相互結合網RDT上での階層マルチキャストによるメモリコヒーレンシ維持手法

西村克信[†] 工藤知宏[††] 西宏章[†]
楊愚魯[†] 天野英晴[†]

国内7大学で共同開発中の超並列計算機JUMP-1は，分散共有メモリを持ちRDT（*Recursive Diagonal Torus*）と呼ぶ木構造を内包するクラスタ間結合網を用いる．木構造を内包する結合網では，メモリコヒーレンシ維持のために階層的にディレクトリを構成する階層ビットマップディレクトリ方式を用いることができる．この方式では様々なディレクトリ長縮約法を用いることができ，JUMP-1で用いられる方式はその一種であると考えることができる．本論文では，階層ビットマップディレクトリ方式を定義し，JUMP-1で用いられる方式のRDT上での実現法を示してそのディレクトリ縮約方式としての性能を評価する．また，RDTルータチップの構成について述べる．

Memory Coherency Control Schemes on the RDT Interconnection Network

KATSUNOBU NISHIMURA,[†] TOMOHIRO KUDOH,[††] HIROAKI NISHI,[†]
YULU YANG[†] and HIDEHARU AMANO[†]

JUMP-1 is currently under development by seven Japanese universities to establish techniques of an efficient distributed shared memory on a massively parallel processor. It provides a memory coherency control scheme called the *hierarchical bit-map directory* to achieve cost effective and high performance management of the cache memory. Messages for maintaining cache coherency are transferred through a fat tree on the RDT (Recursive Diagonal Torus) interconnection network. In this paper, we discuss on the scheme and examine its performance. The configuration of the RDT router chip is also discussed.

1. はじめに

近年，一万を超えるプロセッサ数を持つ規模の超並列計算機についての研究が様々な研究機関により行われている．この中で，文部省重点領域研究の一環として開発が進められている超並列計算機プロトタイプJUMP-1[13),15)]は，キャッシュを用いた分散共有メモリ（Cache Coherent Non Uniform Memory Access model: CC-NUMA）を，超並列計算機上に効率良く実現することを目指している点に特徴がある．この目的のため，JUMP-1は図1に示す4つのRISCプロセッサから成るクラスタ構造を持ち，各クラスタは，MBP[16),17)]と呼ばれる分散メモリ，およびメッセージ通信の管理を行う専用細粒度プロセッサにより，全クラスタで共有できる分散共有メモリを実現する．MBPは，クラスタメモリ上のキャッシュディレクトリを用いて，無効化型と更新型が混在する柔軟性の高いキャッシュプロトコルを実現することができる．

CC-NUMAの効率の良い実現を目指すプロジェクトには，他にもStanford-FLASH[4)]やMIT-Alewife[2)]などがある．これらのプロジェクトは，いずれも規模に依存しないスケーラブルなシステムを目指しているが，具体的に想定している最大規模は数百プロセッサ程度である．このため，これらのシステムではキャッシュディレクトリをライン単位で管理し，無効化型のプロトコルによりキャッシュの一致を制御している．この方法は様々な評価や実システム上での経験[6)]から，数十から数百の規模ならば高い効率を得られることが明らかになっている．

これに対し，JUMP-1はプロセッサ数が一万を超える規模を想定している点が異なる．このように大きな

† 慶應義塾大学理工学部
Faculty of Science and Technology, Keio University
†† 東京工科大学情報工学科
Department of Information Technology, Tokyo Engineering University

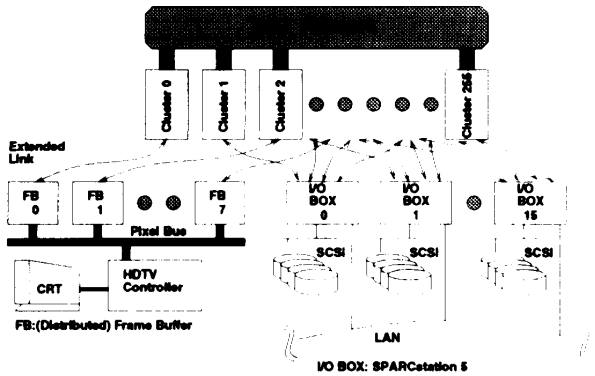

図1 JUMP-1の構成
Fig. 1 Structure of JUMP-1.

規模のシステムでは，ノードを識別するbit数が増加するため，数十バイト程度の大きさのキャッシュラインごとにディレクトリエントリを用意すると，ディレクトリの容量が大きくなり，高速なメモリに格納することが困難になる．さらに，超並列計算機の持つ多数のプロセッサを活用するためには，更新型のプロトコルを用いてデータを積極的に共有させるアルゴリズムを開発することが有効と考えられる．

そこで，JUMP-1では，ページ（4Kバイト）ごとにディレクトリエントリを用意し，データ転送のみをライン単位で行う方式を用いる．さらに，無効化型プロトコルに加えて，更新型プロトコルの混在を許すことで，データを積極的に多数のプロセッサで共有するアルゴリズムを効率良く実現する環境を用意する．しかし，この場合，ページを共有するノード（クラスタ）数は，ライン単位の管理で無効化型プロトコルを用いる従来の方式に比べて大幅に増加する可能性がある．そこで，多数のノードに対して，効率良くメッセージを転送するとともに，多数のノードを表現するディレクトリの容量も縮約する方法が必要になる．

これらを満足する手法として，松本らにより提案されたのが，疑似フルマップディレクトリ方式[18]である．この手法では，超並列計算機を構成する結合網を木構造を構成するものととらえ，階層的にパケットをマルチキャストする．このため，送信元が宛先数分のパケットを送り出す必要がなく，効率良くマルチキャストを行うことができる．また，この階層性を利用してマルチキャスト先を階層化されたクラスタのグループによって指定することによりディレクトリ長を縮約している．

この疑似フルマップ方式は，木構造をなす結合網のそれぞれの節にビットマップを与える手法である階層ビットマップディレクトリ方式に，ディレクトリ長を縮約する手法を組み合せたものと考えることができる．

階層ビットマップディレクトリ方式は単純な木構造の結合網上で実現する場合，木構造の根元付近で混雑が予想されるとともに，木構造の境界ノード付近でマルチキャストに無駄が生じる可能性がある．この問題を解決するため，JUMP-1では，2次元トーラスのFat-tree状の階層構造を持つ結合網RDT (Recursive Diagonal Torus)[19),21)]を提案している．

本論文では，階層ビットマップディレクトリ方式を定義するとともに，新たなディレクトリ縮約法[10),12)]を提案する．また，この縮約階層ビットマップディレクトリ方式をRDT上に実現する方式について述べ，さらに，これらの方式の妥当性に関して予備的な評価を行い，他の方法と比較検討する．

2. 縮約階層ビットマップディレクトリ方式

通常，ライン単位にディレクトリを管理し，無効化型プロトコルを用いる場合，無効化メッセージの宛先は，1ないし2がほとんどであるといわれ[1)]，従来，リミテッドディレクトリ方式[1),2)]，チェインドディレクトリ方式[3),8)]などのディレクトリ構成方式が用いられてきた．前者は共有しているプロセッサ数が一定の数までは，それらのプロセッサを指し示すディレクトリをキャッシュのページ/ラインに持たせ，共有するプロセッサの数がディレクトリ数を超えないようにするか，超えた場合にはブロードキャストに切り替える方法である．後者はディレクトリをリスト構造で持ち，任意の数のプロセッサを示すことができるようにしている．

これに対して，ページ単位にディレクトリエントリを管理し，更新型のプロトコルを用いるJUMP-1では，ページを共有するノード（クラスタ）数が，ライン単位の管理で無効化型プロトコルを用いる従来の方式に比べて大幅に増加する可能性がある．この問題を解決するために松本らが提案したのが，疑似フルマップ方式[18)]である☆．疑似フルマップ方式は，COMAマシンなどで用いられる階層ビットマップディレクトリ方式[9)]のディレクトリを縮約した方式と考えることができる．ここでは，まず階層ビットマップディレクトリ方式を定義する．

この方式では，クラスタ（プロセッサ）が葉に相当する木構造のネットワークを想定する．ここで，木の根からパケットを供給して，同一のパケットを複数の

☆ 本来の疑似フルマップ方式は，遠隔1対1通信，階層マルチキャスト，ローカルマルチキャストを組み合せたディレクトリ方式と通信方式をまとめて指す[18)]が，本論文ではこのうちの階層マルチキャスト方式を検討の対象としている．

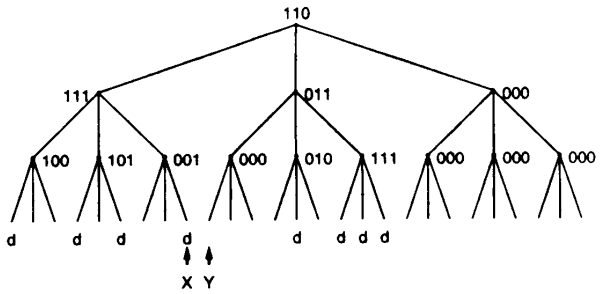

図2 階層ビットマップディレクトリ方式
Fig. 2 Hierarchical bit-map directory scheme.

クラスタにマルチキャストする．木構造のネットワークであるから，各節において送信先の葉が末端にある枝にのみパケットを送れば，必要なクラスタにのみパケットが届くことになる．**図2** に，三進木の根から **d** でマークされた葉にパケットを送る場合について示す．このとき，各節に3bitのビットマップを与えることにより送信先を完全に指定できる．階層ビットマップディレクトリは，マルチキャストを行うためのビットマップであり，これによって同一の経路を同一のパケットが複数回通ることがなくなる．

なお，単純な木構造では木の根付近のトラヒックが大きくなって隘路となってしまうため，結合網は Fat-Tree を内包することが望ましい．また，パケットは実際にはいずれかのクラスタから送信されるため，送信元のクラスタから木の根に容易にパケットを送ることができる必要がある．後に説明するように，RDT はこれらの性質を満たしており，階層ビットマップディレクトリ方式に適した構造を持っているといえる．

2.1 ディレクトリ長の縮約

階層ビットマップディレクトリにおいて必要な bit 数は，m 階層の n 進木において

$$\sum_{k=1}^{m} n^k$$

で与えられる．超並列計算機では，クラスタ（プロセッサ）数が非常に多いため，単純に階層ビットマップディレクトリ方式を用いると，ディレクトリに必要なメモリ量が膨大になり，実現が困難である．もちろんフルマップをもとに階層ビットマップを作成することは可能であるが，これには結合網のトポロジによっては複雑な操作を必要とするし，フルマップに必要なビット数もまた大きく，実現は困難である．一方，階層ビットマップディレクトリを用いた際に，階層ごとにディレクトリを引き直すと，場合によっては大きな時間を要する．そこで以下に示すビット数を縮約する方法が考えられる．

(1) いくつかのクラスタをグループ化して扱うことにより必要なビット数を削減する．

(2) 各節においてその節で適用するディレクトリを管理し，パケットがその節に到達したときにディレクトリを引く．こうすれば，n 進木を用いたとき，各節において1ディレクトリエントリにつき，n bit のビットマップを管理すればよい．

前者について，JUMP-1 では

- ある節以下はブロードキャストとする
- 複数の節で同一のビットマップを用いる．

のいずれか，もしくは両方の組合せによって，階層ごとに n bit（n 進木で）のビットマップを1つだけ持つこととし，以下の3つのディレクトリ縮約方式を採用する．ディレクトリエントリごとに必要な bit 数は，m 階層の n 進木において $m \times n$ bit となる．さらに，ビットマップは，パケットのヘッダ中に持つことができるので，完全にルータ内のみでマルチキャストが可能であり，外部のディレクトリを参照する必要がない．

SM 法 各階層ごとに，その階層のすべての節の縮約前のビットマップの論理和をとり，その階層のすべての節で用いる．

LPRA 法 松本らが文献18)で提案した方法である．パケットは木の根から供給されるが，元々はいずれかの葉に相当するクラスタから送られたものである．根からこの送信元の葉に至るパスをその他のパスと区別して扱う．パケットは，根から下位の節にビットマップに従って順にマルチキャストされるが，ある節で送信元を含まないパスの枝にパケットが送られると，その枝の下の部分にはパケットがブロードキャストされる．すなわち，根から送信元の葉へのパス上の節に各階層のビットマップが適用され，それ以外の節では上位の節からパケットが来ればブロードキャストを行う．

文献18)では，パケットは送信元の葉から木を遡りながらマルチキャストを行うことになっているが，本質的に相違はない．

LARP 法 LPRA と同様に，根から送信元に至るパスをその他のパスと区別して扱う．

LPRA とは逆に，ある節で

- 送信元を含む枝
- 送信元を含まない枝のうちのいずれか

の両方にパケットが送られると，送信元を含む枝の下の部分全体にパケットがブロードキャストされる．それ以外の枝では，同一階層のすべての節で，それらの節の縮約前のビットマップの論理和を用いる．**図3** に三進木を用いた模式図を示す．この図で **s** が

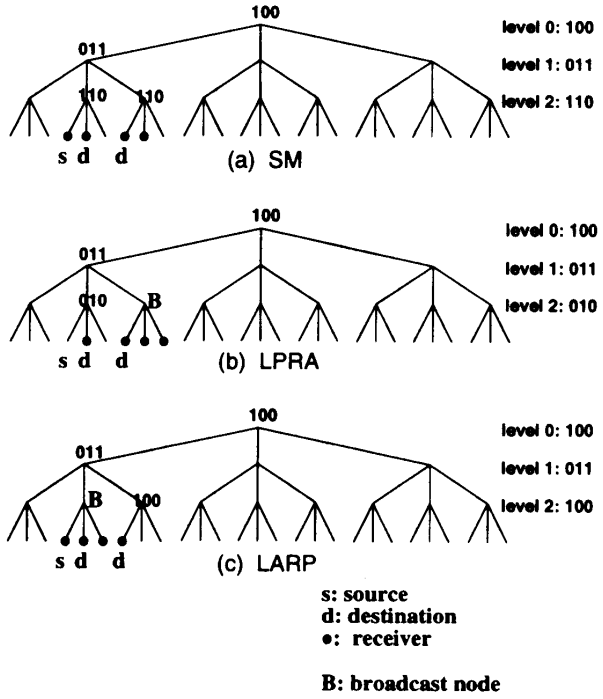

図3 ディレクトリ縮約方式
Fig. 3 Directory reduction schemes.

送信元のクラスタ，dが本来の送り先，●が結果的にパケットが送り付けられるクラスタである．したがって，dのない●の数が少ないほど無駄にパケットを受け取るクラスタ数が少ないことになる．この縮約方式を採り入れた階層ビットマップディレクトリ方式を縮約階層ビットマップディレクトリ方式と呼ぶ．

3. 階層ビットマップディレクトリ方式のRDT上での実現

階層ビットマップディレクトリ方式では木構造状にメッセージを伝達する．しかし，単純な n 進木上に階層ビットマップディレクトリ方式を実現すると，以下の問題点が生じる．

(1) 複数のノードがマルチキャストを行った場合，ルートの近くでメッセージが集中する．

(2) ツリーの葉に左から順に番号を降った場合，ツリー上の位置によっては，番号がとなりあったノードに対してパケットを送るためにも，ツリー全体のルートまで遡った後，多くの宛先に対してパケットを放送する必要が生じる（図2におけるXからYへの転送）．

これらの問題点は，文献18)中に述べられているように，周囲ノードに対する近接マップ，遠隔の1ノードに対する1対1指定を併用することにより，ある程度は解決することができる．

しかし，JUMP-1では2次元トーラスの階層構造を持つ結合網RDT (Recursive Diagonal Torus)[19),21)]を導入することにより，これらの問題の本質的な解決を図っている．

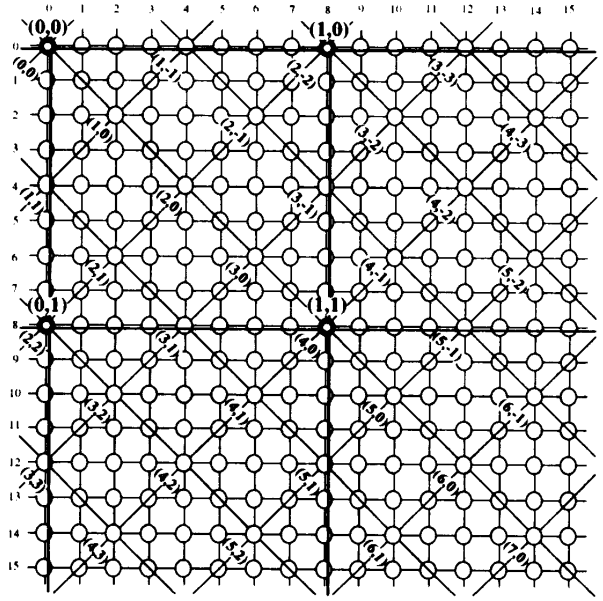

図4 RDTの構成
Fig. 4 RDT (Recursive Diagonal Torus).

3.1 RDTにおけるマルチキャストと木構造の実現

RDTでは，一定のサイズの2次元トーラス（ランク0トーラス）上に以下の方法で，再帰的に上位トーラスを構成する．まず，ランク0トーラスのあるノード (x,y) に対して，±2離れたノードに付加リンクを加えると，この付加リンクにより45度傾いた目の粗いトーラスが構成される．これをランク1トーラスと呼ぶ．ランク1トーラス上に同様のアルゴリズムで次々に上位トーラスを構成し，システムの規模によりトーラスが作れなくなったら終了する．**図4**にランク0から2までのトーラスを示す．この図では特定のノード上にのみ上位トーラスを構成しているが，すべてのノードに対して可能な限り上位トーラスを構成することができる．この構成を完全RDTと呼ぶ．

完全RDTは次数が大きすぎるので，実際のRDTは，ノードによって上位トーラスを分けて持つ．JUMP-1では各ノードは，ランク0トーラスの他にランク1から4までのうち，1つのランクのトーラスを持つ構成 (RDT (2,4,1)/α) をとっている．この構成は，1ステップのルーティングで1から4までのすべての上位トーラスが利用可能であり，65536ノードに対して，次数8で直径12を実現する．

ルーティング法は，ベクトル分解を利用した簡単な方法で実現でき，種々の並列アルゴリズムが完全RDTを念頭において設計することができる．ブロードキャ

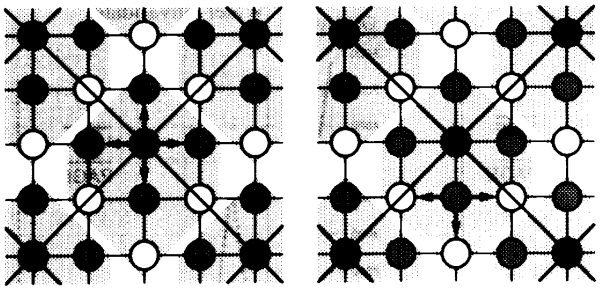

図5 1つ下のランクのノードへの基本転送パターン
Fig. 5 Multicast to the nodes of the lower rank torus.

スト，ソート，ハイパーキューブ，ツリーのエミュレーションが容易に実現可能である[20]．

階層ビットマップディレクトリ方式を実現するためには，RDT の持つツリーのエミュレーション機能を利用する．以下，特に指定しない場合，完全 RDT を対象にアルゴリズムの解説を行うが，実際に JUMP-1 で用いる RDT (2,4,1)/α 用に拡張することは容易である．

RDT によるツリーのエミュレーションは，**図5** に示す亀甲形の領域に対してのマルチキャストにより実現される．まず，ステップ1で四隣すべてのノードにデータを送る．次にステップ2で3ノードが特定方向にデータを送る．この操作は，転送を開始したノードを含めた疑似的な8進木を2ステップで構成することに相当する．

あるランク i トーラスを持つすべてのノードにおいてこのパターンで転送を行うと，このランクのトーラスを持つノードすべてにこのデータが送られる．このパケットを受け取ったノードはさらに，ランク $i-1$ トーラスに対して同様な領域にマルチキャストを行う．このことにより，各ランクに対して8進木を形成していくことができる．

RDT は上位トーラスを複数持つため，結合網全体として Fat-tree[5] が形成される．しかも実際に JUMP-1 で用いられる RDT (2,4,1)/α において各ノードは1ステップのルーティングで一番目の粗いトーラス（最大ランク）のリンクを利用することができるので，木のルートに対して直接パケットを転送することができる．このため，マルチキャストの開始時に木を遡る必要がない．同様の手法は，トーラス/メッシュに限らず，様々な形態の結合網にも同様に適用できる[11]．

3.2 実現法の詳細

RDT (2,4,1)/α におけるマルチキャストは疑似的な8進木により行われるため，階層ビットマップを実

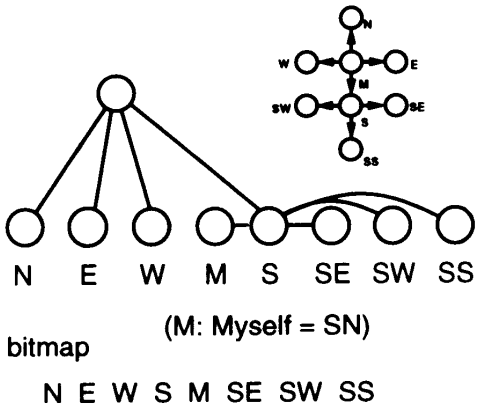

図6 階層ビットマップディレクトリ方式で用いる8進木とビットマップとの対応
Fig. 6 8-ary tree and corresponding bit-map for the hierarchical directory.

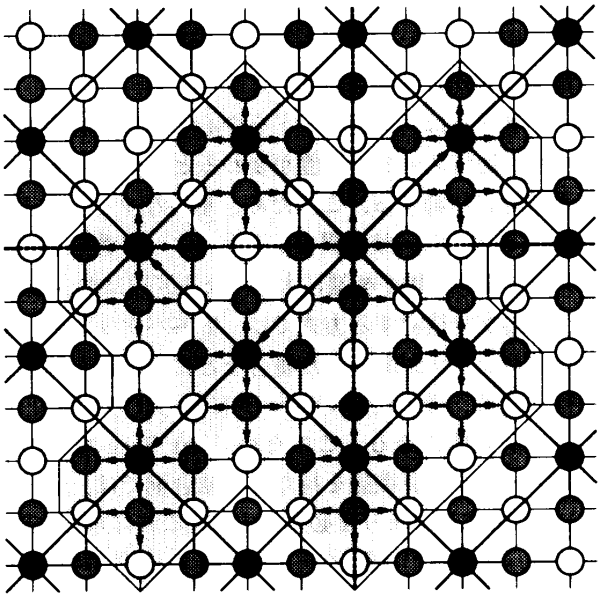

図7 マルチキャストのテリトリ
Fig. 7 "Territory" of a multicast.

現するためには，各階層で8bit のビットマップが必要となる．ここで，基本転送パターンに対し，**図6** に示すビットマップを対応させる．

あるノードが図6に示すパターンで転送するとき，どのランクからマルチキャストするかによって，パケットが届く範囲（テリトリ）が定まる．**図7** にはランク1のテリトリが示されている．ランクが大きいほどテリトリは広がっていき，マルチキャストを行うノードを中心とした不規則な同心円状になる．

この方法には以下に単純なツリー構造や Fat Tree に対して以下の利点がある．

- 上位トーラスが複数個存在するため，単純なツリー構造で問題になる上位ルートでの混雑は起こら

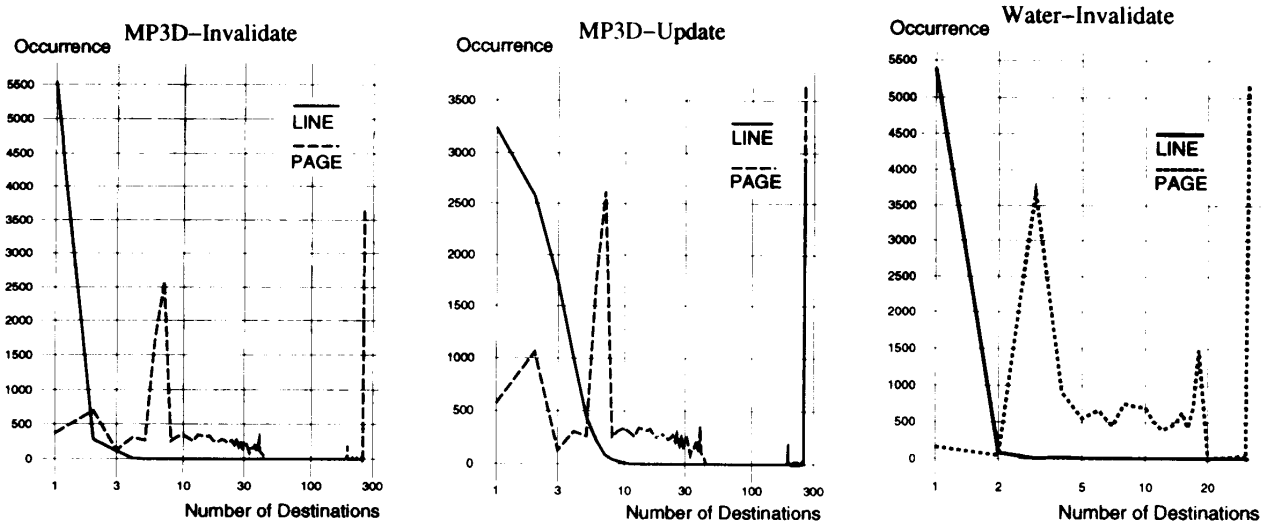

図8 メッセージ宛先数の分布
Fig. 8 Distribution of number of destinations.

ず，複数ノードでマルチキャストが起きても，通信負荷は自然に分散される．

- テリトリは，送信ノードを中心とする同心円状になり，送信ノードの位置が変わるとそれにともない連続的に移動する．このため，単純なツリーやFat Treeのように，枝が異なることによって番号が近接ノード間の距離が極端に大きくなる問題はおきない．
- 本論文で検討する方式では，応答パケットの収集時にバッファがあふれた場合などにMBPが処理を行うため，それぞれのルータに対してMBPとメモリが必要になる．このため，Fat Treeなどの中間ノードが演算機能を持たない結合網では，余分なMBPおよびメモリを必要とする．これに対しRDTでは，木構造の中間ノードが末端ノードに対し自然な形で割り振られる．

しかし，この方法の欠点は，テリトリの形状が不規則でありマルチキャストのためのビットマップを作るのが難しいことである．このため，宛先ノードの相対位置によりどのbitをセットするかを示すテーブルが用意されている．ページを共有するノードが増える際には，ノードidをキーとしてこのテーブルを引きその値をもとにディレクトリの内容を変更する．この際に必要な演算は，SM法では単純な論理和であり，LPRA，LARP法ではそれに加えて若干の条件判断が必要である．

4. ディレクトリ縮約方式の妥当性

4.1 ページを共有するプロセッサ数

前述のように，従来どおりライン単位にディレクトリを管理し，無効化型プロトコルを用いる場合，無効化メッセージの宛先は，1ないし2がほとんどであるとの評価が[1]で行われており，より現実的な方法と仮定に基づく最近の評価[2]でもこれが裏付けられている．そこで，JUMP-1で行うようにページ単位で管理を行う場合にメッセージの宛先がどのようになるかを評価した．

図8に，共有メモリのための並列プログラム集SPLASH[7]に含まれるMP3D問題（1024粒子）を256プロセッサで1反復実行した際の無効化と更新メッセージの宛先プロセッサ数の分布，およびwater問題（粒子数64，実行ステップ数1，F=MAを解くオーダ6）を32プロセッサで実行した際☆の無効化メッセージの宛先プロセッサ数の分布を示す．これは，アドレストレース[14]を基に分析したものである．ここでは，キャッシュラインの大きさを32バイト，ページの大きさを4Kバイトとして，ライン単位とページ単位で管理した場合の宛先プロセッサ数の分布をそれぞれ示している．いずれも横軸が宛先プロセッサ数を対数軸で示したもので，縦軸が発生度数である．

これらの結果から，ライン単位でディレクトリを管理すれば，特に無効化型で宛先プロセッサ数が3程度以下になる確率がかなり高いのに対し，ページ単位の管理では宛先数4~8にピークが見られ，それ以上の宛先を持つ場合もかなり多いことが分かる．更新型を用いる場合もほぼ同様な傾向を持つが，ライン単位で管理した場合の宛先数がやや増える傾向にある．したがって，ページ単位に管理し，更新型のプロトコル

☆ waterでは，トレースが大きくなりすぎたため，250万命令実行した時点でトレースの収集を打ち切った．

表1 各方式のメモリ必要量
Table 1 Memory Requirement.

N	FM	LD	HB	RHB
4096	4096	72	4680	32
32768	32768	90	37448	40

を用いる場合，マルチキャストが必要になることが分かる．

4.2 必要なメモリ量

ノード数 N が4096および32768のシステムについて，フルマップ方式（FM），リミテッドディレクトリ方式（LD），縮約を用いない階層ビットマップディレクトリ方式（HB），縮約階層ビットマップディレクトリ方式（RHB）のそれぞれの方式が1つのエントリを形成するために必要なメモリ量を**表1**に示す．ここで，リミテッドディレクトリ方式は，少なくともあるラインを平均的に共有するプロセッサ数（すなわちメッセージの宛先数）に相当するポインタを持たせるのが一般的である．図8によると，ページ単位の管理を行った場合，ブロードキャストを除いた宛先数の平均をとるとMP3Dは約6，Waterは約5である．このため，ここではポインタ数を6として評価を行った．

表1より，縮約方式はリミテッドディレクトリ方式の半分以下のメモリ必要量でディレクトリを構成することができることが分かる．なお，メモリ要求量についてはチェインドディレクトリ法[3),8)]も有利であるが，この方法は，それぞれのノードでメモリアクセスを必要とすることからレイテンシィの点では不利である．

4.3 受けとられるパケット数の評価

縮約階層ディレクトリ方式は，宛先以外のノードに対しても無駄パケットが送られてしまう点に問題がある．そこでLPRA，SM，LARPの3つについて，8の4乗である4096のクラスタから構成されるRDT上でパケットの宛先の数と分布を変化させたときに，1回のマルチキャストあたり実際にいくつのクラスタにパケットが受けとられるのかを調べた．

本来，JUMP-1ではノードの地理的な距離を考慮したスケジューリング/マッピングが前提となっている．したがって，評価は，実際のアプリケーションプログラムを現実的なマッピングアルゴリズムに基づいてマッピングした状況を仮定して行わなければならない．しかし，このような評価は，アプリケーションの性質，マッピングアルゴリズムの特性が関連するため，種々の条件の下で数多くのシミュレーションを行う必要がある．現在のところ，多数のノード数を仮定して上記のシミュレーションを行う環境がまだ整っていないため，今回は，宛先を定められた分布に従う乱数により決定し，それぞれの場合についてSun OS4.1.3の標準の乱数発生ライブラリを用いて10000回の試行を行い，平均値をとった．

RDTのランク0トーラス上で宛先の送信元からのX方向，Y方向の相対位置の分布がそれぞれ独立な（ランク0トーラス上の1リンクを単位として）標準偏差1の正規分布に従っている場合のパケットを受けとるクラスタ数のグラフを**図9**に示す．この場合，宛先数が32の場合でもパケットが送られるクラスタ数は80~180程度である．また，宛先数が多いときにLARPが有利であることが分かる．

図10は標準偏差を5としたときの同様のグラフである．この場合，パケットが送られるクラスタ数はかなり多くなる．また，宛先数が少ないときにはSMが，宛先数が多いときにはLARPが有利であることが分かる．この傾向は，他のグラフでも認められ，おおむね宛先数が10程度以下であればSMが，それ以上であればLARPが有利である．

前述のように，ページ単位でディレクトリを管理した場合，宛先数6程度にピークがある．そこで宛先数を6に固定し，標準偏差を変化させてパケットを受け取るクラスタ数を調べた結果を**図11**に示す．この結果より，つねにSMがパケットが送られるクラスタ数が最も少なく，標準偏差が大きくなっても380程度にとどまっている．

4.4 レイテンシィの評価

リミテッドディレクトリ方式をはじめとする従来の方式では宛先の数だけ順に1つずつパケットをMBPから送る必要がある．これに対し，縮約階層ビットマップディレクトリでは，MBPから送られたパケットはルータ内でマルチキャストされながら，目的ノードに送られる．したがって，結合網が空いた状態であれば，縮約階層ビットマップディレクトリは，順に1つずつ送る方式に比べて目的ノードに到着するレイテンシィの点で有利になる．

しかし一方，前章の評価によると縮約階層ビットマップディレクトリ方式は，かなりの数の無駄パケットが生じるため，このパケットが結合網やMBPへのリンクの混雑を引き起こし，レイテンシィを引き延ばす可能性がある．

そこで，シミュレーションにより，パケット転送の平均発生間隔を変化させ，目的ノードに到着するまでの平均レイテンシィの評価を行った．すべてのノードは，平均発生間隔に従ってパケットを発生する．このため，発生間隔が短ければ結合網はより混雑することになる．

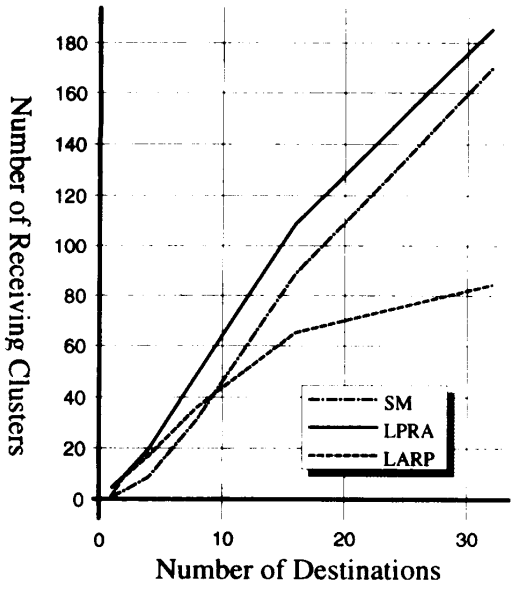

図9 標準偏差1で分布している場合
Fig. 9 Number of receiving nodes vs. number of destinations (SD=1).

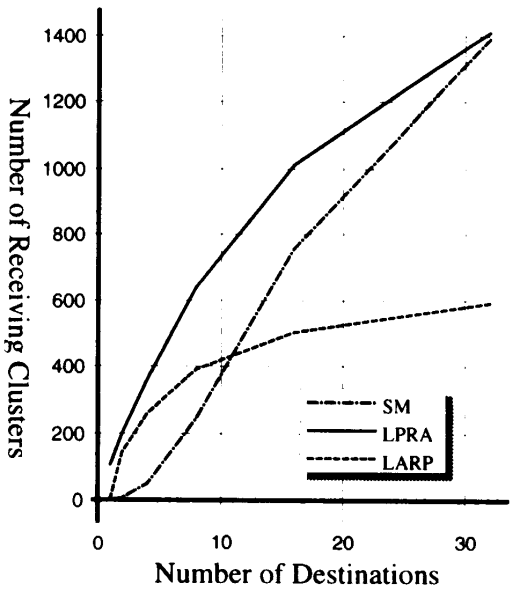

図10 標準偏差5で分布している場合
Fig. 10 Number of receiving nodes vs. number of destinations (SD=5).

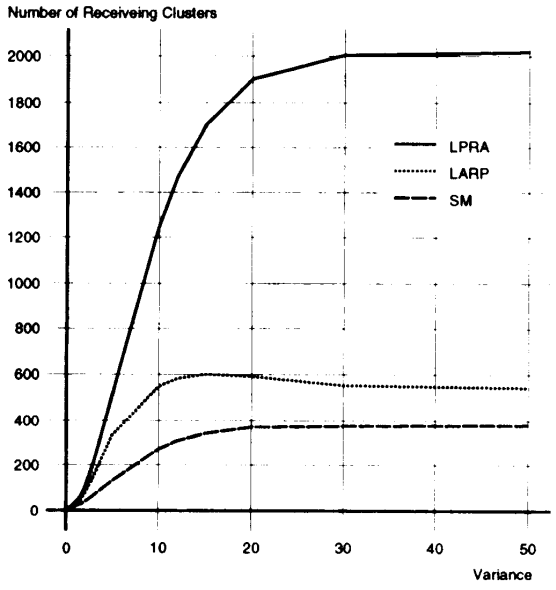

図11 宛先数6で標準偏差を変化させた場合
Fig. 11 Number of receiving clusters vs. SD.

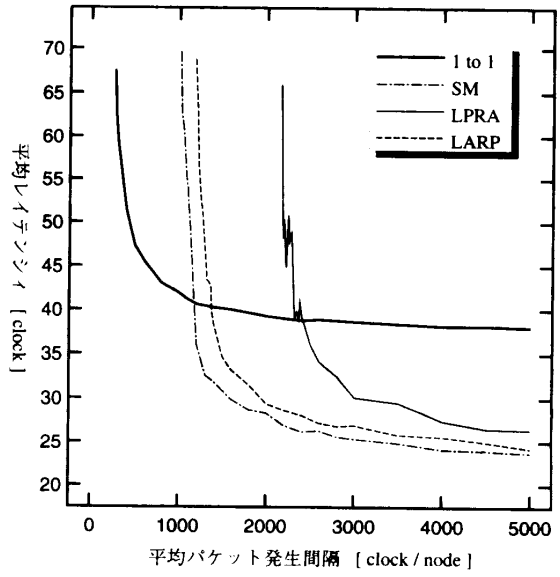

図12 パケット転送間隔とレイテンシ
Fig. 12 Latency vs. message transfer interval.

シミュレーションにおけるルータは，次節に述べるJUMP-1用RDTルータにほぼ忠実に基づいて動作する．ただし，パケットの通過時間は3クロック，パケットの長さは8フリットに固定した．図10に示すかなり無駄パケットが増加する状況（宛先数を6，宛先ノードの相対位置の標準偏差5）とした．ただし，シミュレータの制限によりノード数は $16 \times 16 = 256$ とした．この結果を**図12**に示す．

図12によると，縮約ビットマップディレクトリ方式の平均レイテンシィは，1対1に順に送る従来の方式に比べて約40%改善されている．パケットの発生間隔が短くなるにつれ，いずれの方式でもレイテンシィが悪化するが，SM方式とLARP方式は結合網の性能の限界に近づくまで，1対1に順に送る方式よりも優れたレイテンシィを実現している．これは，従来の方式ではMBPとルータ間のリンクがボトルネックになり，このリンクを待つパケットがルータ内に長時間滞留することで，無駄パケットがあるのと同様程度の混雑が結合網に生じるためと考えられる．

以上のシミュレーションにより，縮約階層ビットマップディレクトリ方式は，無駄パケットによる結合網の

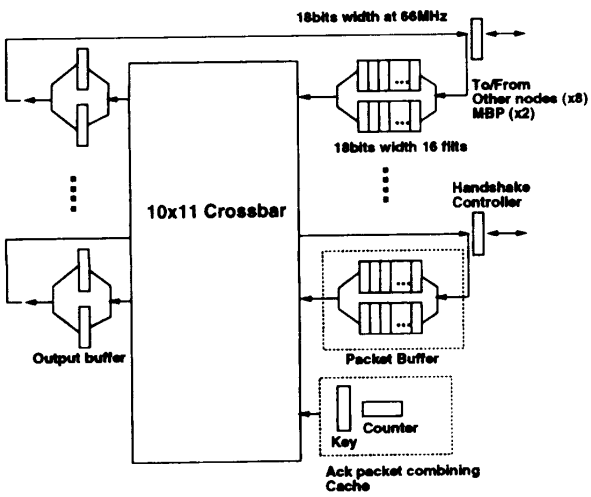

図 13 ルータチップの基本構成
Fig. 13 The diagram of the RDT router chip.

混雑を考慮しても，従来の方式に比べレイテンシィの点で有利であることが分かる．

5. JUMP-1 ルータチップ

現在，本論文で検討した LPRA，LARP，SM すべての方法をサポートした JUMP-1 ルータチップの開発が終了している．この JUMP-1 ルータチップの構成を**図 13** に示す．ランク 0 のリンクに 4 本，上位ランクのリンクに 4 本，MBP0，MBP1 にそれぞれ 1 本ずつリンクが必要なことから，全体は 10 × 10 のクロスバが基本になり，これに応答パケット（ACK パケット）収集機構が加わる．転送の bit 幅は 1 チップあたり各リンクについて 18 bit である．JUMP-1 ではルータチップ 2 個をビットスライスに用いることにより，全体で 36 bit 幅を実現する．転送周波数はプロセッサに同期した 60 MHz であり，システム全体が単一のクロックで動作する．パケット通過時間はパケットのタイプにより 5〜7 クロックである．本ルータチップは扱う最大ランクが 3 で，4096 ノード（16384 プロセッサ）のシステムに対応することが可能である．ランク間の移動には 5 クロック，ランク 3 とランク 0 以外での第 1 回目のマルチキャストには 7 クロック，その他は 6 クロックを要するため，衝突がなければどのノードへのパケットも (69 + パケット長) クロックでマルチキャストが終了する．ラインとチップのピンを有効に利用するため，通信はすべて双方向ラインにより行う．

チップは日立の 0.5 μm BiCMOS ゲートアレイ HG22S125 を利用した．このゲートアレイは，高速であるうえ，ECL 入出力を持つため，容易に転送ラインを直接ドライブすることができる．JUMP-1 の転送パケットは 16 flits (words) であり，3 flits は 2 つのチップについて共通の内容のヘッダとなる．縮約方式で用いるビットベクトルは，この共通のヘッダ部に格納される．転送方式は，Wormhole ルーティングであるが，各入力にはパケット 1 個分を格納することのできるバッファを 2 セット持ち，それぞれが 1 つのチャネルを構成する．このチャネルを使い分けることにより，デッドロックフリーな FIFO 性の保証された階層マルチキャストを実現することができる[20]．本論文で述べた 3 つのマルチキャスト方法はパケットヘッダのビットベクトルに従って，パケット単位で切替え可能である．さらに，各階層で MBP に立ちよるかどうか選択することができ，必要に応じて MBP によりテーブルを引き直してビットマップを構成し直すことにより，縮約を用いない階層ビットマップディレクトリ方式をも実現することができる．総ゲート数は約 90,000 ゲートで，ランダム部は約 50,000 ゲート，ゲート利用率は約 63%である．

6. おわりに

超並列計算機 JUMP-1 の結合網 RDT で用いられる階層ビットマップディレクトリ方式におけるディレクトリ縮約のいくつかの手法を示し，これらの手法で本来の宛先以外のクラスタに送られるパケット数を宛先を確率的に分布させて測定した．また，ページ単位のディレクトリ管理では，宛先数が 6 程度以上となる場合が多いことをアドレストレースに基づく並列アプリケーションの評価結果をもとに示した．これらの結果から，ディレクトリ縮約法の有効性をある程度示すことができた．

今回の結果からは，宛先数が少ないときには SM 法が，多いときには LARP 方が有利となっている．いずれのディレクトリ縮約法を用いるかはパケット中で指定できるため，ページ単位で使い分けることができる．実際にどのような場合にどの手法を使うかは，コンパイラや OS の処理とも関係し，今後検討する必要がある．

また，いずれの方式でも宛先数が多くなると不必要なパケット数が多くなる．これに対処する手法として，JUMP-1 では通常パケットにはアクノレッジが返されることを利用し，不必要なパケットを受け取ったクラスタからはそのアドレスへのパケットが不必要である旨をアクノレッジパケットに付加して送ることにより，アドレスごとに，木の各節の部分で下位の枝中にそのアドレスについてのパケットを必要としているクラスタがあるかどうかの情報を MBP に持たせ，これをも

とに不必要なパケットを削減する手法をとることもできる．これは先に示したディレクトリ縮約手法の2項目である，各節にディレクトリのビットマップを持たせる手法に相当する．

さらに，一万プロセッサを超える規模で分散共有メモリを実現するための転送方式としてはリンクやルータの故障を回避する方法を確立することが重要である．本論文中で提案した手法に耐故障性を導入するためには，マルチキャストを行うための木構造自体に耐故障性を持たせる必要がある．RDTは，複数の上位階層トーラスを持つため，結合網自体には耐故障性があり，1対1転送用には故障ルータやリンクを迂回することのできるルーティング法が提案されている[20]．マルチキャスト用の木構造も上記の手法を拡張して耐故障性を導入する方法が考えられるが，現状ではマルチキャスト用ビットマップを生成する方法が未検討であり，今後の課題である．

本稿で述べた機能は，すべてルータチップに組み込まれており，今後ここで示した3種のディレクトリ縮約手法を実際の並列プログラムを現実的なマッピングアルゴリズムでクラスタに割り当てた場合に関して検討してゆく予定である．さらに，今回のレイテンシィに関する評価は対象システムのサイズが小さく，また確率モデルに基づくもので十分なものとはいえない．さらに大きいシステムに関し，実際のアドレストレースに基づくシミュレーションが必要であり，今後の課題である．

謝辞 ご指導をいただいた東京大学理学部の平木敬博士，松本尚博士をはじめとする重点領域研究超並列プロジェクトメンバ各位に感謝いたします．

東京工科大学情報通信工学科の寺澤卓也氏，情報工学科の好村公一氏，福嶋泰仁氏，後藤修氏，神田和仁氏，上田広満氏にはシミュレーションにご協力いただき感謝します．

ルータチップの設計にはメンターグラフィックス社のユニバーシティプログラムを利用した．また，同チップの製作には日立製作所および日立マイコンのご協力をいただいた．深く感謝いたします．

本研究は，一部文部省科学研究費（重点領域研究(1) 課題番号04235130「超並列ハードウェア・アーキテクチャの研究」）による．

参考文献

1) Agarwal, A., Simoni, R., Hennessy, J. and Horowitz, M.: An Evaluation of Directory Schemes for Cache Coherence, *Proc. 15th ISCA*, pp.280–289 (1988).
2) Chaiken, D. and Agarwal, A.: Software-Extended Coherent Shared Memory: Performance and Cost, *Proc. The 21st ISCA*, pp.314–324 (1994).
3) James, D., Laundrie, A.T., Gjessing, S. and Sohi, G.S.: Distributed-Directory Scheme: Scalable Coherent Interface, *IEEE Computer*, Vol.23, No.6, pp.74–77 (1990).
4) Kuskin, J., et al.: The Stanford FLASH Multiprocessor, *Proc. 21st ISCA*, pp.302–313 (1994).
5) Leiserson, C.E.: Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing, *IEEE Trans. Comput.*, Vol.C-34, No.10, pp.892–901 (1985).
6) Lenoski, D., et al.: The Stanford DASH Multiprocessor, *IEEE Computer*, Vol.25, No.3, pp.63–79 (1992).
7) Singh, J., Weber, W. and Gupta, A.: SPLASH: Stanford Parallel Applications for Shared-Memory, Tech. Report, Computer System Laboratory, Stanford University (1992).
8) Thapar, M. and Delagi, B.: Distributed-Directory Scheme: Stanford Distributed-Directory Protocol, *IEEE Computer*, Vol.23, No.6, pp. 78–80 (1990).
9) Warren, D. and Haridi, S.: Data Diffusion Machien - A Scalable Shared Virtual Memory Multiprocessor, *Proc. International Conference Fifth Generation Computer Systems*, pp.943–952 (1988).
10) 工藤知宏，西村克信，楊 愚魯，天野英晴：超並列計算機JUMP-1のクラスタ間結合網RDTにおける階層マルチキャストによるメモリコヒーレンシ維持手法，情処研報，ARC 107-27, pp.1–6 (1994).
11) 工藤知宏，福嶋泰仁，好村公一，天野英晴，西村克信：超並列計算機用結合網Ring Tree on Meshの提案，信学技報，CPSY 94-61 (SSE94-128), pp.1–6 (1994).
12) Kudoh, T., Amano, H., Matsumoto, T., Hiraki, K., Yang, Y., Nishimura, K., Yoshimura, K. and Fukushima, Y.: Hierarchical Bit-Map Directory Schemes on the RDT Interconnection Network for a Massively Parallel Processor JUMP-1, *Proc. International Conference on Parallel Processing*, pp.I-186-I-193 (1995).
13) Tanaka, H., Muraoka, Y., Amamiya, M., Saito, N. and Tomita, S. (Eds.): *The Massively Parallel Processing System JUMP-1*, Ohmsha (1996). ISBN4-274-90083-5.
14) Terasawa, T. and Amano, H.: Performance Evaluation of the Mixed-protocol Caches with Instruction Level Multiprocessor Simula-

tor, *Proc. IASTED International Conference on Modeling and Simulation MS'94*, pp.1–5 (1994).
15) Hiraki, K., Amano, H., Kuga, M., Sueyoshi, T., Kudoh, T., Nakashima, H., Nakajo, H., Matsuda, H., Matsumoto, T. and Mori, S.: Overview of the JUMP-1, an MPP Prototype for General-Purpose Parallel Computations, *Proc. IEEE International Symposium on Parallel Architectures, Algorithms and Networks*, pp.427–434 (1994).
16) 松本 尚：局所処理と非局所処理を分離並列処理するアーキテクチャ，第 43 回情報処理学会全国大会 (6)，pp.115–116 (1991).
17) 松本 尚，平木 敬：超並列計算機上の共有メモリアーキテクチャ，信学技報，CPSY92-26, pp. 47–55 (1992).
18) 松本 尚，平木 敬：Memory-Based Processor による分散共有メモリ，並列処理シンポジウム JSPP '93 論文集，pp.245–252 (1993).
19) Yang, Y., Amano, H., Shibamura, H. and Sueyoshi, T.: Recursive Diagonal Torus: An Interconnection Network for Massively Parallel Computers, *Proc. 1993 IEEE Symposium on Parallel and Distributed Processing*, pp.591–594 (1993).
20) Yang, Y. and Amano, H.: Message Transfer Algorithms on the Recursive Diagonal Torus, *Proc. IEEE International Symposium on Parallel Architectures, Algorithms and Networks*, pp.310–317 (1994).
21) 楊 愚魯，天野英晴，柴村英智，末吉敏則：超並列計算機向き結合網：RDT，電子情報通信学会論文誌，pp.118–128 (1995).

(平成 7 年 9 月 18 日受付)
(平成 8 年 5 月 10 日採録)

西村 克信

1994 年東京工科大学工学部情報工学科卒業．1996 年慶應義塾大学大学院理工学研究科計算機科学専攻修士課程修了．現在，同大学院博士課程に在籍中．超並列計算機 JUMP-1 用ルータチップの研究に従事．他に，FPGA を用いた教育用マイクロプロセッサシステム等に興味を持つ．

工藤 知宏（正会員）

1991 年慶應義塾大学大学院理工学研究科博士課程単位取得退学．工学博士．現在，東京工科大学情報工学科専任講師．並列処理系アーキテクチャ，離散系シミュレーションの並列化の研究に従事．

西 宏章

1994 年慶應義塾大学理工学部電気工学科卒業．1996 年同大学大学院理工学研究科計算機科学専攻修士課程修了．現在，同大学院博士課程に在籍中．超並列計算機向きネットワーク用ルータチップの研究に従事．超並列計算機ネットワーク，ルータ，VLSI CAD 等に興味を持つ．

楊 愚魯

1984 年中国北京農業機械化学院電気科卒業．1996 年慶應義塾大学大学院理工学研究科計算機科学専攻博士課程修了．工学博士．超並列計算機・結合網の研究に従事．

天野 英晴（正会員）

1981 年慶應義塾大学工学部電気工学科卒業．1986 年同大学大学院理工学研究科博士課程修了．工学博士．現在，慶應義塾大学理工学部助教授．並列計算機の研究に従事．