ハイパフォーマンス
コンピューティング 62－5
（1996. 8. 28）

並列計算機の仮想性能評価システム VIPPES

三島正博† 板倉憲一† 朴泰祐†
中村宏†† 中澤喜三郎†††

あらまし
本研究では，並列計算機の仮想性能評価システム VIPPES（VIrtual Parallel Processor Evaluation System）の提案と実装を行なう．
VIPPES は仮想的な並列処理システム上で実アプリケーションを実行する事を想定し，性能評価を3段階に分けて行なう．各段階での結果は次の段階の入力として引き継がれ，最終的にネットワークシミュレータによって実行時間の測定を行なう．
ユーザーは C 言語及び並列処理ライブラリ PVM を用いて評価用アプリケーションを記述し，ネットワークについての諸特性を記述する事で，実アプリケーションを仮想的な並列処理システム上で実行したときのプロセッサ内外の処理を含めた総合的な性能評価を行なう事ができる．
本稿では，VIPPES の概要，実装 について述べる．又，実際に VIPPES を用いていくつかの仮想的な並列計算機とアプリケーションプログラムの組み合わせで評価を行う．結果，本システムが並列計算機の性能比較をする為に有効である事が分かった．

VIPPES: A Performance Pre-Evaluation System for Parallel Processors

MASAHIRO MISHIMA,† KEN'ICHI ITAKURA,† TAISUKE BOKU,†
HIROSHI NAKAMURA†† and KISABURO NAKAZAWA†††

Abstract
In this paper, we propose and implement a performance pre-evaluation system for parallel processors, named VIPPES (VIrtual Parallel Processor Evaluation System).
In this system, we analyze the behavior of an actual target application program on clock-level simulator of processing units and interconnection network independently, based on trace and profile data of the program execution. The performance evaluation process in VIPPES is broken into three phases: trace creating phase, processor simulation phase and network simulation phase. They are strongly connected with each other to realize precise clock-level simulation in low computational cost.
We describe the concept, design and implementation of VIPPES on workstation cluster systems. Some results of real applications are also shown. Through these experiences, we confirmed the effectiveness of VIPPES for various configurations of parallel processing systems.

1. はじめに

本稿では，並列計算機の仮想性能評価システム VIPPES（VIrtual Parallel Processor Evaluation System）を提案しその実装について述べる．

並列計算機を設計する際，そのシステムで想定されるアプリケーションの処理時間を事前に評価する事は非常に重要である．単体の計算機の場合，クロックレベルでの評価をとる事のできるプロセッサシミュレータが存在すれば正確な性能評価を行なう事ができる．しかし，並列計算機の性能評価は，各 PU（Processing Unit）における内部処理時間の評価に加え，プロセッサ間相互結合ネットワーク上のデータ転送時間及びそれらのオーバーラップを考慮して評価を行なう必要がある．全ての PU の内部処理とネットワーク上でのデータ転送の挙動を同時にシミュレートする大規模なシミュレータは時間的なコストが非常にかかる為，多

† 筑波大学 電子・情報工学系
Institute of Information Sciences and Electronics, University of Tsukuba
†† 東京大学 先端科学技術センター
Research Center for Advanced Science and Technology, University of Tokyo
††† 電気通信大学 情報工学科
Department of Computer Science, University of Electro-Communications

数の PU によって構成されるような並列計算機の仮想性能評価を行なう事は困難である．

現在の大規模な並列処理システムは，分散メモリ型のモデルを使用するものが多く，このようなシステムで科学技術計算のアプリケーションを実行する場合に多く見られるデータパラレル的な処理では，内部計算やデータ転送の順序が決定的な場合が多い．そこでこの点に注目し，VIPPES では PU 内部処理時間とデータ転送時間を分離して独立に評価する．これによって全 PU を同時にシミュレーションするよりもはるかに少ない計算資源の下で全処理時間の評価を行なえる．

以下では，VIPPES の概要，実装について述べる．又，実際に VIPPES を用いていくつかの仮想的な並列計算機とアプリケーションプログラムの組み合わせで評価を行う．最後に今後の課題について述べる．

2. VIPPES の概要

VIPPES は，分散メモリ MIMD 型の仮想的な並列処理システム上で実アプリケーションを実行する事を想定し，PU 内の処理と PU 間のデータ転送の両方をクロックレベルのシミュレーションを中心とした手法で総合的な性能評価するシステムである．

2.1 VIPPES の汎用性

汎用的な並列計算機の性能評価システムを考える場合，以下の事が問題となる．

(1) 並列アプリケーションをどのようにして記述するか．
(2) 並列計算機を構成する PU のアーキテクチャをどのように記述，評価を行なうか．
(3) 並列計算機を構成する PU 間ネットワークをどのように記述，評価を行なうか．

(1) については，PVM (Parallel Virtual Machine)[1]や MPI (Message-Passing Interface)[2]といった汎用的なメッセージパッシングモデルを用いる事で解決する．このようなメッセージパッシングモデルはライブラリの形で実現されており，VIPPES ではこのライブラリを変更する事で，アプリケーションプログラムのソースには変更を加えずに，後で示すようなデータ転送情報の抽出をするプログラムや，プロセッサシミュレータに入力するオブジェクトを作る事ができる．

(2) に関しては汎用的なプロセッサを記述し，シミュレーションを行えるツールが必要となる．このような条件を満たすものとして，VHDL などのハードウェア記述言語及び論理合成ツールがあるが，これらを用いるには二つの問題点がある．一つは，記述や合成を詳細なレベルまで行う必要があり，評価にかかるコストが非常に大きい事である．もう一つの問題点は，FORTRAN や C などの高級言語で記述された並列アプリケーションを評価する場合には，設計した計算機に対して最適なコードを生成するコンパイラが必要となる事である．この点から VIPPES では既存のプロセッサアーキテクチャ，メモリシステムを持つ PU を仮定し，コンパイラも既存のものを使用する．

(3) についても汎用的なネットワークの記述及びシミュレーションを行なう事のできるツールが必要であるが，これには汎用ネットワーク・シミュレータ生成システム INSPIRE (Interconnection Network Simulator with Programable Interaction and Routing for performance Evaluation)[3]を用いる．INSPIRE では，ネットワークの規模，資源，トポロジ，ルーティングアルゴリズムといった諸特性を INSPIRE 用ネットワーク記述言語 NDL (Network Description Language) で記述し，PU の動作は C 言語により手続き的に記述する事ができる．

2.2 VIPPES の構成

全ての PU 及びネットワークを対象として，並列計算機全体のシミュレーションを大規模なシミュレータによって行なう方法は非常にコストがかかる．又，このようなシミュレータの並列化は，各 PU の内部処理とネットワーク上でのデータ転送の依存関係より，シミュレーション全体における時刻の管理が非常に困難になる．

これに対して，VIPPES では PU 内の処理時間とネットワーク上でのデータ転送時間の解析を独立に行なう．並列プログラムの処理時間は PU 内での計算時間とデータ送受信に関する待ち時間に分ける事ができる．前者はデータ転送命令間の時間であり，後者はデータ転送命令にかかる時間である．よって，PU 内での計算をデータ送受信命令を境に分割すると，各ブロックでの計算時間はプロセッサシミュレータによって解析できる．更に，異なる PU の解析は独立して行なえる．全 PU の処理ブロックの処理時間が決定すると，それを基にデータ転送のタイミングが決定し，データ送受信の待ち時間をネットワークシミュレータによって解析する事ができる．ネットワークシミュレーションでは PU 内の処理ブロックが単なる消費時間で抽象化されているので，VIPPES のボトルネックはプロセッサシミュレーションの部分にある．又，プロセッサシミュレーションを正確に行なう為にはデータ受信命令によって行なわれる受信データの格納を行なう必要がある．この為にプロセッサシミュレーションに先だって，何らかの並列システムによる実行トレースからデータ転送情報の抽出を行なう必要がある．

以上の事から，VIPPES では 図 1に示すように 3 つのステップに分けて性能評価を行なう．以下で各ステップでの処理について述べる．

2.3 データ転送情報の抽出

ここでは，プロセッサシミュレーションやネットワークシミュレーションに必要な送受信イベントの履歴と送受信データに関する属性を抽出する．この為に，ワー

クステーションクラスタや並列計算機上で実際にアプリケーションを実行し，データ転送に関するトレース情報を得る．

ここで抽出する情報はデータの送受信が行なわれる際の相手 PU 番号，メッセージサイズ，メッセージタグ，メッセージ内容等であり，データ転送時間や PU の処理時間の測定は行なわない．つまり，実際にターゲットとする並列システムと全く無関係なシステムでこの情報を抽出する事ができる．ただし，この送受信イベントの履歴を基にシミュレーションを行なうので，VIPPES で解析可能な並列プログラムは処理の順序と転送の順序が決定的であるものに限定される．

このステップで得られた情報は，各 PU 毎にトレースファイルに記録され，これを IPTRACE (Inter-PU trace) と呼ぶ．

2.4 プロセッサシミュレータによる評価

このステップではプロセッサシミュレータを用いて，各 PU の内部処理をデータ送受信命令によって分割した各ブロックでの計算時間の評価を PU 毎に独立して行なう．プログラム上でデータ受信が行なわれる時には，IPTRACE から本来受信されるはずのデータを取り出して格納する．これによってプロセッサシミュレーションは正しいデータを基に行なわれる．

IPTRACE を用いる事で各 PU のシミュレーションは独立して行なえるのでプロセッサシミュレーションは分散システム上で容易に行なうことができる．

このステップにかかるコストをさらに下げる方法として，複数の PU を 1 回のプロセッサシミュレーションでモデル化することが考えられる．SPMD 型プログラムでは，扱うデータは違うが，同じ命令列を実行し処理時間が変わらない場合が多い．この様な場合，同じ処理を行なう PU に関しては 1 つの PU の内部処理の解析結果から他の同じ処理をする PU の解析結果を得ることが可能である．つまり，シミュレーションを行なうべき PU の数を減らす事ができ，シミュレーションコストを大幅に抑えることができる．

このステップによって得られた結果は各 PU 毎にプロファイルとして記録され，これを IPPROFILE (Intra-PU profile) と呼ぶ．IPPROFILE 内には，IPTRACE 内のデータ転送に関する情報も含まれており，完全に抽象化された PU の動作が記録される．

2.5 ネットワークシミュレータによる評価

このステップでは抽象化された PU の動作 (IPPROFILE) を基にデータ転送の相互関係をネットワークシミュレータでシミュレーションする．ネットワーク上で各 PU がデータ転送を行なうタイミングやネットワーク上でのメッセージ転送の衝突はこのステップにおいてクロックレベルで正確にシミュレーションされ，結果として総合的な性能評価を得る事ができる．

IPPROFILE は ネットワークの記述とは独立なので，プロセッサへのプロセスのマッピングを変更した場合や同じプロセッサを使用しネットワークのトポロジのみを変えた場合の性能比較などに IPPROFILE を再利用する事できる．この様に，並列システムの解析においてボトルネックとなるプロセッサシミュレーションを省略してできる点は VIPPES の大きな特徴である．

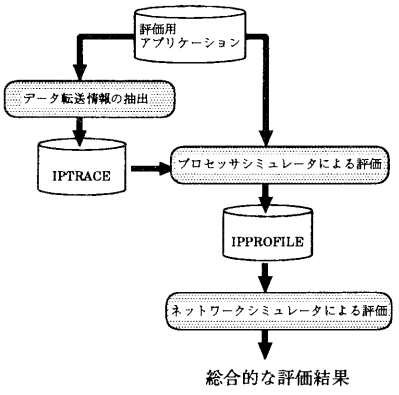

図1 評価の流れ

3. VIPPES の実装

本節では VIPPES を構成する各要素及びシステムの各ステップでの具体的な実装について述べる．

本研究では VIPPES を UNIX ワークステーションクラスタ上に実装し，メッセージパッシングのモデルとして PVM を用いたプログラムを対象とする．VIPPES では 1 PU の動作を PVM 上の 1 プロセスとして想定し，各プロセスで実行される論理 PU 番号はプロセスの生成順とする．

3.1 IPTRACE の生成

ユーザは評価用アプリケーションに対して VIPPES 用のヘッダファイルを付け，IPTRACE 生成用の PVM ライブラリをリンクし，IPTRACE 生成用の実行ファイルを作成する．このライブラリは，アプリケーション上で PVM 関数が呼び出された時に，本来の PVM 関数としての動作の他に，副作用としてメッセージ交換情報の記録を行う．このプログラムをワークステーションクラスタ上の PVM 環境で実行する事によって，IPTRACE を得る (図 2)．

3.2 IPPROFILE の生成

IPPROFILE の生成は SPMD 型プログラムを想定し，二つのステップに分けられる．まず，代表となる PU の評価をプロセッサシミュレータによって行う．次に，同じ動作をする PU の IPPROFILE を代表の IPPROFILE から生成する．どの PU を代表とするかはユーザ の判断にまかせられる．ユーザはこ

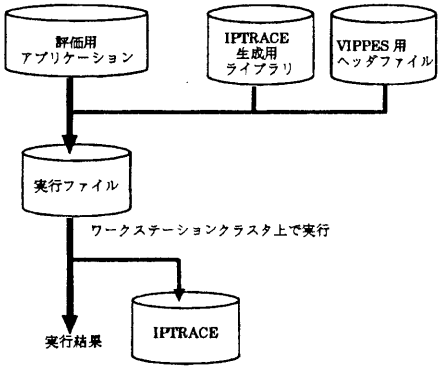

図2 IPTRACE の生成

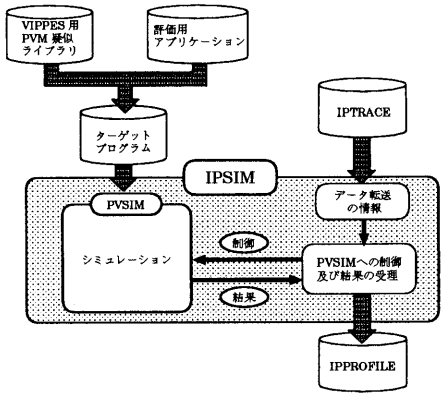

図3 IPSIM による IPPROFILE 生成

れらの情報を記述したコンフィグレーションファイルをシステムに与える．システムはその情報に従い自動的に IPPROFILE を生成する．以下ではそれぞれの処理について述べる．

3.2.1 シミュレータによる IPPROFILE の生成

VIPPES で性能評価を行なう場合，プロセッサシミュレータは最低限以下の要求を満たしている必要がある．

- break-point を設定でき，任意の区間のシミュレーションを行なう事ができる．
- レジスタやメモリの参照と更新を行なう事ができる．
- シミュレーションの結果として実行クロック数が測定できる．

ここでプロセッサシミュレータとして PVSIM[4] を使用する．PVSIM は 本学を中心に実装が行なわれた超並列計算機 CP-PACS (Computational Physics - Parallel Array Computer System)[5] 本体の PU として用いられる PVP-SW (Pseudo Vector Processor based on Slide Windowed Registers) をモデルとして，クロックレベルでシミュレートをする事のできるシミュレータである．PVP-SW は PA-RISC 1.1 アーキテクチャを拡張したもので，PA-RISC 1.1 アーキテクチャと上位互換性を保ちつつ擬似ベクトル処理を行なう事ができる．

PVSIM を使って評価する時には，PVM 関数は単なる break-point として使用するので，シミュレーション実行用にダミーの PVM ライブラリを用意する．そして，ユーザはこのライブラリをリンクし PVSIM 用のターゲットファイルを作成する．送受信の処理 (PVM 関数) に対して break-point を設定し，break-point 間にかかる clock 数を測定する事でデータ転送間に行なわれる PU 内部処理の評価を行なう．次に行なわれる PVM 関数は IPTRACE から分るので，break-point は高々1 つ設定できれば良い．

データ受信関数では IPTRACE から受信データを取り込み，プロセッサシミュレータ上のデータ受信バッファに相当するローカルメモリの更新を行い，あたかも外部からデータが到着しているかのような状況を作り出す．

PVSIM を制御し IPPROFILE を生成する制御プログラム IPSIM (Intra-Pu-SIMulator control program) は以下の処理を行なう (図 3)．

(1) シミュレータの起動．評価用ターゲットファイルの読み込み．
(2) IPTRACE 及び IPPROFILE のオープン．
(3) IPTRACE 内の情報より，測定すべき範囲が分かるので，シミュレータに対し対応する break-point の設定を行なう．break-point は PVM ダミー関数に対して設定される．
(4) 上記範囲のシミュレーションの実行．
(5) 結果の記録．シミュレーションの実行結果から，clock 数の情報を得る．又，データ受信時には IPTRACE より受信データを読みだしメモリ内に格納する．
(6) 継続又は終了．測定すべき範囲が残っていれば再び (3) から処理を継続し，全ての測定が終了していればシミュレータ及びその制御プログラムを終了する．

3.2.2 コピーによる IPPROFILE の生成

SPMD モデルでの代表の PU をオリジナルとし，オリジナル IPPROFILE 内の 内部処理時間と各々の IPTRACE のデータ転送情報から全ての PU の IPPROFILE を生成する (mkpro, 図 4)．

3.3 ネットワークシミュレータによる総合評価

最終的な評価は INSPIRE によって行う．ユーザがネットワークの特性を記述した NDF (Network Description File) と IPPROFILE によって PU の動作を決定する汎用 PBF (PU Behavior File) からネットワークシミュレータを作成し，IPPROFILE を与えてシミュレーションを行なう (図 5)．NDF は評価

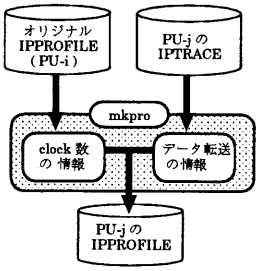

図4　コピーによる IPPROFILE の生成（オリジナル：PU-i）

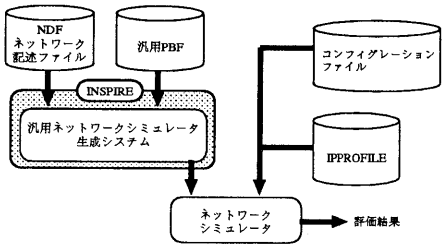

図5　ネットワークシミュレータの生成，実行

を行なう並列計算機のモデルに合わせてユーザーが自由に記述するが，PBF はシミュレーション実行時に IPPROFILE から各 PU の情報を採取し PU の動作を決定する．論理 PU 番号と NDF によって決定されるネットワーク上の PU 番号の対応表をコンフィグレーションファイルから与える事で，同じ NDF 上で自由に PU のマッピングを行なえる．

実際のデータの送受信時にはオーバーヘッドがあり，この時間はネットワークシミュレーション時に補正する．具体的には，汎用 PBF において，各 PU の動作に加え，データ送受信の際の立ち上げオーバーヘッドの時間が内部処理時間として加えられる．

シミュレータの実行結果は，総実行時間，PU 毎の内部処理時間，送受信の回数，送受信時間，立ち上げオーバーヘッドの累積等がある．これらの値より，単なる総処理時間だけでなく，演算時間と通信時間の関係も解析できる．

4. 評　　価

本節では，実際に VIPPES を用いた性能評価例を示す．評価対象の並列計算機モデルは，CP-PACS のアーキテクチャをベースとし，ネットワークのトポロジを以下の 5 種類に変更した時の各アプリケーションの実行時間を比較する．

- ハイパクロスバ（HXB）
- ハイパーキューブ（HC）
- 2 次元双方向トーラス（BTRS）

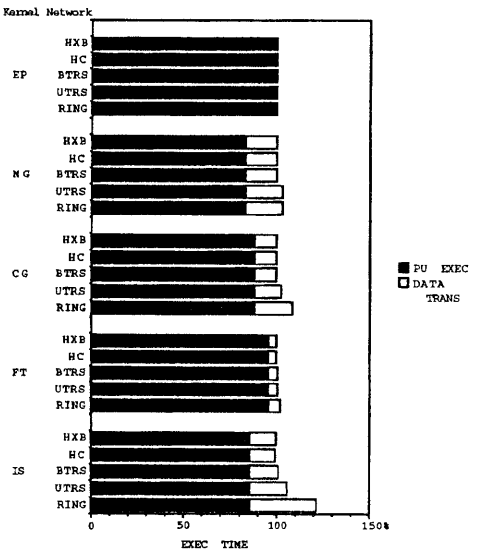

図6　評価結果

- 2 次元単方向トーラス（UTRS）
- 単方向リング（RING）

PU の台数は 8 台とし，評価用のプログラムは NAS Parallel Benchmarks (NASPB)[6] の 5 つのカーネル問題（表 1）の Sample サイズ を用いる☆．

表1　NASPB カーネル問題

カーネル名	問題
EP	合同乗算法による乱数生成
MG	multigrid 法によるポアソン方程式の求解
CG	CG 法による大規模疎行列の最小固有値問題
FT	多次元 FFT による 3 次元偏微分方程式の求解
IS	大規模な並列整数ソート問題

評価結果を図 6に示す．図の縦軸は各カーネル及び評価モデルのネットワークを示し，横軸は HXB モデルの実行時間を 100% とした時の各モデルの実行時間比を表している．グラフの内訳は PU 内処理時間（PU EXEC）及びデータ転送時間（DATA TRANSFER）であり，それぞれ 1 PU 当たりの平均値である．

次に，VIPPES による評価にかかる時間的コストについて考察する．上の評価例を HP 社のワークステーション HP-9000/735（99MHz）を用いて行なった時

☆ 一部のカーネル問題ではシミュレーション時間短縮の為にループ回数を変更してある．

のIPPROFILE生成及びネットワークシミュレータによる評価にかかった時間を表2に示す．表中のexecはシミュレーションクロック数を示し，timeはシミュレーション時間を示す．

表2 HXBモデルの評価にかかった時間

Kernel	1PUのIPSIM		Network Simulator	
	exec (clock)	time (sec)	exec (clock)	time (sec)
EP	793379144	36061	793382665	10380
MG	10332152	621	10948849	276
CG	87482231	5005	91195872	1308
FT	340901949	14382	352615495	5076
IS	1633048	76	1868219	37

プロセッサシミュレーション（IPSIM）は，8 PUで動作が全て事なる場合（EP, CG, IS）には，8 PU分全て実行する必要がある．しかし，これらは完全に並列に行えるので，複数のワークステーションを用いて分散処理することが可能である．例えばこの場合8台のワークステーションがあれば1 PU分の時間で評価できる．又，PUの処理時間がデータの内容に依存しない場合（MG, FT）には1 PU分のIPSIMとコピー処理によって8 PU分のIPPROFILEを得る事ができるので，1台のワークステーションでも1 PU分の時間で評価できる．これらの特徴は大規模な並列計算機をシミュレーションする時に非常に有利である．

IPSIMとネットワークシミュレーションを比較すると，同じクロック数をシミュレーションするのにネットワークシミュレーションの方が約3倍の時間がかかる．これは，IPSIMでは，1 PUのプロセッサの動作をクロックレベルでシミュレーションしている為である．ネットワークトポロジの違いによる比較評価では，IPPROFILEを再利用する事が可能なので，評価時間のかかるIPSIMを省略する事ができる．つまり，再評価にかかる時間はネットワークシミュレータの実行時間のみなので，評価にかかる時間的コストを大幅に抑える事ができる．

以上の点からVIPPESは超並列計算機のシミュレーションを行うのに適したシステムであり，並列計算機のネットワークアーキテクチャの評価に関して優れていることが分る．

5. おわりに

本研究では，並列計算機の仮想性能評価システムであるVIPPESを提案し，UNIXワークステーションクラスタ上での実装を行なった．

本システムはPU及びPU間ネットワークのシミュレーションを行ないアプリケーションプログラムに対しPU内外の処理を含めクロックレベルでの総合的な性能評価を行なう事ができる．

本システムの特徴として性能評価を3つのステップに分ける事があげられる．そして，IPTRACEを用いる事によってPUの内部処理時間の評価は各PUで独立に行う事が可能になる．又，SPMD的な動作をするプログラムに対してはPUの評価を簡略化する事ができる．そして，IPPROFILEを再利用する事によってネットワークの特性が違う並列計算機に対する評価を簡単に行なう事ができる．

評価対象として超並列計算機CP-PACSのネットワークトポロジを変えた仮想的な並列計算機を想定し，いくつかのアプリケーションに対して本システムで実行時間の評価を行った．この結果，ネットワークトポロジによる実行時間の違いが測定できた．又，VIPPESでの評価にかかる時間的コストを測定し，SPMD型プログラムに対し，VIPPESによる評価が非常に低いコストで行なえる事を確認した．

本システムを用いる事により，例えば1000台規模の並列計算機の仮想性能評価を100台規模の並列計算機で行なうと言った事も可能になる．その意味で，VIPPESは超並列計算機上での超並列計算機シミュレータとして用いる事ができる．

本研究の今後の課題として，まず各種ネットワークトポロジとより大規模のアプリケーションを用いて本システムの有効性を検証する事が挙げられる．又，並列化が行なわれていないINSPIREに対しても何らかの並列化による高速化を行なう必要がある．

謝辞

本研究に関し貴重な御意見を頂いた，筑波大学西川博昭助教授ならびに筑波大学アーキテクチャ研究室グループ諸氏に深く感謝します．なお，本研究の一部は文部省科学研究費（08NP0401）によるものである．

参考文献

1) Geist, A. et al.: *PVM 3 USER'S GUIDE AND REFERENCE MANUAL* (1994). ORNL/TM-12187.
2) Message-Passing Interface Forum: MPI:A Message-Passing Interface Standard (1994).
3) 原田智紀ほか: 並列処理ネットワークにのための汎用シミュレータ生成系INSPIRE, 情報研報, Vol. 95, No. 80, pp. 65–72 (1995).
4) 廣野哲: 擬似ベクトルプロセッサの性能評価シミュレータの開発 (1994). 筑波大学平成5年度卒業論文.
5) 中澤喜三郎ほか: 超並列計算機CP-PACSのアーキテクチャ, 情報処理, Vol. 37, No. 1, pp. 18–27 (1996).
6) D.Bailey et al.: THE NAS PARALLEL BENCHMARKS, RNR Technical Report RNR-94-007, NASA Ames Research Center (1994).