計算機アーキテクチャ　90－3
（１９９１．１０．７）

ドラゴンネットを用いた並列ソート

野口泰生　赤星直輝　横田治夫　武理一郎
富士通研究所

データベースにおける並列ソートでのデータ転送量を最小にする並列キーソート法を提案している。並列キーソート法は、並列キーソート、分割値のブロードキャスト、レコードの再配置の３ステージからなる。著者等の提案している全点対全点通信を最適に処理する並列計算機用結合網：ドラゴンネットを利用すればこのソート法は特に効果的である。

並列キーソート（ブロックバイトニックソート）、と分割値のブロードキャストはドラゴンネットのみを利用して実現できる。レコードの再配置における全点対全点通信はドラゴンネットで効率よく処理できる。トランスピュータによる実験により並列キーソート法の有効性を実証した。

A Parallel Key Sort Method Using Dragon Net

Yasuo Noguchi, Naoki Akaboshi, Haruo Yokota, Riichirou Take

Artificial Intelligence Laboratory Fujitsu Laboratories LTD.

Kamikodanaka 1015, Nakahara-ku, Kawasaki 211, Japan

We propose a parallel sorting method called parallel key sort, which minimize communication between processing elements. Parallel key sort works well on a parallel database machine that we have prototyped. The machine consists of processing elements and interconnection network called Dragon net, which executes all-to-all-node communication optimally. Parallel key sort consists of three stages, key sort, broadcast and record reallocation. Using Dragon net, key sort and broadcasting can be executed efficiently and all-to-all-node communication involved in record reallocation can be executed optimally. Parallel key sort exhibited better performance and scalability than that of block bitonic sort on the prototype database machine using Transputers.

1　まえがき

データベースの重要な処理の１つであるソートに関しては、高速化のため並列アルゴリズムの研究が数多くおこなわれている[1]。それらのアルゴリズムはソートハードウェアとして実現されていたり[2]、並列データベースマシンのネットワークの機能として実現する方法が提案されている[3]。

われわれは、ドラゴンネットとよぶ多段結合ネットワークをもちいた並列データベースマシンの研究をおこなっている[4],[5],[6]。本研究報告はこのドラゴンネットで結合されたプロセッシングエレメント上での並列ソートに関するものである。２章で筆者等が用いるドラゴンネットと並列データベースマシンのプロトタイプについて述べる。３章で並列データベースマシン上でデータ転送量を最小にする並列キーソート法について述べる。４章で並列キーソート法の実験結果を示す。

2　ドラゴンネットと並列データベースマシン

2.1　ドラゴンネット

並列データベース演算では、すべてのノードが他のすべてのノードにメッセージをおくる全点対全点通信が多発する。N-ノードの結合網では全点対全点通信においてN^2個のメッセージを生じる。これらのメッセージは大局的なルーチング戦略なしに個々のノードから無秩序に送り出されると、結合網上で多くの衝突をすることになる。この衝突のため結合網の実質的な転送速度がさがり、結合網を有効利用することができない。

著者等の提案しているドラゴンルーチング[7],[8]と呼ばれるプリスケジューリング　ルーチングによれば、全点対全点通信でのメッセージの衝突をさけ結合網を約１００％利用することができる。著者等は、超立方体結合網、バイナリｎキューブ結合網、トーラス結合網に対してそれぞれ最適なプリスケジューリングを発見した。

ドラゴンネットはそれぞれのトポロジーをもったネットワークであり、ドラゴンルーチングを実行するための制御機構を備える。トポロジー、ドラゴンルーチング、制御機構の組み合わせでさまざまなドラゴンネットが存在する。著者等がプロトタイプで用いているドラゴンネットは、図１のようなバイナリｎキューブ結合網＋排他論理和に基づくプリスケジューリング＋分散制御機構の組み合わせのものである。

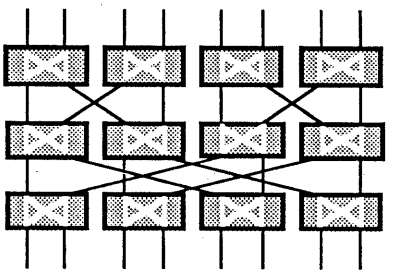

図１　バイナリｎーキューブ結合網

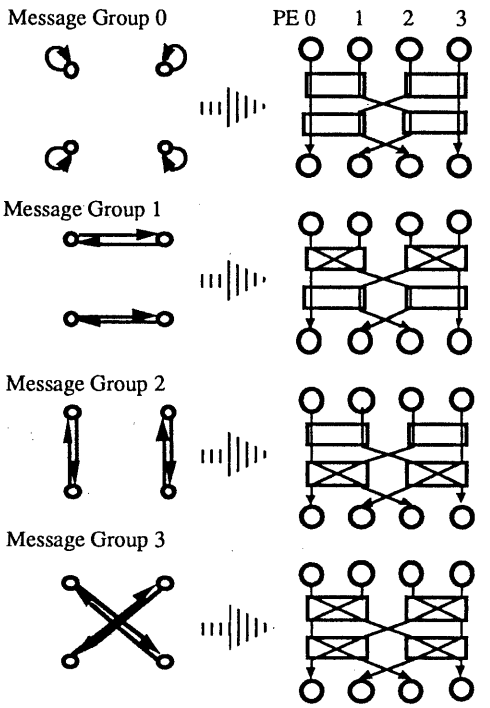

図２　ドラゴンルーチング

このネットワークは図２のようなドラゴンルーチングを実行する。ドラゴンルーチングはN^2個のメッセージをN個のメッセージグループに分割する。ドラゴンルーチングは１メッセージグループを１フェーズで処理する。従ってN-ノードの全点対全点通信を処理は、Ｎフェーズを必要とする。それぞれのグループは０からN-１のｉｄを持つ。グループｓは、ノードｖからノードｖ⊕ｓへ送られるメッセージで構成される。どのメッセージグループもバイナリｎキューブ結合網でメッセージの衝突を起こさない。またどのメッセージグループでもすべてのリンクを有効に使用する。

このネットワークでは分散制御機構によりフェーズが抜けたり、追いぬいたりしないように制御される。各スイッチのなかで、分配器０、分配器１、集約器０、集約器１が並列動作する（図３）。分配器、集約器はそれぞれ局所フェーズを持つ。分配器ｉはパケットをリンクｉ.ｉｎから受けとり、局所フェーズにより決定される集約器に渡す。集約器ｉは局所フェーズにより決定される分配器からパケットを受けとり、リンクｉ.ｏｕｔに出力する。分配器と集約器の通信は同期する。すなわち、両方が通信準備できるまで、通信は保留される。

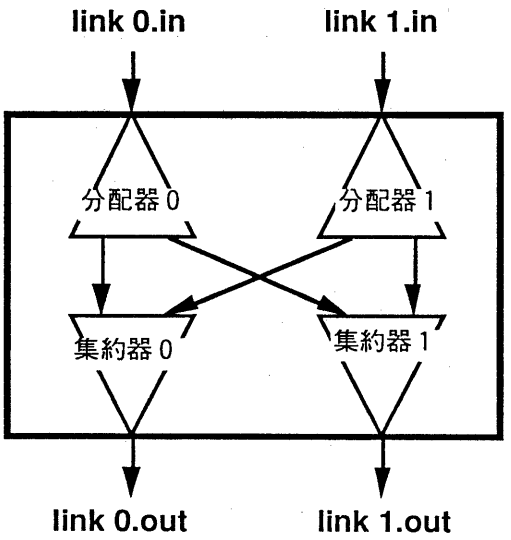

図３　分配器と集約器

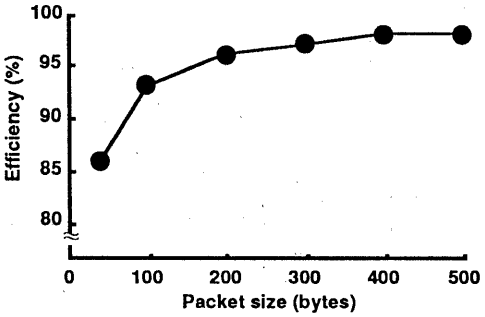

図４　ドラゴンネットの効率

分配器と集約器の局所フェーズはフェーズ切り替タグを含むパケットを受け渡したときに加算される。フェーズ切り替タグは各プロセッシングエレメントがそのフェーズに属する最後のパケットに付ける。局所フェーズがN-１を越えると０に戻される。ドラゴンネットではフェーズ切り替タグ自身は前のフェーズのパスを通る。タグが明示的な行き先を記し、新パスを作りながら進むワームホールルーチングとはこの点で異なる。

図４１はドラゴンネットによるドラゴンルーチングの効率（全転送速度／全バンド幅）をパケット長に対してプロットしたものである。１００バイトを越えるパケットに対して効率は９０％以上、４００バイトを越えるパケットに対して効率は９８％以上である。

ドラゴンネットはドラゴンルーチングの他に、各プロセッシングエレメントが同期をとるためのバリアメッセージをサポートする。バリアメッセージはスイッチの中で待ち合わせを行う。またバリアメッセージは他のメッセージを追い抜かない。プロセッシングエレメントはバリアメッセージをうけとると、すべてのプロセッシングエレメントがバリアメッセージを出したこと、およびそれより前に自分にあてられたメッセージをすべて受信したことを知ることができる。

待ち合わせのとき、両メッセージをオペランドとした、加算、積算等の結合則の成立する演算をおこなえば、各プロセッシングエレメントは、総和、総積などの結果を得ることができる。

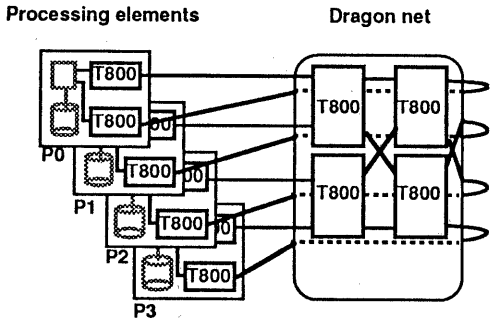

図5　プロトタイプ

2.2　並列データベースマシン

図5は著者等がトランスピュータT800を用いて構成したデータベースマシンのプロトタイプである。プロセッシングエレメント1台にT800を2個使用した。現在プロセッシングエレメントにディスクドライブは接続されていない。したがって実験結果はI／O時間を含まない。

ドラゴンネットは、1個のT800を1個のスイッチとして持ちいて構成した。
T800の4本のシリアルリンクをバイナリ4キューブ結合網を構成するように接続した。分配器と集約器はトランスピュータのソフトウェアで実現されている。トランスピュータではシリアルリンクとALUが並列に動作するので、パケットの送受信とルーチングはオーバラップする。

プロセッシングエレメントの一つのT800はネットワークの入力リンクに接続され、もう1方のT800はネットワークの出力リンクに接続されている。

3　並列キーソート

ソートはデータの比較と移動を繰り返しおこなう処理である。データベースのソートでは、キーになる部分にくらべてレコード全体はかなり大きい。したがってデータ移動は比較よりも多くの時間を費やす傾向がある。特に並列データベースマシンではノード間のレコード全体の転送が大きなオーバヘッドになる。

著者らはネットワーク上でのレコード移動量を最小にする並列分割ソートを提案している[9]。
並列分割ソートは3つのステージから構成される。第1ステージは分割ステージである。このステージでは各プロセッシングエレメントが[9]のアルゴリズムにより、協調的に分割値をもとめる。分割値とはソートが終了した時点でのそれぞれのノードの保持する値の最大値である。第2ステージは再配置ステージである。このステージでは各プロセッシングエレメントは分割値にもとづいて対応するプロセッシングエレメントにレコードを送信する。第3ステージは内部ソートステージである。このステージでは各プロセッシングエレメントが独立に再配置されたレコードをソートする。

並列分割ソートは分割ステージでの処理時間がノード数が増加したときノード数の2乗に比例して増加するという欠点がある。著者等が今回提案する並列キーソートは並列分割ソートのこの部分を改良したものである。

並列キーソートは3つのステージから構成される。第1ステージはキーソートステージである。このステージでは各プロセッシングエレメントが協調的にキーの部分だけをソートする。

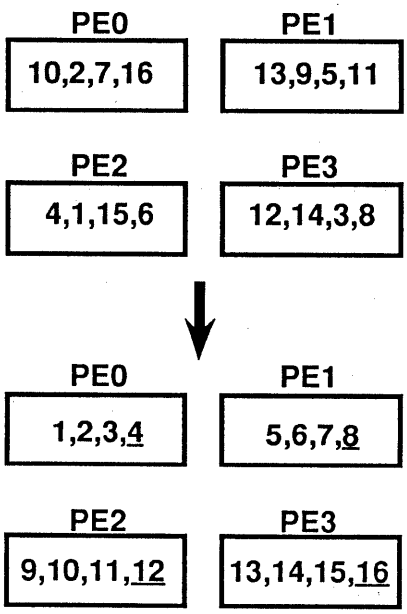

図5　キーソートと分割値

このステージではキー以外のレコードの部分は無視する。これによりソートに不必要なデータ転送を省くことが可能である。キーソートにはブロックバイトニックソートを用いる。ブロックバイトニックソートによればノード数が多くなったときにも処理時間は増加せず減少する。ノード数が多いときには並列分割処理より短時間でキーソート可能である。図5はキーソートを行った結果をしめしている。第2ステージはブロードキャストステージである。このステージでは各プロセッシングエレメントはキーソート後の自分の保持するキーの最大値（図5の下線のついている数値）を分割値として他のすべてのプロッセッシングノードにブロードキャストする。第3ステージは再配置フェーズである。このステージでは各プロセッシングエレメントは分割値にもとづいて対応するプロセッシングエレメントにレコードを送信する。図5の場合には、キーが4以下のものはプロセッシングエレメント0、4を越え8以下のものはプロセッシングエレメント1に集められる。

先にキーがソートされているのでこれをインデックスに用いて、レコードの着信とオーバラップしてレコードを格納する。3つのステージの詳細は以下に述べる。

3.1 キーソート

バイトニックソートは本来ネットワークソータのためのアルゴリズムである[10]。このアルゴリズムではM個のエレメントをO（M log^{2}M）個の比較器を使ってO（log^{2}M）時間でソートする。このネットワークでは全ての比較が2のべき乗の距離はなれているデータ線間でなされるので、バイナリn-キューブのネットワークで実現するのが容易である。比較器の働きはプロセッシングエレメントでシミュレートし、比較器の接続パタンと等価なデータの流れをドラゴンネットで実現した。

限られた数のプロセッシングエレメントでどんな数のエレメントでもソートできなければならないのでブロック化が必要である。ブロック化には2つの手法、2ウエイマージスプリット法[11]とマージ交換法[12]が知られている。2ウエイマージスプリット法は、バイトニックソートの各ステップで2値の比較交換の変りに、2つのストリームをマージし前後に分割（スプリット）して2つのストリームを出力する。マージ交換法はマージ-スプリットの変りにソートを用いて前後に分割を行う。2ウエイマージスプリット法はマージ交換法より高速であるがより多くの局所作業域を必要とする。比較器の機能をネットワークに持たせる場合は局所作業域の制約から後者が用いられる。著者等の方式では比較器の機能をプロセッシングエレメントに行わせる。プロセッシングエレメントには十分なメモリがあり将来的にはディスクまでも局所作業域として使用可能なので高速な2ウエイマージスプリット法を採用した。

ブロックバイトニックソートでは最初に各プロセッシングエレメントが自分の保持するデータブロックを独立にソートする。その後、各プロセッシングエレメントが2ウエイマージスプリットを繰り返すことにより、各プロセッシングエレメントの保持するブロックはソートされたブロックになる。プロセッシングエレメントの数をNとすると2ウエイマージスプリットはlog Nステージ ($S_0, S_1, ..., S_{\log N-1}$) で行う。ステージ$S_i$ はさらにサブステージ ($S_{i0}, S_{i1}, ..., S_{ii}$) をもつ。ステージ$S_i$ はもとのバイトニックソートでは2^{i+1}エレメントのバイトニックマージに相当し、サブステージ S_{ij}はステージS_i でのj番目の比較に相当する。

サブステージ S_{ij}ではプロセッシングエレメントK とプロセッシングエレメント $K \oplus 2^{i+j}$ が対になって2ウエイマージスプリットを行う。対になったプロセッシングエレメントは互いにブロックを交換し、2つのブロックをマージする。マージの後、対のプロセッシングエレメント のうち1方はマージされたブロックの前半部を残し、もう1方は後半部を残し、残りをすてる。ブロックの残しかたは、関数 $R_{ij}(k) = k_{i+1} \oplus k_{i-j}$ によって決定される。k_iは、kのi番目のビット値である。R = 0の時、前半部を残し、R = 1の時、後半部を

残す。

図６は４プロセッシングエレメントの場合でのこの課程を示す。図中、矢印のついた線が２ウエイマージスプリットをあらわす。たとえばステージS_{00}ではプロセッシングエレメントＰ０とＰ１がブロックを交換し、Ｐ０が前半部、すなわち１、２、５、Ｐ１が後半部、すなわち７、８、１１を保持する。同時にプロセッシングエレメントＰ２とＰ３がブロックを交換し、Ｐ２が後半部、すなわち６、９、１０、Ｐ３が前半部、すなわち０、３、４、を保持する。

ブロックバイトニックソートでのプロセッサ間通信は、メッセージの衝突をおこさないｌｏｇＮ個のメッセージパタンでおこなうことができる。これらのメッセージパタンはドラゴンルーチングのメッセージグループのサブセットであるので、ドラゴンネットを持ちいて行うことができる。

図７は４プロセッシングエレメントの場合のメッセージパタンである。ステージS_{00}、S_{11}はドラゴンルーチングのメッセージグループ１によって行う。ステージS_{10}はドラゴンルーチングのメッセージグループ３によって行う。

ソートのエレメント数をM個、プロセッシングエレメントの数をN個とするとき、ブロックバイトニックソートにかかる時間は、内部ソートを除くとO（M／N・ｌｏｇ²N）である。ブロックバイトニックソートを用いない場合、例えば内部ソートの後で、ブロックを２ウエイマージして、さらに再配置する場合でも同じ結果を得ることができる。この場合の所用時間はO（M）となる。図８は、この課程を示す。

後者の場合プロセッシングエレメントの台数効果があらわれない。これははマージの課程で全エレメントが単一のプロセッシングエレメントに集中するからである。ブロックバイトニックソートによれば、プロセッシングエレメントの数を増加させることで処理時間を減少することが可能である。

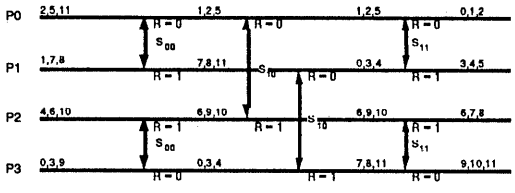

図６　ブロックバイトニックソート

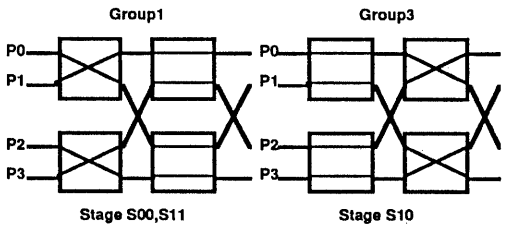

図７　メッセージパタン

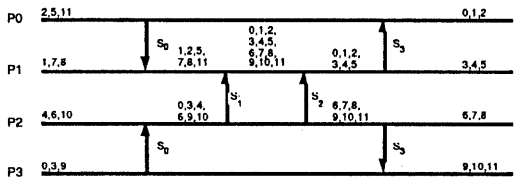

図８　マージ&再配置

3.2　ブロードキャスト

キーソートの終了後、プロセッシングエレメントｉにおける最大値が分割値ｉとなる。全ての分割値は１回のバリア操作でブロードキャストされる。

各プロセッシングエレメントはＮ要素の配列Ａを用意する。プロセッシングエレメントｉはＡ［ｉ］に分割値ｉ、他の要素に０を代入する。各プロセッシングエレメントはこの配列をバリアメッセージとしてネットワークに出力する。

２つの配列がスイッチで出会った時に、これらの配列は要素ごとに足しあわせられる。その結果は両方の配列に上書きされる。各プロセッシングエレメントがこのメッセージを受け取ると、キー

ソートが終了したことを知ると同時に配列の各要素からすべての分割値を得ることができる。

3.3 再配置

レコードの再配置は全点対全点通信を引き起こす。これはドラゴンネットによって実行される。

レコードを転送する前に各プロセッシングエレメントはレコードを格納するための空きスロットを用意する。キーソートによって得られたキーのソート列をインデックスとしてキー値とレコードの格納場所を対応させることができる。レコードの受信とオーバラップしてレコードの格納場所をもとめスロットに格納する。

4. 実験

並列キーソートをプロトタイプ上に実現し、性能測定した。現在プロトタイプにはディスクドライブが接続されていないのでレコードはすべてメモリ上の配列として取り扱った。またレコードは実際にはキーのみをメモリに記憶して、レコードの再配置のときに、キー以外のフィールドを補った。プロセッシングエレメントを2、4、8、16台と変えて測定した。測定条件はレコード96バイト、キー4バイト整数、レコード数120、000件であった。

測定したすべての台数でブロックバイトニックソートより並列キーソートのほうが高速であった（図8）。その差はノード数が多いほうが顕著である。これは処理時間の大部分を占めるキー以外のレコードの転送時間が、ソートエレメント数が固定、プロセッシングエレメント数がNのとき、ブロックバイトニックソートでO（log^2N／N）時間であるのに比べてドラゴンネットではO（1／N）時間であるからである。

並列キーソートとブロックバイトニックソートのスピードアップレシオをそれぞれN＝2のときを2として求めた。並列キーソートでは台数効果の直線性が大幅に改善した（図9）。

5. まとめ

データベースにおける並列ソートでのデータ転送量を最小にする並列キーソート法を提案した。並列キーソート法は、キーソート、分割値のブロードキャスト、レコードの再配置の3ステージからなる。

キーソート（ブロックバイトニックソート）はドラゴンネットを利用して、ソーティングネットワークの機能を各プロセッシングエレメントへマッピングする。このためソートのための特別なネットワークを構成する必要がなく、またネットワークにソート機能をもたせる必要もない。ブロードキャストはドラゴンネットのバリア機能を利用して行う。レコードの再配置における全点対全点通信はドラゴンネットで処理する。

トランスピュータによる実験により並列キーソート法の有効性を実証した。

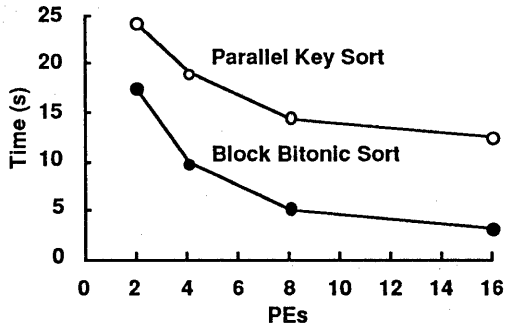

図8 処理時間

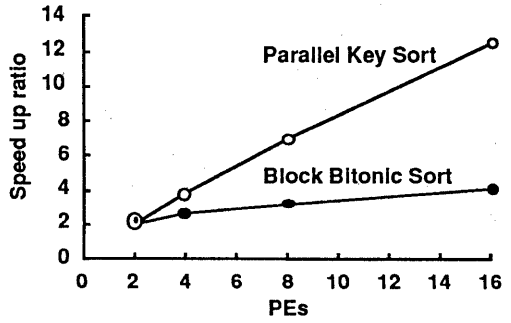

図9 処理時間

参考文献

[1] S. G. Akl, "Parallel Sorting Algorithms," Academic Press, 1985
[2] 安藤, 小宮, 喜連川, 中込, 伏見, "リレーショナルデータベースプロセッサGREOの構成," DE89-37, 1989
[3] 喜連川, 小川, "2 ウエイマージ機能を有するオメガネットワーク," 情報処理学会論文誌, Vol. 32, No. 2, 19991
[4] 武, 野口, "データベース処理向きアーキテクチャの検討," DE89-46, 1989
[5] R. Take, Y. Noguchi, and H. Yokota, "An Architecture for Parallel Database Computing," Proceedings, Transputing, 1991, ISO Press.
[6] Y. Noguchi, R. Take, H. Yokota, N. Akaboshi, "A Parallel Database Machine Using Transputers," Proceedings, Transputer Application, 1991, ISO Press.
[7] 武, "超立方体形ネットワークに於ける全点対全点通信の最適ルーティング法," 情報処理学会第35回全国大会、1987
[8] 野口, 武, 横田, "全点対全点通信用結合網：ドラゴンネット," 電子情報通信学会ワークショップ, 1991
[9] Y. Yamane and R. Take, "Parallel Partition Sort for Database Machine," Proc. International Workshop of Database Machine, 1987.
[10] K. E. Batcher, "Sorting networks and their applications," Proc. AFIPS 1968, Spring Joint Comput. Conf. 1968
[11] G. Baudet and D. Stevenson, "Optimal sorting algorithms for parallel computers," IEEE Trans. Comput. C-27, (Jan)
[12] D. J. Dewitt, D. B. Friedland, D. K. Hsiao, and J. Menon, "A taxonomy of parallel sorting algorithms," Tech. Rep. No. 482, Computer Sciences Department, University of Wisconsin-Madison, Madison, Wisconsin, May 1982.