ハイパフォーマンス
コンピューティング 77－4
（1999. 8. 2）

並列プログラム自動最適化ツール TEA Expert の実並列計算機における評価

板倉憲一[†1] 平野基孝[†2] 朴泰祐[†3]
佐藤三久[†4] 建部修見[†5] 関口智嗣[†5]

TEA Expert は，逐次または並列プログラム中に記述された，最適化要素に対応したパラメータを変化させ，自動的に実行することによって，最適なパラメータセットを決定し，プログラムの最適化を支援するシステムである．ユーザに指定された範囲でパラメータを変更し，実行プログラムの作成・実行・時間計測を繰り返す試行実験を動的に行う．

本稿では，TEA Expert を並列計算機 SR2201 の上に実装し，NPB Kernle CG の性能チューニングに適用した例について述べる．同じアルゴリズムと性能パラメータを持つ Kernel CG のプログラムを 2 種類の方法で作成し，それぞれの最適なパラメータセットを得た．特に，SR2201 固有の擬似ベクトル処理機構を使用する場合には，予想したパタメータセットが最適でないことが分った．

Evaluation of TEA Expert – an automated performance tuning environment – on Real MPP System

KEN'ICHI ITAKURA,[†1] MOTONORI HIRANO,[†2] TAISUKE BOKU,[†3]
MITSUHISA SATO,[†4] OSAMU TATEBE[†5]
and SATOSHI SEKIGUCHI[†5]

TEA Expert is a set of tools which support the performance tuning process for sequential and parallel high performance software by automating exhaustive execution to search the optimal set of tuning parameters.

In this paper, NAS Parallel Benchmarks Kernel CG was tuned on a parallel processor S-R2201 by TEA Expert System. We developed two versions of programs for Kerel CG which have the same algorithm and tuning parameters. In one case, using the pseudo vector processing feature on SR2201, it was shown that TEA Expert System can estimate the optimal parameters which cannot be found by rough human estimation.

1. はじめに

TEA Expert[1)] は逐次または並列プログラムの性能に影響を及ぼす，プログラム中の各種パラメータを最適化するツールである．

一般にプログラムを最適化する場合には，プログラム中のアルゴリズムやデータのアクセス方法などを変更し，演算パイプラインやキャッシュメモリ等を有効に活用する方法を考える．また，並列計算機の場合には，ノード台数，ノードの計算性能，ノード間の通信性能等によって，処理やデータの分割方法を変更し，最短の処理時間を探ることになる．TEA Expert ではこのような性能最適化に関するパラメータを性能パラメータと呼ぶ．

このような性能パラメータには，コンパイラによっては決定できないものも多い．例えば，キャッシュのサイズに依存するものについてはあらかじめコンパイラに組み込んでおくことが困難であり，さらにデータのアクセスパターンを把握する必要があるため，プログラマによるサポートが必要である．現在の多くのコンパイラはプログラムの持つ静的な情報のみに頼っており，キャッシュサイズなど，実行環境によって動的に変化する情報はユーザが与える必要がある．

TEA Expert は，この性能パラメータを与えられた条件の中で変化させてプログラムを実行し，処理時間を評価することによって，最適なパラメータセットを決定するシステムである．

†1 筑波大学 計算物理学研究センター Center for Computational Physics, University of Tsukuba
†2 株式会社 SRA Software Research Associates, Inc.
†3 筑波大学 電子・情報工学系 Institute of Information Sciences and Electronics, University of Tsukuba
†4 新情報処理開発機構 Real World Computing Partnership
†5 電子技術総合研究所 Electrotechnical Laboratory

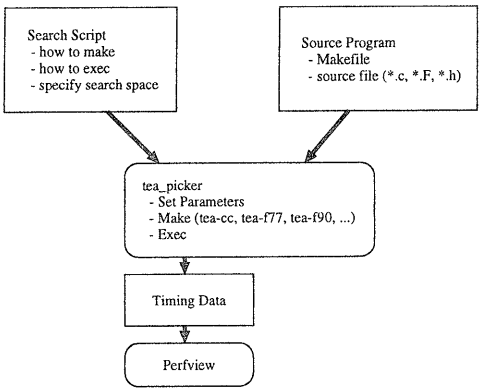

図1　TEA Expert の構成

一般に，プログラムが性能パラメータを考慮して記述されていれば，性能パラメータの変更によって，他のプラットフォームの上で性能を低下させずに実行することが可能である．しかし，新しいプラットフォーム上での最適なパラメータセットを決定する為には，多くの試行が必要となる．TEA Expert はこのような試行と評価を自動化する．

現在，特に，このような技術が必要とされているのは，BLAS などの数値計算の基本カーネルになるルーチン群である．例えば，ATLAS[2] は，行列乗算ルーチン (dgemm) に対し，ブロッキング・サイズなどを変えながら自動的に実行することによって，そのプラットフォームに対応するルーチンを生成するプログラムである．これによって，ベンダ提供のルーチンに匹敵する性能を持つルーチンが生成できることが報告されている．TEA Expert は，このような手法を特定のルーチンに限らず，一般的なプログラムに適用できるように拡張している．

性能解析結果を把握するのももちろん重要であるが，この結果を自動的に反映してチューニングするシステムが期待されている．コンパイラと統合し解析するシステムが最も期待される手法であるが，コンパイラがプログラムの特性を読み切れない場合には，何らかの実行状況に関する情報が必要であり，プログラマからの情報も必要となっている．

2. TEA Expert

2.1 構成と概要

現在，TEA Expert は C と FORTRAN を対象に実装されている．図1に TEA Expert の構成および処理の概要を示す．

- search script
  - 測定対象プログラムの生成方法の記述
  - 測定対象プログラムの実行方法の記述
  - パラメータに関しての探索空間を記述
- source program
  - プログラムをコンパイル，リンクするための Makefile
  - プログラムソースファイル (*.c, *.F, *.h)
- tea_picker
  与えられたパラメータ空間からパラメータを設定し，測定対象プログラムのコンパイル，実行をするプログラム．
  コンパイルには，パラメータをソースプログラムに反映させるプリプロセッサ付きのコンパイラ (`tea_cc`, `tea_f77`, `tea_f90`) を使用する．
- timing data
  パラメータ値と実行時間をレコードとして持つデータベース
- perfview
  timing data を可視化するグラフツール

以下，各部の詳細についての述べる．

2.2 評価対象 source program

TEA Expert の対象となる source program 中では，性能パラメータや，時間計測範囲を明示的に示す．この指示は，C 言語の時は `#pragma TEA`，FORTRAN 言語の場合は `!$ TEA` で始まる directive によって行われる．現在サポートされている directive には以下のものがある．

- 測定区間の指定．
  `#pragma TEA begin`
  `#pragma TEA end`
  TEA Expert は，ここで指定された区間の実行時間が最小になるようにパラメータを探索する．ソースのこの個所には，性能測定のためのタイマールーチンと結果の出力ルーチンが埋め込まれる．
- パラメータの指定．
  `#pragma TEA parameter` *param*
  param の識別子が性能パラメータであることを宣言する．そのパラメータは tea_picker で指定された値に置き換えられる．
- ループの unroll の指定．
  `#pragma TEA unroll(`*p1,p2,...*`)  latency(`*p*`)`
  loop unrolling は数値計算ルーチンにおいて，性能に影響を及ぼす重要なプログラムの最適化技法であるため，TEA Expert ではその変形をサポートする．但し，直後に unroll が可能な for ループがなくてはならない．unroll のパラメータが複数の場合はループ交換が可能であるとみなし，多重のループのブロッキングを行う．また，`letency` は，unroll された文間のスケジューリングのためのパラメータである．

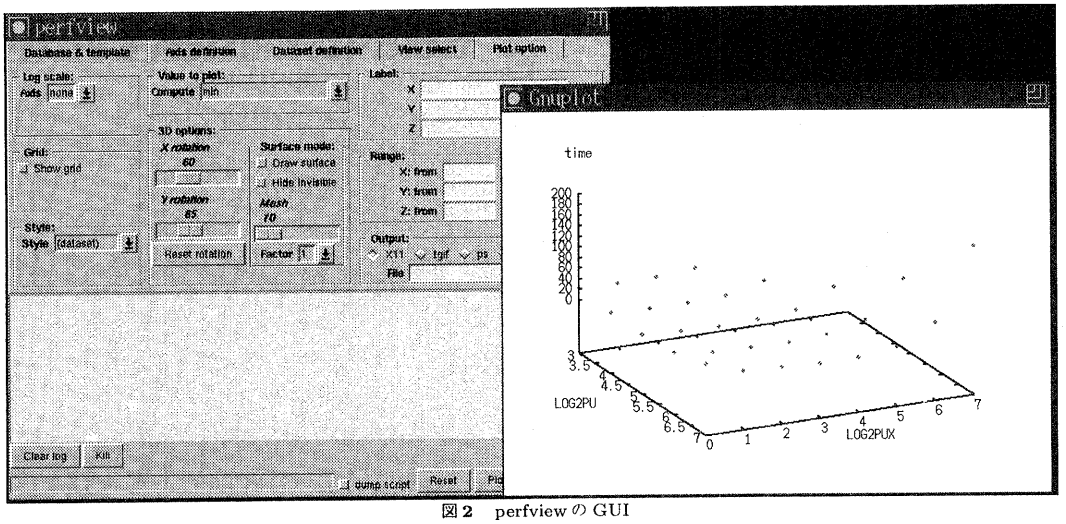

図2　perfview の GUI

2.3　パラメータ探索空間設定 search-script

serach-script では評価対象プログラムのコンパイル及び実行方法と，与えられた性能パラメータの探索空間を記述する．

Execute　*program-name arg1 arg2 ...*

プログラムの実行方法を指定する．指定のない場合は，a.out を引数なしで実行する．

Make　*make commands*

プログラムをコンパイルするためのコマンドを指定する．指定されない場合には，make コマンドを実行する．

Search　*parameter range*

性能パラメータの探索空間を指定する．*parameter range* は，以下の形式で，これらの積 (&)，和 (|) を指定できる．

- 範囲指定 param=[min,max,step]
- 列挙指定 param={value1,valu2, ...}
- 値指定 param = value

2.4　最適パラメータサーチプログラム tea_picker

tea_picker は，serach-script で与えられた性能パラメータを変化させ，プログラムを実行させる．

まず，tea_picker は，Search で指定された探索空間の中から未試行の性能パラメータ値を設定し，ファイル **TEA_PARAM** にその値を保存する．次に，Make で指示された手順に従って実行モジュールを生成する．この時に，性能パラメータの値を埋め込むプリプロセッサ付きコンパイラ (**tea-cc**, **tea-f77**, **tea-f90**) を用いる．このプリプロセッサでは時間測定ルーチンを埋めこみ，**TEA_PARAM** からパラメータの値を設定し，指示された loop unrolling を行う．最後に，Execute で指定された手順によりプログラムを実行する．実行時に性能パラメータの値を引数等に利用する場合は、**TEA_PARAM** から得ることができる．実行されたプログラムの結果は，パラメータの値と実行時間の組として，実行結果データベースに記録される．

2.5　性能パラメータ可視化ツール perfview

perfview は gnuplot を利用した，実行結果を可視化するためのツールである．このツールでは，実行結果データベース内のデータに対しパラメータを変化させた場合の実行時間をグラフとして表示する．指定された形式でデータを作成すれば，perfview は一般的な性能データ表示に利用できるが，TEA Expert が生成する実行性能データベースは，perfview に直接入力できるフォーマットである．

perfview は 2 つのパラメータを変化させて，3 次元の表示をすることもできる．**図2** にその GUI を示す．X window 上では，3 次元空間で回転などをさせながら結果を見ることができる．TEA Expert は，バッチ的に実行時間を評価するので，広範囲のパラメータ空間の網羅的な自動的に検索できるが．しかし，効率的な探索空間を設定するには，探索空間内を粗く評価し，perfview によってパラメータと実行時間の関係を可視化することが有効である．

3.　TEA Expert による並列プログラムの最適化例 –NPB Kernel CG–

本節では，NPB version 1 Kernel CG[3] の超並列計算機上での最適化を，TEA Expert によって行う．

3.1　NPB Kernel CG の並列化

Kernel CG は正値対称な大規模疎行列の最小固有値を CG (Conjugate Gradient) 法によって求めるベ

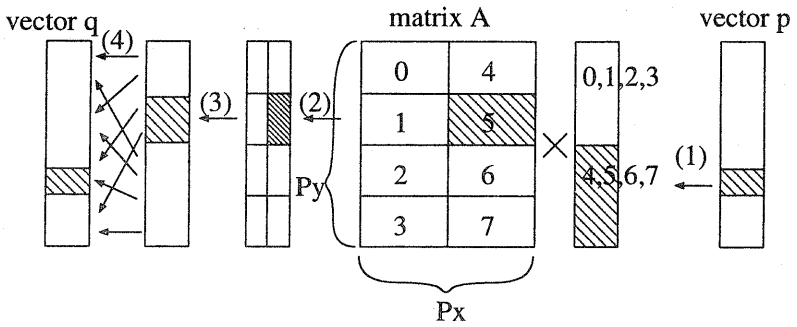

図3　matvec の処理とデータ転送パターン

```
Search (L2PU in {3}) * (L2PUX in {0, 1, 2, 3});
Search (L2PU in {4}) * (L2PUX in {0, 1, 2, 3, 4});
Search (L2PU in {5}) * (L2PUX in {0, 1, 2, 3, 4, 5});
Search (L2PU in {6}) * (L2PUX in {0, 1, 2, 3, 4, 5, 6});
Search (L2PU in {7}) * (L2PUX in {0, 1, 2, 3, 4, 5, 6, 7});
```

図4　serach script

ンチマークである．Class A の問題サイズでは 疎行列のサイズは 14000 × 14000, 非零要素率は約 1% であり一行に平均 140 個ある．以下に Kernel CG のメインループを示す．

```
do k=1,KERNITR
   call matvec(a, colidx, rowstr, p, q)
   alpha=rho/dotpro(p, q)
   z(1:N)=z(1:N)+alpha*p(1:N)
   rho0=rho
   r(1:N)=r(1:N)-alpha*q(1:N)
   rho=dotpro(r, r)
   beta=rho/rho0
   p(1:N)=r(1:N)+beta*p(1:N)
enddo
```

このメインループにおいて最も処理時間のかかる部分は行列・ベクトルの積 (matvec) である．matvec の処理は仮想的な 2 次元 PU 平面 ($P = P_x \times P_y$, P は PU 台数) に行列をブロック-ブロック分割して行なう．この P_x, P_y の組合せによりデータ転送時間の最適化が行える [4]．

2 次元ブロック-ブロック分割した matvec の並列処理は以下の手順で行われる．

(1) Collection:　P_y 台の PU に分散されている被乗数ベクトル p を collective 通信によりまとめる

(2) Multiplication:　行列 A の部分行列と p の部分ベクトルの乗算を行なう

(3) Summation:　P_x 台の PU に分散されている部分ベクトルに対し collective 通信により部分和を求め部分ベクトルを作る

(4) Shuffle:　部分ベクトル q の位置を合せるため Shuffle 転送を行なう

図3 に，8 PU ($P_x = 2, P_y = 4$) での matvec の処理を示す．

matvec で行われる演算量は，疎行列の非零要素がランダムに現れるため，P_x, P_y にはほぼ無関係であると予想される．これに対し，通信時間に関しては，定性的には P_x の増加は Summation 時間の増加を招き，P_y の増加は Collection 時間の増加を招く．全ての通信におけるデータ転送の回数とデータ転送量を表 1 に示す．このため，Summation の時間と Collection の時間がほぼ同じような割合いで，トレードオフ関係にあるならば，最適な P_x を見つける必要がある．

表1　matvec におけるデータ転送

処理パターン	転送回数	総転送量
Collection	$log_2 P_y$	$N/P \times (P_y - 1)$
Summation	$log_2 P_x$	$N/P_x \times log_2 P_x$
Shuffle	1	N/P

3.2　データ転送時間の最適化

まず，並列計算機 SR2201 において，C 言語と MPI 通信ライブラリを用いて，2 次元ブロック-ブロック分割に基づく Kernel CG プログラムを実装した．そして，全体の PU 台数 P と X 方向の PU 台数 P_x を性能パラメータとし，図 4 の search script で各 PU 台数における最適な P_x を求めた．この結果のグラフを図 5 に示す．

この結果から，最適な 2 次元ブロック-ブロック分割の方法は，$P = 8$ の時に $P_x = 4$, $P = 128$ の時に

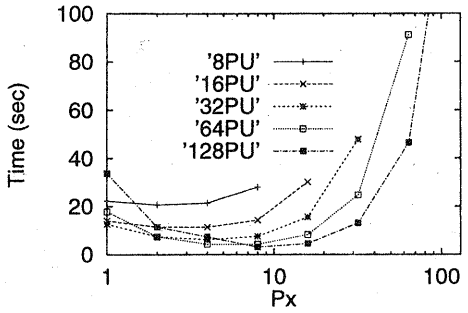

図5　C言語+MPI通信の結果

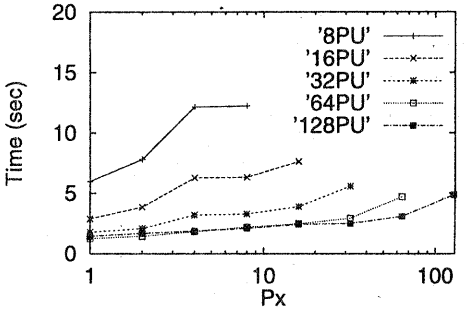

図6　FORTRAN言語+native通信の結果

$P_x = 8$ 等となることが分る．このことから，予想通り，演算量は P_x によらず一定☆であり，Summationの時間と Collection の時間のオーダが同じであると考えられる．そして，この2つの通信時間のバランスによって，最適な P_x が決定される．

ここで評価に用いたプログラムでは，

```
#ifdef _TEA
#pragma TEA parameter L2PU
#pragma TEA parameter L2PUX
#else
#define L2PU 4
#define L2PUX 1
#endif
```

という記述を使って，2つの性能パラメータ L2PU, L2PUX を導入した．それぞれのパラメータは $\log_2 P$, $\log_2 P_x$ を表わす．tea-cc 等のコンパイラではコンパイル時に，特別なマクロ_TEA が定義される．これにより，一つのプログラムを TEA Expert 環境でのコンパイルと，そうでない場合の両方に対応させることができる．このため，プログラムのデバッグ時には TEA Expert の環境ではなく，通常の cc や，デバッグツールが利用可能である．

このようにして，デバッグの完了したプログラムと性能パラメータの探索空間を指定したスクリプトを TEA Expert の環境に与えることで，図5を作成するために必要な試行実験を自動的に行える．一般的な性能チューニングの過程では，このような試行実験作業は，パラメータを変える度に，エディタでマクロの定義 (#define) や Makefile を書き変えるか，それと同等の事を行う専用のシェルスクリプトを利用していたが，この TEA Expert 環境を使うことにより，非常に効率良く実験を進めることができた．

次に，同じアルゴリズムを FOTRAN と SR2201 の native な通信ライブラリを用いて記述したプログラムに対し，再び TEA Expert による評価を行なう．これは，SR2201 の特徴である擬似ベクトル処理機構[5]と PU 間のゼロコピー通信[6]を活かし，プログラムを SR2201 専用にチューニングした場合を想定している．FORTRAN の場合は

```
!$ TEA parameter L2PU
!$ TEA parameter L2PUX
```

の記述により，性能パラメータを指定する．

このプログラムに対して，C言語の場合と全く同じ search script で各台数における最適な P_x を求めた．この結果のグラフを図6に示す．

図6の結果では，全ての PU 台数において，$P_x = 1$ が最も処理時間が短い結果となり，2次元ブロック-ブロック分割ではなく，単純な1次元行ブロック分割が最適であることを示している．

SR2201 の native な通信ライブラリによって，通信時間の全実行時間に対する割合は削減される．そこで，単純な転送パターンから転送オーバヘッドと転送スループットを求め，表1を考慮して通信時間を理論的に求めてみると，1回の通信時間は，図7のようになり，やはり2次元ブロック-ブロック分割が有効であると考えられる．

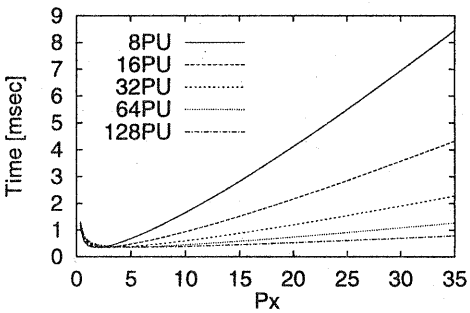

図7　native 通信による matvec 1回の通信時間（理論値）

しかし，擬似ベクトル処理機構を用いると，matvec

☆ 最終的なコンパイルには gcc を用いており，後で述べる擬似ベクトル処理機構は使用していない．

の中で行われる Multiply の処理時間 (図 3 の (2)) が P_x によって変化することが分っている[4]. これは, 疎行列 A の 1 行の平均非零要素数が約 140 であることと, SR2201 用の FORTRAN コンパイラでは, Multiply の最内ループにおいて要素数が 50 以上の時のみ擬似ベクトル処理機構が有効になり, それ以下の場合には 1 要素あたりの処理時間が長くなってしまうことが原因である. つまり, $P_x = 2$ 以上では, 平均非零要素数は 70 以下となり, 擬似ベクトル処理機構が有効になる 50 を上回る行が少くなるため, P_x の増加と共に Multiply の処理時間が増加する. これによって FORTRAN 言語と native 通信ライブラリを用いたプログラミングでは $P_x = 1$ が最適という結果となる.

このように, 並列計算機の基本的な性能指標であるデータ転送時のオーバヘッドとスループットから, 通信時間のモデルを立て, アプリケーションプログラムの通信時間を見積もることはある程度可能である. しかし, そのマシン固有の特性は, 予測するのが困難である. SR2201 では, 擬似ベクトル処理機構がマシン固有の特性として現れたが, 他の一般的な RISC プロセッサでは, キャッシュメモリの容量やウェイ数等の構成が固有の特性として現れやすいだろう. また, 最近多く用いられている SMP 構成のノード内処理においては, 共有メモリと各キャッシュメモリ間のデータ転送性能も問題となりやすい.

最大性能を発揮するようなプログラムやサブルーチン/ライブラリ等を作成する場合には, このような各計算機での固有の特性を加味して, いくつかの性能パラメータをあらかじめ決定するところまでをプログラマが行い, 最適なパラメータセットは, 利用する計算機での実測による評価から決定する方法が有効であると言える.

4. おわりに

TEA Expert は, 最適な性能パラメータセットを自動的に探索することより, 性能チューニングを支援するシステムである. プログラマは, 性能に影響のあるプログラム中のパラメータを明示的に指定する. 更にそのパラメータの取りうる範囲を指定し, 探索空間を決定する. この 2 つの情報を基にして, TEA Expert システムは実行時間の測定を自動的に行い, 最適なパラメータセットを決定する.

本稿では, 実際に並列計算機 SR2201 上での NPB version 1 Kernel CG ベンチマークの性能チューニングを TEA Expert システムを用いて行った. C 言語と MPI 通信ライブラリを用いたプログラム及び FORTRAN 言語と native な通信ライブラリを用いたプログラムの両方に対して, 同じ性能パラメータを用いて性能チューニングを行った結果, それぞれの場合で, 最適な性能パラメータが違うことが分った. データ通信時間のように, ユーザがプログラムで明示的に記述している部分の処理時間の見積もりは比較的やさしいが, SR2201 の擬似ベクトル処理機構のように, コンパイラがそれを使用するか否かを暗黙のうちに決定する部分の処理時間の見積もりは難しく, 擬似ベクトル処理機構を利用する FORTRAN を用いたプログラムでは予想した 2 次元ブロック-ブロック分割が最適な構成とはならないことが分った.

ユーザがプログラム中でコントロールできる性能パラメータによってプログラムは最適化できる可能性は高いが, 最適な性能パラメータセットを理論的に決定することは難しい. 今後, プロセッサの内部や, SMP 構成のノードの動作等は更に複雑になるであろう. このような状況に対して, 数値ライブラリ等を TEA Expert のようなシステムによって実測し, 性能チューニングすることが必要になると考えられる.

謝　辞

日頃, 性能評価技術に関し議論して頂く TEA グループのメンバー諸氏に感謝いたします. TEA グループは, 研究技術組合新情報処理開発機構・電子技術総合研究所・筑波大学を中心とする性能評価に関する研究グループである. また, 本研究の一部は筑波大学計算物理学研究センターの協力によるものである.

参考文献

1) 佐藤 三久、建部 修見、関口 智嗣、朴 泰祐:「自動適応並列プログラム性能最適化ツール TEA Expert」, 情報処理学会 HPC 研究会報告 72-3, pp. 13–18 (1998).
2) Whaley, R. and Dongarra, J.: Automatically Tuned Linear Algebra Software, Technical report, University of Tennessee (1997). http://www.netlib.org/utk/project/atlas.
3) Bailey, D., Barszcz, E., Barton, J., Browning, D., Carter, R., Fatoohi, R., Fineberg, S., Frederickson, P., Lasinski, T., Schreiber, R. and Simon, H.: "The NAS Parallel Benchmarks", Technical report, NAS (1994). RNR-94-007.
4) 板倉憲一, 松原正純, 朴泰祐, 中村宏, 中澤喜三郎:「超並列計算機 CP-PACS における NPB Kernel CG の評価」, 情報処理学会論文誌, Vol. 39, No. 6, pp. 1757–1765 (1998).
5) Nakamura, H., Imori, H., Nakazawa, K., Boku, T., Nakata, I., Yamashita, Y., Wada, H. and Inagami, Y.: "A Scalar Architecture for Pseudo Vector Processing based on Slide-Windowed Registers", *Proc. of ACM International Conference on Supercomputing '93*, pp. 298–307 (1993).
6) (株) 日立製作所:「リモート DMA 転送 FORTRAN 版 使用の手引」, 6A20-3-312 (1996).