2.2. 基礎性能

2.2.1. スカラおよびスレッド並列性能評価

富士通株式会社　　青木 正樹

ここでは、スカラ（1CPU コア）性能およびスレッド並列性能の評価結果について報告する。

1. 性能評価のポイント

スカラおよびスレッド並列性能の評価分析のポイントは、以下の 4 点。

・コンパイラの最適化/並列化の能力
・CPU の演算器性能
・メモリアクセス性能（バンド幅、レイテンシ）
・スレッド並列時のオーバヘッド

各々の性能要件は、アプリコードがどのようにコーディングされているかにより異なるが、定性的に上記観点を評価分析すれば、おおよその性能評価/分析は可能である。

2. 評価対象 CPU および評価対象コード

今回の評価に用いた CPU を以下に示す。

大分類	分類
SPARC 系	SPARC64V　1 チップ/ 1 コア 1.3GHz
	SPARC64Ⅵ　1 チップ/2 コア 2.28GHz
	SPARC64Ⅶ　1 チップ/4 コア　2.5GHz
IPF	Montecito　1 チップ/2 コア　1.6GHz
X86 系	Xeon/Woodcrest　1 チップ/2 コア　3.0GHz
	Xeon/Clovertown　1 チップ/4 コア　2.6GHz
	Opteron/Barcelona　1 チップ/4 コア　2.3GHz
VPP	VPP5000　1PE　9.6GFLOPS

今回の評価に用いたコンパイラを以下に示す。

・ 富士通製：Parallelnavi Language Package for Linux および Soraris V3 系
・ 他社コンパイラ：Intel コンパイラ 10 系、SUN Studio 11

今回の評価に用いたコードを以下に示す。

コード	目的	備考
Netlib/vectord	コンパイラの自動並列化能力	公開
実コード 254 本（高ｺｽﾄﾙｰﾌﾟ）	コンパイラの自動並列化能力	非公開
STREAM Benchmark	メモリ性能（バンド幅）	公開
LMBench	メモリ性能（レイテンシ）	公開
EuroBen Benchmark	カーネルループの演算性能	公開
OpenMP Micro Benchmarks	OpenMP オーバーヘッド時間	公開
姫野ベンチ	アプリコード評価	公開

各コードの詳細は以下のとおり。

【Netlib/vector】

・ http://netlib.org/benchmark/vectord
・ 本来は、20 年近く前に自動ベクトル化コンパイラの評価用に作成されたコード。
・ しかし、ベクトル化と類似の自動並列化の評価にも利用できると判断。
・ 全 135 ループからなる。

【実コード 254 本（高コストループ）】
・　富士通のコンパイラ開発部隊が保有する実コード群。
・　通常は、出荷前の性能試験に利用。
・　ベクトル機の時代からの BMT コード＆著名 BMT コードから構成。
・　各コードの実行コスト 90%以上のルーチンを抜き出し、翻訳テストにも利用。総ループ 7000 以上。

【STREAM Benchmark】
・　http://www.cs.virginia.edu/stream/
・　HPC のアプリにとり、メモリバンド幅は最重要。STREAM で公開されているのは総バンド幅
・　Web には、全ての値（真の実力）が公開されていない場合がある。

STREAM（公表データ一例）

システム	CPU	CPUS	MB/s	1 コアあたりの MB/s	B/F 比
Fujitsu VPP5000	vector	1	37544.0	37544.0	3.91
IBM System p5 595	powoer5+ 2.3GHz	64	206243.0	3222.5	0.35
Fujitsu Enterprise M9000	SPARC64_VI 2.4GHz	128	227059.0	1773.9	0.18
Fujitsu PRIMEQUEST＊	Montecito 1.6GHz	64	82755.0	1293.0	0.20

＊は、未公開だが実測値。

【LMBench】（ここではメモリレイテンシに特化し説明）
・　http://www.bitmover.com/lmbench/
・　ポインターチェイン（次のアドレスは前のメモリアクセスが終らないと決まらない ture dependency の関係を利用）の単純コード。
・　1 プロセス/整数ロード実行。（一般的には一番近いメモリアクセスの時間）
・　L1 ⇒ L2 ⇒（L3⇒）メモリのレイテンシ変移をみることができる。

【Euro Ben Benchmark】
・　http://www.euroben.nl/
・　科学技術計算用コンピュータの単体 CPU 性能と MPI 並列性能を測定するためのベンチマークプログラム。
・　ベンチマーク手順の合理化と規格化の促進のために、オランダ Utrecht Univ.の High Performance Computing Group（HPCG）により Working Group が 1990 年に設立され、現在は V5.0 が公開されている。OpenMP・MPI 版もあり。
・　以下の 3 つの Module からなる。
　Module1: 素性能（演算性能，メモリ性能，関数性能）
　Module2: 基本数学アルゴリズム性能（行列積，連立方程式，固有値，FFT など）
　Module3: 微分方程式,ポアソン方程式など
・　新しいチップの測定結果が公開されている。

【OpenMP　MicroBenchmarks】
・　http://www2.epcc.ed.ac.uk/computing/research_activities/openmpbench/openmp_index.html
・　OpenMP でサポートされている各ディレクティブのオーバヘッド時間を計測するコード
　⇒スレッド並列時のオーバヘッド時間の素性能
　　（各ベンダの実現方法にもよるが、自動並列化のオーバヘッドと考えてもよい）

3. コンパイラ評価

Vector/実コードを用い、自動並列化能力を評価した。

・並列化有無の判断は、コンパイラ出力の MSG による。

・ Euro Ben Benchmark を用い、X86 用コンパイラの逐次実行性能を評価。

■評価まとめ

・富士通コンパイラはトップランナーでなくなった。（Vector 評価より）

・しかし、実コードでは高水準。ベクトル化技術継承。（実コード評価より）

・近い将来、自動並列化能力は、自動ベクトル化同等以上となる。（予測）

・X86 コンパイラの逐次実行性能は、富士通コンパイラ対 Intel に対して凸凹。

3.1 Vector コードによる自動並列化能力評価結果（まとめ）

	コンパイラ	並列ループ数	ベクトル化比
参考	ベクトル化(VPP)	85	0.98
推定	理想の並列化コンパイラ	87	1.00
1 位	SUN	72	0.83
2 位	富士通	67	0.77
3 位	INTEL	61	0.70
4 位	PGI	10	0.10

各オプション

富士通　-Kfast,parallel_strong -Qt -Et

Sun　-fast -parallel -loopinfo -reduction

Intel　-O3 -ipo -xW -parallel -par-threshold0 -par-report3 -vec-report5

PGI　-fastsse -Mconcur=innermost -Minfo

3.2 Vector コードによる自動並列化能力評価結果（ループ毎詳細抜粋）

	VPP	FJ	SUN	PGI	INTEL	理想(推定)
並列化個数	85	87	72	10	61	87
ループ番号						
S111	○	○	○	×	○	○
S112	○	×	×	×	×	×
S113	○	○	○	×	○	○
S114	○	○	×	○	×	○
S115	○	○	×	×	○	○
S116	○	×	○	×	×	○
S118	×	○	×	×	○	○
S119	○	○	×	×	○	○
S121	○	×	×	×	×	×
S122	○	○	○	×	×	○
S123	×	×	×	×	×	×
S124	×	×	×	×	○	×
S125	○	○	×	×	○	○
S126	×	×	○	×	×	○

※FJ は富士通を表す。以降同様。

【Vector コード：FJ のみ自動並列化】

最大/最小位置検索

```
S331
F  並列化可

     3950    1                DO 1 NL = 1,NTIMES
     3951    1                J  = -1
     3952    2  pp u           DO 10 I = 1,N
     3953    2   p u             IF(A(I) .LT. 0) J = I
     3954    2   p          10  CONTINUE
     3955    1                CHKSUM = DBLE(J)
     3956    1                CALL DUMMY(LD,N,A,B,C,D,E,AA,BB,CC,CHKSUM)
     3957    1             1   CONTINUE
```

【Vector コード：FJ 自動並列化不可】

複雑(？)な配列添え字

```
S174
F  並列化不可

     2273    1              DO 1 NL = 1,2*NTIMES
     2274    2  s u          DO 10 I= 1, N/2
     2275    2  s u            A(I) = A(I+N/2) + B(I)
     2276    2  s u      10   CONTINUE
     2277    1              CALL DUMMY(LD,N,A,B,C,D,E,AA,BB,CC,1.D0)
     2278    1          1    CONTINUE
```

【Vector コード：ベクトル化のみ可能】

ループからの飛び出し

```
S332
VPPのみ 並列化可

     3977    1                DO 1 NL = 1,NTIMES
     3978    1                INDEX = -1
     3979    1                VALUE = -1.D0
     3980    2  s u            DO 10 I = 1,N
     3981    3  s u              IF ( A(I) .GT. T ) THEN
     3982    3    u                INDEX = I
     3983    3    u                VALUE = A(I)
     3984    3    u                GOTO 20
     3985    3  s u              ENDIF
     3986    2  s u         10 CONTINUE
     3987    1              20 CONTINUE
     3988    1                CHKSUM = VALUE + DBLE(INDEX)
     3989    1                CALL DUMMY(LD,N,A,B,C,D,E,AA,BB,CC,CHKSUM)
     3990    1              1  CONTINUE
```

収集・拡散ループ

```
S342
VPPのみ 並列化可

     4039    1              DO 1 NL = 1,NTIMES
     4040    1              J = 0
     4041    2  s u          DO 10 I = 1,N
     4042    3  p u            IF(A(I) .GT. 0.D0)THEN
     4043    3  m u               J    = J + 1
     4044    3  s u              A(I) = B(J)
     4045    3  p u            ENDIF
     4046    2  p u       10  CONTINUE
     4047    1              CALL DUMMY(LD,N,A,B,C,D,E,AA,BB,CC,1.D0)
     4048    1           1   CONTINUE
```

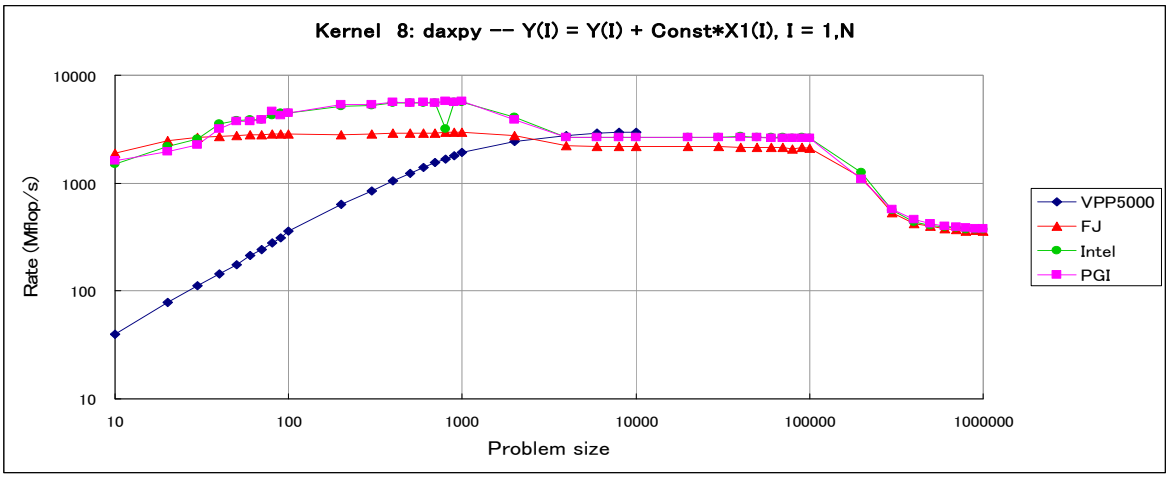
3.3 実コードによる自動並列化能力評価結果（まとめ）

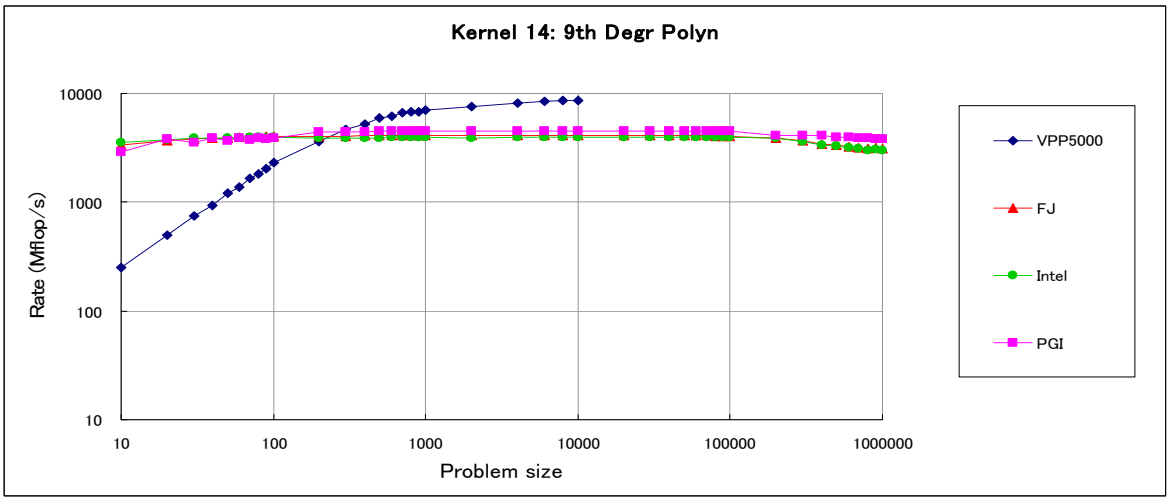
	コンパイラ	並列ループ数	ベクトル化比
参考	ベクトル化(VPP)	5590	1.00
1 位	富士通	5020	0.90
2 位	Intel	4643	0.83
3 位	SUN	3015	0.54
4 位	PGI	1130	0.20

富士通のコンパイラ開発評価に使用しているコード群であるため、富士通コンパイラ優位なコードである可能性十分あり。今後、詳細分析を行なう必要がある。

3.4 コンパイラ　スカラ性能比較

Euro Ben Benchmark を使用し、IA コンパイラベンダの各コンパイラのスカラ性能を評価した。
実行マシン：Woodcrest 3.0GHz





基本ループの最適化能力は、各ベンダほぼ同等。

4. CPU の演算性能

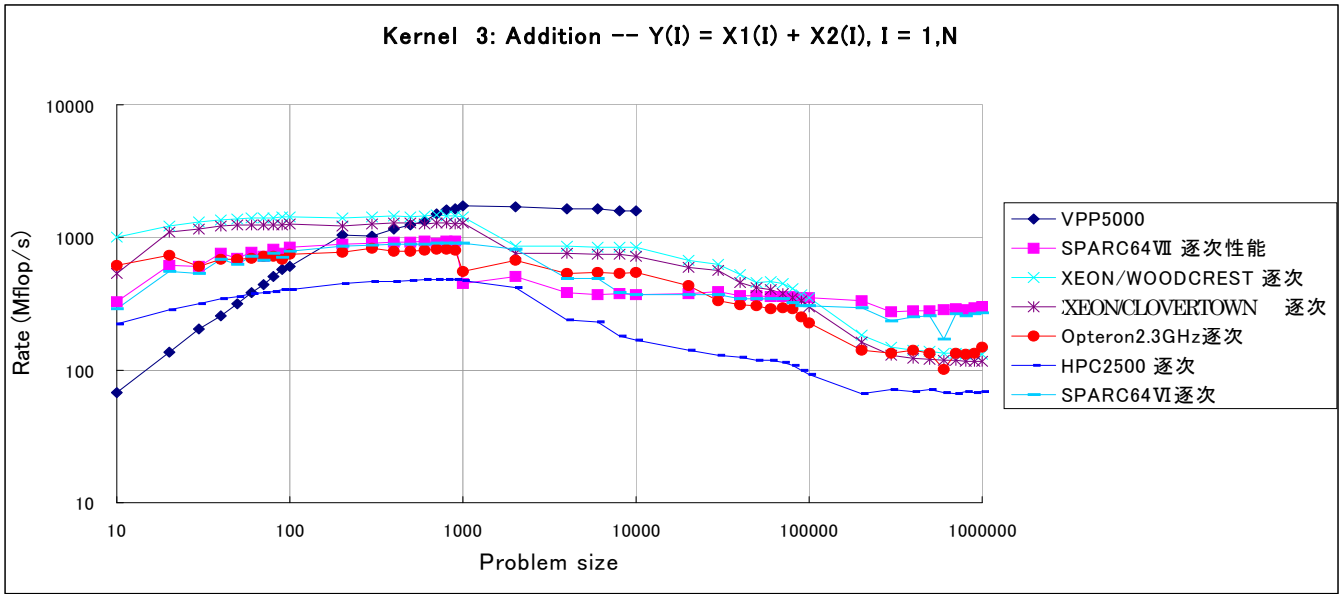
Euro Ben Benchmark を用い、各 CPU のスカラ（1CPU コア）の性能を実測。カーネルループは代表的な 4 種を評価。

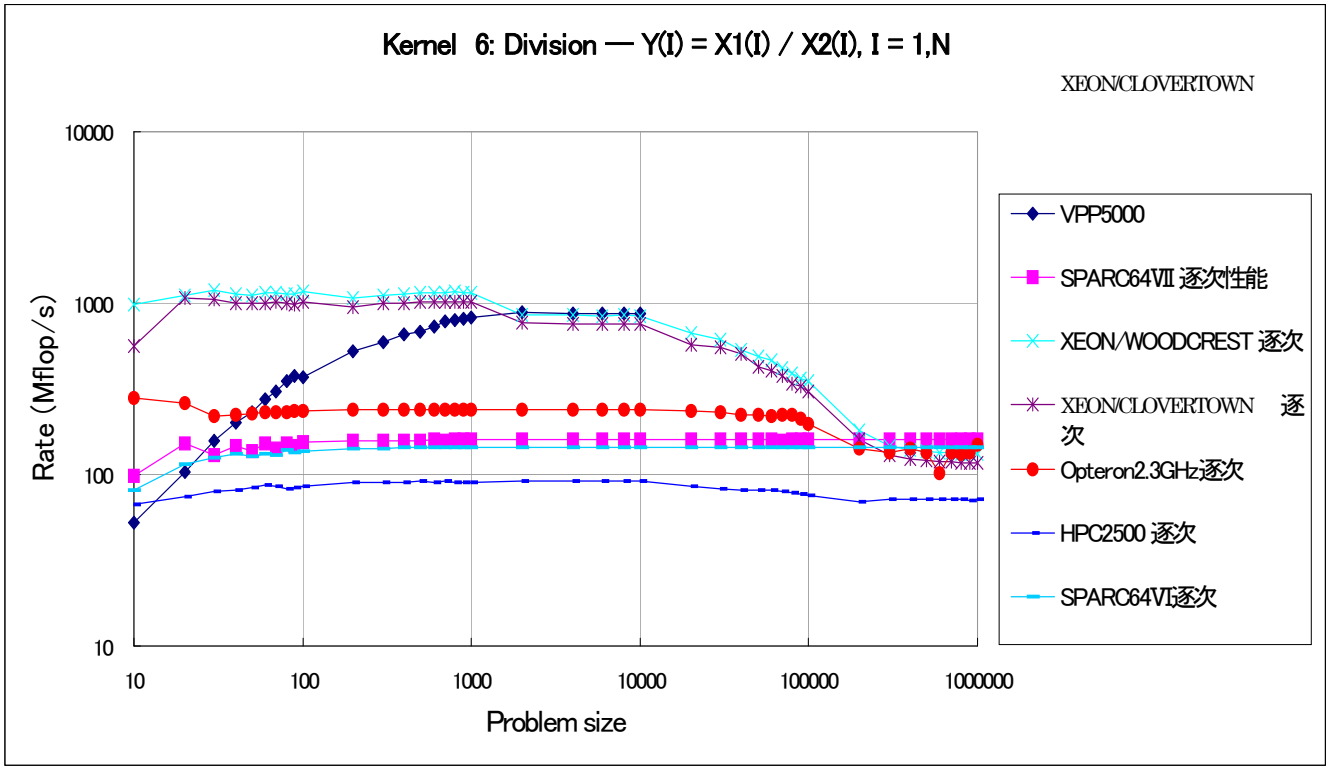
■評価まとめ

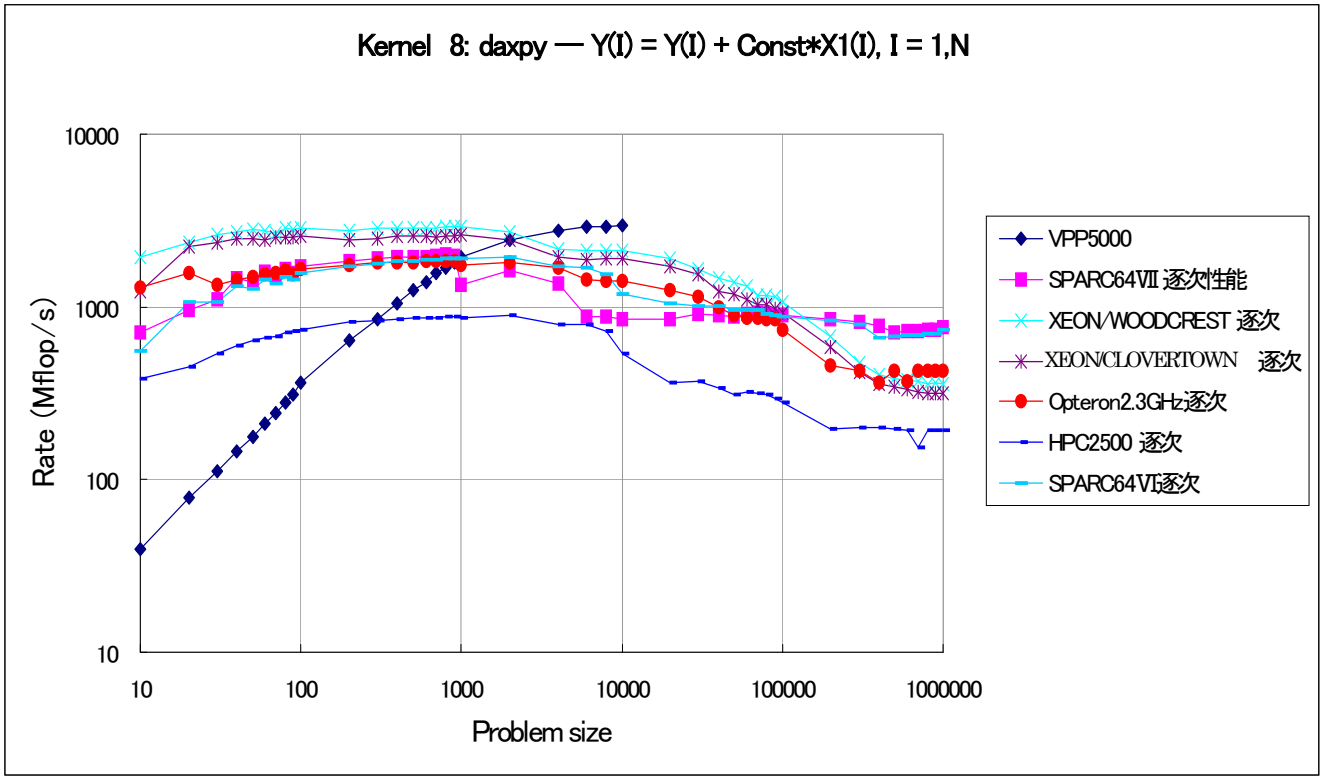
・データが L1 上にある場合、Xeon 系チップが速い。特に DIV は、同一値が連続した場合は特殊ルートで非常に高速。（Reg1=reg2/reg3---->reg4=reg2/reg1 の場合）

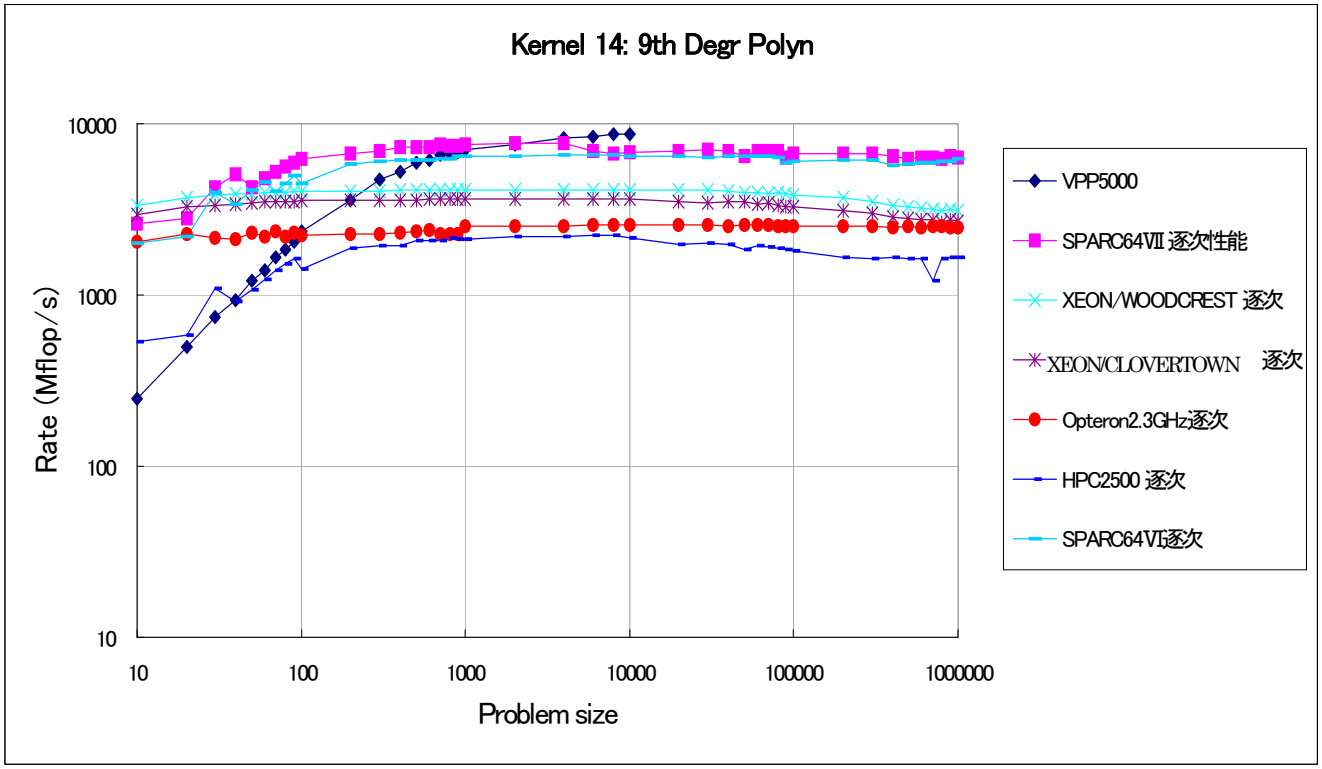
・SPARC(2.28GHz)は、演算が多いと速い。

・ピーク性能比：ベクトルはほぼ計算通り。スカラは低い。（効率悪い）






Kernel 8: daxpy — Y(I) = Y(I) + Const*X1(I), I = 1,N
10000
1000
100
10
Rate (Mflop/s)
10
100
1000
10000
100000
1000000
Problem size
VPP5000
SPARC64VII 逐次性能
XEON/WOODCREST 逐次
XEON/CLOVERTOWN 逐次
Opteron2.3GHz逐次
HPC2500 逐次
SPARC64VI逐次


Kernel 14: 9th Degr Polyn
10000
1000
100
10
Rate (Mflop/s)
10
100
1000
10000
100000
1000000
Problem size
VPP5000
SPARC64VII 逐次性能
XEON/WOODCREST 逐次
XEON/CLOVERTOWN 逐次
Opteron2.3GHz逐次
HPC2500 逐次
SPARC64VI逐次

5. メモリアクセス性能

STREAM によるバンド幅の評価、LMBench によるレイテンシ評価を行なった。

■評価まとめ

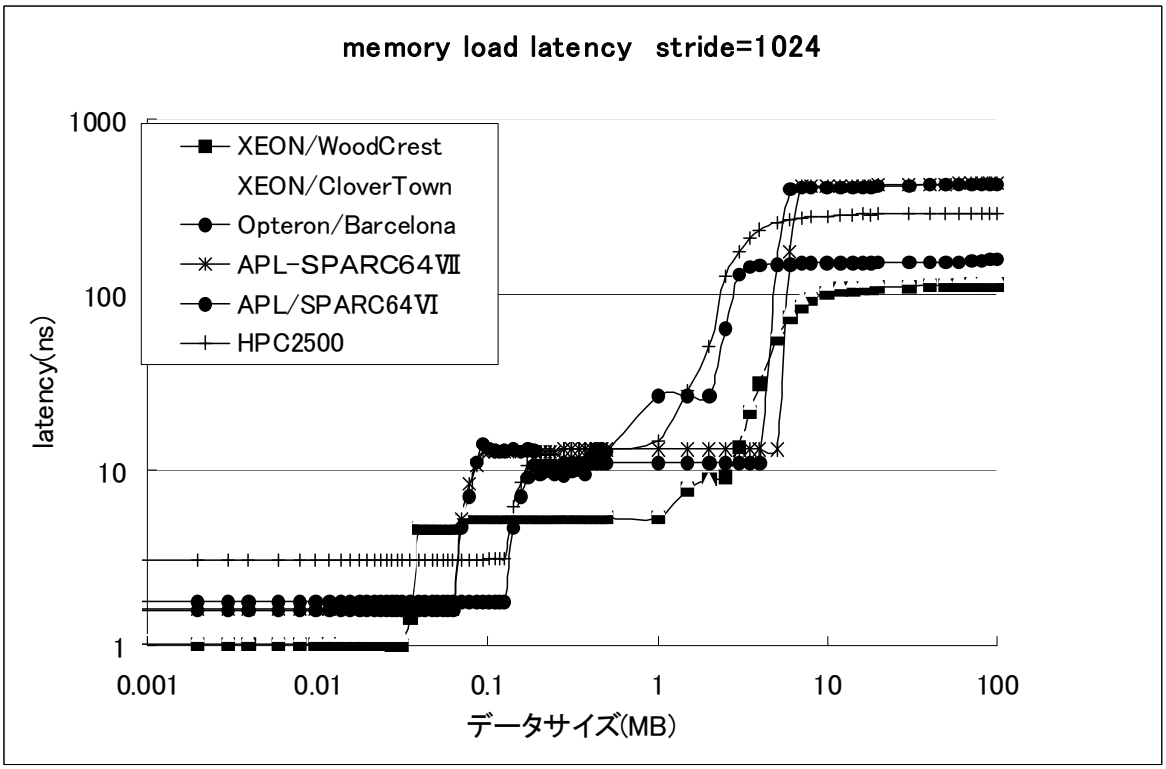
小規模 SMP ノードはレイテンシが良い。

【バンド幅】

システム	ノードあたりのチップ数	ノード全体のバンド幅	1 チップのバンド幅
VPP5000	1	37.5 GB/s	37.5 GB/s
IBM P575	8	86.1 GB/s	10.4 GB/s （注 1）
HPC2500	128	60 GB/s	1.7 GB/s
PQ580/IPF	32	77.1 GB/s	5.4 GB/s
APL/SPARC64VI	32	133.4 GB/s	7.6 GB/s
PG/Woodcrest	2	5.4 GB/s	3.3 GB/s
Opteron/Barcelona	16	16.0 GB/s	5.9 GB/s

(注 1)公表値。他は今回実測。

【レイテンシ】



6. スレッド並列のオーバヘッド（素性能）評価

OpenMP MICRO Bench を用い代表的な 4 つのディレクティブ（スレッド並列のオーバヘッド時間）を評価した。

■評価まとめ

・SPARC64 系 CPU は、ハードウェアバリア機構および共用キャッシュ機構の効果で高速。
・他は、メモリ(キャッシュ)レイテンシとキャッシュ機構に依存。

	SPARC64Ⅶ	SPARC64Ⅵ (2.28GHz)		Opteron (1.9GHz)		WoodCrest (3.00GHz)		CloverTown (2.66GHz)	HPC2500 (1.3 GHz)
	4 並列	2 並列	4 並列	4 並列 (FJ)	4 並列 (Intel)	2 並列	4 並列	4 並列	4 並列
PARALLELDO	0.30	3.90	9.61	1.93	1.89	0.72	2.15	2.20	2.05
DO	0.10	1.80	6.72	1.08	1.05	0.34	0.79	0.87	0.26
バリア	0.10	1.82	6.69	1.07	1.00	0.36	0.75	0.87	0.27
REDUCTION	0.50	4.16	17.46	3.29	3.35	1.20	3.17	3.35	4.43

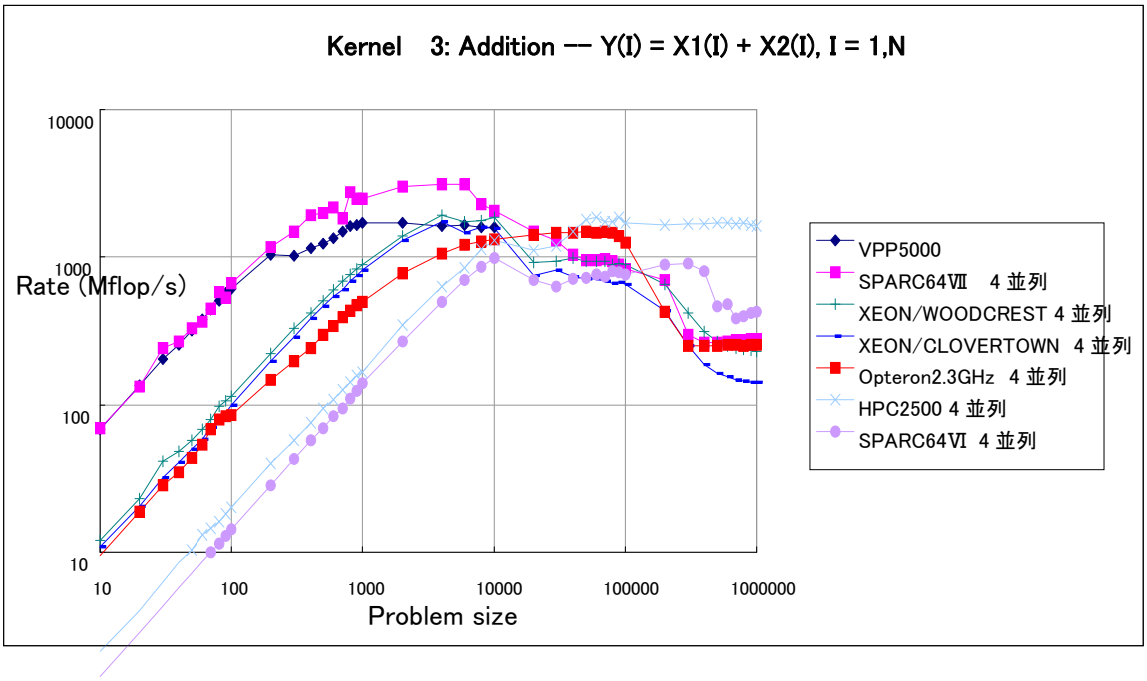
単位：μ秒

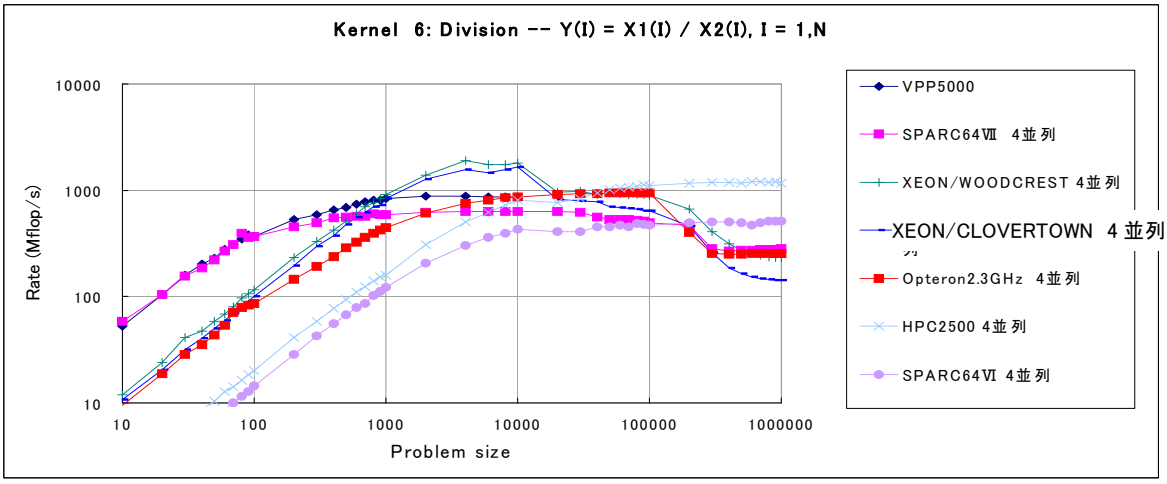
7. スレッド並列の性能評価

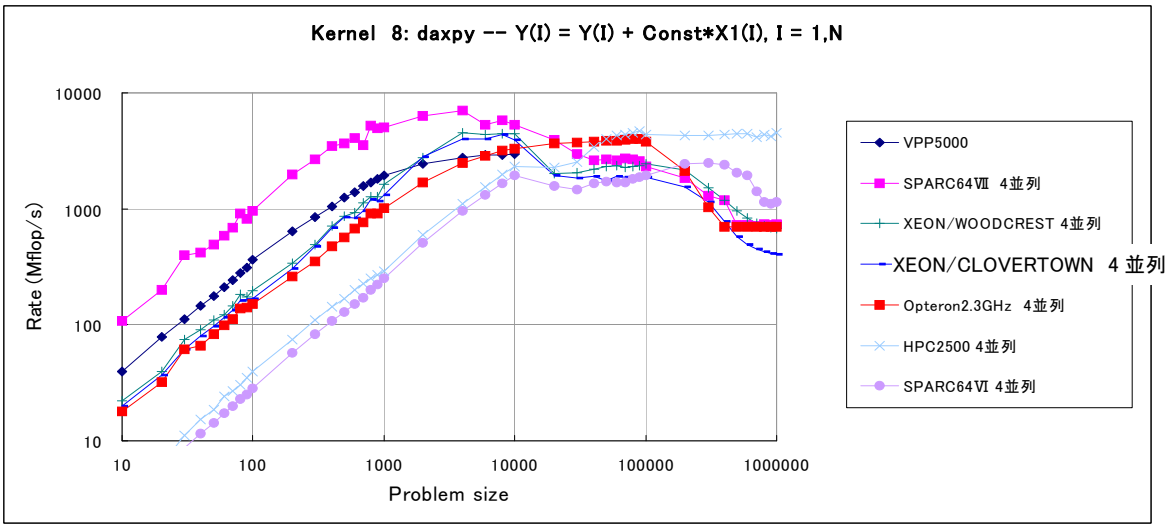
各 CPU の 4 スレッド実行時のカーネルループ実行時間評価。Euro Ben Benchmark を用い並列性能を実測。カーネルループは、代表的な 4 種を評価。

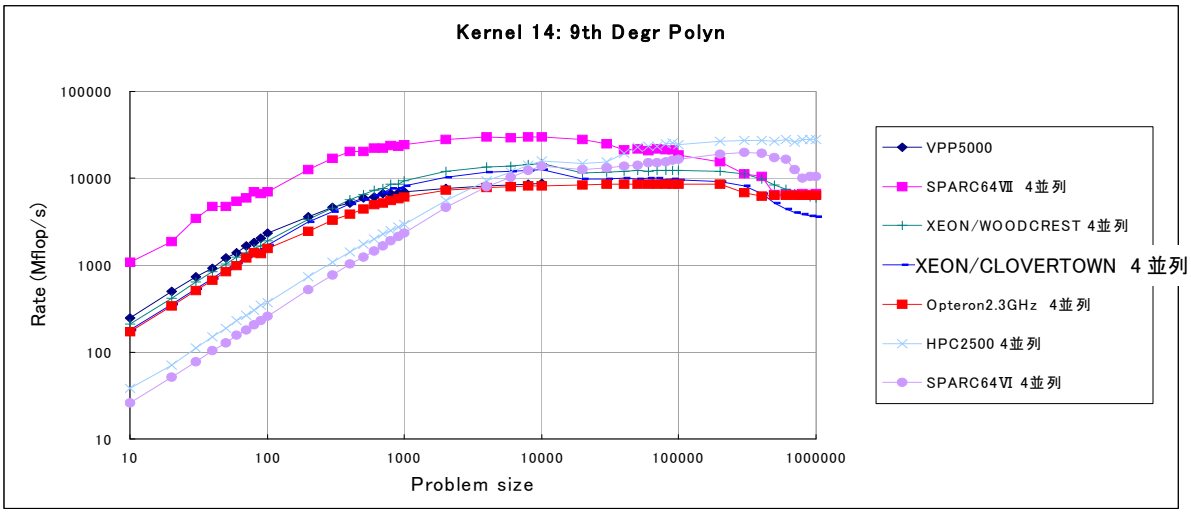
■評価まとめ

・キャッシュ機構で細粒度性能決まる。
・ベクトル並みの効率のためには、それなりのハードウェア機構必要。




Kernel 6: Division -- Y(I) = X1(I) / X2(I), I = 1,N
Rate (Mflop/s)
10000
1000
100
10
10
100
1000
10000
100000
1000000
Problem size
VPP5000
SPARC64Ⅶ 4並列
XEON/WOODCREST 4並列
XEON/CLOVERTOWN 4並列
Opteron2.3GHz 4並列
HPC2500 4並列
SPARC64Ⅵ 4並列


Kernel 8: daxpy -- Y(I) = Y(I) + Const*X1(I), I = 1,N
Rate (Mflop/s)
10000
1000
100
10
10
100
1000
10000
100000
1000000
Problem size
VPP5000
SPARC64Ⅶ 4並列
XEON/WOODCREST 4並列
XEON/CLOVERTOWN 4並列
Opteron2.3GHz 4並列
HPC2500 4並列
SPARC64Ⅵ 4並列


Kernel 14: 9th Degr Polyn
Rate (Mflop/s)
100000
10000
1000
100
10
10
100
1000
10000
100000
1000000
Problem size
VPP5000
SPARC64Ⅶ 4並列
XEON/WOODCREST 4並列
XEON/CLOVERTOWN 4並列
Opteron2.3GHz 4並列
HPC2500 4並列
SPARC64Ⅵ 4並列

8. 姫野ベンチ評価

アプリコード評価として、姫野ベンチの評価分析を実施した。

【評価したモデル】

ソースコード（Fortran90 ＋ OMP）	
サイズ	要素数
XL	(1024 × 512 × 512)
L	(512 × 256 × 256)
M	(256 × 128 × 128)
S	(128 × 64 × 64)

【カーネルコード】

```
do k=2,kmax-1
  do j=2,jmax-1
    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)
      GOSA1=GOSA1+SS*SS
      wrk2(I,J,K)=p(I,J,K)+OMEGA *SS
    enddo
  enddo
enddo
```

【評価マシン諸元】

	SPARC Enterprise M9000	HPC2500	HX600	PRIMERGY RX200 S3	IPF
プロセッサ	SPARC64VII	SPARC64V	Barcelona	Woodcrest	Itanium2
周波数 (GHz)	2.5	1.3	2.3	3.0	1.6
L2 キャッシュ (MB)	6	2	0.5	4	0.25
L3 キャッシュ (MB)	-	-	2	-	9

8.1 単精度版の性能

・1CPU コア／コンパイラ毎の性能　　単位：MFLOPS

Grid size	M9000	HPC2500	HX600	PRIMERGY				IPF
				FJ	Intel	PGI	Pathscale	Intel
XS(64x32x32)	2096	513	761	2102	2312	1836	1937	2856
S(128x64x64)	2269	480	1414	1532	1386	1428	1508	1672
M(256x128x128)	2428	492	1094	1389	1321	1351	1349	1561
L(512x256x256)	2421	475	993	1260	1218	1234	1227	1515
XL(1024x512x512)	2294	472	607	-	-		-	-

・ ピーク性能比

	M9000	HPC2500	HX600	PRIMERGY	IPF
単精度演算 ピーク性能 (MFLOPS)	10000	5200	18400	24000	6400

Grid size	M9000	HPC2500	HX600	PRIMERGY				IPF
				FJ	Intel	PGI	Pathscale	Intel
XS(64x32x32)	21%	10%	4%	9%	10%	8%	8%	45%
S(128x64x64)	23%	9%	8%	6%	6%	6%	6%	26%
M(256x128x128)	24%	9%	6%	6%	6%	6%	6%	24%
L(512x256x256)	24%	9%	5%	5%	5%	5%	5%	24%
XL(1024x512x512)	23%	9%	3%	-	-		-	-

8.2 倍精度版の性能

・ 1CPU コア／コンパイラ毎の性能　　単位：MFLOPS

Grid size	M9000	HPC2500	HX600	PRIMERGY				IPF
				FJ	Intel	PGI	Pathscale	Intel
M(256x128x128)	1648	301	476	691	720	820	750	-

・ ピーク性能比

	M9000	HPC2500	HX600	PRIMERGY	IPF
倍精度演算 ピーク性能 (MFLOPS)	10000	5200	9200	12000	6400

Grid size	M9000	HPC2500	HX600	PRIMERGY				IPF
				FJ	Intel	PGI	Pathscale	Intel
M(256x128x128)	16%	6%	5%	6%	6%	7%	6%	-

8.3 多重実行性能

・ CPU チップ／コンパイラ毎の性能　　単位：MFLOPS

単精度実数型

Grid size	M9000	HPC2500	HX600	PRIMERGY				IPF	
				FJ	Intel	PGI	Pathscale	Intel	
M(256x128x128)	777	540	768	594	531	533	588	-	4 多重
M(256x128x128)	2391	549	1060	1389	1321	1351	1349	1561	1 多重

倍精度実数型

Grid size	M9000	HPC2500	HX600	PRIMERGY				IPF	
				FJ	Intel	PGI	Pathscale	Intel	
M(256x128x128)	378	295	165	282	282	326	285	-	4 多重
M(256x128x128)	1648	301	476	691	720	820	750	-	1 多重

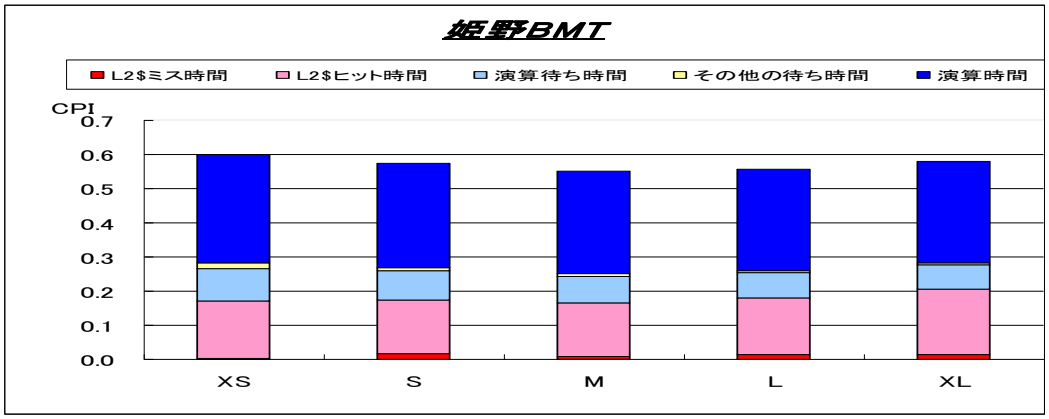
8.4 コード詳細分析

SPARC Enterprise M9000 を用いてカーネルループを詳細分析。
単精度版：1CPU コア実行

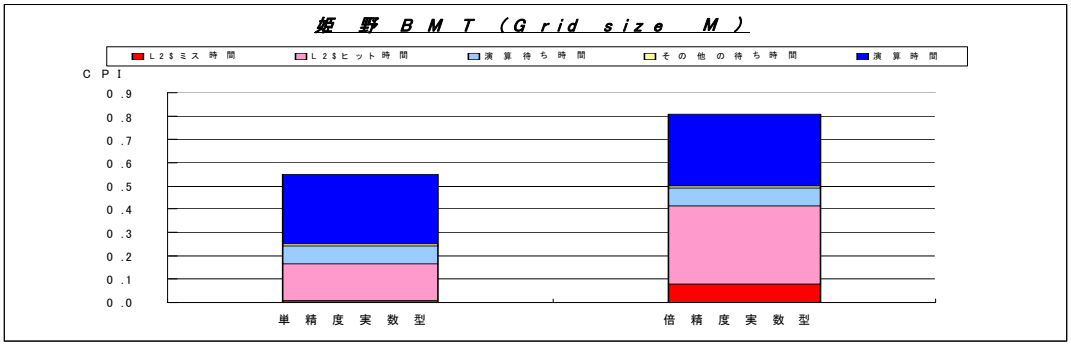
Grid size	MFLOPS	メモリ量(MB)	ﾛｰﾄﾞｽﾄｱ数(1 回あたり)	演算命令率	LD/ST 率	演算時間率	演算待ち時間率	ﾒﾓﾘｱｸｾｽ時間率			ﾒﾓﾘｽﾙｰﾌﾟｯﾄ(MB/s)
								L2$ﾋｯﾄ時間率	L2$ﾐｽ時間率	合計	
XS(64x32x32)	2096	3.5	32 (31+1)	50%	42%	53%	16%	28%	0%	29%	-
S(128x64x64)	2269	28				53%	15%	27%	3%	30%	3863
M(256x128x128)	2428	224				55%	14%	29%	2%	30%	3972
L(512x256x256)	2421	1792				53%	13%	30%	2%	32%	3883
XL(1024x512x512)	2294	14336				51%	12%	33%	2%	35%	3716

・メモリ量から、N 並列実行の場合、キャッシュに乗ってしまうため評価注意。
・メモリアクセス命令と演算命令の比率ほぼ同じ。＝アクセス性能と演算性能は両方重要。
・単精度版の場合、予想以上にメモリアクセス“のみ”がネックとなっている時間は多くない。
・2.5GFLOPS の性能のためには、バンド幅 4GB/s が必要。

PA イベント情報：各モデル毎 CPI グラフ



PA イベント情報：単精度版と倍精度版



CPI＝0.25 が MAX（1 マシンサイクルで 4 命令実行）

8.5 姫野ベンチ評価まとめ

・演算器とメモリアクセスのバランスのとれた CPU が速い

・2.5GFLOPS の性能出すためには、4GB/s のバンド幅が必要

＝　5GFLOPS ならば、8GB/s

・2.5GHz 程度のマシンサイクルならば、4GB/s でバランスしている（ﾒﾓﾘｱｸｾｽが見えない）
ただし、倍精度データになったら倍のバンド幅が必要

・レジスタ数の多い CPU が速い（IPF，SPARC）

・IPF：128 個、SPARC：32 個、X86-64：16 個

・レジスタの write-port 数は 4 個必要。HPC2500 の write-port 数は 2 個。

・キャッシュ容量は多い方が良い

・より少ない並列数でキャッシュに乗る

・IBM の POWER 系チップならば S モデルが L3 に乗る

・SPARC Enterprise M9000 の L1 が倍大きかったら、性能は 1.5 倍
⇒今後は、高速な L1 にいかに乗せるかのコンパイラ技術も必要

9. 自動並列化

「自動並列の性能」は懐疑的に思っている人も多い。一方、「ベクトル機の自動ベクトル化は性能が出やすい」というのが万人の認識。（ベクトル化は技術が確立している）

では、自動並列化で、なぜ性能でないのか？

①スレッド並列のオーバヘッド

バリア時間

フォルスシュアリング（キャッシュの仕組みにより異なる）

②解析能力（コンパイラの出来）

⇒理想的には①、②を解決すれば、自動ベクトル化並みの自動並列化が可能となる。

9.1 今後の HPC ハードウェアのトレンド

・　単体性能向上の行き詰まり

シリコンの集積度向上を命令レベルの並列処理に投入する効果減少。消費電力の制約が大きく、クロック向上は困難、キャッシュ容量の増大によりさらに電力増。

解決手段　⇒　チップ内に<u>複数の CPU コア</u>

・　今後は、並列システムによる高性能実現

CPU コア数で数千、数万以上のシステム。MPI で単純にプロセスを増すだけでは、性能向上は困難。

解決手段　⇒　<u>チップ内スレッド並列でプロセス並列度を抑える</u>

9.2 理想のスレッド並列化（①、②解決）　ベクトル化との比較

$$性能向上率＝\frac{1}{(1-V)+V/\alpha}$$

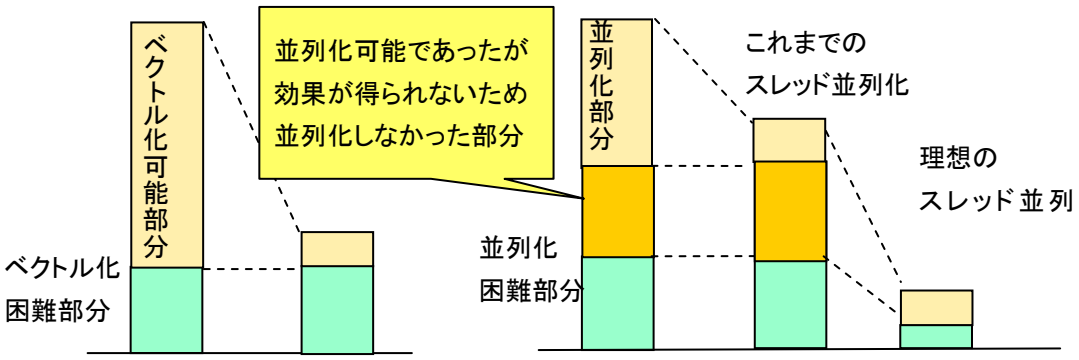
V：ベクトル化率

α：ベクトル化した場合の性能向上

$$性能向上率＝\frac{1}{(1-P)+P/n}$$

P：並列化率

n：並列化した場合の性能向上



9.3 チップ内（マルチコア）自動並列化で効果をだすためには

以下の要件を満足するシステムでなければならない。

・ 並列化率を高くできるか？

➔これまで、外側ループで並列化を行ってきた（疎粒度）。ベクトル化と同様に内側ループを並列化（細粒度）できれば、ベクトル並みに高い自動並列化率が可能。

・ 並列処理のオーバヘッドが十分小さくできるか？

➔内側ループを並列化すると、バリア同期の回数が非常に増加する。バリア同期の高速化が必須。また、CPU コアに接続しているキャッシュ間のデータ転送オーバヘッド削減が必要（共用キャッシュなら可能）。そして、それぞれのコアに十分なメモリバンド幅が必要。（メモリバンド幅ネックでは並列処理も意味なし）

9.4 共用キャッシュの評価

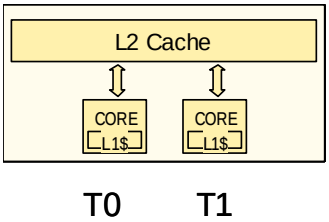
2 スレッド並列で共用キャッシュの効果を実測した。

【実測環境】

SPARC64VI（2.4GHz）

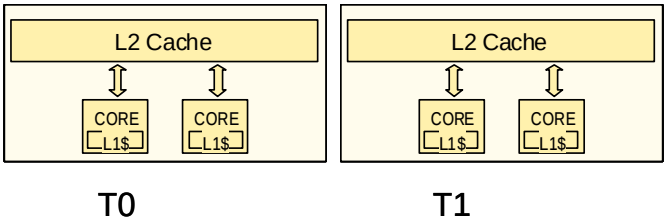
2 コア／1 チップ、L2：6MB　1 チップ内の 2 コアは L2 共用

①共用キャッシュでの実行

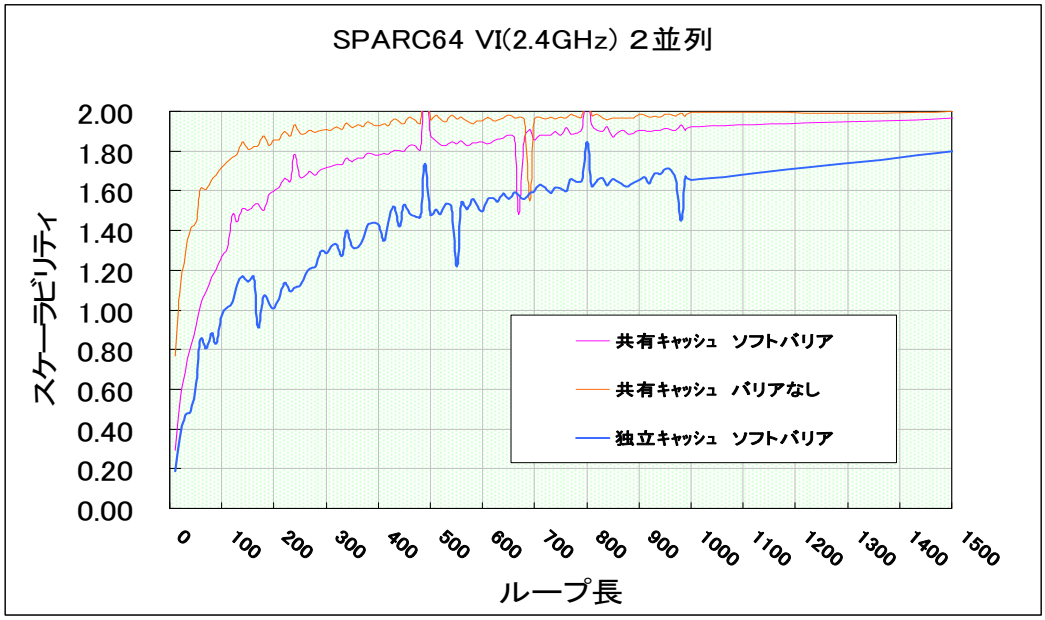


②独立キャッシュでの実行

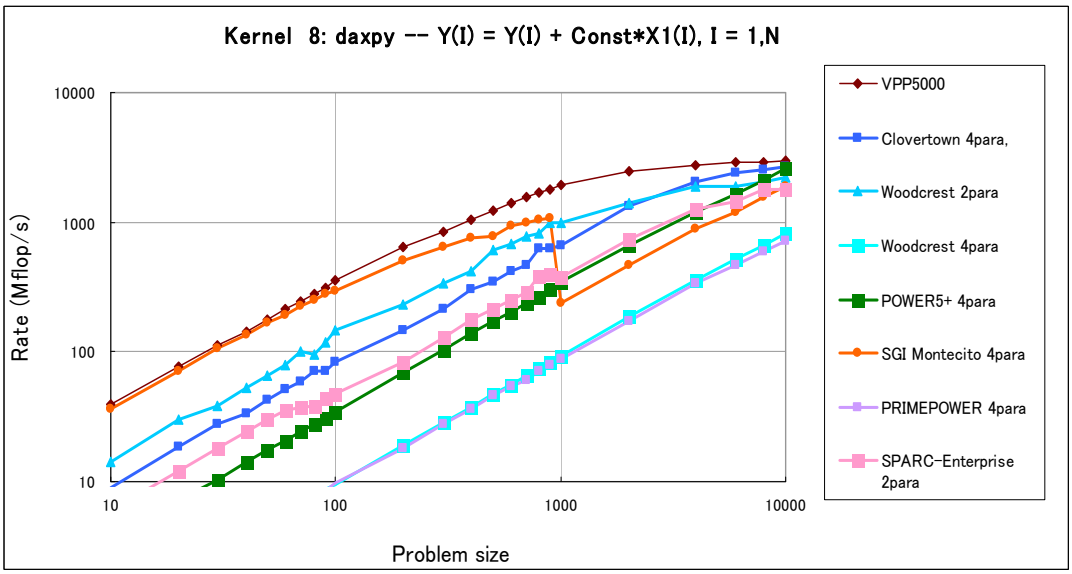
（総容量では 2 倍のキャッシュ）



【実測結果】評価コード：DAXPY



＜各種 CPU での評価＞



■評価果まとめ

スレッド並列時、コア間共用キャッシュは以下の効果がある。

①異なるスレッド（コア）間で同一キャシュラインの書き込み時に発生する false sharing の軽減

⇒細粒度のスレッド並列時、効果大

②あるスレッド（コア 1）でメモリからキャッシュに書き込んだデータが他のスレッド（コア 2）でキャッシュ上のデータとして再利用可能。

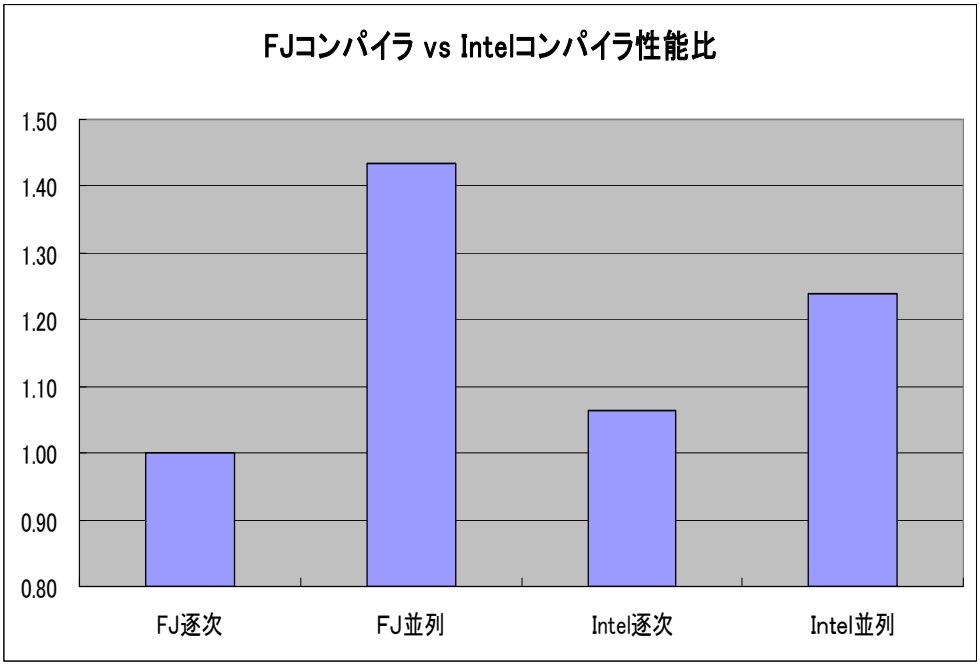
⇒ランダムアクセス時

```
do i=1,n
   a(i)=a(i)*b(l(i))
enddo
```

9.5　IA サーバー上でのコンパイラ自動並列化 比較評価

富士通製と Intel 製の 2 つのコンパイラでコンパイラの自動並列化効率を評価した。
評価の結果、コードにより凸凹あるが、逐次では Intel 製優位。並列では富士通製優位であった。

・マシン
　PRIMERGY RX200 S2、　CPU：Woodcrest（2 コア/1 チップ×2）、　OS：RedHat ES4.0
・コンパイラ
　富士通コンパイラ　options: 逐次 -Kfast –static –Kcmodel=small
　　並列 -Kfast,parallel -static -Kcmodel=small
　Intel　Fortran Compiler 10.0.017　options: 逐次 –xP –O3 –no-prec-div -static
　　並列 –xP –O3 –no-prec-div -static -parallel
・評価プログラム
　実コードのうち、いずれかの評価コンパイラのスケーラビリティが 1.2 倍以上となる厳選 27 本。



並列：4 スレッド並列

以上