計算機アーキテクチャ 128－9
ハイパフォーマンス 70－9
コンピューティング
（1998. 3. 5）

SMP クラスタ COMPaS の性能評価

田中良夫† 松田元彦† 佐藤三久†

我々は Pentium Pro を 4 台搭載した対称型マルチプロセッサ (SMP:Symmetric Multi-Processor) であるサーバ型 PC8 台を Myrinet と 100Base-T Ethernet で結合した SMP クラスタ COMPaS を構築した．また，Myrinet 上に低オーバヘッドで高いバンド幅を提供するリモートメモリ転送に基づくユーザレベルの通信レイヤを設計・実装した．本稿ではいくつかのベンチマークプログラムによる COMPaS の性能を示す．COMPaS の性能は，データの局所性およびノード間通信の負荷の大きさに依存する．主記憶へのアクセスが集中するようなプログラムにおいてはメモリバスの性能がネックとなるが，データの局所性が高く，キャッシュを有効に利用することができる場合には高い性能を得ることができる．

Performance Evaluation of SMP Cluster, COMPaS

YOSHIO TANAKA ,† MOTOHIKO MATSUDA † and MITSUHISA SATO†

We have built an eight node SMP cluster called COMPaS (Cluster Of Multi-Processor System), each node of which is a quad-processor Pentium Pro PC. We have designed and implemented a remote memory based user-level communication layer which provides low-overhead and high-bandwidth. In this paper we report on the performance of COMPaS. The performance of COMPaS is affected by data size and access patterns. Although the performance is limited by the low memory bus bandwidth of PC-based SMP nodes for some memory intensive workloads, we can get the high performance on COMPaS if we can get the high data locality and take the advantage of cache,

1. はじめに

我々は 4 プロセッサ (Pentium Pro, 200MHz) の PC サーバ 8 台を Myrinet[1] および 100Base-T Ethernet で結合した SMP クラスタ **COMPaS(Cluster Of Multi-Processor System)** を構築した[5),6)]．SMP クラスタにはコンパクトな大きさで多くのプロセッサを搭載でき，保守，管理も容易になるという利点がある．複数のスレッドがメッセージパッシングによりノード間通信を行なうと，バッファに対して排他処理が必要となることやメッセージのコピーによりメモリバスに対する負荷が増加するなどの問題が生じる．これらの問題を解決するために，我々は SMP クラスタにおけるノード間通信の手段としてリモートメモリ転送に基づく Myrinet 上のユーザレベル通信レイヤとして **NICAM**[7] を作成した．NICAM は低オーバヘッドかつ高いバンド幅のリモートメモリ転送および同期プリミティブを提供する．NICAM は Myrinet のネットワークインタフェース (NI) 上の DMA エンジンを用いてデータの転送を行い，データ転送は CPU に負荷をかけずに NI 上のマイクロプロセッサによって行なわれる．NICAM は Myrinet の NI 上に Active Message(AM)[4] 機構を実装して通信を行なう．メッセージパッシングにおいては，メッセージの送り手と受け手はメッセージを送受信する際に陰に同期をとることができるが，リモートメモリ転送の場合には各ノードはリモートメモリに非同期に書き込みを行なう．データ並列プログラムなどではすべてのノードが同期的に計算を進めていくため，高速な同期プリミティブを提供する必要がある．NICAM は CPU-NI 間のオーバヘッドを削除することにより，高速なバリア同期を提供する．

SMP クラスタのアーキテクチャは分散メモリアーキテクチャと共有メモリアーキテクチャの中間的な形態であり，各ノード間ではメッセージパッシングやリモートメモリ転送により通信を行ない，各ノード内のプロセッサ間では共有メモリを介して通信を行なう．SMP クラスタで高い性能を得るためには，両方のアーキテクチャの利点を引き出す必要がある．SMP システムが安価となり広く普及しつつある現在，SMP クラスタはクラスタシステムの 1 つの有効な選択肢となっているが，プログラミングおよび性能特性に関して未知の部分が多い．

† 新情報処理開発機構
Real World Computing Partnership

我々は文献[5]においてSMPクラスタ上でのプログラミングモデルとして共有/分散融合プログラミングモデルを提案し，100Base-T Ethernet上でのmpichとSolarisスレッドの組合わせでいくつかのベンチマークプログラムを実装・実行し，COMPaSの性能に関して報告を行なった．結果としてはメモリバスの性能がボトルネックとなり，高い性能を得ることはできなかった．本稿ではMyrinet上でのNICAMとSolarisスレッドを用いてラプラス方程式の陽解法，行列の乗算，radixソートおよびCGカーネルを実装・実行した結果を報告する．行列の乗算のプログラムにおいては，SMPの利点を活かすためにキャッシュを有効に利用するようなプログラミングを行なった．

本稿では，NICAMの概略とその性能，ベンチマークプログラムによるCOMPaSの性能，およびCOMPaSの性能特性に関して報告する．次節では我々が作成したCOMPaSの仕様およびノードプロセッサの基本性能を示す．3節ではNICAMの仕組みおよびノード間通信の基本性能に関して説明する．4節ではCOMPaSの性能測定に関する実験結果を示し，5節ではCOMPaSの性能特性に関して考察し，最後にまとめを行なう．

2. COMPaSの仕様とノードプロセッサの基本性能

本節ではCOMPaSの仕様およびメモリバスのバンド幅やスレッド間のバリア同期などのノードプロセッサの基本性能に関して述べる．図1にCOMPaSの仕様を示す．COMPaSは4台のPentium Proを搭載したPCサーバ東芝GS700を8台100Base-T EthernetとMyrinetにより接続したシステムである．GS700は4-Way SMPのPentium Pro PC(200MHz, 450GX chip-set, 16KB L1キャッシュ, 512KB L2キャッシュ, 128MB 主記憶)である．各ノードのオペレーティングシステムはSolaris2.5.1である．

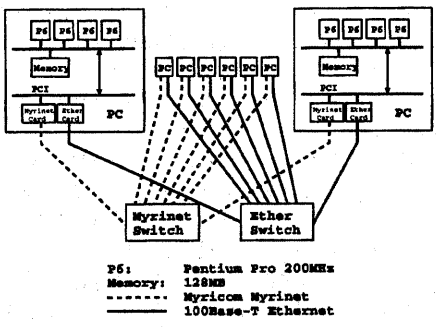

図1　COMPaSの仕様

2.1 メモリバスの性能

表1にバスのバンド幅を，表2に複数のスレッドでbcopyを行なった場合のバンド幅を示す．表2の2番目の列はスレッドあたりのバンド幅の平均を示し，3番目の列は全スレッドのバンド幅の合計を示している．全スレッドのバンド幅の合計はスレッド数には関係なく一定であり，複数のスレッドを起動した場合にはスレッドあたりのバンド幅はスレッド数に反比例して低くなることがわかる．

表1　バスの性能

read(MB/s)	write(MB/s)	copy(MB/s)
203.5	97.9	70.5

表2　複数スレッドでbcopyを行なった場合のバンド幅

スレッド数	バンド幅 (MB/s)	全体のバンド幅 (MB/s)
1	71.31	71.31
2	37.06	74.12
3	24.75	74.25
4	18.56	74.24

2.2 スレッド間のバリア同期

表3にスレッド間でのバリア同期にかかる時間を示す．本バリアはスピンロックを用いており，OSによって提供されるmutex変数やcondition変数は用いていない．本バリアは4スレッドで2マイクロ秒以下と，高速なバリア同期を提供することができる．Solarisのmutex変数を用いたバリア同期の場合は4スレッドで約180マイクロ秒かかってしまう．

表3　スレッド間のバリア同期

スレッド数	2	3	4
バリア同期の時間 (μsec)	1.222	1.761	1.960

3. NICAM: Myrinet上のユーザレベル通信レイヤ

SMPクラスタにおいては，各ノード内で複数のスレッドが動作する．ノード間通信をメッセージパッシングにより行なうと，メッセージバッファに対する排他処理が必要となる．また，メッセージのコピーによりメモリバスに対する負荷が増加してしまう．リモートメモリ転送はこれらの問題を回避することができる．我々はリモートメモリ転送に基づくMyrinet上のユーザレベル通信レイヤであるNICAMを設計，実装した．NICAMは低オーバヘッドで高いバンド幅のデータ転送および同期機能を提供する．本節ではNICAMの仕組みおよび性能に関して述べる．

3.1 Active Message を用いた通信プリミティブ

NICAM は NI 上の Active Message(AM) の仕組みを用いたリモートメモリ転送を行なう通信レイヤである．リモートメモリ転送には

- 通信に関わる排他処理を最小限にする
- バスへの負荷を抑えることができる

という利点がある．また NICAM は NI 上の DMA エンジンを用いてリモート転送を行なうことにより，データ転送に CPU が関与しないようにしている．SMP クラスタにおいては CPU オーバヘッドはバスを占有し，他の CPU に影響を与えてしまうため，NICAM は CPU オーバヘッドを低く抑えられるように設計されている．実装に際しては AM モデルを単純にするため，CPU-NI 間の呼び出しも AM を用いている．

3.2 NICAM の性能

図 2に NICAM を用いた場合の point-to-point のバンド幅および通信時間を示す．

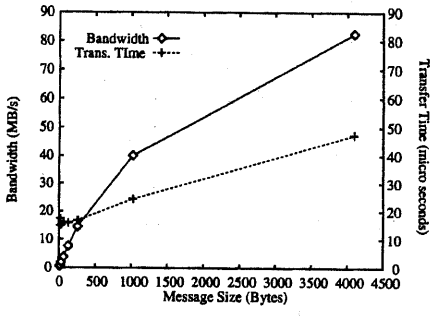

図 2　NICAM によるリモートメモリ転送の性能

短いメッセージを転送する際の最小レイテンシは約 16 マイクロ秒であり，最大のバンド幅は約 82.5MB/s である．表 4に COMPaS 上で NICAM および PM[2] を用いた場合のバリア同期にかかる時間を示す．

表 4　NICAM および PM のノード間の同期時間

Num. of Nodes	2	4	8
NICAM(μsec)	23.79	31.45	38.67
PM(μsec)	13.47	30.37	47.42

PM は Myrinet 上のユーザレベルのメッセージパッシングライブラリであり，クラスタシステムにおいて高い性能を示している．NICAM の結果は *am_barrier()* を用いたものである．PM はバリア同期のためのプリミティブを提供していないため，PM の結果は point-to-point 通信を shuffle exchange アルゴリズムを用いて組み合わせてバリア同期をとったものである．CPU-NI 間の通信コストのため 2 ノードの場合の同期時間は NICAM の方が長くなっているが，NI 間で処理が行なわれる各ステップに必要なコストは約 7.5 マイクロ秒と PM の約 17 マイクロ秒に比べて半分以下となり，8 ノードの場合には NICAM の方が高速に同期がとれる．

4. 実験結果

共有/分散融合プログラミングは，*reduction* などのいくつかの命令に関しては特殊な処理が必要となるものの，基本的には分散プログラミングの局所計算 (ノード内の計算) 部分を共有 (マルチスレッド) プログラミングによって処理するという形で容易に実装することができる．我々は以下の 4 種類のベンチマークを共有/分散融合プログラミングにより実装し，COMPaS の性能を測定した．

ラプラス方程式の陽解法 (2 次元)

反復法に基づく方程式の解法であり，我々の実装はヤコビ法を用いている．分散プログラミングの場合，ノード間通信は隣接間転送のみとなる．実験では行列のサイズは 640×640 とした．

行列の乗算

並列計算機における行列の乗算には行分割，列分割およびブロック分割などのデータの分割方法と，ijk, jki などの演算順序の組み合わせにより様々なアルゴリズムが存在する．今回の実装ではデータの局所性を最大限に引き出すようにブロッキングとタイリングを組み合わせた方法を採用した．実験では行列のサイズは 1800×1800 とした．

radix ソート

radix ソートは各データを n 進数で表し，各桁ごとのソートを繰り返すことにより全体のソートを行なう方法である．実験では 4M 個の整数のソーティングを行なった．

疎行列 CG カーネル

Nas Parallel Benchmarks に基づいて実装を行なった．実験でのサイズは Class A である．

実験に際しては 1, 2, 4, 8 の 4 通りのノード数に関し，各ノード数ごとにスレッド数を 1, 2, 4 の 3 通りに変化させて実行時間を計測した．本実装では起動したスレッドは Solaris が提供している *processor_bind()* 関数を用いて別々のプロセッサに割り当てられるようになっている．ノード数とスレッド数の積が動作するプロセッサ数 (スレッド数) となる．1 ノードの場合の実験結果は共有メモリシステム上でのマルチスレッドプログラミングの結果と同じである．1 スレッドの場合の実験結果は分散プログラミングの結果を表している．

図 3と図 4にラプラス方程式の陽解法の実行時間を示す．図 3は Myrinet 上で NICAM を用いた場合の実行時間を，図 4は 100Base-T Ethernet 上で mpich を用いた場合の実行時間を示す．図 3および図 4はそれぞれ 1 ノード，2 ノード，4 ノード，8 ノードの場合の結果を示し，X 座標はノード数とスレッド数の積，つまり全プロセッサ数を表す．1 ノードの場合に

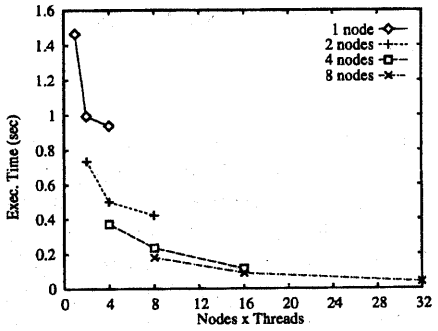

図3　ラプラス方程式の陽解法の実行時間 (N=640, NICAM, Myrinet)

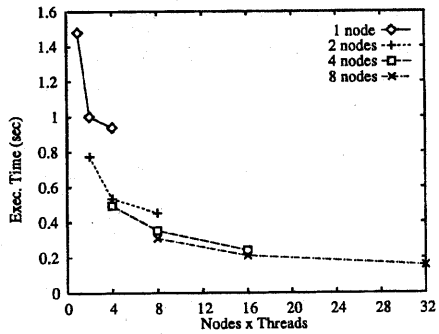

図4　ラプラス方程式の陽解法の実行時間 (N=640, mpich, 100Base-T)

スレッド数に対するスケーラビリティが悪い．これは，データサイズが大きくて主記憶に対するアクセスが多数発生し，メモリバスの性能がボトルネックになっているためである．ラプラス方程式の陽解法ではノード間通信は隣接間転送のみであるが，ノード数が増えると計算量が減るのに対してノード間通信は変わらないため，ノード間通信の性能差が顕著に表れる．例えば8ノードの場合，Myrinet上のNICAMを用いた実行時間は100Base-T Ethernet上でmpichを用いた場合に比べて約1/2から1/3程度になっていることが分かる．

図5から図8に各ベンチマークプログラムをMyrinet上でNICAMを用いて実行した場合の速度向上率を示す．X座標は図 3と同じである．idealは理想的な速度向上率を示したものである．

ラプラス方程式の陽解法では，8ノードの場合に理想的な速度向上率以上の結果が得られている．これは，ノード間通信の高速化に加えデータサイズが小さくなってキャッシュを有効に利用することができた結果である．行列の乗算ではかなり高い速度向上率が得られている．行列の乗算はラプラス方程式の陽解法に比べるとはるかに大きなデータを扱っているが，アルゴリズムを工夫してデータの局所性を高く引き出して

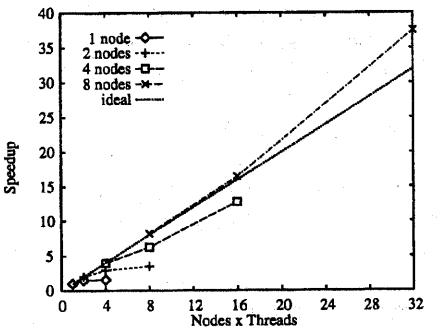

図5　ラプラス方程式の陽解法の速度向上率 (N=640, NICAM, Myrinet)

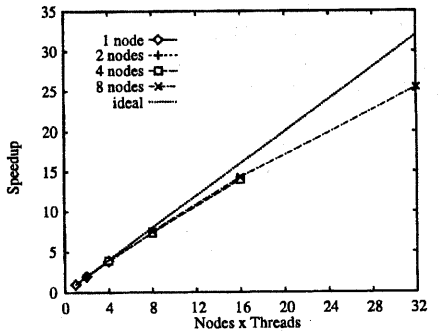

図6　行列の乗算の速度向上率 (N=1800, NICAM, Myrinet)

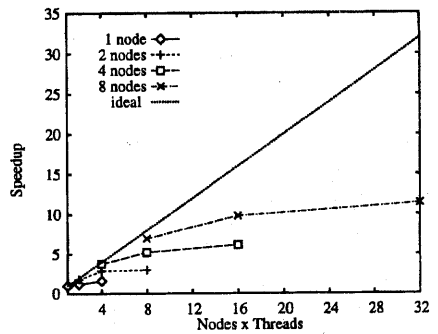

図7　radix ソートの速度向上率 (4M int, NICAM, Myrinet)

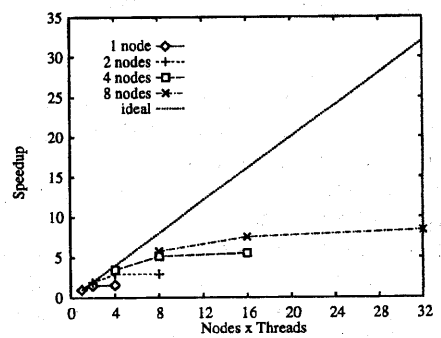

図8　CG カーネルの速度向上率 (Class A, NICAM, Myrinet)

いるため，1ノードの場合に高い並列化効率を得ることができる．また，ノードを固定してスレッドを増加させる場合とスレッドは増加させずにノードを増やす場合とでほぼ同じ結果を示している．これはもともと重くないノード間通信の処理がNICAMを使うことによってほとんどオーバヘッドとして見えなくなっているためである．8ノード・4スレッドの場合でも約80%程度の並列化効率を得ることができる．これは，スレッド内でキャッシュを有効に利用する実装による高性能計算と，NICAMによる高性能ノード間通信により可能となる．radixソートでは1スレッドの場合にはノード数に対して高いスケーラビリティが得られている．radixソートではノード間で交換するデータ量が大きくなる(2ノードの場合は平均4MB，8ノードの場合は平均256KB)ため，ノード間通信の性能が全体の性能に大きな影響を与えるが，NICAMを用いた高速通信により高い性能が得られていると考えられる．しかし，スレッド数に対するスケーラビリティは高くない．これは，データサイズが大きいため，メモリバスの性能がボトルネックとなっているためと考えられる．CGカーネルではスレッド数に対するスケーラビリティが極めて悪い．4スレッドを用いた場合の実行時間は2スレッドのものとほとんど変わらない．CGカーネルのデータサイズはとても大きく，かつデータに対するアクセスが1度走査するだけと局所性が低いため，主記憶へのアクセスが集中し，メモリバスの性能がボトルネックとなってしまうためである．

5. 考　　察

5.1 データの局所性の効果・影響

ラプラス方程式の陽解法の場合，図5に示したようにノード数が少ない場合にはスレッド数を4にしても速度向上率はかなり低いが，ノード数が8になると速度向上率は理想値を越える．特に，32プロセッサの場合には約37.5という速度向上率が得られている．これは，プロセッサ数が32を越えると各スレッドあたりのデータサイズがちょうどキャッシュにフィットする程度となり，主記憶に対するアクセスが減ってキャッシュを有効に利用することができ，SMPの利点を引き出すことができるためである．

このようにデータサイズが小さくなれば高い性能が得られるが，たとえデータサイズが大きくてもアルゴリズムを工夫することにより，データの局所性を引き出して450GX chip-setのようなバスの性能が弱いSMPシステムにおいても，SMPの利点をいかすことができる．キャッシュ上のデータに対してアクセスする回数を増やす(主記憶へのアクセスを減らす)ことにより，データの局所性を高めることができる．行列の乗算においては，データサイズは大きい(1800×1800の場合には行列1つあたりで約25MB)が，行列をブロック化することにより高いデータの局所性が得られ，結果としてスレッド数に対して高いスケーラビリティを得ている．

通常のマルチスレッドプログラミングにおいて高いスケーラビリティを得ることができれば，共有/分散融合プログラミングモデルにおいてノード数が増加しても高いスケーラビリティを得ることができる．Pentiumを用いたSMPシステムのバス性能はまだ十分ではなく，このようなバスの性能が弱いSMPシステムを用いたクラスタ上で高い性能を得るためには，ノード内での計算アルゴリズムを工夫して高い局所性を得ることが重要である．

5.2 メモリバスボトルネックによる影響

CGカーネルのように主記憶に対するアクセスが集中するようなプログラムでは高い効率が得られなかった．ある時点での全スレッドのバスのバンド幅の合計はメモリバスの性能(バンド幅)で抑えられるため，主記憶へのアクセスが集中するような場合にはメモリバスの性能がボトルネックとなってしまう．この問題を回避するためにはキャッシュを有効に利用する必要があるが，Pentium ProのL2キャッシュは物理アドレスキャッシュであるため，実際の大きさ(COMPaSでは512KB)の半分程度の大きさしか有効でない点に注意する必要がある．

5.3 ノード間通信の割合

表5に行列の乗算における1ノードおよび8ノードの場合の並列化効率を示す．Nはノード数，Tはスレッド数を表わす．

表5　行列の乗算の並列化効率 (N=1800, NICAM, Myrinet)

	T = 1	T = 2	T = 4
N = 1	1.00	0.99	0.97
N = 8	0.94	0.89	0.80

1ノードの場合にはスレッド数に対して高いスケーラビリティがあるが，8ノードの場合にはスレッド数に対するスケーラビリティが低下する．8ノードの場合の実行時間をブレークダウンした結果，1,2,4スレッドの場合の乗算にかかる時間はそれぞれ17.09秒，8.59秒，4.46秒であった．このように，8ノードの場合でも乗算演算に関してはスレッド数に対して高いスケーラビリティが得られている．一方ノード間通信の時間はスレッド数に依存せずに一定であり，その値は0.83秒であった．しかし，全実行時間に対する通信時間の割合を見てみると，1スレッドの場合は約4.6%であるのに対し，4スレッドの場合は15.7%となる．

行列の乗算は演算量に比べると通信量が少ないベンチマークではあるが，スレッド数が増えると各スレッドが担当する領域が小さくなって演算時間が短縮され，ノード間通信にかかる時間がオーバヘッドとして見えてくるために，プロセッサ数が多くなるとスレッド数

に対するスケーラビリティが低くなってしまう．

6. 結　　論

我々は Pentium Pro 4 台を搭載した PC サーバ 8 台を Myrinet および 100Base-T Ethernet で結合した SMP クラスタ COMPaS を作成した．本稿では COMPaS の基本性能を示し，COMPaS 上で共有/分散融合プログラミングモデルにより実装したベンチマークの実行結果を示した．また，SMP クラスタ向けの通信レイヤである NICAM の概要および性能に関しても報告した．NICAM は Myrinet の NI 上の Active Message を用いたリモートメモリベースの通信レイヤであり，低オーバヘッドかつ高いバンド幅を可能とする．

ラプラス方程式の陽解法のように複数のスレッドを起動してデータサイズが小さくなり，キャッシュに収まりきるようになればメモリバスの性能がネックとなっていても，SMP の利点を引き出してマルチスレッド処理により高い効率を得ることができる．また，データサイズが大きくてもブロック化による行列の乗算のようにキャッシュを効果的に利用することができれば，同様にマルチスレッド処理により高い効率を得ることができる．radix ソートや CG カーネルのようにデータサイズが大きく，キャッシュを効果的に利用することができない場合には主記憶へのアクセスが集中し，メモリバスの性能がボトルネックとなって高い性能は得られない．現在の Pentium Pro を搭載した SMP クラスタを構築する際のガイドラインとして以下のようなことが言える:

- メモリバスの性能が大きな影響を与える．Pentium Pro のようにメモリバスの性能が低い場合には，主記憶へのアクセスが集中するようなアプリケーションにおいては性能が劣化する可能性がある．
- 450GX chip-set のようなバスの性能が低い SMP システムにおいて SMP の利点をいかして高い性能を得るためには，データの局所性を引き出すことが重要である．データの局所性を引き出すということは，キャッシュ上のデータに対するアクセスの割合を増やすということである．データサイズが小さければデータの局所性は高くなるが，たとえデータサイズが大きくてもキャッシュにロードしたデータを何度もアクセスするようにアルゴリズムを工夫することにより，高いデータの局所性を引き出すことができる．
- SMP クラスタにおいては，CPU オーバヘッドの少ない，マルチスレッドセーフなノード間通信の手段が必要である．また，メモリバスに対するアクセスも極力避ける必要がある．我々は Myrinet の NI 上での Active Message および DMA の機能を用いたリモートメモリ転送を行なう NICAM により，この問題を解決した．

リモートメモリ転送機能と複数のスレッドを用いる 1 つの方法として，ノード間通信と演算とをオーバーラップする方法が考えられる．各スレッドはノード間通信の際に同期をとらず，自分の担当領域に新しいデータが到着したら演算を開始する．このような実装により，通信レイテンシなどを隠蔽することができれば，より高い効率を得ることが可能となると考えられる．今後バスの性能が高い SMP システムおよびそれらを用いたクラスタが普及してくると考えられる．そのようなクラスタで高い性能を得るためにも，SMP クラスタの性能特性について知見を得ることは重要である．

謝　　辞

本研究にあたり，クラスタの構築に関してご指導頂いた新情報処理開発機構並列分散システムソフトウェア研究室の皆様に感謝致します．

参 考 文 献

1) “Pentium Pro Cluster Workshop”, http://www.scl.ameslab.gov/workshops/.
2) N. J. Boden, et al, “Myrinet – A Gigabit-per-Second Local-Area Network,” IEEE MICRO, Vol. 15, No. 1, pp. 29–36 (1996).
3) H. Tezuka, et al, “PM: An Operating System Coordinated High Performance Communication Library”, High-Performance Computing and Networking, Lecture Notes in Computer Science, Vol.1225, pp.708–717, Springer-Verlag (1997).
4) W. Gropp and E. Lusk, “MPICH Working Note: Creating a new MPICH device using the Channel interface”, Argonne National Laboratory Technical Report (1995).
5) T.von Eicken, et al, “Active Messages: a Mechanism for Integrated Communication and Computation”, Proc. 19th Int'l Symp. on Computer Architecture, pp. 256-266 (1992).
6) 田中良夫 他, “SMP クラスタでの共有/分散融合プログラミング”, 情報処理学会ハイパフォーマンスコンピューティング研究会報告, Vol.97, No.75, pp. 61–66 (1997).
7) Y. Tanaka, et al, “COMPaS: A Pentium Pro PC-based SMP Cluster and its Experience”, Proc. Int'l Workshop on PC-NOW'98 (1998) (to appear).
8) 松田元彦 他, “SMP クラスタ向けネットワーク・インタフェース上 AM 通信”, 情報処理学会計算機アーキテクチャ研究会報告, Vol. 97, No. 76, pp. 55–60 (1997).