2 バイトニックソート

単調増加する数列と単調減少する数列を接合した数列、またはその数列を巡回シフトした数列をバイトニック列という。数列$\{a_1, a_2, \ldots, a_{2^n}\}$がバイトニック列であるとき、$1 <= i <= n$に対して
$d_i=\min(a_i, a_{n+1})$, $e_i=\max(a_i, a_{n+1})$ならば以下のことが成り立つ。

1 数列$\{d_1, d_2, \ldots, d_n\}$,$\{e_1, e_2, \ldots, e_n\}$はバイトニック列である。

2 $\max(d_1, d_2, \ldots, d_n) <= \min(e_1, e_2, \ldots, e_n)$

このことは長さ2^iのバイトニック列に2^{i-1}回の比較を行ない大小関係のある2個の長さ2^{i-1}のバイトニック列に分割できることを示す。この操作を再帰的にi回繰り返すと長さ2^iのバイトニック列からソート列を得る。長さ2^iのバイトニックマージャは$i*2^{i-1}$個の比較器を用いてこのソートを行なうネットワークである。長さ2^nのバイトニックソートネットワークは長さ2のバイトニックマージャ(比較器)から長さ2^nのバイトニックマージャまで順番に結合したものである。長さ2^iのバイトニックマージャは長さ2^{i-1}の昇順にソートするバイトニックマージャと長さ2^{i-1}の降順にソートするバイトニックマージャと結合する。長さ2^iのバイトニックマージャは長さ2^{i-1}の昇順ソート列と降順ソート列を長さ2^iのバイトニック列と見なしてソートする。バイトニックソートネットワーク全体では$2^{n-2}n(n+1)$個の比較器を持つ。図1に長さ8のバイトニックソートネットワークを示す。

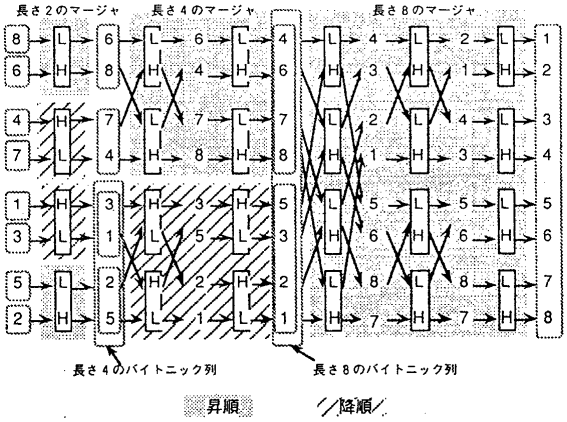

図1 長さ8のバイトニックソートネットワーク

バイトニックソートでは要素数が多いと比較器も多く必要である。このため定数個の比較器で任意のエレメント数のソートができない。これを解決するために比較要素をブロック化する[2]。ブロックバイトニックソートでは一つのエレメントは複数エレメントからなるブロックである。処理に先だってブロック内はソートしてある。比較器は2つのブロックをマージし1つのソート列を作りそれを前半部と後半部に分割して出力する(マージ&スプリット)。
バイトニックソートを並列計算機に移植する場合、各段のマージ&スプリットをPEにマッピングする。比較器間の結合はPE間通信でシミュレートする。

3 1PE-1エレメントマッピング

1PE 1エレメントマッピングでは2^n個のPEを持つ超立方体結合計算機で2^n個のブロックをバイトニックソートする。1つのPEは1個のブロックを持つ。最初にブロック内をソートする。ブロックのマージ&スプリ

Mapping Bitonic Sorting on Parallel Processors
Y. Noguchi, T. Hagiwara, N. Akaboshi and R. Take
Fujitsu Laboratories Ltd.

ットはnステージ(S_0, S_1, ..., S_{n-1}) で行う。S_i はさらサブステージ (S_{i0}, S_{i1}, ..., S_{ii}) を持つ。S_i はもとのソートネットワークでは長さ2^{i+1}のバイトニックマージャに相当しS_{ij}はその中のj番目のマージ&スプリットに相当する。1PE-1エレメントマッピングでは、S_{ij}においてP_kと$P_{k'}$ ($k'= k\oplus 2^{i-j}$)が対になって1つのマージ&スプリットを行う。この対は互いにブロックを交換しマージする。マージの後、前半部、後半部の1方を残し他方を捨てる。動作の制御は関数 $R(i,j,k) = k_{i+1}\oplus k_{i-j}$ による。k_iはkの2進数表現(b_{n-1}, ..., b_i, ..., b_1, b_0)でのb_iを表わす。R=0のときP_kが前半部を残し$P_{k'}$が後半部を残す。R=1のときP_kが後半部を残し$P_{k'}$が前半部を残す。プロセッサ間通信は、超立方体の隣接ノード間に限定されメッセージの衝突をおこさない。

4 1PE-1比較マッピング

1PE-1比較マッピングでは2^n個のPEを持つ超立方体結合計算機で2^{n+1}個のブロックをソートする。1つのPEは2つのブロックを持つ。1つのブロックの大きさは1PE-1エレメントマッピングの1/2である。最初に2つのブロックまとめてソートし、前半と後半の2つのブロックにする。ブロックのマージ&スプリットはn+1ステージ(S_0, S_1, ..., S_n) で行う。S_i はさらにn+1個のサブステージ (S_{i0}, S_{i1}, ..., S_{ii}) を持つ。ただしS_n (最後のステージ) は、n 個のサブステージ (S_{n0}, S_{n1}, ..., S_{nn-1}) を持つ。1PE-1比較マッピングでは、S_{ij}においてP_kと$P_{k'}$ ($k'= k\oplus 2^j$)が対になって2つのマージ&スプリットを行う。$R(k,i,j) = k_{i+1}\oplus k_i\oplus k_j$とする。R=0 の時$P_k$は$P_{k'}$に大ブロックを送り、$P_{k'}$は$P_k$に小ブロックを送る。R=1 の時は逆のブロックを送る。ブロックの交換のあとP_kと$P_{k'}$はそれぞれマージスプリットをおこなう。1PE-1比較マッピングではソート終了時にP_iに $i + (N/2 - 1) * (i_0 - i_{n-1})$ 番目のブロックが得られる。プロセッサ間通信は、超立方体の隣接ノード間に限定されメッセージの衝突をおこさない。

5 比較

エレメント数をE個, PE数$N=2^n$個のときの1PE-1エレメントマッピング、1PE-1比較マッピングの内部ソートを除く処理時間を表にしめす(図2)。1PE-1比較マッピングでは1PE-1エレメントマッピングに比べてブロック長が1/2であるためサブステージあたりの処理時間が1/2になる。また2倍のブロック数を扱うためサブステージ数は増加する。1PE-1比較マッピングの最初のマージ&スプリットは内部ソートと兼用される。所要時間の差は、$a*E*n*(n-1)/4N$ となり, n>1(N>2) で1PE1比較マッピングが1PE-1エレメントマッピングより高速である。

6 実験結果

両マッピングを並列計算機に移植して処理時間を実測した。使用した計算機は16個のトランスピュータT800でバイナリ4-キューブのネットワークに結合している。図3は、32バイト整数100万件に対するソート時間である。合計時間で10%以上の高速化がはかられている。

	時間/サブステージ	サブステージ数	合計時間
1PE-1エレメント	a*E/N	n*(n+1)/2	a*E*n*(n+1)/2N
1PE-1比較	a*E/2N	(n+1)*(n+2)/2 - 1	a*E*n*(n+3)/4N

図2 処理時間の比較

	内部ソート	マージ	合計
1PE-1エレメント(s)	5.07	5.62	10.69
1PE-1比較(s)	5.07	4.17	9.23

図3 処理時間の実測値

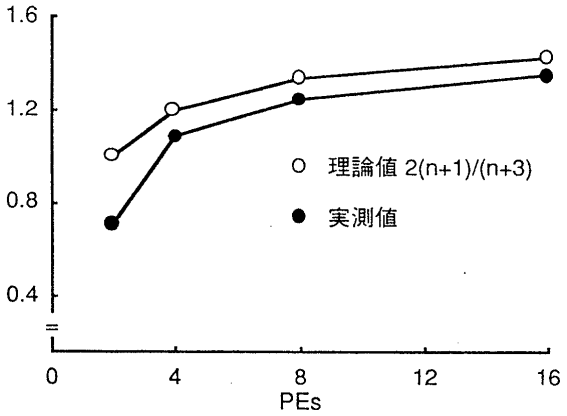

図4 内部ソートを除く処理時間の性能向上比

図4はマージ処理時間の1PE-1エレメントマッピングと1PE-1比較マッピングの内部ソートを除く処理時間の比である。横軸はPE数である。5節の解析結果より速度向上比が約10%悪い。内部ソートを除く処理時間はPE間通信時間とPE内処理時間からなる。1PE-1エレメントマッピングではマージ&スプリットのときブロックの半分を捨てる。このため1PE-1エレメントマッピングのPE内処理時間は1PE-1比較マッピングのPE内処理時間の2倍よりは小さい。このためサブステージあたりの処理時間が5節で仮定した1/2より大きくなる。実測ではこの値が0.56である。

参考文献

[1] K. E. Batcher, "Sorting networks and their applications," Proc. AFIPS 1968, Spring Joint Comput. Conf. 1968.

[2] S. G. Akl, "Parallel Sorting Algorithms," Academic Press, 1985

[3] Y. Noguchi, R. Take, H. Yokota, N. Akaboshi, "A Parallel Database Machine Using Transputers," Proceedings, Transputer Application, 1991, ISO Press.