

一般化されたコンバイニング機構の評価

田 中 清 史[†] 松 本 尚[†]
対 木 潤^{††} 平 木 敬[†]

大規模並列計算機システムにおいて、効率の良い並列処理を実現するためにはネットワーク上の通信量をできる限り削減することが必要である。一般化されたコンバイニング機構は、従来のコンバイニング技法を最大限に一般化したものであり、ネットワークのスイッチングノードにおける到着条件、演算機能、マッチング条件の3項目の一般化により構成される。到着条件の一般化はコンバイニングの成功率を最大限に高めることを目的とし、演算機能の一般化はあらゆるタイプのコンバイニングを可能とする。更にマッチング条件の一般化により、コンバイニング数、マッチング Key が柔軟に選択可能となる。

本稿では一般化されたコンバイニング機構の実現方法を示し、並列計算機お茶の水5号において本機構の性能評価を行ない、その有効性を検証する。

Performance Evaluation of Generalized Combining

KIYOFUMI TANAKA,[†] TAKASHI MATSUMOTO,[†] JUN TSUIKI^{††}
and KEI HIRAKI[†]

On a large-scale parallel computer system, the number of messages through an interconnection network should be reduced for the efficient parallel processing. *Generalized Combining* is a mechanism extended by ultimately generalizing existing combining methods, and consists of three generalizations on switching nodes; generalization of arrival requirement, generalization of processing function and generalization of matching requirement. Generalization of arrival requirement improves the success rate of matching as highly as possible. Generalization of processing function enables every types of combining. Generalization of matching requirement makes it possible to combine not only any number of messages, but also messages with any matching keys.

In the paper, we give an implementation method of Generalized Combining, evaluate the performance of generalized combining on OCHANOMIZ-5, parallel computer prototype, and verify its effectiveness.

1. はじめに

大規模な分散メモリ型並列計算機において、プロセッサ間通信や同期処理、共有メモリ管理など本質的に非局所的な処理が存在する。これらはネットワーク上の通信を必要とするが、一般に高速な処理を行なう要素プロセッサのデータ要求に対してネットワークは高レイテンシである。またネットワーク内のトラフィックが混雑した場合、バンド幅の上限により性能が著しく低くなる。このようなネットワークの通信処理能力の限界が高速な並列実行のボトルネックとなる。このため効率の良い並列実行を実現するためには、できるだけレイテンシの低い高速なネットワークを使用すると同時に、ネットワーク内のメッセージの数がバンド幅を大きく越えない

ように通信量を低く抑えることが重要である。

通信量の削減法の一つとして、Gottlieb [1] は、ハードウェアレベルで動的にメッセージ数を低く抑える機能を有するコンバイニングネットワークを提案した。これは同じメモリブロックへの同種のリクエストは一つにまとめられることを指摘したものであるが、この方法においてコンバイニングが成立するためには、リクエスト同士がスイッチングノードで衝突するという全く偶発的な事象が起こる必要がある。

我々は、従来のコンバイニング技法を一般化し、コンバイニングの成功率を最大限に高め、更に偶発的でなく計画的なコンバイニングを実現することを目的とした、一般化されたコンバイニング機構を提案してきた [8]。本機構は Gottlieb の方法とは異なり、コンバイニングの成立のために必ずしもメッセージ同士の衝突を必要としないことが特徴的である。

本稿では一般化されたコンバイニング機構の低コストな実装方式を述べ、部分的に実現したネットワークを用いてプログラムの実行結果及び本機構の有効性を示す。

[†] 東京大学大学院理学系研究科情報科学専攻
Department of Information Science, Faculty of Science, University of Tokyo
^{††} 富士通株式会社
FUJITSU Ltd.

2. 関連研究

共有メモリをサポートする大規模並列計算機システムでは、同一メモリ変数(同一メモリブロック)へのアクセス集中を発端としてネットワークが飽和する“ホットスポット”現象 [2] が起こる可能性がある。このアクセス競合を軽減する方法として、複数の同一メモリブロックへのリクエストがネットワークの内部スイッチングノードで衝突した場合、それらを一つにまとめて通信量を削減する“コンバイニング”が考案された。

Gottlieb [1] はこのようなコンバイニングネットワークを提案したが、各スイッチングノードでのコンバイニング数が2に限定され、コンバイニングのためのキューの長さも固定されたものであった。このような限定された機能の下では、コンバイニング可能なメッセージ同士の到着時間のずれがキューの長さを越えた場合、あるいはノード上での該当リクエストの数がコンバイニング可能な数を超過した場合にコンバイニングの成功率が低下する。

Philip Bitar [3] はこの到着時間のずれによるコンバイニング率の低下を緩和するために、スイッチングノード上での待ち時間を設定し、その時間内は先行するリクエストをノード内で待機させることが効果的であることを指摘した。しかし、リクエストのコンバイニングなどは静的に予測不可能であるため、設定した待ち時間が大きすぎた場合には、逆にレイテンシの増大のみを引き起こす危険性がある。

コンバイニングネットワークを実現する際に重要な要素となるのが、スイッチングノードにかかるハードウェアコストである。Pfister [2] らによれば、ハードウェアのサイズおよびコストはコンバイニング回路のない場合と比較して6～32倍である。Tzeng [4] はコンバイニングの対象となるメッセージのための専用ネットワークを通常のネットワークと分離して低コスト化を図った。しかし二つの異なるネットワークが必要なことと、通常のネットワークの各ステージでコンバイニングネットワークへの接続が必要であることから、この方法が低コストであるとはいえない。

3. 一般化されたコンバイニング

一般化されたコンバイニング機構は、以下の3項目の一般化を実現する。

- 到着条件の一般化
- 演算機能の一般化
- マッチング条件の一般化

3.1 到着条件の一般化

メッセージがスイッチングノードに到着する時間に対して、マッチングが行なわれる(コンバイニングが成立する)時間は、“過去”、“現在”、“ノード上での遅延時間内”および“未来”の4つのカテゴリーに分類され

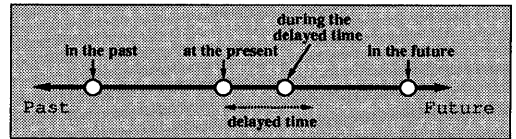


図1 Time frame of matching point of combining.

る。ここで、“過去”はメッセージがノードに到着したときコンバイニングが既に確定していることを意味する。“現在”は到着と同時にマッチングが行なわれることを意味する。“遅延時間内”はノードに到着してからノードを出発するまでの通過時間内にマッチングが行なわれることを意味する。従来のコンバイニング技法はこのタイプに含まれる。更にこの“遅延時間(待合せ時間)”はゼロから無限大まで可変(設定可能)であるとする。ゼロの場合、メッセージは後続する到着メッセージを待つことなくノードを通過する。無限大の場合は対象となるメッセージが全て到着するまでノード上で待機する。最後に“未来”はメッセージがノードを通過した後でマッチングが行なわれることを意味する。図1に4種類のマッチングポイントを示す。

3.2 演算機能の一般化

従来のコンバイニングネットワークは少数の種類の演算機能、すなわちメモリに対するFetch&Op [1] のバリエーションのみを提供するものであった。一般化されたコンバイニング機構において、演算機能の一般化によりスイッチングノードは到着メッセージに対してあらゆる演算が適用可能であるものとする。これにより、共有メモリ管理におけるAcknowledgeメッセージのコンバイニング [5] やバリア同期、更にスイッチングノードが数値演算機能を持つことによりネットワークによるリダクション数値演算が可能となる。

3.3 マッチング条件の一般化

マッチング条件の一般化は以下の2つからなる。

a) コンバイニング数の一般化

コンバイニングの対象となるメッセージの数はゼロから無限大まで指定可能とする。指定した数のメッセージが全て到着した時点でコンバイニングされたメッセージが発火する。ただし全てのメッセージが到着する以前に設定した遅延時間を越えた場合は指定した数のコンバイニングは成立しない。このため指定した数のコンバイニングを必ず成立させたい場合は遅延時間を無限大に設定する必要がある。

b) マッチングKeyの一般化

任意のマッチングKeyを任意数指定可能とする。この一般化により、従来のコンバイニングが使用したメモリアドレスのみならず、プロセッサIDやコンバイニングの相手を明示的に指定するKeyを選択することが可能となる。

3.4 特徴

- 3種類の発火条件

コンバイニングされたメッセージが発火するための条件は3種類存在する。遅延時間内のコンバイニングにおいては、遅延時間(待合せ時間)が経過した場合と指定したコンバイニング数のメッセージが全て到着した場合である。この場合、いずれか一方が満たされたときに発火する。更に未来及び過去のコンバイニングにおいては、マッチングポイントと発火ポイントは完全に独立している。先行メッセージは後続メッセージを待つことなく単独で発火する。

- 不完全マッチングの存在

遅延時間内のコンバイニングにおいて、先行メッセージが到着してから遅延時間を経過した時点でコンバイニングの対象となる全ての後続メッセージが到着しなかった場合、意図したメッセージ数のコンバイニングは成立しない。メモリアクセスリクエスト同士のコンバイニングなどは静的に予測不可能なためこの不完全マッチングが存在する。

- データの溢れ

一般化されたコンバイニング機構において、各スイッチングノードは到着するメッセージを格納するためのメモリ空間を有することを仮定している。しかしノード上に到着して待機するメッセージの数は静的に予測不可能である。特に設定した遅延時間やコンバイニング数が大きい場合にはこのデータ数のメモリ量に対する爆発が起こる可能性がある。これに対する対処法は以下の通りである。現実的には大規模並列システムはNUMA型で構成されることから、溢れを起こしたスイッチングノードに最も近い要素プロセッサが存在する。溢れを検出したスイッチングノードは、その最も近傍のプロセッサに割り込みメッセージを送信して処理を依頼する。

4. 一般化されたコンバイニングの実現

一般化されたコンバイニング機構を並列計算機プロトタイプお茶の水5号(OCHANOMIZ-5) [9,10] の2進ツリー状ネットワークに実装する。

4.1 到着条件の一般化の実現

メッセージのノード上での遅延時間(待合せ時間)を4段階で指定する。

- レベル0

待合せ時間はゼロ。すなわちメッセージはノード上で待機しない。(ただしスイッチングに要する遅延時間は存在する。)このレベルのメッセージはコンバイニングの対象とならない。

- レベル1

待合せ時間はゼロ。ただしスイッチングノードの通過時にコンバイニングユニットを経由する。メッセージがリクエストタイプの場合ウェイトバッファを参照し、先行リクエストならばエントリをセットしてリクエストをフォワードする。後続リク

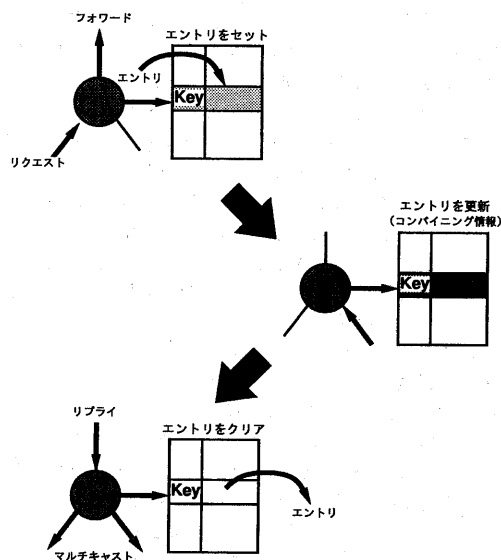


図2 レベル1のメッセージに対する操作

エストならばコンバイニング情報を付加してエントリを更新する。メッセージがリプライタイプの場合、エントリを参照し必要ならばリプライをマルチキャストする。この際エントリをクリアする。このレベルは予測不可能型コンバイニングに対する、未来のマッチングを実現するためのものであり、ウェイトバッファ内に該当エントリが存在する間は後続する同種のリクエストはコンバイニングされる。図2にメッセージ通過の際の操作を示す。

- レベル2

メッセージはノード上で有限時間待機することを意味する。実際の待ち時間は4種類指定可能とする。各時間はスイッチングノードの再構成により可変である。予測不可能型コンバイニングであるFetch&Addのためのレベルである。

- レベル3

待ち時間が無限大であることを意味する。コンバイニングの対象となるメッセージが全て到着するまで待機し、全てが到着した時点でコンバイニングされたメッセージが発火される。予測可能型コンバイニングのためのレベルである。

従来のコンバイニングはメッセージ同士がスイッチングノード上で衝突する場合に限定されていたが、上記のレベル1の場合には衝突の必要がない。予測不可能型のコンバイニングに対する低オーバーヘッドかつ到着時間差に対する制約が最も緩和された方法である。

4.2 再構成による演算機能の一般化の実現

各ノードで全ての演算機能を同時に実現するのはハードウェアの制約から困難である。再コンフィグレーション可能なFPGAを使用して、アプリケーションの高速

実行に最も適した演算機能を再構成によって実現する。実際には、Fetch&Add やリダクション数値演算などの数値演算回路部分が再構成の対象となる。

4.3 マッチング条件の一般化の部分的実現

お茶の水5号のネットワーク構造は2進木であることから3つ以上のメッセージのコンバイニングの可能性は低いことが予測される。(レベル1あるいはレベル3のメッセージを使用する限り起こり得ない。)このことからコンバイニング数を2以下に制限する。よってコンバイニング数の指定はレベル指定(level-0 or others)で置き換え可能である。マッチングKeyとして、階層化されたElastic Barrierの場合はプロセッサID、その他の場合は物理メモリアドレス空間を使用する。

4.4 コンバイニングの演算機能

一般化されたコンバイニング機構を実現するネットワークがサポート可能であり、大規模並列計算機システムの効率の良い実行を支援する機能に以下のようなものがあり、これらをお茶の水5号で実装する。

- 分散共有メモリにおけるリモートメモリアクセスに伴う Acknowledge メッセージのコンバイニング MBP(Memory-Based Processor) [5] で提案されたコンバイニングである。共有メモリのコンシステンシ管理において Acknowledge メッセージをコンバイニングによって効率良く回収してネットワークの負荷を軽減する。この場合、到着する Acknowledge メッセージ数はスイッチングノードでマルチキャストした時点で既知であるので、コンバイニング数をその数にして、かつ待合せ時間を無限大に設定しコンバイニングを行なう。
- 階層化された Elastic Barrier [7]
Elastic Barrier [6] を大規模並列システムに拡張したものであり、多重発行した同期コマンドを制御するコントローラと、ハードウェアによるバリアのコンバイニングツリーによって実現する。バリア同期の性質から待合せ時間は無限大、マッチングKeyはプロセッサグループIDである。
- メモリベース Elastic Barrier [5]
共有メモリとコンバイニングネットワークによるバリア同期である。ハードウェアによる Elastic Barrier よりもオーバーヘッドが大きくなるが、グループ構成に起因するスケジューリングの制約を受けず、より多重のグループ構成が可能となる。待合せ時間は無限大、マッチングKeyは同期ポイントに対応して割り当てられた共有変数の物理メモリアドレスを指定する。
- 同期不可分命令 (Test&Set, Fetch&Add) のコンバイニング
Test&Set はレベル1、Fetch&Add はレベル2のメッセージで実現する。Fetch&Add に関しては待合せ時間を設定することにより、Gottlieb [1] の方法よりも計画的にコンバイニングの成功率を上げ

ることが可能である。

● リダクション数値演算機能

通信データを利用してネットワークがシストリックアレイ的にリダクション数値演算を行なうことにより計算の高速化を実現する。総和、最大(小)値、行列積、ソーティングなどをサポートする。待合せ時間は無限大であり、再構成によって使用する数値演算回路を提供する。

● リモートメモリアクセスのリード及びインバリデートリクエストのコンバイニング

同一アドレスへのリードリクエストのコンバイニングをレベル1のメッセージで行なう。同様にインバリデートリクエストのコンバイニング [8] を行ない、Home クラスタでの処理を軽減する。これらのコンバイニングのマッチングKeyは物理メモリアドレスを使用する。

上記の中で階層化された Elastic Barrier のみは専用ハードウェアによって実現し、その他はネットワークスイッチングノード内のコンバイニングユニットによって実現する。各メッセージタイプとメッセージレベルの対応を表1に示す。

4.5 ネットワークスイッチングノードの構成

ネットワークの内部スイッチングノードは、コンバイニング機構を実現するために以下の要素から構成される。

- スイッチ要素およびそのコントローラ (Router)
ノードの親、2つの子、およびコンバイニングユニットの間で、4×4クロスバースイッチを構成する。
- 外部メモリで構成されるマッチングストア
レベル1～3用のウェイトバッファ、およびレベル2における待合せのためのコンバイニングキューを構成するメモリ空間を外部のSRAMによって確保する。なお、バッファ探索のオーバーヘッドを削減するためにハッシュ表を構成する。
- 演算器とコントローラ
スイッチからコンバイニングユニットに入ってくるデータをレジスタにセットし、レジスタメモリ演算を行なう。再構成により異なる演算を可能とする。
- 溢れ検出回路
各ハッシュキーに割り当てられた空間ごとに溢れを検査する。溢れを検出した場合は、割り込みパケットを生成し、近傍のプロセッサへ送る。
- 階層コントローラ (HC)、グループレジスタ
階層化された Elastic Barrier を実現するために、階層コントローラと同期グループを指定するグループレジスタを持つ。

4.6 必要ハードウェア量

お茶の水5号のスイッチングノードはSRAMベースのFPGAを使用している。コンバイニングユニットを

表1 Levels of message types

Level	Waiting Time	Message Type	Remark
0	zero	regular messages	no-combining
1	zero	Read, Invalidate, Test&Set	future combining
2	specified time	Fetch&Add	four different prepared times
3	infinity	Ack, Barrier, Reduction computing	complete combining

付加した場合のハードウェアの増加量は、付加しない場合と比較して50%程度であった。(Xilinx社のLCAを使用し、内部のfunction generatorおよびCLBの使用率により算出した。ここで、外部のSRAMは含まれていない。) Pfister [2] らの見積り(6~32倍)と比較すると、本方式は十分低コストであるといえる。

5. 性能評価

現在、メモリのリードリクエスト、Test&Setおよび階層化されたElastic Barrierの3種類のコンバイニングについて動作している。ここで、リードリクエストとTest&Setについてはレベル1のメッセージで実装し、階層化されたElastic Barrierについては独立したハードウェアによるレベル3の実装となっている。

5.1 リードリクエストとTest&Setのコンバイニングによる性能向上の上限

最初の例は、8台のプロセッサが物理メモリの同一アドレスを連続して読み出す場合である。実際のアプリケーションでは起こり得ない状況であるが、予測不可能型のコンバイニングによる性能向上の上限を知るために実測した。結果は表2の通りである。コンバイニングをレベル1で実装した場合、コンバイニングを行わない(レベル0)場合に対して実行時間が66%短縮された。

次にTest&Setを使用して各プロセッサが同回数Lockを獲得するプログラムの実行時間を測定した。クリティカルセクション内でカウンタ変数をインクリメントしている。結果は40.6%の実行時間の短縮が得られた(表3)。コンバイニングが成功しても一方はLockの獲得に失敗してリクエストを再発行するので、リードリクエストの場合と比較すると向上率は低くなる。

5.2 行列の巾乗

行列の M 乗を計算するプログラムの実行時間を測定した。ここで行列は 64×64 で、 $M = 32$ である。実行の流れを図3に示す。

表2 Execution time of read requests

	Total time (sec)	speed up
without combining	34.44	
with combining	11.80	-66%

表3 Execution time of Test&Set requests

	Total time (sec)	speed up
without combining	236.86	
with combining	140.66	-40.6%

最初に、バリア同期をソフトで実現した場合の結果を図4に示す。図の横軸は計算に参加したプロセッサ数、縦軸は実行時間である。ここで、ソフトウェアバリアはTest&Setとカウンタ変数によって実現されており、各プロセッサは同期ポイントに到達したときにカウンタを1インクリメントする。カウンタの値がプロセッサ数と同数になったときバリアが成立する。

図の凡例は上から、コンバイニングを行わない場合、リードリクエストのみコンバイニングを行なった場合、リードリクエストとTest&Setの両方でコンバイニングを行なった場合である。8台の実行の場合、コンバイニングを全く行わない場合に対して、リードおよびTest&Setのコンバイニングを行なったときに実行時間が2.9%短縮された。図中の6台の場合、リードおよびTest&Setのコンバイニングを行なった場合が最も悪い結果を示しているが、これはスイッチングノード内のコンバイニングユニットの使用において競合が起こっているためと考えられる。

次にバリア同期にハードウェアバリアを用いた場合の結果を図5に示す。ソフトウェアバリアと比較すると実行時間に関して良い結果を示した。ここで、コンバイニングに焦点を絞るためにElastic BarrierのReal Requestのみを使用した。ハードウェアバリアを使用した場合、8台実行のときを比較すると、実行時間が最

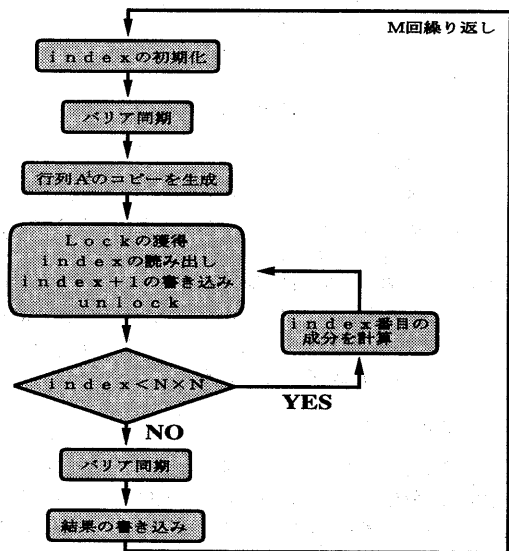


図3 行列の巾乗計算のフローチャート

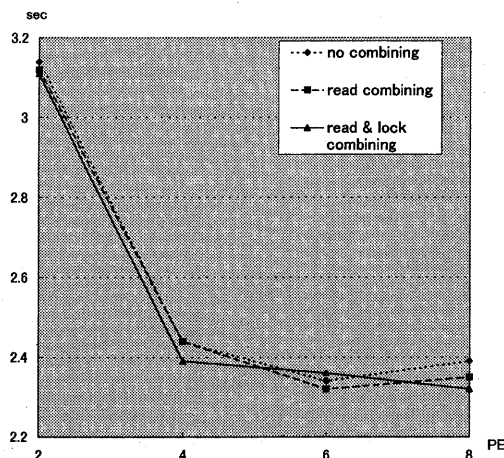


図4 ソフトウェアバリア使用

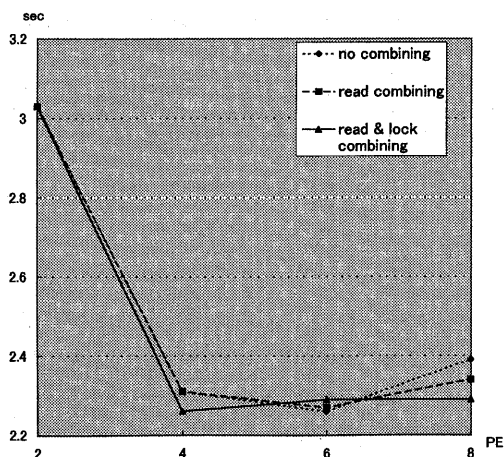


図5 ハードウェアバリア使用

大4.2%短縮した。

今回実装したレベル1のメッセージは、コンバイニングの対象とならないレベル0のメッセージと比較すると、コンバイニングユニットの通過及びウェイトバッファの参照の時間だけレイテンシが大きくなる。それにもかかわらず、コンバイニングを行わない場合に対してより良い実行結果を示したことは、レベル1のコンバイニングの可能性を示している。

6. おわりに

本稿では、大規模並列計算機上で効率の良い通信を支援する一般化されたコンバイニング機構の実現方式を示した。今回の実装において一般化されたコンバイニングを部分的に実現し、3種類のコンバイニングについて実

行結果を示した。いずれの場合もコンバイニングによる性能向上が得られた。また、本方式は低コストのハードウェアで実現可能であることを示した。

今後は Acknowledge メッセージのコンバイニングなど他の種類のコンバイニングを実装し、より実用的な規模のアプリケーションを用いて一般化されたコンバイニング機構の総合的な評価、検証を行なう予定である。

謝辞

本研究は通商産業省 RWC プロジェクトの一環として行われた。お茶の水5号の開発に当たり協力していただいた日本サンマイクロシステムズ株式会社ならびにデジタルテクノロジー株式会社に深く感謝の意を表します。また、Mentor Graphics社と Synopsys社の University Programを用いた。両者に深く感謝します。

参考文献

- 1) Allan Gottlieb, Ralph Grishman, Clyde P. Kruskal, Kevin P. McAuleffe, Larry Rudolph, and Marc Snir: The NYU Ultracomputer-Designing an MIMD Shared Memory Parallel Computer. IEEE Transactions on Computers, 32(2):175-189 (Feb. 1983).
- 2) Gregory F.Pfister and V.Alan Norton: "Hot Spot" Contention and Combining in Multistage Interconnection Networks. IEEE Transactions on Computers, 34(10):943-948 (Oct. 1985).
- 3) Philip Bitar: Combining Windows: The Key to Managing MIMD Combining Trees. The workshop on Scalable Shared Memory Multiprocessors. (May 1990)
- 4) Nian-Feng Tzeng: A Cost-Effective Combining Structure for Large-Scale Shared-Memory Multiprocessors. IEEE Transactions on Computers, 41(11):1420-1429 (Nov. 1992)
- 5) 松本尚, 平木敏: 超並列計算機上の共有メモリアーキテクチャ. 電子情報通信学会技術研究報告, CPSY92-26, pp.47-55 (Aug. 1992).
- 6) [] 松本尚: 細粒度並列実行支援機構. 情報処理学会研究報告, ARC Vol.89, No.60, pp.91-98 (Jul.1989)
- 7) 松本尚, 平木敏: 拡張された Snoopy Spin Wait と階層化された Elastic Barrier. 第47回情報処理学会全国大会講演論文集 (4), pp.43-44 (Oct. 1993).
- 8) 田中清史, 対木潤, 松本尚, 平木敏: 一般化されたコンバイニング機構. 情報処理学会研究報告, ARC Vol.96, No.13, pp.31-36 (Jan. 1996)
- 9) 田中清史, 対木潤, 松本尚, 平木敏: 並列計算機プロトタイプお茶の水5号. 第3回FPGA/PLD Design Conference & Exhibit 応用技術論文集, pp.505-514 (Jul. 1995).
- 10) 田中清史, 対木潤, 松本尚, 平木敏: 汎用並列計算機プロトタイプお茶の水5号の予備評価. 電子情報通信学会技術研究報告, CPSY96-53, pp.47-54 (Aug. 1996).