# ハイパクロスバ・ネットワークにおける並列ソート処理*

1T−5

板倉憲一、朴泰祐、中村宏、中澤喜三郎†
筑波大学 電子・情報工学系‡
{itakura,taisuke,nakamura,nakazawa}@arch.is.tsukuba.ac.jp

## 1. はじめに

MIMD 方式の大規模並列システムにおいてデータベース処理のような非数値処理を行なう場合、プロセッサ(PU)間の交信は不均一、広範囲なものになる場合が多い。

ハイパクロスバ・ネットワーク (HXB) [1] はクロスバスイッチを用いた多段のネットワークである。特に3次元の HXB は比較的小規模なクロスバスイッチを用いて平均距離及び最大距離が短く様々な交信パターンに対応し数千ノード程度の並列処理システムに有効である。

本研究では HXB の特性を確認するために、非数値処理の例として並列ソート処理の性能評価を行なう。並列ソートのアルゴリズムとしては代表的な bubble、bitonic、radix の3種類を用いる [2][3]。また、他のネットワークアーキテクチャとして Hyper Cube(HC)、Torus Mesh(TM) との比較も行う。

## 2. ハイパクロスバ・ネットワーク

HXB は HC を拡張した Base m n-cube をさらに拡張したネットワークである。$n$ 次元の HC は $2^n$ 個の PU を結合するが、Base m n-cube は $m^n$ 個の PU を結合する。各 PU は $n$ 本の腕を出し、m 進数で表記された PU 番号が1桁だけ違う $m$ 台の PU にクロスバスイッチを介してつながる。ハードウェア量は PU 数 $P = m^n$ に対して $m \times m$ のクロスバが $m^{n-1}n$ 個必要なのでクロスバのスイッチ数は $mP\log_m P$ 個となる。

HXB は各次元に対して独立な $m$ をとれるように Base m n-cube を拡張したものである。3次元の HXB では、$x, y, z$ 方向の PU 数をそれぞれ $X, Y, Z$ 個とすると、全 PU 数は $X \cdot Y \cdot Z$ となる。3次元の HXB は3次元の HC のリンク上に PU を全 $X$(または $Y, Z$)個配置し、それらの間をクロスバ結合したトポロジになっているので、HC の特性を包含しており同様に3次元 TM の特性も包含している。クロスバのスイッチ数は $XYZ(X+Y+Z)$ 個となる。

## 3. 並列ソートアルゴリズム

ここでは、3つの並列ソートアルゴリズム、bubble、bitonic、radix についてネットワークの転送パターンを中心に述べる。各アルゴリズムの処理時間は step 数、step 当たりの通信時間、step 当たりの PU 処理時間から求まり、buble は通信時間が小さく、bitonic は step 数がやや小さい。radix は step 数が極めて小さいが通信時間や PU 処理時間が大きいという特性を持つ。PU の台数を $P$、総データ数を $N$、PU 当たりのデータ数を $n(=N/P)$ とする。処理の前提としては各プロセッサの持っているデータ数 $n$ は等しく、ソート後も全プロセッサは同じデータ数 $n$ を持つものとする。また、bubble 及び bitonic のアルゴリズムでは初期状態として PU 内のデータがソートされている必要がある。これは PU 毎に逐次クイックソートによって一回だけ行なう。また、各アルゴリズムの処理時間のオーダを表1に示す。

### 3.1 bubble sort (odd-even sort)

並列化 bubble sort は、全プロセッサを1次元方向に並べ、隣同士でデータの交換を行う。ネットワークの転送パターンは1次元の隣接転送のみでありどのようなネットワークでも無衝突転送が可能である。処理 step 数は $P$ であり、各 step は2フェーズからなる。最初のフェーズで各 PU は自分の $n/2$ のデータをマージソートし、次のフェーズでは隣接 PU 間で $n/2$ のデータを交換し合いマージソートをする。

### 3.2 bitonic sort

並列化 bitonic sort で $p$ 個の PU から成る bitonic 列をソートするためには、1回の shuffle 転送と1回の butterfly 転送 ($\log_2 p$ 回の exchange) と $\log_2 p+1$ 回のマージソートを行なう。処理全体のステップ数は $\log_2 P$ である。butterfly 転送のパターンは HC 及び HXB では無衝突転送が可能である。shuffle 転送のパターンは HC であっても HXB でもそのまま転送するとネットワーク中の経路で衝突する。しかし、HXB では最大でも無衝突転送の約2倍の時間で転送は完了する。このことは計算機シュミレーションで確認した。TM では shuffle 転送はもちろん、butterfly 転送においても経路上で衝突し、回線の仮想化をしていないとデッドロックを起こす。

### 3.3 radix sort

並列化 radix sort はマスタ・スレーブ方式で行なう。マスタは各スレーブから送られてくる部分インデックスを集め、グローバルインデックスを作成する。各スレーブはこれを元に適当な相手とデータ交換を行なう。従って転送パターンは部分インデックスのための $P$対1の collection、グローバルインデックスのための1対 $P$の broadcast、各スレーブが持っているデータを担当スレーブに転送する $P$対 $P$の complete exchange がある。HXB 上で各 PU のデータ転送量を $D$としたときの転送時間のオーダは以下のようになる。collection 転送は全 PU が一斉に転送を行ない、マスタは来た順に処理を行なえば $O(P \cdot D)$ の転送時間となる。HXB では broadcast 転送が容易に行なえる [1] のでその転送時間は $O(D)$、complete exchange は各 PU が相対的に位置の等しい PU を相手に送信すると無衝突転送を $P$回行なうので $O(P \cdot D)$ の転送時間となる。また、PU の内部

*Parallel Sorting on Hyper-Crossbar Network

†Ken'ich ITAKURA, Taisuke BOKU, Hiroshi NAKAMURA, Kisaburo NAKAZAWA
‡Insitute of Information Sciences and Electronice, University of Tsukuba

処理は基数で分類した数え上げ ($O(n)$) とグローバルインデックス作成 ($O(P \cdot 2^r)$) である。

## 4. 評価と考察

ここではHXBの並列計算機を想定する。まずPVM上で並列プログラムを作成し正当性を確認した後、1つのPUに対応するプログラムからネットワークに関する部分を取り除きワークステーションで実行させてCPU時間を測定する。ネットワークの転送時間についてはシミュレータ[4]を用いて解析を行なう。ネットワークの性能は転送立ち上げ over head を5$\mu$sec、throuput を200Mbyte/secとした。データは1M個の整数(4byte)とする。PU台数を増やした時のPU処理時間を図1に、転送時間を図2にそれぞれ示す。

まず、PU内部処理時間を見るとbubble sortはプロセッサ台数が増えると$n$が減るので処理時間は減少するが絶対的な時間は遅い。bitonic sortは全体的に良い性能を示す。これはstepが$O((log_2 P)^2)$であるためである。radix sortについては今回の実装方法ではPU処理時間のうち$2^r$に依存する部分が大きいので$n$がある程度大きくないと不利である。この点については改良の余地があると考えられる。$n$が$P$に対して十分大きい時には処理時間は$O(1/r)$となる。また、$P$少ない時のbitonic sort処理時間のほとんどは初期のクイックソートの時間である。

データ転送についてはbitonic sortの毎回のメッセージ長がN-halfを越えているので転送時間は回数ではなく転送データ量に依存する。これに対してradix sortは短いメッセージを多く転送するので回数に依存する。PU数が500以上の場合にはbitonic sortは転送メッセージ量が減ることで速くなるが、radix sortは転送回数が増えることで遅くなっている。ネットワークの性能によって全体の時間は変わってくるが多くのパラメータで実験したところPU数が500以上の時にはbitonic sortがもっとも良い性能を示すことがわかった。

ネットワーク別に考えるとHXBはbitonic sortのshuffle転送を最も効率良く行なえる。これと同様のことを1024PUのHCで実現するためにはWormHoleのようなルーティングが必要で大規模システムではハードウェア的に実現は難しい。TMと比べるとbubble sortの隣接転送以外はHXBの方が高い性能を示すので[1][4]、bitonic sortとradix sortはHXBの方が転送時間が短くなる。

## 5. おわりに

本研究ではHXBを用いた並列計算機上での並列ソートの処理性能を評価した。用いたアルゴリズムはbubble sort,bitonic sort,radix sortである。問題規模を一定としPU台数が32から1024の範囲でPUの内部処理時間とネットワークの転送時間に分けて処理時間を求めた。PU数が数十台の規模の時にはradix sortが有効であるが、500台以上の規模ではbitonic sortが有効である。またHXBではbitonic sortで現れるshuffle転送とbutterfly転送を効率的に処理できるので、500台以上のPUを用いたマシンで並列ソート処理を行なう場合に極めて有効なネットワークトポロジであることが分かった。

## 謝辞

本研究に当たり、貴重な御意見をいただいた筑波大学西川博昭助教授に深く感謝します。

## 参考文献

[1] 斉藤 哲也 他「超並列計算機のネットワークの実現可能性と性能評価」、情処研報 92-ARC-95-4

[2] 岡田 英明 他「超並列計算機における各種ソートアルゴリズムの実装と評価」、情処研報 92-ARC-97-12

[3] Selim G. Akl 著、阿江忠、山下雅史、相原玲二 訳「並列ソーティングアルゴリズム」、啓学出版

[4] 石橋 規子 他「大規模並列処理ネットワークにおけるランダム転送性能の評価」、情処全大46回

表1:各アルゴリズムの処理時間のオーダ

| | step数 | 転送時間 | PU処理時間 |
|---|---|---|---|
| bubble | $P$ | $O(n)$ | $O(n)$ |
| bitonic | $1/2((\log_2 P)^2)$ $+\log_2 P$ | $O(n)$ $+O(\log_2 n)$ | $O(\log_2 n)$ |
| radix | $d/r$ | $O(P \cdot R)$ $+O(P)$ | $O(n)$ $+O(P \cdot R)$ |

$d$:keyのbit数、$r$:基数のbit数、$R$:基数$(2^r)$
bubble,bitonicは1回の内部ソート$O(n \log n)$を行なう。

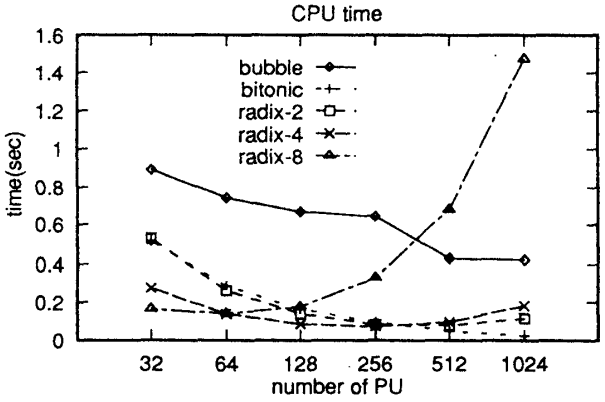


図1:総データ数固定のPU処理時間

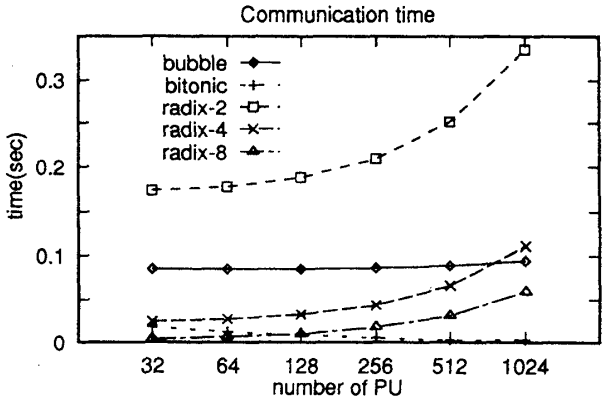


図2:総データ数固定の通信時間