Himeno BMT によるハイパフォーマンスコンピュータの性能評価

渡部 善隆　　南里 豪志　　藤野 清次

概要

HimenoBMT は物理・工学の広い範囲で現れる Poisson 方程式を 3 次元の一般座標系による差分法により離散化し Jacobi 反復法により近似解を求めるプログラム群であり，科学技術計算における性能評価試験プログラムとして広く利用されている．本稿では HimenoBMT の Fortran プログラム群について，それらの特徴を分析するとともに，倍精度版とあわせて行なったベクトル並列計算機，スカラ SMP 計算機，PC クラスタなどの性能評価試験の結果について報告する．

Performance Evaluation of High Performance Computers by Himeno BMT

Yoshitaka Watanabe　　Takeshi Nanri　　Seiji Fujino

Abstract

HimenoBMT is a widely used benchmark program for the three-dimensional Poisson equation by Jacobi iteration with finite difference discretization. This paper describes some features of HimenoBMT and gives performance evaluation results for high performance computers.

1　はじめに

HimenoBMT は理化学研究所・情報基盤研究部・情報環境室長の姫野龍太郎氏が作成したベンチマークテストであり，熱伝導場や非圧縮性流体など物理・工学の広い範囲で現れる Poisson 方程式を 3 次元の一般座標系による差分法により離散化し，Jacobi 反復法により近似解を求めるものである [1].

HimenoBMT プログラムは Fortran と C のソースコードを web サイトからダウンロードすることができる．また PC/AT 互換機用と Machintosh 用の実行形式ファイルも入手可能である．2003 年 6 月現在公開されているソースコードを表 1 に示す.

本稿では，Fortran 90 の逐次版，FORTRAN 77+MPI および Fortran 90+OpenMP の並列版について HimenoBMT の特徴を分析するとともに，倍精度版とあわせて行なったベクトル並列計算機，スカラ SMP 計算機，PC クラスタなどの性能評価試験結果について報告する．したがって，以下の評価・考察はその他のプログラムには必ずしも該当しない場合があることに注意されたい.

表 1: HimenoBMT ソースコード

	プログラム言語 ＋ 並列化手段	配列宣言 動的	配列宣言 静的
逐次版	FORTRAN 77		○
	Fortran 90	○	
	C	○	○
並列版	FORTRAN 77 + MPI		○
	FORTRAN 77 + OpenMP		○
	Fortran 90 + OpenMP	○	
	C + MPI		○
	C + OpenMP	○	

2　HimenoBMT

2.1　計算サイズ

一般に知られているように，計算する配列のサイズ (ループ長) が小さい場合，ベクトル計算機では演算性能は出にくく，反対にスカラ計算機ではキャッシュの効果で性能を発揮する．しかし配列のサイズ

九州大学情報基盤センター
Computing and Communications Center, Kyushu University

が大きくなるにしたがい，ベクトル計算機の性能は一定値に近づく反面，スカラ計算機はキャッシュに納まらなくなると極端に性能が落ちる場合がある．HimenoBMT では，計算機の性能を公平に測定する目的から 5 通りの計算サイズを用意している．表 2 は各サイズの i,j,k の値と，大阪大学の NEC SX-5 で測定したハードウェア情報による使用記憶容量である[*1]．i,j,k は x,y,z 方向の分割数に対応する．プログラムでは i,j,k の値に 1 を加えた 3 次元配列 14 個分に相当する部分が記憶容量の大半を占める．

表 2: HimenoBMT の計算サイズ

サイズ	$i \times j \times k$	記憶容量
XS	64 × 32 × 32	48MB
S	128 × 64 × 64	64MB
M	256 × 128 × 128	272MB
L	512 × 256 × 256	1856MB
XL	1024 × 512 × 512	14448MB

2.2 性能測定方法

HimenoBMT の Fortran プログラムは, 各計算サイズにおいて, まず Jacobi 反復を 3 回行ない, 反復に要した経過時間を組込みサブルーチン `SYSTEM_CLOCK` (MPI 版は `mpi_wtime` 関数) によって測定する．その後，実行時間が 1 分程度となるように反復回数を設定し直して再実行を行ない，経過時間と演算数から 1 秒間に実行可能な浮動小数点演算の回数: MFLOPS(Mega FLoating Operations Per Second) 値を求める．

計算の核となる部分は図 1 に示す 3 重ループである (空白・大文字/小文字・継続行の位置を修正している)．ループ内の演算は加算 14 回，減算 7 回，乗算 13 回，合計 34 回であり，これにループの回転数 `(kmax-2)×(jmax-2)×(imax-2)` をかけたものを 1 回の Jacobi 反復に要する浮動小数点演算数としている．なお，以下に示す性能評価結果の値はすべて表示桁以降を切り捨てた MFLOPS 値である．

2.3 配列の値

3 重ループの計算の後，配列 `p` が Fortran 90 プログラムでは

```
p(2:imax-1,2:jmax-1,2:kmax-1) &
     = wrk2(2:imax-1,2:jmax-1,2:kmax-1)
```

によって更新される．その他の変数 `a`, `b`, `c`, `wrk1`, `bnd`, `OMEGA` の値は反復に依存しない．配列の値の設定は 3 重ループを行なう部分とは別のサブルーチンに記述されており，配列 `b` と `wrk1` の値はすべての要素が 0，`c` と `bnd` はすべて 1，`a` は 1 または 1/6 に設定されている．主要 3 重ループの変数 `b`, `wrk1`, `c`, `bnd`, `a` を定数に置き換えたものを図 2 に示す．幾つかの処理系で変数の値を変えて実験した限り，反復に依存しない規則的な変数であることを考慮した最適化は特に行なわれていなかった．

2.4 倍精度版の作成

HimenoBMT の主要な計算は単精度で行なわれる．大規模数値計算の多くが倍精度で行なわれることを考慮して，ダウンロードした単精度プログラムを修正した倍精度版ともあわせて性能評価を行なった．倍精度への変更点は，変数の宣言部分と組込み関数の変更，定数の書き換え (`1.0` → `1.0D0` など) である．

図 3 は XS サイズの逐次版において Jacobi 反復を 10,000 回まで繰り返し，変数 `GOSA` の値をプロットした収束状況の図である．計算は FUJITSU GP7000F モデル 900 で行なった．図中の “quadruple precision” は，倍精度版のプログラムに 4 倍精度への精度拡張オプションを付加して作成した実行可能ファイルを用いて測定した結果である．単精度逐次版の XS

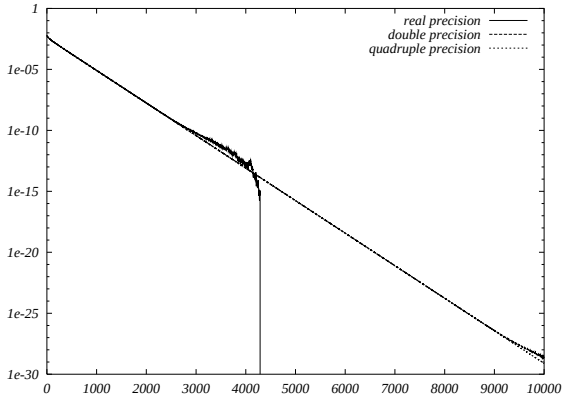

図 3: Jacobi 反復法の収束状況 (XS サイズ)

サイズでは，4290 回以降 `p` の値が更新されなくなる，すなわち `GOSA` の値が 0 になることがわかった．

2.5 MPI プログラムの配列分割

MPI プログラムでは，i, j, k 方向をそれぞれ $\hat{i}$, $\hat{j}$, $\hat{k}$ 分割し，$\hat{i} \times \hat{j} \times \hat{k}$ 並列実行する．また，配列も $\hat{i} \times \hat{j} \times \hat{k}$ 分割される．このため，翻訳時に $\hat{i}$, $\hat{j}$, $\hat{k}$ の値をパラメータとして指定する必要がある．性能測定では，表 3 に示す分割指定を行なった．

2.6 最適化オプション

翻訳時に指定する最適化オプションよる実行性能の違いを調査した．表 4 は FUJITSU PRIMEPOWER 850 における M サイズ, L サイズの結果，表 5 は FU-

[*1] プログラム全体の実行に要した記憶容量であり，特に XS, S サイズについては配列に必要な値よりも大きめの数値になっている．

```
GOSA = 0.0
do k = 2,kmax-1     !
do j = 2,jmax-1     !--> kmax, jmax, imax はサイズ (XS,S,M,L,XL) によって変化
do i = 2,imax-1     !
  s0 = a(I,J,K,1)*p(I+1,J,K) + a(I,J,K,2)*p(I,J+1,K) + a(I,J,K,3)*p(I,J,K+1) &
      +b(I,J,K,1)*(p(I+1,J+1,K)-p(I+1,J-1,K) - p(I-1,J+1,K)+p(I-1,J-1,K))    &
      +b(I,J,K,2)*(p(I,J+1,K+1)-p(I,J-1,K+1) - p(I,J+1,K-1)+p(I,J-1,K-1))    &
      +b(I,J,K,3)*(p(I+1,J,K+1)-p(I-1,J,K+1) - p(I+1,J,K-1)+p(I-1,J,K-1))    &
      +c(I,J,K,1)*p(I-1,J,K) + c(I,J,K,2)*p(I,J-1,K) +c(I,J,K,3)*p(I,J,K-1)+wrk1(I,J,K)
  ss = (s0*a(I,J,K,4)-p(I,J,K))*bnd(I,J,K)
  GOSA = GOSA+SS*SS
  wrk2(I,J,K) = p(I,J,K) + OMEGA*SS
end do
end do
end do
```

図 1: Jacobi 反復の主要 3 重ループ (逐次版)

```
GOSA = 0.0
do k = 2,kmax-1     !
do j = 2,jmax-1     !--> kmax, jmax, imax はサイズ (XS,S,M,L,XL) によって変化
do i = 2,imax-1     !
  s0 = p(I+1,J,K) + p(I,J+1,K) + p(I,J,K+1) + p(I-1,J,K) + p(I,J-1,K) + p(I,J,K-1)
  ss = s0/6.0 - p(I,J,K)
  GOSA = GOSA+SS*SS
  wrk2(I,J,K) = p(I,J,K) + OMEGA*SS
end do
end do
end do
```

図 2: 図 1 の反復に依存しない変数を書き下したプログラム

表 3: MPI プログラムの分割指定

並列度	$\hat{i}$	$\hat{j}$	$\hat{k}$
1	1	1	1
2	2	1	1
4	2	2	1
8	2	2	2
16	4	2	2
32	4	4	2

JITSU VPP5000/64 における結果である．プログラムは逐次単精度版である．VPP5000/64 の `-Wv,-Me` は多重ループの融合を促進する翻訳時オプションである．このオプションの指定によって図 1 の 3 重ループが一重化され，ベクトル長が長くなることより性能向上が得られる．

表 4: 最適化オプションによる性能差; PRIMEPOWER 850

オプション	M	L
`-O0`	20	20
`-O1`	427	432
`-O2`	466	472
`-O3`	487	490
`-Kfast`	462	466
`-Kfast_GP=2,prefetch=4 -O4`	481	486

表 5: 最適化オプションによる性能差; VPP5000/64

オプション	M	L
`-O0`	1.7	1.8
`-O1`	426	482
`-O2`	766	840
`-O3`	2703	3293
`-O4`	2681	3310
`-O5`	2680	3303
`-O4 -Wv,-Me`	3555	4386

2.7 動的/静的配列宣言による差

Fortran 90 で記述された HimenoBMP プログラムは，配列を動的に宣言する．そのため，配列の大きさは実行時に確定する．一方，FORTRAN 77 で記述された MPI 版プログラムは配列の大きさが翻訳時に確定しているため，コンパイラによっては配列の大きさに応じた最適化が行なわれる．このため，測定結果の表にしばしば見られるように，MPI 版の 1 並列の結果の方が逐次版より高い数値となることがある．

3 測定結果

3.1 FUJITSU GP7000F モデル 900

九州大学情報基盤センターのスカラ SMP 計算機 GP7000F モデル 900 の 32CPU までを使用した

(2003 年 4 月)．仕様は以下の通りである．

プロセッサ	SPARC64 GP; 300MHz; 64CPU
主記憶容量	64GB
1 次キャッシュ	128KB
2 次キャッシュ	8MB
OS	Solaris 7 Generic_106541-24
コンパイラ	Fujitsu Fortran Compiler Driver Version 5.1
最適化オプション	`-Kfast_GP=2,V9,prefetch=4 -O4`

表 6, 表 7 に単精度および倍精度プログラムの測定結果を示す．“2”, “4” などの数字は並列度を表す．自動並列化は，翻訳時オプション `-Kparallel` を指定して作成した実行可能ファイルを用いた結果である．単精度，倍精度とも，自動並列化を含め並列化による

表 6: GP7000F: 単精度

	XS	S	M	L	XL
逐次	153	70	68	64	62
自動並列化 1	77	50	49	46	46
自動並列化 2	159	105	97	91	90
自動並列化 4	292	228	195	182	182
自動並列化 8	551	602	393	364	361
自動並列化 16	1037	1164	782	749	741
自動並列化 32	1665	2162	1678	1445	1424
OpenMP 1	105	61	59	56	55
OpenMP 2	225	131	117	109	108
OpenMP 4	412	290	238	216	215
OpenMP 8	763	802	477	432	429
OpenMP 16	1300	1515	951	862	855
OpenMP 32	1990	2634	2054	1705	1684
MPI 1	198	81	76	69	66
MPI 2	364	167	152	153	140
MPI 4	718	714	298	298	285
MPI 8	1313	1382	581	578	548
MPI 16	2078	2763	1244	1134	1084
MPI 32	1923	5134	4813	2230	2209

性能向上が見られた．ただし，サイズ毎の性能はばらつきが大きく，特に M サイズ以降サイズが大きくなるにしたがい数値が低くなる傾向が見られた．

3.2 FUJITSU VPP5000/64

九州大学情報基盤センターの分散メモリ型ベクトル並列計算機 VPP5000/64 の 32PE までを使用した (2003 年 5 月)．仕様は以下の通りである．

プロセッサ	9.6Gflops/PE; 64PE
主記憶容量	8GB/PE または 16GB/PE
1 次キャッシュ	128KB
2 次キャッシュ	2MB
ネットワーク	クロスバー; 3.2GB/秒/PE×2
OS	UXP/V V20L10 X02111
コンパイラ	Fujitsu UXP/V Fortran V20L20 Driver L02091
最適化オプション	`-Kfast`

表 7: GP7000F: 倍精度

	XS	S	M	L	XL
逐次	118	41	38	36	36
自動並列化 1	109	40	37	35	35
自動並列化 2	133	70	64	62	60
自動並列化 4	256	138	127	122	119
自動並列化 8	486	306	254	245	239
自動並列化 16	861	985	509	490	472
自動並列化 32	1367	1708	1096	956	934
OpenMP 1	94	37	35	33	33
OpenMP 2	174	79	71	68	67
OpenMP 4	333	157	140	135	134
OpenMP 8	623	372	281	269	267
OpenMP 16	1048	1238	559	536	531
OpenMP 32	1543	2074	1223	1055	1041
MPI 1	149	46	40	37	35
MPI 2	276	89	90	86	79
MPI 4	532	201	179	179	167
MPI 8	1043	887	363	361	334
MPI 16	1727	2018	713	683	673
MPI 32	2051	4020	1483	1359	1164

表 8, 表 9 に単精度および倍精度プログラムの測定結果を示す．倍精度の XL サイズにおける”—” は記憶

表 8: VPP5000: 単精度

	XS	S	M	L	XL
逐次	1524	2628	3537	4363	4332
MPI 1	3048	3858	4061	4497	4573
MPI 2	2161	3842	5754	7454	8415
MPI 4	3845	8820	12048	14352	16745
MPI 8	4682	14810	22476	28459	32913
MPI 16	2928	12266	29058	36771	49473
MPI 32	2249	13834	46818	70246	95122

表 9: VPP5000: 倍精度

	XS	S	M	L	XL
逐次	1481	2336	3378	3911	—
MPI 1	2467	3183	3576	4214	—
MPI 2	2502	4248	5489	6172	7591
MPI 4	3718	7732	10592	11891	15316
MPI 8	4696	13381	19308	24673	30559
MPI 16	2970	12119	26677	34022	41861
MPI 32	2262	13889	44589	65157	84201

容量不足で実行できなかったことを意味する．単精度，倍精度ともに計算サイズが大きくなるにしたがって性能が高くなることがわかる．また，倍精度の値が単精度に比較して若干低くなっている．

3.3 FUJITSU PRIMEPOWER 850

九州大学情報基盤センターのスカラ SMP 計算機 PRIMEPOWER 850 の 8CPU までを使用した (2003 年 3 月)．仕様は以下の通りである．

プロセッサ	SPARC64 V; 1.35GHz; 16CPU
主記憶容量	24GB
1 次キャッシュ	256KB
2 次キャッシュ	8MB
OS	Solaris 8 Generic_114665-02
コンパイラ	Fujitsu Fortran Compiler Driver Version 5.3 P-id: 912528-01
最適化オプション	`-Kfast`

表 10, 表 11 に単精度および倍精度プログラムの測定結果を示す．倍精度の XL サイズにおける “—” は

表 10: PRIMEPOWER 850: 単精度

	XS	S	M	L	XL
逐次	453	454	463	467	507
自動並列化 1	448	452	457	461	464
自動並列化 2	904	899	915	927	949
自動並列化 4	1678	1735	1749	1820	1875
自動並列化 8	3347	2646	2830	2868	3607
OpenMP 1	452	454	463	467	470
OpenMP 2	905	901	917	930	975
OpenMP 4	1679	1763	1776	1824	1923
OpenMP 8	3329	2642	2972	2813	3711
MPI 1	543	544	583	590	553
MPI 2	1037	1157	1164	1175	1181
MPI 4	2093	2056	2054	2057	2193
MPI 8	4093	2958	3027	2875	3726

表 11: PRIMEPOWER 850: 倍精度

	XS	S	M	L	XL
逐次	355	355	328	337	—
自動並列化 1	350	340	336	340	—
自動並列化 2	695	677	653	650	—
自動並列化 4	1228	1116	1132	1174	—
自動並列化 8	2803	1481	1483	1595	—
OpenMP 1	350	348	334	342	—
OpenMP 2	687	678	635	653	—
OpenMP 4	1212	1138	1090	1189	—
OpenMP 8	2700	1255	1456	1660	—
MPI 1	312	311	320	333	—
MPI 2	553	540	540	579	—
MPI 4	1020	945	1000	1023	—
MPI 8	2537	1421	1421	1629	—

記憶容量不足で実行できなかったことを意味する．PRIMEPOWER 850 は GP7000F の後継機にあたる．GP7000F と比較するとサイズ毎の値の差が少ない傾向にある．

3.4 NEC SX-7

東北大学シナジーセンターの分散・共有メモリ型ベクトル並列計算機 SX-7 の 1 ノード (共有メモリ型) を使用した (2003 年 4 月)．仕様は以下の通りである．

CPU	8.83Gflops/CPU; 32CPU
主記憶容量	256GB
キャッシュ	128KB
OS	SUPER-UX 13.1
コンパイラ	FORTRAN90/SX Version 2.0
最適化オプション	省略値

表 12, 表 13 に単精度および倍精度プログラムの測定結果を示す．VPP5000 と同様，単精度，倍精度とも

表 12: SX-7: 単精度

	XS	S	M	L	XL
逐次	1437	2339	2989	3419	3649
OpenMP 1	1374	2304	2904	3175	3430
OpenMP 2	2963	4911	6153	6917	7345
OpenMP 4	5430	9435	11926	13667	14572
OpenMP 8	8707	18240	23802	26626	28755
OpenMP 16	13709	31862	43095	48415	51399
MPI 1	1742	2590	3135	3006	3482
MPI 2	1654	3755	5482	6143	6920
MPI 4	2205	6617	10589	11953	13501
MPI 8	4016	10939	20576	23704	26743
MPI 16	2497	9398	21979	37676	43358

表 13: SX-7: 倍精度

	XS	S	M	L	XL
逐次	1369	2159	2810	3278	3547
OpenMP 1	1421	2350	2877	3427	3716
OpenMP 2	2770	4772	6002	6774	7220
OpenMP 4	5388	9269	11839	13603	14646
OpenMP 8	9701	17899	23406	26485	28373
OpenMP 16	10722	32566	44590	44590	51353
MPI 1	1859	2705	3341	3384	3629
MPI 2	1957	3791	5615	6273	7148
MPI 4	3087	6786	10879	12408	13813
MPI 8	4133	11200	20375	23639	27312
MPI 16	2640	10007	23897	38204	46729

に計算サイズが大きくなるにしたがって性能が高くなることがわかる．倍精度と単精度の値はほぼ同じである．また，MPI プログラムの小さなサイズにおいては，並列度をあげると性能の低下が見られた．

3.5 Hewlett Packard hpserver zx2000

PC クラスタとして，hpserver zx2000 における性能を測定した．仕様は以下の通りである．

プロセッサ	Itanium2 900MHz; 12CPU
主記憶容量	512MB/CPU
1 次キャッシュ	32KB
2 次キャッシュ	256KB
3 次キャッシュ	1.5MB
ネットワーク	Myricom Myrinet2000 M3F-PCI64B (SRAM 2MB) GM1.6.4; MPICH-GM 1.2.5..10
OS	Red Hat Linux Advanced Workstation 2.1
コンパイラ	Intel Compiler version 7.1
最適化オプション	`-O3`

表 14，表 15 に単精度，倍精度の結果をそれぞれ示す．“—” は記憶容量不足のため実行できなかったこ

表 14: zx2000: 単精度

	XS	S	M	L	XL
逐次	1021	1037	931	—	—
MPI 1	1028	1054	1367	—	—
MPI 2	1887	1739	1831	14	—
MPI 4	3204	3301	3417	65	—
MPI 8	5602	6372	6974	8782	—

表 15: zx2000: 倍精度

	XS	S	M	L	XL
逐次	684	715	1.9	—	—
MPI 1	692	728	41	—	—
MPI 2	823	1244	1284	—	—
MPI 4	1636	2331	2531	13	—
MPI 8	3516	4329	5213	66	—

とを意味する．単精度の L サイズ，倍精度の M, L サイズの一部で極端な性能低下が見られる理由は，一部の記憶領域をディスクで代替したためと考えられる．MPI プログラムでは各 CPU が記憶領域を分割して保持するため，例えば L サイズの単精度プログラムでは 8 並列において実行可能となる．

3.6 FUJITSU FMV-W600

比較のため，現時点において広く普及しているパーソナルコンピュータのひとつである Intel Pentium プロセッサ搭載の計算機を用いて性能を測定した．仕様は以下の通りである．

プロセッサ	Pentium4 3.06GHz
主記憶容量	1GB
1 次キャッシュ	8KB
2 次キャッシュ	512KB
OS	Microsoft Windows XP Professional Version 2002
コンパイラ	Fujitsu Fortran Compiler Driver Version 3.0.10.1
最適化オプション	`/Kfast,prefetch`

表 16 に測定結果を示す．L, XL サイズは記憶容量不足のため実行できなかった．

表 16: FMV-W600

	XS	S	M	L	XL
単精度	285	263	237	—	—
倍精度	177	158	166	—	—

4 考察

スカラ計算機において倍精度の結果が単精度と比較して一律性能が劣化する理由は，単純にひとつの配列要素に必要なビット数が倍になるため，キャッシュに納まることのできる要素数が少なくなるためだと考えられる．表 17 に L サイズ (zx2000 は M サイズ) の MPI 8 並列で測定した倍精度の単精度に対する性能比を示す．現在の大規模数値計算のほとんどが倍

表 17: 単精度と倍精度の性能差 (L サイズ, MPI 8 並列)

	GP7000F	VPP5000	PRIMEPOWER 850	SX-7	zx2000
比率	62%	86%	56%	100%	74%

精度で行なわれていることを考慮すれば，倍精度版 HimenoBMT の提供および測定結果の公開が強く望まれる．

GP7000F, PRIMEPOWER 850 の結果から，HimenoBMT は自動並列化オプションの指定だけで OpenMP, MPI 版と同等の性能が得られることがわかった．したがって，このようなプログラムには共有メモリ型の計算機が最適であると思われる．

ベクトル計算機では，サイズが大きくなるほどベクトル長の長い計算が可能となり，性能が向上することが確認できた．逆に XS サイズの計算では Itanium2 プロセッサがベクトル計算機の半分以上の性能を達成しており，小さな計算サイズにおけるベクトル計算機の利点は少なくなりつつある．

スカラ計算機では，キャッシュを活用することにより性能が大きく変化する．今回の性能測定においても，コンパイラ，ライブラリのバージョンによって倍近い数字の差が生じることもあった．今回の数値実験を通して，計算機性能においては，コンパイラと計算機構造とプログラミングの 3 つの技術が複雑に絡んでいることを再認識した．したがって，汎用的と思われる高速な数値計算アルゴリズムを提案する際には，計算サイズ，並列化を行なった場合には並列度との関係を複数の計算機環境で試み，それらの結果を開示することにより，より提案手法の信頼性が増すのではと思われる．

参考文献

[1] `http://w3cic.riken.go.jp/HPC/HimenoBMT/`