計算機アーキテクチャ 117－15
（1996. 3. 5）

CP-PACS パイロットモデルにおける LINPACK ベンチマークの高速化

曽根 猛　服部 正樹　朴 泰祐　中村 宏　中澤 喜三郎

筑波大学 電子・情報工学系
〒305 つくば市天王台 1-1-1

Tel: 0298-53-6912　Fax: 0298-53-5206
E-mail: {sone,hattori,taisuke,nakamura,nakazawa}@arch.is.tsukuba.ac.jp

本研究では，大規模科学技術計算を目的とした超並列計算機 CP-PACS のプロトタイプであるパイロットモデル (PILOT-1) の性能評価を実測によって行う．CP-PACS のノードプロセッサは擬似ベクトル機構が付加された PA-RISC1.1 アーキテクチャのスーパースカラプロセッサであるが，PILOT-1 ではこの追加機構はサポートされていない．PILOT-1 はこれらのノード間をハイパクロスバ・ネットワークで結合した分散メモリ型並列計算機である．

今回は，LINPACK ベンチマークを評価対象として取り上げた．ループアンローリング手法，ブロードキャスト機能，FMPYADD 命令等を用いることにより，LINPACK ベンチマークの高速化を行い，単純なガウスの消去法に比べて約 3 倍の高速化が達成された．また，前進消去の部分では，スーパリニア的な台数効果が得られた．

LINPACK Benchmark Evaluation on CP-PACS Pilot-Model

Takeshi SONE　Masaki HATTORI　Taisuke BOKU
Hiroshi NAKAMURA　Kisaburo NAKAZAWA

Institute of Information Sciences and Electronics
University of Tsukuba
1-1-1 Tennoudai, Tsukuba 305

Tel: 0298-53-6912　Fax: 0298-53-5206
E-mail: {sone,hattori,taisuke,nakamura,nakazawa}@arch.is.tsukuba.ac.jp

In this study, we evaluate the actual performance of the prototype of CP-PACS named PILOT-1. CP-PACS, a massively parallel processor, is aimed to solve large scale scientific problems. PILOT-1 is a parallel processor with distributed memory architecture, and equipped with superscalar processors based on PA-RISC 1.1 architecture. Pseude vector prcessor feature, which is supported in CP-PACS, is ommited in PILOT-1. All nodes are connected via Hyper-Crossbar Network.

We evaluated the performance of LINPACK Benchmark on PILOT-1. Utilizing loop unrolling technique, broadcast feature and FMPYADD instructions, its performance gets about three times faster compared with the simple Gaussian Elimination method. In forward elimination part, PILOT-1 achieves a super linear speedup.

1 はじめに

本研究では，超並列計算機 CP-PACS (Computational Physics by Parallel Array Computer System) [1] のプロトタイプである CP-PACS PILOT-1 (以下 PILOT-1 と省略) の性能を実測に基づき評価する．

CP-PACS は現在，筑波大学を中心に開発が進められており，計算物理学の分野における大規模科学技術計算を主たる目的とする．CP-PACS は，システム中の各 PU (Processing Unit) をハイパクロスバ・ネットワーク (以下 HXB と省略)[2] で結合した分散メモリ型超並列計算機である．PILOT-1 も同様に HXB を実装した分散メモリ型並列計算機で，筑波大学計算物理学研究センターに設置されており，現在稼働中である．PILOT-1 における並列プログラミングは基本的に CP-PACS と同じである．現在，PILOT-1 上で CP-PACS 用の各種プログラムの開発が進められている．

今回は，PILOT-1 の性能評価に LINPACK ベンチマーク [4] を用いる．LINPACK ベンチマークでは，各種計算機における性能が報告されているため，これを評価することで，他の計算機と比較検討を行うことができる．

2 CP-PACS PILOT-1

PILOT-1 のシステムの概要について説明する．PILOT-1 は分散メモリ型の MIMD 方式の並列計算機で，図 1 に示すように 14 台の PU と 2 台の IOU (内 1 台は SIOU) の計 16 台のノードから構成されている．IOU (Inupt/Output Unit) は分散磁気ディスク記憶装置などを接続するためのアダプターが備わっている PU であり，SIOU (Supervisor Input/OutputUnit) はコンソールが接続された IOU である．

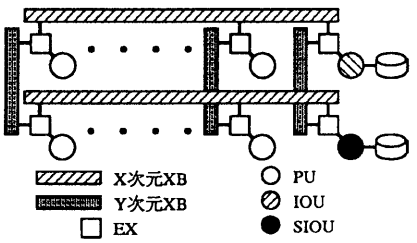

図 1: CP-PACS PILOT-1 の構成

2.1 ノードプロセッサ

ノードプロセッサは，Hewlett Packard 社の PA-RISC1.1 アーキテクチャを基にした RISC プロセッサを使用している．2 命令並列実行可能なスーパースカラ方式で，クロック周波数は 90 MHz，主記憶容量は 32 MB (IOU は 64 MB) となっている．また，全く依存関係のない浮動小数点データの乗算と加算を同時に処理する FMPYADD 命令が用意されているため，1 ノードあたりの最大性能は，180 MFLOPS となる．ダイレクトマップ方式の 2 レベルキャッシュを採用しており，1 次は命令 8 KB ／データ 16 KB のライトスルー方式，2 次は命令 1 MB ／データ 1 MB のライトバック方式のとなっている．しかしながら，CP-PACS そのものとは異なり，擬似ベクトル処理機構 [3] は装備されていない．PILOT-1 上では C 及び Fortran によるプログラミングが可能である．これらの言語に並列プロセス生成・プロセス間通信用のライブラリ関数が追加されており，それらを用いて明示的にメッセージパッシングによる並列プログラムを作成する．

2.2 ネットワーク

PILOT-1 ではノード間の相互結合網に 2 次元 HXB を採用している．2 次元 HXB は 2 次元直交座標上の各格子点にノードを配置し，各次元方向のノードをクロスバスイッチ (XB) で結合したネットワークである．ノード番号が 2 つの次元とも異なるノード間のデータ転送は，エクスチェンジャ(EX) と呼ばれる小規模なクロスバスイッチにより，複数の次元の XB を経由して行う．また，メッセージ転送は wormhole ルーティングを採用しており，ネットワーク中でのメッセージ間のデッドロックを回避するため，固定ルーティング方式を用いている．ネットワークの最大転送スループットは 100MB/s であるが，NIA (Network Interface Adapter) とメモリ間のシステムバスの最大スループットが 80MB/s となっているため，実質的な最大転送スループットは 80MB/s である．

2.3 リモート DMA 転送

PILOT-1 のノード間通信は，リモート DMA 転送と呼ばれる高速通信機構を採用している (図 2)．

一般的なメッセージ送受信では，送信ノードのユーザメモリ空間上のデータが OS 内の通信バッファにコピーされ，それが NIA を通じて受信ノードの通信バッファに転送される．通信バッファ上のデータは，受信プロセスが受信命令を発行することにより，ユーザメモリ空間に再度コピーされて受信処理が完了する．それに対して，リモート DMA 転送では，送受信プロセスのユーザメモリ空間内で，データが直接メッセージとして転送される．これにより，OS 空間とユーザ空間との間での冗長なデータのコピーがなくなるため，ネットワークの実効転送スループットが向上する．さらに，NIA に対する送信起動操作は，システムコールを用いず，ある制約の下で，ユーザ・アプリケーションから直接起動できるようになっているため，メッセージ転送時の立ち上げオーバーヘッドも短縮される．このときには，計算領域を直接，通信領域として使うのが最も理想的である．

リモート DMA 転送は高速にノード間のデータ転送を行うことができるが，送信側プロセスから受信ノードのメモリに直接データを書き込むため，書き込みのタイミングに

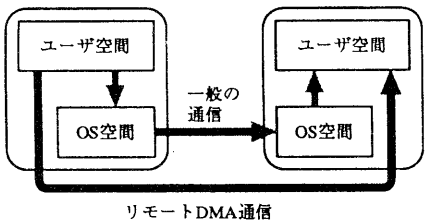

図 2: Combuf 通信

よっては，受信ノード上の有用なデータに対するオーバーライトの問題が生じる．さらに，キャッシュとメモリの整合性の問題もある．今回の評価ではメッセージ送出時及び到着時に，それぞれのノード上の OS によりキャッシュのフラッシュ及びパージを行うことで，この問題を解決している．また，リモート DMA 転送では受信領域のセキュリティの問題から，送信プロセスは予め受信 PU に対して，送信権の獲得を行わなければならない．その他に，通信領域は物理メモリに割り当てられた連続領域上になければならないという制約がある．

3 LINPACK ベンチマーク

LINPACK ベンチマークは，ガウスの消去法に基づく連立一次方程式の求解問題である．プログラムの概要は図 3 に示すように，ピボット処理と積和計算の 2 重ループから構成される．内側の 2 重ループが，計算量の大部分を占めるため，並列計算機では内側の 2 重ループを並列化することにより，処理の高速化が可能となる．また，PILOT-1 のノードプロセッサには FMPYADD 命令が用意されているので，内側の 2 重ループ内の積和計算で，FMPYADD 命令を効率的に使用することにより，処理の高速化が期待できる．今回の評価では，行列サイズを 1000 元とした．

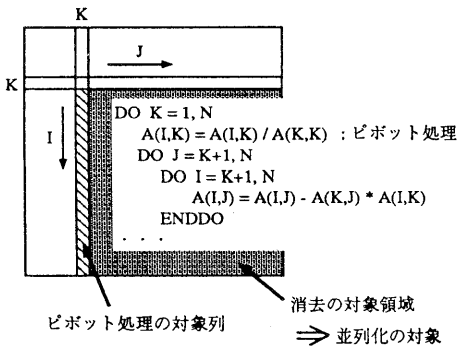

図 3: LINPACK ベンチマークの概要

4 並列化手法と処理の高速化

4.1 並列化手法について

分散メモリ型 MIMD 方式の並列計算機で LINPACK ベンチマークを評価した場合，各 PU へのデータの分散方法を行方向，列方向の両方でサイクリックに分割した方が効率的であるという報告がなされている [5][6]．しかし，今回の評価では PILOT-1 の持つブロードキャスト機能を有効に利用するため，このマッピング手法は用いない．

PILOT-1 のブロードキャストは，各次元方向の XB の出力を全て開放することにより実現されており，ノード数によらず常に通信コストが一定である．また，ブロードキャスト用のリモート DMA 領域に関しては，送信権の獲得がシステムによりチェックされないため，その分のオーバヘッドを省略できる (バリア同期等によりユーザが通信領域の確保を保証しなければならない)．そこで，今回の評価ではブロードキャスト機能を用いるため，列方向のサイクリック分割を採用した．このとき，最外ループをアンローリングする (多段同時消去) ことにより，ブロック・サイクリック分割となる．

4.2 多段多列同時消去

ガウスの消去法は，図 3 に示すように 3 重ループとなっている．一般に，ループ・アンローリング手法による主記憶へのロード／ストア回数の削減や，キャッシュを用いたブロッキング手法を用いることにより，処理の高速化がなされる．ガウスの消去法に関しては，最外ループをアンローリングすることによる多段同時消去や，積和計算の 2 重ループの外側のループをアンローリングすることによる多列同時消去を行うことにより，主記憶へのロード／ストア回数を削減できる (図 4)．

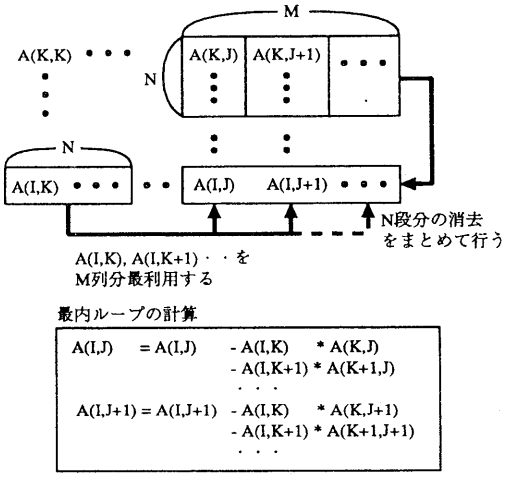

図 4: N 段 M 列同時消去

表 1: ロード／ストア回数と演算数

ロード回数	M+N+(N×M)/n
ストア回数	M
演算数	2×N×M

N 段同時消去では，消去対象となる A(I,J) のロード／ストア回数を 1/N に削減でき，M 列同時消去では，消去の際に用いられる A(I,K) のロードの回数を 1/M に削減できる．このとき，1 イタレーションあたりのロード／ストアの回数と浮動小数点の演算数は表 1 のようになる．ここで，n は最内ループにおけるベクトル長である．このとき，表 1 からロード／ストア回数に対する演算数の比率 R は以下の式で表される．

$$R = \frac{2NM}{2M + N + (N \times M)/n} \quad (1)$$

ここで，N=2M のとき，R は最大となる (ベクトル長 n が十分大きいとき)．PILOT-1 のノードプロセッサでは，ユーザが使用できるレジスタ数が 28 個なので，N+M+N×M≤28 と N=2M からレジスタを最大限効率的に使用する場合には，M=3，N=6 となり，6 段 3 列同時消去が最もよいことになる．

また，単一プロセッサにおいて，キャッシュを使ったブロッキング手法として，縦ブロックガウス法 [7] がある．これは消去の対象領域をキャッシュ上に保持することにより，処理の高速化を達成している．PILOT-1 では，1 次キャッシュが 16 KB なので，サイズが 1000 の行列では 2 列分しかキャッシュに保持することができない．さらに，並列化を行った場合，上記のようなブロッキング手法を用いることは難しい．このことから，先に示したレジスタの有効利用はガウスの消去法を並列化した場合には，非常に重要になる．

5 評価

プログラムは Fortran と通信ライブラリを用いて作成した．評価は SIOU を除く 15 ノードを用いて行った．

5.1 多段多列同時消去

N 段 M 列同時消去を行うプログラムを N と M のそれぞれの組合せに対して作成し，評価を行った．図 5 に結果を示す．

各ケースにおいて，ループ J (積和計算の外側のループ) のアンローリング回数を多くすることにより性能が向上しているが，アンローリングの回数がある値を過ぎると性能が低下している．これは，ロードしたデータがレジスタから追い出されることにより，データの再ロードが必要となるためである．しかしながら，N+M+N×M≤28 を満たす N と M の組を考えると，4 段同時消去では 4 列，5 段と 6 段同時消去では 3 列となるが，6 段同時消去のときは 5 列消去の場合が最も性能がよく，342 MFLOP/s を示している．4.2節では R を最大とする N と M を求めたが，実際には，PILOT-1 のノードプロセッサは，1 クロックに最大で 2 演算 (FMPYADD 命令) しか行えないため，R の値を 2 より大きな値としてもあまり効果がない．そこで，積和計算の最内ループに関して，アセンブラ・リストを出力することにより，使用されている命令を調べてみた．その結果を表 2 に示す．

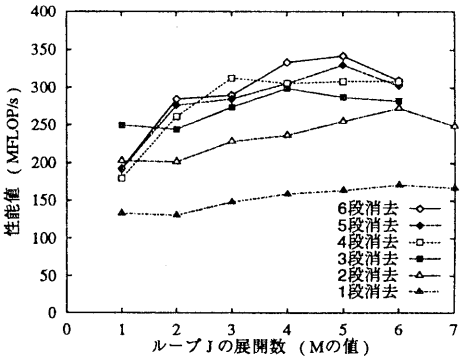

図 5: N 段 M 列同時消去による演算性能の変化

表 2: 最内ループの使用命令数 (6 段同時消去)

列展開数 (M)	1	2	3	4	5	6
FLOAD	7	8	11	19	27	39
FSTORE	1	2	3	4	5	6
FMPYADD	0	5	1	7	13	13
FMPY	6	7	17	17	17	23
FADD	6	7	17	17	17	23
LDO	7	6	9	11	10	21
ADD	0	1	2	3	4	5
ADDIBF	1	1	1	1	1	1
OR	0	1	0	0	2	0

表 2 において，整数系命令 (FLOAD，FSTORE 等) に対する浮動小数点演算命令の比率は，1 列の場合から順に 0.75，1.06，0.97，1.08，0.96，0.82 となる．6 段 5 列同時消去は，4 列同時消去に比べて，整数系命令数に対する浮動小数点演算数の比率が小さくなっているが，これは，レジスタから追い出されたデータの再ロードのためである．ループ・アンローリングの回数を多くした場合，ロードしたデータがレジスタから追い出されて，データの再ロードの必要が生じるが，それらのデータは 1 つ前のイタレーションでレジスタから追い出されたデータなので，常にキャッシュヒットする．そのため，5 列同時消去は 4 列同時消去に比べて，整数系命令数に対する浮動小数点演算数の比率が小さくなっているが，データのロードのコストが小さく済んでいるため，全体の性能としては最もよい

性能を示していると思われる．

最外ループKのみをアンローリングした場合(M=1)を比較すると，3段同時消去までは性能が向上するが，3段同時消去と4段同時消去では4段同時消去の方が性能が低くなっている．これは，3段同時消去までは積和計算の最内ループがコンパイラによりソフトウェアパイプライン化されているためである．このことは，アセンブラ・リストを出力することにより確認した．このため，1段〜3段ではループJをアンローリングして2列同時消去を行った場合に，性能が低下している．

図5では，6段同時消去までしか示していないが，さらに7段，8段同時消去を行った場合は，6段同時消去の場合より常に性能が下回っていた．並列計算機において，ガウスの消去法を行う際，ピボット列を担当するノードが積和計算の2重ループの処理の途中で，次のピボット列の処理を行うことにより，ピボット処理及びデータ転送の遅延時間を隠蔽することができる．しかし最外ループをアンローリングすることによりN段同時消去を用いた場合，ピボット処理をN列分同時に行う必要が生じる．このとき，ピボット処理と同時に(N-1)列分に消去処理を行わなければならなくなり，この部分の処理が大きくなるに従い，ピボット処理を先行することによる隠蔽ができなくなる．このために，7段，8段同時消去を用いても6段同時消去の場合に比べて性能が向上していない．

5.2 ブロードキャスト機能

図5に示した結果は，全てPILOT-1のブロードキャスト機能を用いた場合の結果である．ここで，ブロードキャスト機能を用いず，1対1通信を繰り返すことにより，ピボット列のデータ転送を行った場合との比較を表3に示す．表3の値は，6段5列同時消去を用いた場合の結果である．表で，1対1転送の繰り返しの括弧中に示した値は，リモートDMA転送領域の送信権獲得にかかる時間を除いた場合のものである．

表3: ブロードキャスト機能による性能向上

	Broadcast	1対1転送
MFLOP/s	342.0	77.08 (243.3)
処理時間(秒)	1.955	8.675 (2.748)
前進消去(秒)	1.614	8.374 (2.467)
後退代入(秒)	0.341	0.301 (0.281)

1対1転送の繰り返しの場合，リモートDMA転送領域の送信権の獲得の確認のためのオーバーヘッドが非常に大きく，性能が低下していることが分かる．送信権獲得のオーバーヘッドを除いても，ブロードキャスト機能を用いた場合は1対1転送を繰り返した場合に比べて，約4割の性能向上が見られる．

5.3 キャッシュによるブロッキング

並列計算機で，ガウスの消去法を用いる場合，積和計算の2重ループ部分が並列化される．このとき，最内ループで繰り返し使われるデータはピボット列に対応する部分である．この部分をキャッシュ上に保持することにより，データのロードのコストを削減できる．PILOT-1の2次キャッシュは1 MBあるため，15ノードで実行したときは，計算に必要な全データが2次キャッシュ上に存在する．ここでは，16 KBの1次キャッシュを効率的に使用することを考える．1次キャッシュは16 KBなので，2048要素を1次キャッシュに保持することができる．このとき，行列Aの配列サイズを1000とすることにより，図6に示すように48要素分ずつ各列の先頭のキャッシュ・エントリーがずれる．

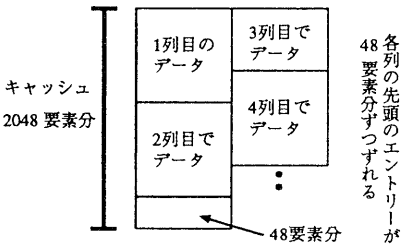

図6: キャッシュによるブロッキング

このとき，積和計算の2重ループで列方向に消去する際，48要素消去したら次に列の消去を行い，全ての列に対する消去が終了した後，次の48要素の消去を開始する．このようにすることで，消去領域のデータによるピボット列のキャッシュからの追い出しを減らすことができる．これを6段5列同時消去の場合に適用した結果を表4に示す．このとき約4%性能が向上した．

表4: キャッシュのブロッキングによる性能向上

	Non Blocking	Blocking
MFLOP/s	342.0	356.7
処理時間(秒)	1.955	1.875
前進消去(秒)	1.614	1.534
後退代入(秒)	0.341	0.341

5.4 FMPYADDの活用

表2に示したように，各ケースにおいてFMPYADD命令の使用頻度が異なる．図5で最もよい性能を示している6段5列同時消去では30個の積和計算のうち13個しかFMPYADD命令が使用されていない．FMPYADD命令を活用することにより，さらに性能向上が期待できる．そこで，PILOT-1のノードプロセッサがPA-RISC1.1アーキテクチャと互換であることを利用し，積和2重ループ部

分を HP9000 model735 の Fortran コンパイラを用いて分割コンパイルした．そして，PILOT-1 において，そのオブジェクト・ファイルをリンクすることにより実行モジュールを作成した．先に示した 6 段 5 列同時消去でキャッシュのブロッキングを行った場合に適用した結果を表 5 に示す．

表 5: HP-UX Fortran コンパイラを使用した場合

	PILOT-1	HP-UX
MFLOP/s	356.7	385.6
処理時間 (秒)	1.875	1.734
前進消去 (秒)	1.534	1.343
後退代入 (秒)	0.341	0.391

HP-UX の Fortran コンパイラでコンパイルした場合，最内ループにおける使用命令数は FLOAD 28，FSTORE 5，FMPYADD 27，FMPY 3，FADD 3，LDO 5，ADDIB 1 の計 72 命令であった．このとき，PILOT-1 のコンパイラでコンパイルした場合との大きな差は，FMPYADD 命令の数であり，2 倍の数の FMPYADD 命令が生成されている．これにより，全体の性能としては 1 割強の性能向上が達成できている．このことから，コンパイラの性能が向上することにより，さらに PILOT-1 の性能を引き出すことが可能であることが分かる[1]．

5.5 台数効果について

図 7に台数効果のグラフを示す．グラフはノード数が 2 の時を 1 として，それに対する比率を表したものである．実行モジュールは 5.4節で示したものを用いた．

全処理については，後退代入の部分が並列に実行できないため，ノード数が増加するに従い台数効果が得られなくなる．しかしながら，前進消去部分のみであればスーパリニアな台数効果が得られており，15 ノードの時は，496.4 MFLOP/s の性能を示している．これは，ノード数が増えるに従いキャッシュが有効に働き，8 ノードの時には行列 A が 2 次キャッシュに収まるためである．

6 おわりに

本研究では，超並列計算機 CP-PACS のプロトタイプである PILOT-1 の性能評価を LINPACK ベンチマークを用いて行った．多段多列同時消去，ブロードキャスト機能，キャッシュによるブロッキング，FMPYADD 命令の活用等により，PILOT-1 における LINPACK ベンチマークの高速化を行い，15 ノードで 385.6 MFLOP/s の性能を得た．このとき，単純なガウスの消去法に比べて約 3 倍の性能向上が確認された．また，前進消去の部分に関しては，十分な台数効果が得られ，この時スーパリニア現象が確認された．

[1]CP-PACS では，コンパイラは疑似ベクトル機能に対応して，モジュロ・スケジューリングが採用されているため，命令間の依存関係をなくすことにより FMPYADD 命令の有効利用が期待できる．

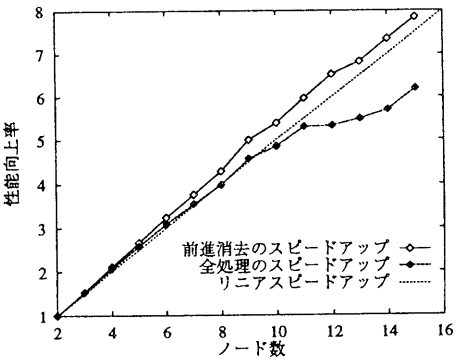

図 7: 台数効果

今回の評価では，行列 A を各ノードに配置させる際に列方向のサイクリック分割を用いたが，ブロードキャストを複数回用いることにより，マルチキャストと同じ機能を実現できる．このことを利用して，行と列の両方向にサイクリックに分割した場合の評価を行う必要がある．

謝辞

本研究に進めるにあたり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アーキテクチャ研究室諸氏に深く感謝します．なお，本研究の一部は創成的基礎研究費 (07NP0401) 及び文部省科学研究費 (奨励 (A)07780222) の補助によるものである．

参考文献

[1] 中澤喜三郎 他：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.18-28, 1996 年 1 月

[2] 田中輝雄 他：識別子を用いたデータ転送方式を基本とする MIMD 型並列計算機アーキテクチャ，並列処理シンポジウム JSPP'89 論文集，1989 年

[3] H. Nakamura, et al. : A Scalar Architecture for Pseudo Vector Processing based on Slide-Windowed Registers, Proc. of International Conference on Supercomputing, pp.298-307, 1993

[4] Dongarra, J. J. : Performance of Various Computers Using Standard Linear Equations Software, Computer Science Department, University of Tennessee, TN 37996-1301, September 1995

[5] 高橋：分散メモリ型計算機 PAX 上におけるガウスの消去法，JSPP'91，pp.333-340，1991 年

[6] 建部修見：分散メモリ型並列計算機による LU 分解，情処研報，HPC-57-10，pp.55-60，1995 年

[7] 小国力 他：行列計算ソフトウェア：WS、スーパーコン、並列計算機，丸善，1991 年