Vol.2014-ARC-213 No.30
Vol.2014-HPC-147 No.30
2014/12/10

キャッシュの効果を加えたルーフラインモデルの拡張による プログラムの性能見積り

南一生[†1]　井上俊介[†2]　千葉修一[†3]　横川三津夫[†4]

プログラムの実行性能限界を見積もるために，プロセッサのピーク性能，メモリバンド幅， Operational Intensity（Flop/Byte）をパラメータとしたルーフラインモデルが提案されている．ルーフラインモデルは，メモリネックのプログラムの場合に見積り性能と実測性能が良く一致するが，キャッシュアクセスが増えてくると，見積り性能と実測性能が乖離してくる．本報告では，キャッシュアクセスが増大するカーネルプログラムに対し，コーディングに基づく実行性能の見積もり方法を提案する．また，いくつかのカーネルループに対し，スーパーコンピュータ「京」上の実行性能の評価を行った結果，本方法が実効性能見積もりに適用できることを明らかにした．

Performance Estimation of the Programs by the Extension of the RoofLine Model adding the Cache Effect

KAZUO MINAMI[†1]　SHUNSUKE INOUE[†2]
SHUICHI CHIBA[†3]　MITSUO YOKOKAWA[†4]

The Roofline models have been proposed in order to estimate the marginal performance of programs based on some features of computer systems such as peak performance, memory bandwidth, and operational intensity. The estimated performance by the model is in good agreement with the measured performance in the case that programs access memory devices directly. However, a difference between the estimated performance and the measured performance appears in the case that cache accesses of the program increase. In this paper, we extended the roofline model to a new one which can apply to a performance estimation of programs in which many cache accesses occur. It is shown that the new model can estimate the sustained performance of various kernel loops on the K computer by comparing with measured performance.

1. はじめに

計算機は，単体プロセッサの時代から，プログラムに内在する並列性をコンパイラで解釈することにより，高速処理を実現してきた．計算機が並列アーキテクチャへと変化してからは，数千から数万に及ぶプロセス間の並列性をプログラム上で明示して利用することにより超高速計算を実現してきている．

一方，プログラミングにおいては，高級言語とコンパイラが開発され，研究者やプログラマは，定式化・離散化に基づく式に忠実に，かつ物理現象に沿った素直なプログラミングを行なうことが一般的であった．しかし，現在では，計算機アーキテクチャの変化によるプログラミングの変化が生じている．演算器の計算能力が高くなる一方でメモリからのデータ供給能力が相対的に不足している問題（メモリーウォール問題）に対処するために，データ供給能力の高いキャッシュメモリを設け，キャッシュメモリに置いたデータを何回も再利用し演算を行なう方法が取られている．こうする事により演算器の能力を十分使い切る事ができる．このように「超並列性を引き出すプログラミング」と「プロセッサの単体の実行性能引き出すプログラミング」は，現代のスーパーコンピュータ，特に 8 万個余に及ぶプロセッサを備え，数々の数値計算のための新機能が導入されている「京」のようなスーパーコンピュータにおいては，ユーザ，研究者，プログラマが意識すべきプログラミング上の重要点である．プロセッサの単体の実行性能引き出すプログラミングのためのプロセッサ単体性能チューニングにおいては，キャッシュを有効利用するためのプログラミング上の様々なテクニックが提示されている[1][2]. このようなテクニックを駆使し，CPU 本来の性能を引き出し，シミュレーション時間を最小化することが，限られた計算資源の有効活用，また，研究期間の短縮に資するものであると考える．しかし，プロセッサ単体性能のチューニングを進める場合，対象とするプログラムの期待しうる性能値，つまり限界性能が容易に予測できないことが問題となっている．Samuel Williams らは，メモリバンド幅，CPU ピーク性能， Operational Intensity(Flop/Byte)を用いた性能見積り手法 : Roofline Model（ルーフライン）モデルを提案した[3]. 著者らも，「京」ではルーフラインモデルをベースに，プログラムのコーディングを精査することにより性能見積りを実施し，その見積りを元に CPU 単体性能チューニングを実施している[4]．しかし，ルーフラインモデルは，メモリバンド幅ネックの場合には見積り性能と実測性能が良く一致するが，キャッシュアクセスが増えてくると，見積り性能と実測性能が乖離してくることが分かってきた．

†1 独立行政法人 理化学研究所 計算科学研究機構 運用技術部門
†2 株式会社)富士通システムズ・イースト 解析シミュレーション部
†3 富士通株式会社 次世代テクニカルコンピューティング開発本部
†4 国立大学法人神戸大学　大学院システム情報学研究科

Vol.2014-ARC-213 No.30
Vol.2014-HPC-147 No.30
2014/12/10

本論文では,「京」上のプロセッサ単体性能の評価・チューニングにより明らかになった事例をベースに，メモリバンド幅ネックの状態からキャッシュアクセスが増大した状態まで適用可能な，プログラムのコーディングからその性能限界値を見積もるモデルを提唱し，そのモデルを「京」上で実験し確認する．

2. 「京」の CPU 概要

「京」での CPU 単体性能について議論するために「京」の CPU の概要[5]について述べる．一つの計算ノードは，一つの CPU（富士通製 SPARC64 TMVIIIfx），16GB のメモリ，計算ノード間のデータ転送を行うインターコネクト用 LSI（ICC: Inter-Connect Controller）で構成されている．CPU は，8 つのプロセッサコア，コア共有の 2 次キャッシュメモリ（6MB/12 way/ Write back 方式），メモリ制御ユニットを持っている. CPU の LSI の大きさは縦 22.7mm×横 22.6mm である．各コアは，L1 データキャッシュ(32KB/2way/ Write back 方式)，4 つの積和演算器，256 本の倍精度浮動小数点レジスタを持っており，一つの SIMD 命令により，2 つの積和演算器を同時に動作させることができる. 2 つの SIMD 命令を同時に実行することにより一つのコアはクロックサイクル毎に 8 個の浮動小数点演算ができる．従ってコアの理論性能は 16GFLOPS，CPU（8 コア）の理論性能は，単精度/倍精度とも 128GFLOPS となる．また，コア間の並列処理の同期を取るためのハードウェアバリア機構，計算に必要なデータを事前にキャッシュに取り込むプリフェッチ機構，プログラマブルなキャッシュ制御を可能とするセクタキャッシュ機構，など科学技術計算のための様々な機構を備えている．理論メモリバンド幅は 64GB/秒，B/F 値は 0.5 である. L2 キャッシュの理論バンド幅は 256GB/秒, B/F 値は 2.0 である．L1 キャッシュの理論バンド幅はコア毎に 64 GB / 秒，B/F 値は 4.0 である．またメモリ及び L2 キャッシュは 1 ライン(128byte)毎にアクセスされる．CPU の DGEMM 性能は 123.6GFLOPS（効率 96.6 %），コア間のハードウェアバリア性能は 49nsec であった．また，STREAM ベンチマークコードの triad によるメモリアクセス性能は，46.6GB/秒であった．

3. 「京」でのルーフラインモデルの検証

3.1 ルーフラインモデル

ルーフラインモデルは, Samuel Williams の論文[3]中のステンシルの例[6]や疎行列・ベクトル積の例[7]でも議論されているようにキャッシュからのロードは考慮しないメモリアクセスに則った見積もりとなっている．ルーフラインモデルにおいて，ハードウェアが持つ実効的なメモリバンド幅を B，ハードウェアの持つピーク性能を F，アプリケーションの演算強度を X=f/b（アプリケーションの要求フロップス値：f，アプリケーションの要求バイト値：b）とすると，アプリケーションの性能は min（F，B*X）で表される[3]．ここで，下式が成り立つ．

$$B*X=B*(f/b)=B*F*f/(b*F)=(B/F)/(b/f)*F \quad (1)$$

すなわち，B*X は，ハードウェアの持つ B/F 値をアプリケーションの要求 b/f 値で除した値にピーク性能を掛けた値とみなすことができる．また，両辺を F で除すればピーク性能比を求めることができる．本論文では，以下のように右辺の B,F,b,f 用いた方式で議論を展開する．

アプリケーションの性能　　：min(F, (B/F)/(b/f)*F)
アプリケーションのピーク性能比：min(1.0 , (B/F)/(b/f))

3.2 検証用テストプログラム

従来のルーフラインモデルは，3.1 に示したようにメモリ性能を基礎とした，性能の見積もり手法である．ここでは，メモリ性能のみならず，全ての配列が L2/L1 キャッシュに載った状態についても，ルーフラインモデルが適用可能か検証する．そのために，全ての配列がメモリに載った状態(on メモリ)， L2 キャッシュに載った状態(onL2)，L1 キャッシュに載った状態(onL1)の 3 種類のテストプログラムを作成し, それぞれ配列の量を変化させ性能を測定した．メモリ及び L2 キャッシュアクセス性能テストのプログラムを 1 に示す．このプログラムをベースとして，(右辺)*c1(i,j)+z のように項を追加し要求バイトの大きさを変えずに演算数を増やし，要求 b/f 値を変化させた．このような手法を採用したのは，4.2 節に後述するように，M&A simd 演算器の有効利用を図り，不要な性能低下要因を導入しないためである. j のループに対しブロック分割による 8 スレッド並列化を実施している．またプログラムでは，on メモリ，onL2 となるように，M，N の大きさを調整している．次に L1 キャッシュアクセス性能テストのプログラムを図 2 に示す．このループでは j の添字を取り除いているが，コンパイラの最適化の状態を L2 キャッシュのテストと同じ状態にして評価を実施するためである．このプログラムも onL1 となるように, M, N の大きさを調整している．また，本プログラムも j のループに対しブロック分割による 8 スレッド並列化を実施している．

```
do j = 1,N
  do i = 1,M
   c10(i,j) = z+c1(i,j)*x
  enddo
enddo

do j = 1,N
  do i = 1,M
   c10(i,j) = ((z+c1(i,j)*x)* &
                c1(i,j)+z)* &
                c1(i,j)+z
  enddo
enddo
```

図 1 メモリ・L2 キャッシュ基礎性能テストプログラム

```
      do j = 1,N
        do i = 1,M
         c10(i) = z+c1(i)*x
        enddo
      enddo
```

図 2 L1 キャッシュ基礎性能テストプログラム

3.3 ルーフラインモデルのテスト結果

メモリ性能テストの結果より，メモリの実効バンド幅は 46.3GB/sec と測定された．ルーフラインモデルによる見積もり値は，実測値と良く一致していることが確認されたが．従来のメモリ性能を基礎としたルーフラインモデルの検証であるため，ここに結果の図等は掲載しない．実効 B/F 値は，実効メモリバンド幅/理論メモリバンド幅×理論メモリ B/F 値で求められ，実効 B/F 値=46.3/64*0.5=0.36 となる．次に，L2 キャッシュ性能テストの結果では，L2 の実効バンド幅のピークは 159GB/sec と測定された．また，L1 キャッシュ性能テストの結果では，L1 の実効バンド幅のピークは 304GB/sec と測定された．L2/L1 の実効 B/F 値もメモリと同様に計算され，L2 の実効 B/F 値は 1.24 となり，L1 の実効 B/F 値は 2.38 となる．したがって，L2 においては，B/F 値 =1.24 以上，L1 においては，B/F 値=2.38 以上では見積もり値と合致しているはずである．また，L2 においては，B/F 値=1.24 以下，L1 においては，B/F 値=2.38 以下では性能が 100%以上に見積もられるので 1 を上限とする．この考えでルーフラインモデルにより性能を見積もったグラフを図 3 に示す．図 3 の横軸は B/F 値を縦軸はピーク性能比を表す．

ルーフラインモデルによる見積もり値は，実測値と良く一致しているといえるが，ピーク性能比が 100%に近い所は多少の差異がでている．

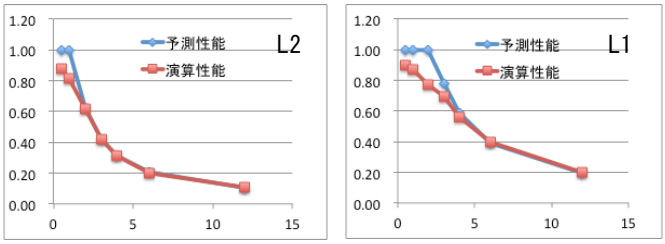

図 3 L2/L1 キャッシュ基礎性能テストの結果

3.4 ルーフラインモデルの適用可能ケースと適用限界

著者等の研究において，「京」上のルーフラインモデルによる性能見積もりとチューニングの結果について Seism3D[8][9]と FFB[10]を例に報告されている[4]．ここに示された事例のように，メモリバンド幅に性能が支配されるような場合には，従来のルーフラインモデルで性能限界を良く見積もることができる．しかし図 4 に示すプログラムのように，メモリバンド幅のアクセスに比してキャッシュへのアクセスが増大してくるとルーフラインモデルでの見積もり値と実測値の乖離が大きくなってくる．

図 4 の例では，on メモリの配列要素は 5 であり，cdiv の配列が onL2 となるため，メモリアクセス対 L2 キャッシュアクセスの比が 5：21 となり，メモリアクセスに比べ L2 キャッシュのアクセス比率が，増大している．図 4 の演算数は 43 であるため，ルーフラインモデルをそのまま適用すると，要求 B/F 値は 40/43＝0.093，ハードウェアの実効 B/F 値 0.36 を用いると見積もり性能は，0.36/0.93=0.39（39%）となる．この値は、実測ピーク性能比：14.7%とは大きな乖離がある．このような，メモリアクセスとキャッシュアクセスの混合状態での限界性能の見積もりが，従来のメモリ性能のみをベースとするルーフラインモデルでは難しいことが分かった．

```
do k=1,96
   do n=1,16769
      scl(n,k,l)=(                    &
           +cdiv(0,n,l,1)*vx(n    ,k,l) &
           +cdiv(1,n,l,1)*vx(n+1  ,k,l) &
           +cdiv(2,n,l,1)*vx(n+131,k,l) &
           +cdiv(3,n,l,1)*vx(n+130,k,l) &
           +cdiv(4,n,l,1)*vx(n-1  ,k,l) &
           +cdiv(5,n,l,1)*vx(n-131,k,l) &
           +cdiv(6,n,l,1)*vx(n-130,k,l) &
           +cdiv(0,n,l,2)*vy(n    ,k,l) &
                         :
           +cdiv(0,n,l,3)*vz( n   ,k,l) &
                         :
           )*fact
    enddo
enddo
```

図 4 Roofline モデルと実測性能の差が生じるプログラム例

4. メモリとキャッシュ混合状態での性能限界値見積もりモデル

4.1 限界性能見積りモデル

3 章では，メモリベースでのルーフラインモデルの成功パターンと限界について述べた．また，L2,L1 キャッシュの両方について，それぞれのキャッシュアクセスのみの場合には，ルーフラインモデルが適用可能であることを示した．ここでは，メモリとキャッシュアクセスが混合している場合について，ルーフラインモデルを拡張し限界性能の見積もりモデルを提案する．

メモリと L2 キャッシュの混合性能モデルでは，データの移動時間を基礎としたモデル化を行う．このモデル化では，以下のパラメタを使用する．

(1)メモリについて

・メモリデータ転送量：Mdata

・実効的なメモリバンド幅：Mband,

・メモリに載っているデータの移動にかかる時間：Mtime

(2)L2 キャッシュについて

・L2 データ転送量：L2data

・実効的な L2 バンド幅：L2band

・L2 に載っているデータの移動にかかる時間：L2time

Mtime および L2time は，それぞれ以下の関係がある．

・Mtime=Mdata/Mband，L2time=L2data/L2band

メモリアクセスを一定に保ち, L2 アクセスを増大させた場合，当初のメモリデータ転送ネックの場合は，プログラムの実行時間は Mtime となると予測される，その後，L2 アクセスを増大させ L2 データ転送ネックとなると，プログラムの実行時間は L2time になるものと予測される. したがってプログラムの実行時間は，max(Mtime,L2time)で表されると予測される．このモデルでは，実行時間が Mtime から L2time へ変化する交差点が存在することになる. プログラムの演算量：Ca，演算ピーク性能：Ppeak とすると演算ピーク性能比：Cp は，以下のように計算される.

Cp=Ca/(max(Mtime,L2time)*Ppeak)

この式の中で max(Mtime,L2time)の項については, メモリデータ転送ネックの場合は，Mtime>L2time であるが，L2 の転送量が増えてくるとある時点で Mtime<L2time になると考えられる．ここで，性能評価対象のプログラムのメモリアクセスと L2 アクセスと L1 アクセスの比を考え，メモリアクセス:m，L2 アクセス：n，L1 アクセス：l，演算数：k の割合であるとして，そのようなプログラムを，mM-nL2-lL1-kF と呼ぶことにする.

ここで Mtime>L2time から Mtime<L2time に切り替わる交差点を求めてみる．この交差点は，Mtime=L2time なので，ループの回転数を Nloop とすると交差点の条件は以下のようになる.

Nloop*m/Mband=Nloop*(m+n)/L2band

n=((L2band/Mband)-1)*m

また 3.1 節に述べたように，アプリケーションの性能は，ハードウェアの持つ B/F 値をアプリケーションの要求 B/F 値で除した値にピーク性能を掛けた値となるため，配列要素が倍精度の場合に，Mtime>L2time の領域と，Mtime<L2time の領域では, それぞれ以下のように求められる.

(1) Mtime>L2time の領域

Cp=(Mband/Ppeak)/(m*8/k)

(2) Mtime<L2time の領域

Cp=(L2band/Ppeak)/((m+n)*8/k)

メモリと L1 キャッシュの混合した場合も，基本的にはメモリと L2 キャッシュの混合の場合と同様の考えで性能見積りモデルを考えることが出来る.

4.2 プロセッサ単体性能に影響を与える要因

著者等の先行研究において， (1)プリフェッチの有効利用，(2)ラインアクセスの有効利用，(3)キャッシュの有効利用，(4)効率の良い命令スケジューリング，(5)SIMD 演算器の有効利用，の５つが高いプロセッサ単体性能を得るための方法であることを示し，また，要求 B/F 値が高いアプリケーションについては，これらのうち(1)(2)(3)(4)の順番で性能要因としての影響が大きい事を示した[4]. 要求 B/F 値が低いアプリケーションについては，(2) (3) (4)(5)が性能要因としての影響が大きくなる.

このような観点から，5.1 節に示した性能見積もりモデルを，要求 B/F 値が高い場合から，低い場合に適応させるためには，限界性能の見積もり対象となるプログラムは，プログラムの要求 B/F 値が高いケース，低いケース，それぞれについて，上記の性能要因を満たしている場合に，見積もりに近い性能が得られるものと考える．見積もり性能まで性能が達しない場合は, 上記の要件を満たしていないか, なんらかの問題が発生していると考えられる．このような場合は，要件を満たすようにチューニングする，問題を取り除くためのチューニングを行う, 等の対処が必要となる.

5. メモリとキャッシュ混合状態での性能実験結果

5.1 メモリ・L2 キャッシュ・L1 キャッシュの混合性能テストプログラム

混合性能テストとして，配列がメモリと L2 キャッシュの両方に載った混合状態，メモリと L1 キャッシュに載った混合状態の 2 種類のテストプログラムを作成し，メモリとキャッシュに載る配列の量を変化させ性能を測定する.

メモリ・L2 キャッシュアクセス混合性能テストのプログラムを図 5 に示す．図 5 のプログラムはメモリのみ考慮した時の要求バイト：B=3×8=24，要求演算数：F=2，であり要求 B/F 値：B/F=24/2=12 となる．本プログラムは，k のループに対しブロック分割による 8 スレッド並列化を実施している．N1=4000，N2=60，N3=80 を設定し全体を 60 回実行して計測している．j の差分一つ分は 4000×8 バイト=32KB は離れているので各配列:C への差分アクセスは L2 アクセスとなる．図 5 の L2 に対する要求バイトは，2×8=16 となる．また演算数は 2 である．このプログラムを 3 つのメモリアクセス，2 つの L２アクセス，演算が 2 であるので，3M-2L2-2F と呼ぶ事とする．このプログラムをベースとして， 3.2 と同様に(右辺)* c(I,j+2,k)+c(I,j-2,k)のように項を追加し L2 のアクセスと演算数を増やしている．このプログラムを 3M-4L2-4F と呼ぶ．また(右辺)*c(I,j-2,k)+c(I,j-2,k)のように項を追加したプログラムを 3M-3L2-4F と呼ぶ. ここに示したテストプログラムの一般形を mM-nL2-kF と書く事とする.

メモリ・L1 キャッシュアクセス混合性能テストは 2 つのタイプのテストを実施している．まず simd-short のプログラムを図 6 に示す. short とは L1 のアクセスを, c(i+1,j,k), c(i+2,j,k)のように 1 つ単位で増減させているからである. この場合，i の差分一つ分は 8 バイトしか離れていないことになり，各配列:c への差分アクセスは L1 アクセスとなる．図 6 の L1 に対する要求バイトは，2×8=16 となる．また演算数は 2 である．このプログラムを３つのメモリアクセス, 2 つの L1 アクセス, 演算が 2 であるので, 3M-2L1short-２F と呼ぶ事とする．本プログラムのスレッド並列化の方

Vol.2014-ARC-213 No.30
Vol.2014-HPC-147 No.30
2014/12/10

法および N1,N2,N3 の値，全体の実行回数は，メモリ・L2 キャッシュアクセス混合性能テストプログラムと同じである．この 3M-2LIshort-２F のプログラムをベースとして，(右辺)* c(i-2,j,k)+c(i+2,j,k)のように項を追加し L1 のアクセスを追加し演算数を増やしたプログラムを 3M-４L1short-４F と呼ぶ．また(右辺)*c(i-2,j,k)+c(i-2,j,k)のように項を追加したプログラムを 3M-3L1short-４F と呼ぶ．このような項の追加により，L1 アクセスの増加と演算数の増加を調整している．次に simd-long のプログラム(図 7) について説明する．ベースのプログラムは，simd-short アクセスのプログラムと同じである．このプログラムをベースとして(右辺)*c(i-12,j,k)+c(i+12,j,k) のように,L1 アクセスを１２ 単位で増減させた項を追加し，L1 アクセスと演算数を増やしているので，simd-long と名前をつけている．プログラムの名前の付け方の一般形は，mM-lL1short-kF のようになり，simd-short のプログラムと同様である．

```
do k = 1,N3
  do j = 1,N2
    do i = 1,N1
       a(i,j,k) = c(i,j-1,k)+c(i,j,k)*c(i,j+1,k)
    enddo
  enddo
enddo
```

図 5 メモリ・L2 アクセス混合性能テストプログラム

```
do k = 1,N3
  do j = 1,N2
    do i = 1,N1
       a(i,j,k) = c(i-1,j,k)+c(i,j,k)*c(i+1,j,k)
    enddo
  enddo
enddo
```

図 6 メモリ・L1 アクセス混合性能テスト simd-short

```
do k = 1,N3
  do j = 1,N2
    do i = 1,N1
       a(i,j,k) =   &
         (c(i-1,j,k)+c(i,j,k)*c(i+1,j,k))* &
          c(i-12,j,k)+c(i+12,j,k)

    enddo
  enddo
enddo
```

図 7 メモリ・L1 アクセス混合性能テスト simd-long

5.2 メモリ・Ｌ２キャッシュ混合テスト結果

本節では 5.1 節のモデルに従い,メモリ・L2 キャッシュ混合テスト結果を整理する．メモリと L2 キャッシュの混合性能テストの結果を表 1 に示す．列毎の項目の定義は 5.1 節にならって以下のように計算した

Mdata[Byte]＝m*ループの回転数,
L２data[Byte]=(m+n)*ループの回転数,
Mband[Byte/sec]=Mdata/実行時間,
L2band[Byte/sec]= L2data/実行時間

表 1 より，Mband の最高値は 46GB/sec であり，この値を Mband_peak とする．また表 1 より L2band の最高値は 146GB/sec であり，この値を L2band_peak とする．この値は，3.3 節 の L2 基礎テストの章で得られた 159GB/sec より低めに出ているが，「京」では，メモリと L2 キャッシュについて同一のハードウェアでデータ転送を受け持つようになっており，互いのバンド幅が影響を受けるため，メモリアクセスをしない L2 基礎テストの値とは差が生じている．Mtime，L2time はそれぞれ,Mtime=Mdata/Mband_peak で計算し，L2time=L2data/L2band_peak で計算した．単位はそれぞれ sec である．Mtime>L2time から Mtime<L2time に切り替わる交差点の条件は，n=((L2band/Mband)-1)*m であるので，L2band を L2band_peak，Mband を Mband_peak の値を使用すると，交差点の条件は，n=2.17m である．この条件をテストプログラムの名称で表わすと ，3M-6L2 と 3M-8L2 の間となる．表 12 の B/F 値は，Mtime>L2time の場合は，8*m/k を示し，Mtime<L2time の場合は，8*(m+n)/k を示している．Ca は k*ループの回転数で計算し，Cp は，Mtime>L2time の領域では，Cp= (Mband/Ppeak)/(m*8/k)で，Mtime<L2time の領域では，Cp=(L2band/Ppeak)/((m+n)*8/k)で計算した．ただしピーク性能比 1.0 を超える事はないので Cp が 1.0 以上の値は 1.0 が上限としている．
Mtime>L2time の領域で B/F 値を横軸とした，性能の見積もり値と実測値の比較を図 8 に示す．また，Mtime<L2time の領域で B/F 値を横軸とした，性能の見積もり値と実測値の比較を 8 に示す．表 1，図 8 を見ると，ピーク性能比 1.0

	m	n	k	実測性能	実行時間	Mdata	L2data	Mband	L2band	Mtime	L2time	B/F値	Ca	Cp
3M-2L2-2F	3	2	2	0.029	0.63	2.765E+10	4.608E+10	4.389E+10	7.314E+10	0.60	0.32	12.00	2.304E+09	0.030
3M-3L2-4F	3	3	4	0.057	0.63	2.765E+10	5.530E+10	4.389E+10	8.777E+10	0.60	0.38	6.00	4.608E+09	0.060
3M-4L2-4F	3	4	4	0.060	0.60	2.765E+10	6.451E+10	4.608E+10	1.075E+11	0.60	0.44	6.00	4.608E+09	0.060
3M-5L2-6F	3	5	6	0.084	0.64	2.765E+10	7.373E+10	4.320E+10	1.152E+11	0.60	0.50	4.00	6.912E+09	0.090
3M-6L2-6F	3	6	6	0.082	0.66	2.765E+10	8.294E+10	4.189E+10	1.257E+11	0.60	0.57	4.00	6.912E+09	0.090
3M-6L-12F	3	6	12	0.166	0.65	2.765E+10	8.294E+10	4.254E+10	1.276E+11	0.60	0.57	2.00	1.382E+10	0.180
3M-6L2-24F	3	6	24	0.322	0.67	2.765E+10	8.294E+10	4.127E+10	1.238E+11	0.60	0.57	1.00	2.765E+10	0.359
3M-6L2-48F	3	6	48	0.645	0.67	2.765E+10	8.294E+10	4.127E+10	1.238E+11	0.60	0.57	0.50	5.530E+10	0.719
3M-8L2-8F	3	8	8	0.097	0.74	2.765E+10	1.014E+11	3.736E+10	1.370E+11	0.60	0.69	11.00	9.216E+09	0.104
3M-10L2-10F	3	10	10	0.106	0.85	2.765E+10	1.198E+11	3.253E+10	1.410E+11	0.60	0.82	10.40	1.152E+10	0.110
3M-12L2-12F	3	12	12	0.111	0.97	2.765E+10	1.382E+11	2.850E+10	1.425E+11	0.60	0.95	10.00	1.382E+10	0.114
3M-8L2-16F	3	8	16	0.197	0.73	2.765E+10	1.014E+11	3.787E+10	1.389E+11	0.60	0.69	5.50	1.843E+10	0.207
3M-10L2-20F	3	10	20	0.212	0.85	2.765E+10	1.198E+11	3.253E+10	1.410E+11	0.60	0.82	5.20	2.304E+10	0.219
3M-12L2-24F	3	12	24	0.227	0.95	2.765E+10	1.382E+11	2.910E+10	1.455E+11	0.60	0.95	5.00	2.765E+10	0.228
3M-14L2-28F	3	14	28	0.236	1.07	2.765E+10	1.567E+11	2.584E+10	1.464E+11	0.60	1.07	4.86	3.226E+10	0.235
3M-16L2-32F	3	16	32	0.169	1.70	2.765E+10	1.751E+11	1.626E+10	1.030E+11	0.60	1.20	4.75	3.686E+10	0.240
3M-18L2-36F	3	18	36	0.184	1.76	2.765E+10	1.935E+11	1.580E+10	1.106E+11	0.60	1.33	4.67	3.917E+10	0.231
3M-8L2-32F	3	8	32	0.389	0.74	2.765E+10	1.014E+11	3.736E+10	1.370E+11	0.60	0.69	2.75	3.686E+10	0.415
3M-8L2-64F	3	8	64	0.758	0.76	2.765E+10	1.014E+11	3.638E+10	1.334E+11	0.60	0.69	1.38	7.373E+10	0.830
3M-8L2-128F	3	8	##	0.860	1.34	2.765E+10	1.014E+11	2.063E+10	7.565E+10	0.60	0.69	0.69	1.475E+11	1.000

表1 メモリ・L2混合性能テストの結果

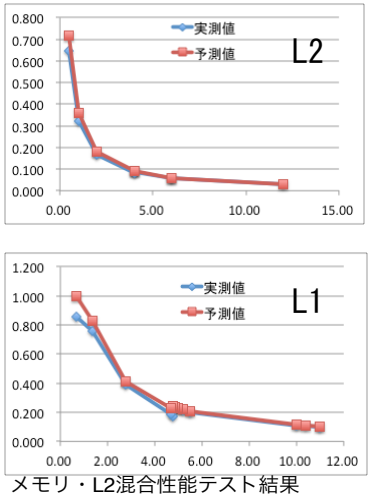

図８ メモリ・L2混合性能テスト結果

近辺と n＞5m の領域では誤差が大きくなる傾向があるが，その他の領域では，性能を十分見積もり可能であると判断できる．n>5m の領域で誤差が大きくなる理由は，L2 キャッシュ溢れである．L2 アクセスする配列が一つ分で 4000×8B=32KB であり，このテストでは m=3 であり n は 15 以上である．これだけで 32KB×15=480KB となり 1 コア分の L2 キャシュ量：750KB の 64%を占めることになる．L2 キャッシュは，メモリアクセスする配列分も必要であり，キャッシュ溢れの可能性が生じてくる．実際，測定データを見ると，3M-14L2-28F では L2 キャッシュミス率：0.80%，キャッシュミス数:1.48E+08 であるのに対し，3M-16L2-32F では L2 キャッシュミス率：1.51%，キャッシュミス数：3.64E+08 へと増大している．ピーク性能比 1.0 近辺での誤差については，ピーク性能に近い性能の達成が，DEGEMM 等の特別にチューニングされたプログラムに限られる事情を考慮すれば納得できる．

5.3 メモリ・L1 キャッシュ混合のテスト

本節では 4.1 節のモデルに従い,メモリ・L1 キャッシュ混合テスト結果を整理する．

ただし，メモリと L2 キャッシュの混合性能テストの場合は，Mdata や L2data の値をプログラムから正確に見積もることが出来るが，メモリと L1 キャッシュの混合性能テストの場合は，L1data の値を正確に見積もる事が難しくなる．L1 キャッシュのテストプログラムには，5.1 に示したように回転 i に対して C(i-1,j,k)，C(i,j,k)，C(i+1,j,k) の様な項が存在する．同様に回転 i+1 に対しては，C(i,j,k), C(i+1,j,k), C(i+2,j,k) の様な項が存在する．この回転 i と回転 i+1 は同じ項が存在するために，同じ項に対する回転 i+1 のアクセスは，L1 キャッシュへのアクセスでなくレジスタ上の値を使用できる可能性がある．このように，メモリと L1 キャッシュの混合テストの場合は，どの項が L1 アクセスとなり，どの項がレジスタアクセスとなるのか正確に見積もる事ができなくなる．したがって L1data の値を正確に見積もる事ができなくなり，L1time，Cp の値も見積りできなくなる．しかし，メモリデータ転送ネックの場合は，プログラムの実行時間は Mtime となり，L1 アクセスを増大させ L1 データ転送ネックとなると，プログラムの実行時間は L1time になる状況および Mtime と L1time の交差点が存在することは，メモリと L2 キャッシュの混合性能テストと同様と考えられる．

i+1,i+2 の様な項を使用する simd-short のプログラムについては，上記に述べたような事情で見積りが難しくなると考えられるが，i+12,i+24 のように離れた場所をアクセスする simd-long のデータについては，上記に述べた状況は比較的起こりにくいと考えられる．そこで，見積りには simd-long のデータを使用し，simd-short のデータについては，メモリアクセス一定の時に，どこまで項を増やすと性能に影響が出てくるかを実験的に求めるために使用する事とする．

メモリと L1 キャッシュの混合性能テストの結果のうち simd-short の結果を表 2 に示す．B/F 値はメモリアクセスのみの B/F 値を示している．予測性能は，メモリバンド幅ベースの見積り値を示している．また実行時間(sec)，メモリバンド幅(GB/sec),「京」のシステムで提供されているプロファイラにより実測された値である．性能の実測値(ピーク性能比)は，演算数を k×ループ回転数で計算し，実行時間で除した値である．表 2 を見ると simd-short の場合に，3M-32L1-32F まではメモリベースの見積りが可能な事を示

	m	n	k	B/F値	予測性能	実測性能	実行時間	メモリバンド幅
3M-2L1-2F	3	2	2	12.000	0.030	0.029	0.620	44.850
3M-4L1-4F	3	4	4	6.000	0.060	0.058	0.620	44.940
3M-6LI1-6F	3	6	6	4.000	0.090	0.089	0.610	45.320
3M-8L1-8F	3	8	8	3.000	0.120	0.118	0.610	45.460
3M-10L1-10F	3	10	10	2.400	0.150	0.148	0.610	45.150
3M-12L1-12F	3	12	12	2.000	0.180	0.174	0.620	44.900
3M-16L1-16F	3	16	16	1.500	0.240	0.232	0.620	44.450
3M-20LI1-20F	3	20	20	1.200	0.300	0.286	0.630	43.880
3M-24L1-I24F	3	24	24	1.000	0.360	0.343	0.630	44.130
3M-28L1-128F	3	28	28	0.857	0.420	0.400	0.630	43.870
3M-32L1-32F	3	32	32	0.750	0.480	0.457	0.630	44.030
3M-36L1-36F	3	36	36	0.667	0.540	0.506	0.640	43.550
3M-40L1-40F	3	40	40	0.600	0.600	0.545	0.660	42.270
3M-44L1-44F	3	44	44	0.545	0.660	0.535	0.740	37.370
3M-48L1-48F	3	48	48	0.500	0.720	0.508	0.850	32.760
3M-52L1-52F	3	52	52	0.462	0.780	0.442	1.060	26.220
3M-56L1-56F	3	56	56	0.429	0.840	0.485	1.040	26.740
3M-60L1-60F	3	60	60	0.400	0.900	0.540	1.000	27.590

表2 メモリ・L1混合性能テストの結果(simd-short)

	m	n	k	実測性能	実行時間	Mdata	L1data	Mband	L1band	Mtime	L1time	B/F値	Ca	Cp
3M-2L1-2F	3	2	2	0.029	0.620	2.765E+10	2.599E+10	4.459E+10	4.192E+10	0.60	0.11	12.00	2.304E+09	0.030
3M-4L1-4F	3	4	4	0.059	0.610	2.765E+10	4.332E+10	4.532E+10	7.101E+10	0.60	0.18	6.00	4.608E+09	0.060
3M-6L1-6F	3	6	6	0.089	0.610	2.765E+10	6.064E+10	4.532E+10	9.941E+10	0.60	0.25	4.00	6.912E+09	0.090
3M-8L1-8F	3	8	8	0.116	0.620	2.765E+10	7.797E+10	4.459E+10	1.258E+11	0.60	0.32	3.00	9.216E+09	0.120
3M-10L1-10F	3	10	10	0.148	0.610	2.765E+10	9.529E+10	4.532E+10	1.562E+11	0.60	0.40	2.40	1.152E+10	0.150
3M-12L1-12F	3	12	12	0.177	0.610	2.765E+10	1.126E+11	4.532E+10	1.846E+11	0.60	0.47	2.00	1.382E+10	0.180
3M-16L1-16F	3	16	16	0.236	0.610	2.765E+10	1.473E+11	4.532E+10	2.414E+11	0.60	0.61	9.50	1.843E+10	0.236
3M-18L1-18F	3	18	18	0.133	1.220	2.765E+10	1.646E+11	2.266E+10	1.349E+11	0.60	0.68	9.33	2.074E+10	0.237
3M-20L1-20F	3	20	20	0.149	1.210	2.765E+10	1.819E+11	2.285E+10	1.504E+11	0.60	0.75	9.20	2.304E+10	0.238
3M-24L1-24F	3	24	24	0.179	1.210	2.765E+10	2.166E+11	2.285E+10	1.790E+11	0.60	0.90	9.00	2.765E+10	0.240
3M-28L1-28F	3	28	28	0.210	1.200	2.765E+10	2.512E+11	2.304E+10	2.094E+11	0.60	1.04	8.86	3.226E+10	0.242
3M-32L1-32F	3	32	32	0.161	1.790	2.765E+10	2.859E+11	1.545E+10	1.597E+11	0.60	1.19	8.75	3.686E+10	0.243

表3 メモリ・L1混合性能テストの結果(simd-long)

している．つまり n<10m 程度までは，メモリベースの見積りを使用できる事を示している．

simd-long の結果を表 3 に示す．Mdata，Mband，L1data，L1band，Mtime，L1time の定義と計算方法および単位については 4.1 節に準じる．L1data については，本節に述べたように正確に見積もることができない. 理論値を求めた後，実測値と比較したところ５％程度，理論値の方が大きめになっていることが分かった．そこで 3M-32L1-32F の実測値が，理論値の 94%であったため，全ケースについて理論値の 94%を補正後の理論値とした．表 3 の L 1 band の最高値は 241GB/sec であり，この値を L1band_peak とする．この値は, 3.3 の L1 基礎テストの章で得られた 304GB/sec より低めに出ているが，「京」では，L２キャッシュと L１キャッシュについても同一のハードウェアでデータ転送を受け持つようになっており，互いのバンド幅が影響を受けるため，L2 アクセスをしない L１基礎テストの値とは差が生じている．

Mtime>L1time から Mtime<L1time に切り替わる交差点の条件は, 4.1 節に示した式と同様に n=((L1band/Mband)-1)*m であるので，n=4.23m である．この条件をテストプログラムの名称で表わすと，3M-12L1long-12F と３M-16L1long-16F の間となる．表 3 の B/F 値は，Mtime>L1time の場合は，8*m/k を示し，Mtime<L1time の場合は，8*(m+n)/k を示している．Ca，Cp の計算方法も 4.1 節に準ずる．

表 3 を見ると 3M-16L1long-16F の実測値と見積もり値:Cp は, よく一致している. しかし, それ以降の 3M-18L1long-8F から 3M-32LIlong-32F のデータは, 3M-16L1long-16F と B/F 比がほとんど同じ値であるにも係らず，実測値と見積もり値:Cp が一致していない．調査の結果，これらの一致しないケースは，最適化においてソフトウェアパイプラインリングとアンロール[11][12]が適応されなくなった事が原因であると判明した．それは，3M-18L1long-8F のケースにソフトウェアパイプラインリングとアンロールの最適化を外すオプションを指定しても性能は，ほとんど変化せず 14%程度の性能となる事，3M16LI6 に対し，同様に最適化を外すと性能が 15%程度まで落ちる事から分かる．

これらのケースの性能上限値は 25%であるが，上記の状況により上限値までの性能が得られていないが，性能改善は可能という事である

5.4 性能見積り方法

ここまでの議論を整理し，以下のように性能上限値の見積り方法を提案する．

(1) プログラムのアクセス量の計算

プログラムのメモリ・L2 キャッシュ・L１キャッシュアクセスバイトを計算する. 計算する項目は以下の通りとする．

a) メモリまでアクセスする配列のバイト数(m)
b) L2 までアクセスする配列のバイト数(nL2)
c) L1 までアクセスするバイト数のうち short アクセスするバイト数(nL1_S)
d) L1 までアクセスするバイト数のうち long アクセスするバイト数(nL1_L)

ここでいう short/long とは 5.1 節で示した意味である．

(2) 条件の判断と性能値の計算

ここまでの議論をまとめると性能見積もり値の計算手順は以下のようになる. L1 についての性能見積りについては, まず，L１の short アクセス量が限界まで達していない事を確認する．限界を超えていない場合は，5.3 で示したように，メモリベースの見積りで良いので nL1_S の影響は考慮しなくても良い．限界を超えている場合は，今回の論文では議論の対象としない．L1 アクセス量については，m 個分の L1 アクセス分は，5.1 節 のモデル化の中で考慮されているので，nL1_L と nL2 を加算して評価する． f)g)については，5.3 で議論したとおりであるが，n については，long アクセスの対象である nL1_L のみを使用すれば良い．L2 についての性能見積り h)i)については，5.2 に記述した通りである．以下に手順をまとめる．

e) nL1_S<10m である事を確認
・ 成立する場合は L1 への short アクセスは考慮しない．
・ 成立しない場合は見積りできない．
f) (nL1_L+nL2)<4.23m の場合
・これをメモリベースの見積りと呼ぶ．----<eM>
g) (nL1_L+nL2)>4.23m の場合
・これを L1 ベースの見積りと呼ぶ． ----<eL1>
h) nL２<2.17m の場合
・メモリベースの見積りを行う． ----<eM>
i) nL２>2.17m の場合
・これを L2 ベースの見積りと呼ぶ． ----<eL2>
j) 最終的な見積り値の確定
・<eM><eL2><eL1>のうち一番性能が低いものを見積り値とする．

ここで<eM><eL2><eL1>は以下のように計算する．

<eM>：Mband/Ppeak=46/128=0.36 ， Cp=0.36/(m*8/k)
<eL2>：L2band/Ppeak=146/128=1.14,
Cp=1.14/((m+nL2)*8/k)
<eL1>：L1band/Ppeak=241/128=1.88,
Cp=1.88/((m+nL2+nL1_L)*8/k)

6. 実例

6.1 実例１

実例１のプログラムは図 4 に示されたプログラムである．このプログラムのパラメタは，m=5，nL2=21，nL1_S=6,

nL1_L=12, k=43 である. nL1_S<10m の条件を満たすので，L1 の short アクセスは考慮しない．(nL1_L+nL2)>4.23m が成り立つので L1 ベースの見積りとなる．L1 ベースの見積り値は，1.88/((5+21+12)*8/43)=1.88/7.07=0.266 となる．また, nL２>2.17m が成り立つので L2 ベースの見積りとなる．L2 ベースの見積り値は，1.14/((5+21)*8/43)=1.44/4.83=0.236 となる．L1 ベースの見積りと L2 ベースの見積りの小さい方を使用するので，最終的な見積り値は，0.236，23.6%となる．このプログラムの実測値は，14.7%であり見積り値より大分低い値となっている．ここで 4.2 に述べた性能要件を満たしていないか，なんらかの問題が潜在していると予想し調査を実施した．その結果，L1 ミス dm 率を確認すると 13.36%であった．L1 ミスの発生には，アクセス要求(dm)時に発生する，ハードウェアプリフェッチ時に発生する，ソフトウェアプリフッチ時に発生する，の３種類あるが，要求(dm)時に発生する L1 ミスが少ない方が良い．L1 ミス dm 率は，L1 ミスのうち，要求(dm)時に発生する L1 ミスの割合である．この値が大きめの場合は，キャッシュスラッシング[11][12]の可能性が疑われる．今回も，キャッシュスラッシングの可能性を疑い,配列のマージ[11][12]を実施した．その結果，実測性能は 19.7%まで向上し，見積もり性能まで近づくことができた．また L1 ミス dm 率は 3.85%まで低下した．ここで議論した実例 1 のサマリを表 4(a)に記載する．

6.2 その他の実例

その他の実例のサマリを表 4(b)(c)(d)に示す．ここでこれらの詳細は示さないが，(b)は，メモリベースの見積もりとなる．見積もり性能値は，20.8%となるが，最初の実測値は，12.9%であり見積もり性能値に遠く及んでいなかった．そこで 6.1 と同様に，L1 ミス dm 率を確認すると 15.11%であった．ここでも配列マージのチューニングを実施することにより，実測性能が 19.3%まで向上し，L1 ミス dm 率も 8.81%まで低下した．

(c)もメモリベースの見積もりとなる．見積もり性能値は，4.5%となり，実測値は，3.8%である．この例は，L1 ミス dm 率も 0.31%と低く，これ以上の性能向上の試みは実施しない．

(d)は，L2 ベースの見積もりとなる．見積もり性能値は，32.4%となるが，実測値は，25.7%であり，多少の性能差が見受けられる．L1 ミス dm 率を確認すると 14.44%であり，L1 キャッシュスラッシングの発生が疑われる．ただ，このプログラムのケースは，今までの配列マージのテクニックが使用できない．また通常のキャッシュスラッシングの解消テクニックである，ループ分割を使用すると，ロード・ストアの増加を招く可能性が大きい．また，パディング(参照)のテクニックも使用できない状態である．従って，これ以上の性能向上については断念するが，本来は，30%程度の性能を得られるプログラムであると考える．

	m	L2	L1_S	L1_L	演算数	見積もり性能	実測性能	tune後実測性能
(a)	5	21	12	6	43	0.236	0.147	0.197
(b)	13	2	12	8	60	0.208	0.129	0.193
(c)	11	2	0	2	11	0.045	0.038	—
(d)	3	8	8	0	25	0.324	0.257	—

表 4 実例のサマリ

7. まとめ

プログラムの限界性能を見積もるためにルーフラインモデルが提唱されている．本報告では,「京」上で，まずルーフラインモデルが，onL2/onL1 キャッシュの場合にも成り立っている事を確認し，次にルーフラインモデルを拡張したメモリとキャッシュアクセスが混在している状態での性能見積もりモデルを提案した．さらに「京」上で提案した性能見積もりモデルが成り立つことを確認した．最後に性能見積もりモデルを実際のプログラム例に適用し，提案した見積もりモデルでプログラムの限界性能値を見積もることができることを示した．今後は，このような見積もりモデルを，intel マシン等，他のアーキテクチャにまで広げていきたい．

謝辞

本報告に際しご討論頂き貴重な助言を頂いた，富士通株式会社の青木正樹氏，井上晃氏をはじめとした性能評価チームの皆様，理化学研究所 AICS 運用技術部門ソフトウェア技術チームの諸氏に感謝いたします．本報告の結果は、理化学研究所のスーパーコンピュータ「京」を利用して得られたものです．

参考文献

[1] Matteo Frigo,Volker Strumpen：Cache Oblivious Stencil Computation,ICS '05 Proceedings of the 19th annual international conference on Supercomputing,Pages 361-366 ,ACM New York, NY, USA ,2005

[2] 近藤正章，岩本貢，中村宏：キャッシュラインを考慮した 3 次元 PDE solver の最適化手法,情報処理学会研究報告．計算機アーキテクチャ研究会報告 2001(22), 91-96, 2001-03-08

[3] S. Williams, A. Waterman, and D. Patterson: Roofline: an insightful visual performance model for multicore architectures. Commun. ACM, 52:65–76, 2009.

[4]南一生，井上俊介，堤重信，前田拓人，長谷川幸弘，黒田明義，寺井優晃，横川三津夫.："「京」コンピュータにおける疎行列とベクトル積の性能チューニングと性能評価"ハイパフォーマンスコンピューティングと計算科学シンポジウム論文集, pp.23-31 (2012)

[5] 雑誌 FUJITSU2012-5 月号 (VOL.63, NO.3)特集：スーパーコンピュータ「京」

[6] Datta, K., Murphy, M., Volkov, V., Williams, S., Carter, J., Oliker, L., Patterson, D., Shalf, J., and Yelick, K. Stencil computation optimization and autotuning on state-of-the-art multicore architectures. Conference (Austin, TX, Nov. 15–21). IEEE Press, Piscataway, NJ, 2008, 1-12.

[7] Williams, S., Carter, J., Oliker, L., Shalf, J., and Yelick, K. Lattice Boltzmann simulation optimization on leading multicore platforms. In Proceedings of the IEEE International Symposium on Parallel and Distributed Processing Symposium (Miami, FL, Apr. 14–18, 2008), 1–14.
[8] T. Furumura and L. Chen, “Parallel simulation of strong ground motions during recent and historical damaginge earthquakes in Tokyo, Japan”, Parallel Computing, 31, pp149-165, 2005.
[9]古村孝志, “差分法による３次元不均質場での地震波伝播の大規模計算”, 地震 2, 61 巻, S83-S92, 2009.
[10]「乱流音場解析ソフトウェア FrontFlow/Blue」: http://www.ciss.iis.u-tokyo.ac.jp/riss/project/device/
[11]南一生:配信講義, CMSI 計算科学技術特論Ｂ, アプリケーションの性能最適化 2 (CPU 単体性能最適化), http://www.cms-initiative.jp/ja/events/2014-haishin
[12]青木正樹,プログラムのチューニング方法, http://www2.itc.nagoya-u.ac.jp/riyou/tuning.pdf