計算機アーキテクチャ　107－27
（１９９４．　７．　２３）

超並列計算機でのコヒーレンシ維持のための マルチキャスト手法の検討

工藤知宏†　　松本尚‡　　平木敬‡　　西村克信§　　楊愚魯§　　天野英晴§

†東京工科大学情報工学科
‡東京大学理学部情報科学科
§慶應義塾大学理工学部

超並列計算機において分散共有メモリを実現するためには、キャッシュのディレクトリを管理する必要がある。しかし、全体を構成するプロセッサ数が非常に多いため、フルマップ方式ではベクタ長が非常に大きくなる。

この問題を解決する手法に疑似フルマップディレクトリ方式がある。この方式では、階層マルチキャストに近傍マルチキャスト、１対１通信などを組み合わせてディレクトリを実現する。階層マルチキャストとは、木構造を構成する結合網を利用し、その階層性を利用してマルチキャストを行なう手法であり、マルチキャスト先を階層化された集団によって指定することによりディレクトリ長を小さくすることができる。

本報告では、階層マルチキャスト手法のいくつかを比較検討する。

A Study on Multicast Schemes for Memory Coherency Control of a Massively Parallel Processor

T. Kudoh†　　T. Matsumoto‡　　K. Hiraki‡　　K. Nishimura§　　Y. Yang§
H.Amano§

†Tokyo Engineering University
‡University of Tokyo
§Keio University

To realize a cache coherent distributed shared memory on a large scale parallel machine, directories of cache blocks/pages should be maintained. In a massively parallel computer with more than 10000 nodes, the full-map directory scheme is unrealistic because of a large amount of required memory for the directory.

The pseudo full-map directory scheme is one of the schemes which reduce the size of the directories. In this scheme, a hierarchical multicast scheme is used combined with other schemes such as multicast to neighbors and 1-to-1 communications.

In this report, we examine some hierarchical multicast schemes for the pseudo full-map directory scheme.

1 はじめに

超並列計算機においてキャッシュを用いた分散共有メモリを実現するためには、ページ、またはキャッシュブロック毎のディレクトリを管理する必要がある。即ち、キャッシュへの書き込み時に、無効化または更新のメッセージをそのページまたはブロックを共有している全てのキャッシュに対して送らなくてはならない。超並列計算機では、全体を構成するプロセッサ数が非常に多いため、この無効化/更新をブロードキャストによって行なうと、それぞれのプロセッサが膨大な数の不必要なパケットを受けとることになってしまう。一方、あるページ/ ブロックを共有しているプロセッサの数は1から全プロセッサ数までの広い範囲をとり得るため、ビットベクタにより共有しているプロセッサを示すフルマップ方式ではベクタ長が非現実的に大きくなってしまう。

これらの問題に対処しつつディレクトリ管理を行なう手法としては、リミテッドポインタ方式[6]、チェインドディレクトリ方式[7][8]が知られている。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ポインタをキャッシュのページ/ブロックに持たせ、共有するプロセッサの数がポインタ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。

一方、本報告でとりあげる**疑似フルマップディレクトリ方式**[2][3]は、階層マルチキャストに、近傍マルチキャスト、1対1通信等の方法を組み合わせてディレクトリを実現する手法である。**階層マルチキャスト**とは、超並列計算機を構成するネットワークを階層性を持つ木構造を構成するものととらえ、この階層性を利用してマルチキャストを行なう手法であり、マルチキャスト先を階層化された集団によって指定することによりディレクトリ長を小さくすることができる。

[2][3]中で示した階層マルチキャストの方式をここではLPRA(Local Precise Remote Approximate)法と呼ぶ。本報告では、さらにSM(Single Map)法、LARP(Local Approximate Remote Precise)法を提案し、これらの手法の比較検討を行なう。

2 超並列計算機JUMP-1と疑似フルマップディレクトリ方式

現在、国内のいくつかの大学が共同でJUMP-1と呼ばれる超並列計算機の開発を行なっている[4]。JUMP-1では4プロセッサからなるクラスタ毎にMBP[1]と呼ばれる分散メモリ及びメッセージ通信の管理を行なう専用細粒度プロセッサを持ち、RDT[10]と呼ばれるプロセッサ間(クラスタ間)結合網を用いる。

JUMP-1では、ディレクトリに必要なメモリ量を削減し、ディレクトリをMBPでキャッシングして高速処理を行なうためにキャッシュブロックではなくページ単位でディレクトリを管理し、データ転送のみをブロック単位で行なう手法を用いる[3]。この場合、キャッシュブロック単位で管理する手法と比べ、共有するプロセッサ(クラスタ)数が増加する。したがってリミテッドポインタ方式ではポインタ数が不足した時に極端な性能の低下を招くし、チェインドディレクトリ方式ではリスト構造を持つディレクトリの参照に時間がかかってしまい、疑似フルマップディレクトリ方式が有利になると考えられる。

疑似フルマップディレクトリ方式で用いられる階層マップでは、コピーを持つプロセッサ(またはクラスタ)を直接指定するのではなく、管理に要するbitを減らすために、コピーを持つプロセッサのグループを指定する。この場合、コンシステンシィを維持するのに必要なパケットをグループ内の全てのプロセッサに対して送るため、実際にコピーを持っていないプロセッサでは、送られたパケットは捨てられる。すなわち無駄なパケットが生じる。この無駄なパケットの数をできるだけ少なく押えることが重要である。

3 階層マルチキャストのRDT上での実現

n進木上に階層マルチキャストを実現すると、以下の問題点が生じる。

- 複数のノードがマルチキャストを行なった場合、ルートの近くでメッセージが集中する。
- ツリー上の位置により、運が悪いノードは隣接ノードに対してパケットを送るため、ツリー

全体のルートまで遡った後、全ノードの半分にに対してパケットを放送する必要が生じる。

これらの問題点は、[3] 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、ある程度解決することができる。しかし、RDT上に実現する場合、複数の上位トーラスを利用することにより、本質的な解決を図ることができる。

3.1 RDTにおけるマルチキャストと木構造の実現

RDTは階層的にトーラスを組み合わせた形状を持つ。このため、上位ランクのトーラスの一つ一つのノードが、1つ下位のランクのトーラスを構成するノードのいくつかをそれぞれ自分の枝として持ち、このとき下位のランクの全てのノードが上位ランクいずれかのノードの枝となれば木構造が実現されたことになる。

RDTで、あるランクから一つ下のランクへのマルチキャストに用いる基本転送パターンを図1に示す。まず、ステップ1で四隣全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。

あるランク i トーラスを持つ全てのノードにおいて、このパターンを実行すると、ランク $(i-1)$ トーラスを持つ全てのノードにこのデータが送られる。

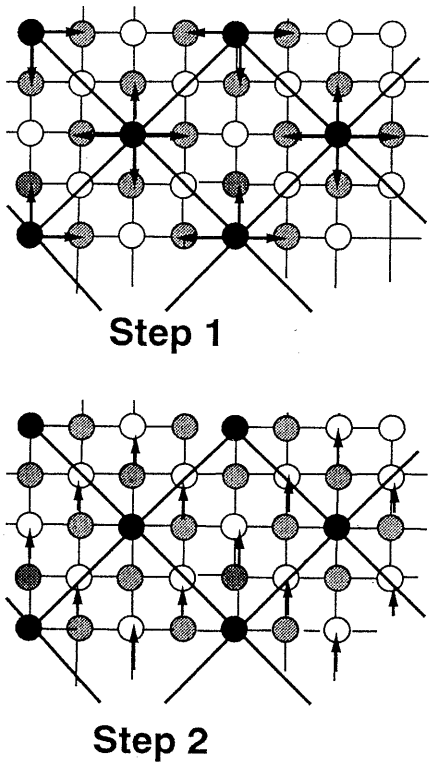

図 1: 一つ下のランクのノードへの基本転送パターン

3.2 実現法の詳細

階層マルチキャスト方式の階層は、RDTにおける各ランクのトーラスを用いる。そして、上位の階層から下位の階層へのメッセージのマルチキャストには、前述の通り、図1に示すマルチキャストの基本転送パターンを用いる。このため、RDTにおけるマルチキャストは8進木により行なわれると考える事が出来る。

マルチキャストには、各階層で8 bitのビットマップが必要となる。ここで、図2のように基本転送パターンに、ビットマップを対応させる。この場合、「自分自身」はWEに相当する。この変則的な8進木のツリーが階層化マルチキャスト機構に対応し、それぞれの階層のビットマップの組み合わせが階層化マップに対応する。

あるノードが図2に示すパターンで転送する時、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3にランク1のテリトリを示す。ランクが大きい程テリトリは広がっていき、マルチキャストを行なうノードを中心とした不規則な同心円状になる。

この方法の利点として、以下の事が挙げられる。

- 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、ノードを中心とする同心円状になり、ノードの位置によって移動する。このため、単純なツリーの場合のような枝が異なるために近接ノード間の距離が大きくなる問題はおきない。

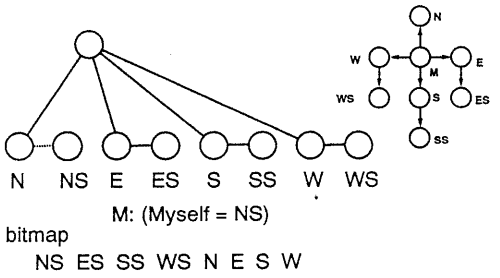

図 2: マルチキャスト用の 8 進木とビットマップとの対応

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、あらかじめ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を用意する。

4 階層マルチキャストの手法

前節で、RDT 上に木構造を構成する方法について述べた。本節ではこの木構造を用いた階層マルチキャストのいくつかの手法について比較検討する。

ここで、パケットは木の根から供給されるものと考える。プロセッサ (クラスタ) は木の葉であるから、通常の結合網では木の根に到達するにはかなりのルーティングを必要とするが、RDT では多くの木があり、どのノードにも必ず近傍にいずれかの木の根に相当するノードがあるため、データが木の根から供給されると考えて差し支えない。

根からパケットが供給される時、全ての節についてそれぞれに枝にパケットを送るかどうかを示すビットマップを指定すればフルマップを実現したことになる。しかし、これには全体で葉の数をこえるビット数の指定が必要になる。そこで、本報告で扱う階層マルチキャスト方式では、

- 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、もしくは両方の組み合わせによって階層毎に一つのビットマップを持つ。このビットマップそれぞれは RDT では 8 ビット、n 進木では n ビットである。

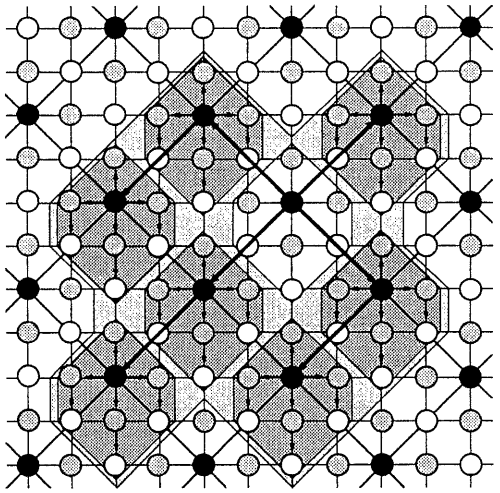

図 3: マルチキャストのテリトリ

ここでは、以下の 3 つの階層マルチキャストの方式を検討する。

LPRA 法: 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プロセッサ (クラスタ) から送られたものである。そこで送信元を含む枝とそうでない枝を区別し、ある節で送信元を**含まない**枝にパケットを送ると、その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という手法が、LPRA 法である。即ち、根から送信元の葉への経路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なう。

文献 [2][3] 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なうことになっているが、本質的に相違はない。

SM 法: 同一階層の全ての節で同一のビットマップを用いる。

LARP 法: 送信元を含む枝とそうでない枝を区別し、ある節で送信元を**含む**枝にパケットを送ると、その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全ての節で同一のビットマップを用いる。

図4に三進木を用いた模式図を示す。

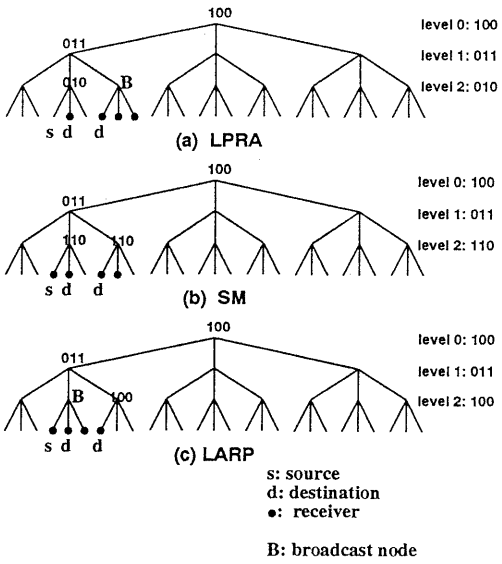

図 4: 階層マルチキャスト方式

4.1 受けとられるパケット数の予備的な評価

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時に、一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、疑似フルマップは、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な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なう必要がある。現在の所、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な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 回の試行を行ない、平均値をとった。

したがって今回の評価は予備的な評価であり、ある意味ではマッピングの効果が限定されている最悪の状況を仮定した評価である。したがって、各手法がどのような場合に有効になるかを最終的に結論づけるものではない。より現実的な評価については今後の課題である。

図5に、宛先が一様に分布している時に、宛先数を1から16まで変化させた場合の、実際にパケットを受けとるクラスタ数のグラフを示す。パケットを受けとるクラスタ数から宛先の数を引いた数が不必要なパケットの数となる。いずれの手法でも、宛先数が多いと不必要なパケット数が非常に多くなる。宛先数32の場合、いずれの方法でも全体の90%を超えるクラスタにパケットが送られてしまうことになる。しかし、宛先が一様に分布するというのは、JUMP-1 に関しては**ほとんどあり得ない仮定**であり、この結果は最悪の場合を想定したものである。

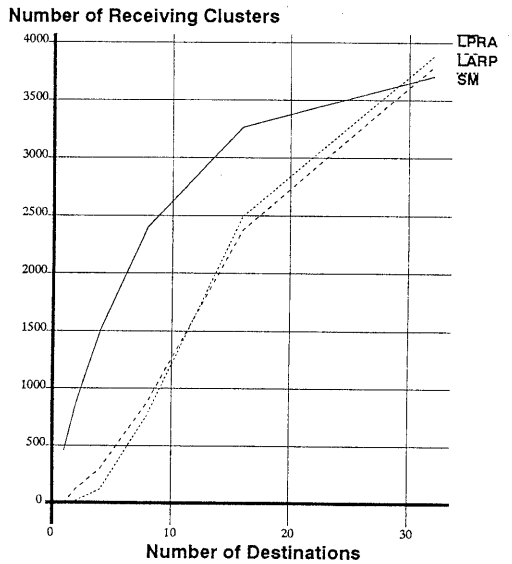

図 5: 宛先が一様に分布している場合

次に、RDT のベース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 (ベーストーラス上の1リンクを単位として) 分散1の正規分布に従っている場合について同様のグラフを図6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、宛先数が多い時に LARP

が有利であることが分かる。

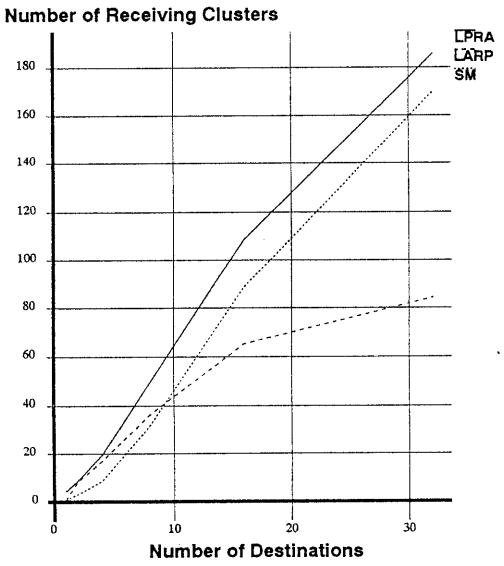

図 6: 分散 1 で分布している場合

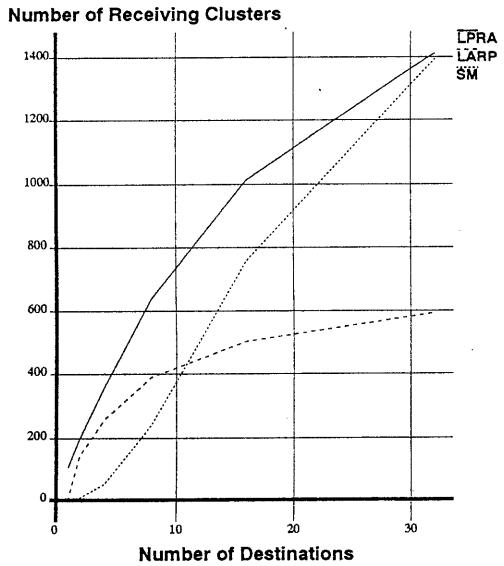

図 7: 分散 5 で分布している場合

図 7は分散を 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、宛先数が少ない時にはSMが、宛先数が多い時には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。

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ている [6]。しかし、我々のこれまでの研究から、ページ単位でディレクトリを管理した場合には、無効化型プロトコルに基づく典型的なアプリケーションでも、無効化メッセージの宛先数は 4〜5 程度までは増加することがわかっている [5]。アプリケーションの種類及び実装法を変えることにより、無効化型プロトコルでも宛先数はさらに増える可能性がある。また、更新型プロトコルを利用すれば、宛先数は確実に増加し、16 をはるかに越える数字となる。

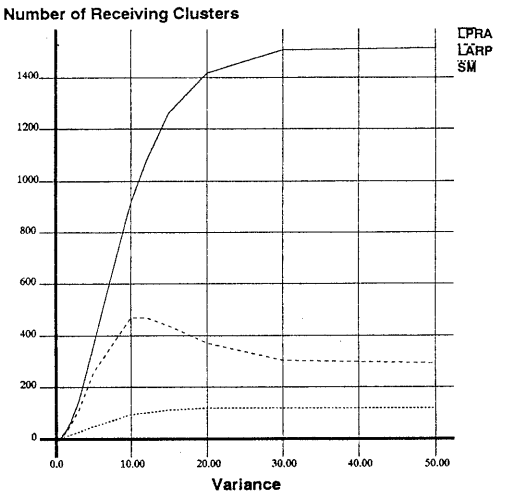

図 8: 宛先数 4 で分散を変化させた場合

まず、宛先数を控え目にとり、4 の場合について、分散を変化させてパケットを受けとるクラスタ数を調べた結果を図 8に示す。宛先数 4 では、常に SMがもっともパケットが送られるクラスタ数が少なく、分散が大きくなっても 120 程度に留まっている。

宛先数がさらに多い 16 の場合についての同様のグラフを図 9に示す。この場合はLARP がもっとも有利であるが、いずれの方法でも分散が大きい

とパケットが送られるクラスタ数が非常に大きくなる。

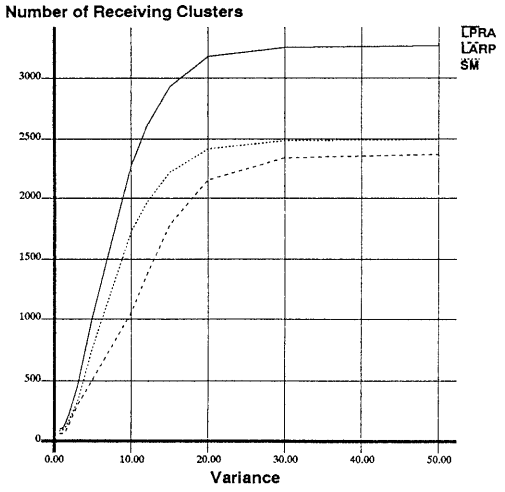

図 9: 宛先数 16 で分散を変化させた場合

5 おわりに

疑似フルマップディレクトリ方式で用いる階層マルチキャストのいくつかの手法についてマッピングの効果が限定的な場合を想定し、宛先を確率的に分布させてパケット数を測定した。今回の結果からは、宛先数が少ない時には SM 法が、多い時には LARP 方が有利となっているが、本文中に述べたように、今回の評価は、現実の状況と異なる予備的な評価に過ぎない。今回の結果のみから方式間の優劣を断定することは危険である。

さらに、今回の結果は、正規分布に基づくパケット数のみの評価であるため、LPRA が有利になる状況が見い出せないが、これは、遠隔地に対する交信が各ノードでのブロードキャストが多くなるためである。しかし、実装上はブロードキャストの方がマップを用いる場合より容易であり、Ack の収集時にもはるかに有利である。したがって、より実際的な状況を考慮した上で、宛先数に応じてそれぞれの手法を切替えて用いることが考えられる。さらに、この 3 つの手法は、ほとんど同一の機構で実装することができるので、現在、これらを切替えて用いることのできるルータを設計中である。

また、今回の評価は、階層マルチキャスト手法単独で行なった。実際の疑似フルマップディレクトリ方式では、階層マルチキャストに近傍マルチキャスト、1 対 1 通信などの手法を組み合わせて用いる。これについては、現在近傍マルチキャストの手法の検討などを行なっている。

一方、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と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、木の各節の部分で、アドレス毎に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かどうかをキャッシングしておく手法が考えられる。

今回の評価は、宛先を乱数によって発生させることにより行なったが、実際にはアプリケーション毎に宛先の数/分布が異なる。そこで、さらに精密なデータをとるためには、並列計算機上でアプリケーションを実行する際のアドレストレースを用いる方法がある。現在、インストラクションレベルの並列計算機シミュレータ MILL[9] によって生成したトレースデータを基に評価を行なっている。

謝辞

有意義な討論を頂いた東京大学理学部の松岡聡博士、猪原茂和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ー各位に感謝する。

また、本研究は、現三菱電気の加藤和彦氏の慶應義塾大学在学中の研究成果に基づくところが大きい。

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] 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 回情処全大 (6), 1991.

[2] 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, CPSY 92-36, 1992.

[3] 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, 1993.

[4] 文部省重点領域研究「超並列原理に基づく情報処理基本体系」第 2 回シンポジウム予稿集. 1993.

[5] 文部省重点領域研究「超並列原理に基づく情報処理基本体系」第 4 回シンポジウム予稿集. 1994.

[6] Agarwal A., Simoni R., Hennessy J., and Horowitz M. An evaluation of directory schemes for cache coherence. In *15th ISCA*, 1988.

[7] James D.V., Laundrie A. T., Gjessing S., and Sohi G. S. Distributed-directory scheme: Scalable coherent interface. *IEEE Computer* Vol.23 No.6, 1990.

[8] Thapar M. and Delagi B. Distributed-directory scheme: Stanford distributed-directory protocol. *IEEE Computer* Vol.23 No.6, 1990.

[9] T. Terasawa and H. Amano. Performance evaluation of the mixed-protocol caches with instruction level multiprocessor simulator. In *IASTED International Conference of MODELLING AND SIMULATION*, 1994.

[10] Y. Yang, H. Amano, H. Shibamura, and T.Sueyoshi. Recursive diagonal torus: An interconnection network for massively parallel computers. In *Proc. of 1993 IEEE Symposium on Parallel and Distributed Processing*, 1993.