Vol.2012-ARC-202 No.28
Vol.2012-HPC-137 No.28
2012/12/14

# 時間的局所性を考慮した疎行列のキャッシュ適合性

冨森苑子[†1]　田邊 昇[†2]　高田雅美[†1]　城 和貴[†1]

エクサスケールマシンは複雑なメモリシステムとなることが予想されている．同マシンへの適用を視野に入れた疎行列ライブラリの実現に向け，本報告では疎行列のキャッシュへの適合性分類において，疎行列の形状から得られる列インデックス列の空間的局所性と時間的局所性に関する指標を併用することを提案する．さらに，提案指標をフロリダ大学の疎行列コレクションを用いて評価した．その結果，空間的局所性だけで説明できなかった諸現象が説明可能になった．L1 キャッシュヒット率と時間的局所性の指標値との間に相関を確認できた．さらに，両指標を用いたメモリアクセス機構選択基準を与える方程式を導出した．

# Sparse matrices suitability for cache memory based on temporal locality

SONOKO TOMIMORI[†1]　NOBORU TANABE[†2]
MASAMI TAKATA[†1]　KAZUKI JOE[†1]

In Japan, memory system of ExaFLOPS machines is expected more complex. In this paper, we propose a new characteristic of sparse matrices about spatial locality and temporal locality of row-index sequences in order to classify suitability for cache memory systems. Moreover, we evaluate proposal characteristic using University of Florida Sparse Matrix Collection. As a result, it became possible to explain phenomenons which could not be explained only by spatial locality. It is confirmed that there are the correlations between temporal locality and the general purpose cache (L1) hit rate. Moreover, we developed the equation for access method selection based on proposed characteristics.

## 1. はじめに

近年，2018 年頃のエクサスケールマシンの実現に向けた検討[1]が日本でも行われるようになった．エクサスケールマシンではメモリバンド幅を十分に確保できなくなり，それをカバーするための複雑なメモリシステムの採用が予想されている．大規模疎行列を係数とする連立一次方程式(疎行列ベクトル積)に帰着される応用ではメモリバンド幅への要求が高い．国内の重点アプリケーションの多くがこのクラスに属するため，高速化へのニーズが高い．

また，近年ではグラフ処理に代表されるビッグデータ処理が注目を集めている．これらにおいても，巨大で複雑な非零要素配置を有する疎行列で表現される処理が必要になる．Web サイトの重要性を与える PageRank などの大規模非定型情報の検索[2]や嗜好分析・リコメンーションにおいて，疎行列処理の大規模化と高速化が求められている．

現時点での世界第二位のスーパーコンピュータである京は 2 階層のキャッシュをベースにした比較的シンプルなメモリシステムを有する．しかし，その上での最適化でさえ容易ではない．ごく少数の選抜アプリケーションのみに最適化のプロが配置され，そのアプリケーション固有の性質を利用した極限に迫る最適化が行われている．一方，選抜されなかったアプリケーションや，寿命が短いアプリケーションでは，ユーザーの科学者たちは性能チューニングに時間と人手を投資できない．疎行列処理に帰着される応用だけを取ってみても，フロリダ大学の疎行列コレクション[3]をみれば明らかなように，その非零要素配置は多種多様である．そこで重要になるのが最適化済みのライブラリや自動最適化コンパイラである．複雑なメモリシステムを有するエクサスケールマシンでは，自動化された最適化機構の重要性が一段と高まる．

以上のような認識から筆者らは，アプリケーション固有の最適化を避けた二種類の統一的調整機能や，アクセス機構選択による疎行列向けの新しい自動チューニング手法を提案し，それらに基づく大規模疎行列やメモリシステムの特性を考慮した高速な汎用疎行列ライブラリの構築に着手した．[4]

その第一段階として疎行列の空間的局所性に関する特性指標を提案し，その指標と GPU の L1 キャッシュヒット率の間に相関を調査した．[5]その結果，強い相関を示すグループが多数派であるが，少数ながら示さない疎行列も存在することを確認した．このことは，疎行列のキャッシュへの適合性を事前判定するためには，空間的局所性の指標のみでは不十分であることを意味している．

本論文では上記の問題点を補うために，空間的局所性に関する指標に加え，時間的局所性を併用した指標を提案する．さらにその指標によって各種疎行列の特性を測定し，GPU の L1 キャッシュヒット率の間に相関を調査する．

本報告の構成は以下の通りである．2 章では参照の局所

†1 奈良女子大学
Nara Women's University
†2 (株)東芝
Toshiba corporation

Vol.2012-ARC-202 No.28
Vol.2012-HPC-137 No.28
2012/12/14

性に関して整理する．3 章では筆者らの先行研究で提案した疎行列の空間的局所性に基づく特性指標について紹介する．4 章ではその不足を補う時間的局所性に基づく特性指標の併用について提案する．5 章では提案指標を用いた評価実験について述べる．6 章では関連研究について述べ，7 章でまとめと今後の課題について論ずる．

## 2. 参照の局所性

本章ではキャッシュメモリがメモリアクセスの高速化のために利用している参照の局所性に関して整理する．

### 2.1 空間的局所性

空間的局所性とは「メモリ上のある項目が参照されると，その近くの項目もまもなく参照される確率が高いという性質」を言う．

通常，キャッシュメモリはこの性質を有効活用するために，アドレスが近いデータのブロック（キャッシュライン）を設け，ヒット・ミスの判定や，ミス時のデータ移動，外部メモリアクセスをライン単位で行なう．京や GPU を含む，多くのプロセッサが 128 バイトのラインサイズを有するキャッシュを搭載している．バースト長をある程度以上長くすることでデータ転送効率が良くなり，ミス 1 回あたりのペナルティの大きさや，有限サイズのキャッシュメモリ容量に収まるライン数を確保することの利点（後述する時間的局所性の活用）との間のトレードオフを多くのベンチマークにより評価した結果から，現時点ではラインサイズは 128 バイトに設定されるケースが多い．ただし，容量が大きくなるほど最適なラインサイズは大きくなる傾向にあるので，将来はさらに大きくなる可能性がある．

しかし，Graph500 ベンチマークや疎行列ベクトル積のような幾つかのランダムアクセス性の強いアプリケーションには，この空間的局所性が乏しい．このため，例えば 1 個の 4 バイトしか必要としない時にも，使わない近傍の 124 バイトを含む 128 バイトを外部メモリから転送する必要があり，非常に効率の悪い状態をキャッシュラインは引き起こす．よって，疎行列処理の高速化においては，疎行列アクセスにおける空間的局所性の制御が極めて重要である．

特に，キャッシュライン内部の空間的局所性はソフトウェアによる制御は困難で，この問題の解決には，キャッシュを排除したベクトル型スーパーコンピュータのメモリシステムか田邊らが提唱してきたキャッシュベースシステムと組合せ可能な DIMMnet-2 の Gather 機能や，Gather 機能付きハイブリッドメモリキューブ[9]などのハードウェアレベルでのサポートが必要である．空間的局所性や後述する時間的局所性も低い疎行列の場合は，これらのハードウェアを用いた解決方法が有効であり，3 章で紹介する筆者らの先行研究や本研究は，そのような疎行列の特定を目指すものである．

### 2.2 時間的局所性

時間的局所性とは，「メモリ上のある項目が参照されるとまもなくそれが再び参照される確率が高い，という性質」を言う．

通常，キャッシュメモリは上記の性質を有効活用するため，多くのラインから構成され，直前のアクセス以外のアクセスもキャッシュ上に存在する確率を高める．

時間的局所性に関する理論的研究の歴史は古く，Denning らは 1968 年に現在時刻 t とウィンドウサイズ T で定義されるワーキングセット W(t,T)等の概念を提唱し，1972 年には平均ワーキングセットサイズを参照列から 1 パスで求めるアルゴリズムを提案している．

行列処理においては，例えば密行列積のように行列を小さな部分行列に分割するタイリングによって，メモリアクセスの時間的局所性を改善する手法が有効な場合がある．しかし，多様な非零要素配置を有する疎行列処理においてはタイリングが困難な場合が多い．疎行列処理においては行列によっては行番号と列番号の入替えを行なうことで，メモリアクセス順序を変更し，時間的局所性を向上できる．つまり，時間的局所性はソフトウェア的に制御ができると考えられる．

## 3. 空間的局所性に基づく行列特性指標

### 3.1 先行研究における提案の概要

筆者らは疎行列のキャッシュへの適合性分類に資する疎行列の特性に関する新しい指標として「列インデックス列の空間的局所性」を提案した．図 1 にその概念図を示す．行列の特性値を表す場合の定義を以下に示す．「非零要素のみを CRS 形式で格納し，列ベクトル読み出しに用いるインデックス配列を先頭から順に読み出した際に，下位 5bit 以外が継続して一致している回数をカウントする．不一致が生じた時のカウント値を出力し，1 からカウントしなおす．その出力されたカウント値の数列の平均を取ったものを，列インデックス列の空間的局所性と定義する．」

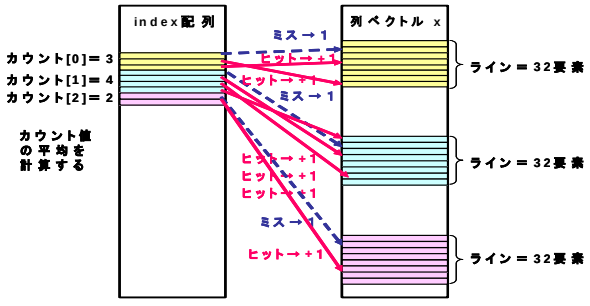



図 1　先行研究の提案指標(列インデックス列の空間的局所性)

Figure 1　The proposed property (spatial locality of row-index sequences)

### 3.2 先行研究における結果の概要

筆者らは上記の提案指標と GPU の L1 キャッシュヒット

Vol.2012-ARC-202 No.28
Vol.2012-HPC-137 No.28
2012/12/14

率の間に相関を調査した．その代表的な結果を図に示す．

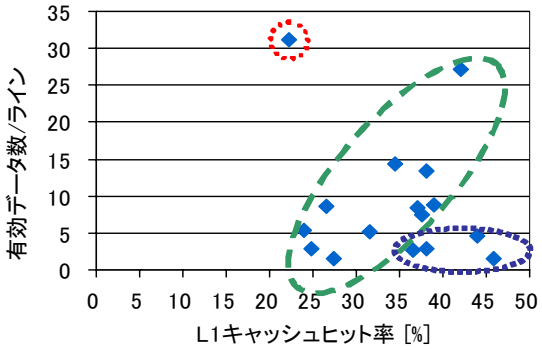



図 2　L1 ヒット率と有効データ数/ラインの関係(Fold 法前処理後)

Figure 2　The relation between L1 cache hit rate and the number of valid data/line (Pre-processed by Fold-method)

図 2 より明らかなように，提案指標と GPU の L1 キャッシュヒット率の間に正の相関を示すグループ(緑で囲んだ行列グループ)が多数派であるが，少数ながら相関を示さない疎行列(青で囲んだグループ)や，1 つだけ孤立したもの(赤で囲んだ行列)も存在することを確認した．このことは，疎行列のキャッシュへの適合性を事前判定するためには，空間的局所性の指標のみでは不十分であることを意味している．

## 4. 時間的局所性を考慮した行列特性指標

### 4.1 行列特性指標に関する提案

3 章に記載された方法によってキャッシュライン内部の空間的局所性については考慮されているので，これにライン単位の時間的局所性を組み合わせることを提案する．ライン単位の時間的局所性を測定するには，列インデックス列の値をライン内の項目数に相当するビット数(128 バイトラインでデータ型が単精度浮動小数ならば 128/4＝32 に対応する 5bit)だけ右にシフトしたラインの識別子となる line_ID[t]を入力アドレス列として用いればよい．

具体的には，参照間隔のヒストグラムを生成する Denning のアルゴリズムをシンプルにし，そこから得られる時間的局所性の評価指標を疎行列の列インデックス列に対して測定し，疎行列のキャッシュへの適合性を判断する．ラインの参照間隔とは時刻 t にアクセスしたラインを最後にアクセスした時刻 TIME[line_ID[t]]との時間差のことである．アルゴリズムにおいては時刻 t を進める前に TIME[line_ID[t]]に現在時刻 t を記録する．

時間的局所性に関する評価指標としては平均参照間隔または，これにラインサイズを乗じた平均ワーキングセットサイズを用いる．これらの計算には必ずしも Denning のようにヒストグラムを作成する必要がない．よって，疎行列ライブラリのフロントエンド部に組込む際の判定時間の短縮の観点から，デフォルトではヒストグラムは作成しない．

なお，空間的局所性指標と時間的局所性指標は同じ列インデックス列を入力として計算可能なので，両者は並行して計算することができる．特に，列インデックス列がファイルに入っている場合は別に計算するよりも並行して計算した方が処理時間の短縮が可能である．

### 4.2 自動最適化への応用

平均ワーキングセットサイズはバイト数で表現され，どれくらいのキャッシュ容量を準備すればヒット率が安定するかを示す．よって，これと実行プラットフォームのキャッシュ容量の大小関係を比較することで，時間的局所性の観点からのキャッシュの効き具合や，自動最適化においてキャッシュを用いる方向を優先すべきか否かを判断できると予想される．

空間的局所性の指標値と時間的局所性の指標値から，最適化の方向性をどうするかについて定性的な観点で整理すると表 1 のようになる．表 1 は図 2 で表示されている位置が対応するようにならべてある．

表 1　二つの局所性指標を組合せた最適化戦略

Table 1　Optimization strategy.

| | 時間的局所性:低 | 時間的局所性:高 |
|---|---|---|
| 空間的局所性 高 | オーダリングの模索 | キャッシュ優先 |
| 空間的局所性 低 | メモリ側 Gather 優先 | L1 に入る：キャッシュ優先 |
| | | L1 に入らない：メモリ側 Gather |

図 2 において，時間的局所性の指標の導入により，3 章に記載の指標だけではなぜヒット率が異様に低くまたは高くなったのか説明できなかった赤線グループや青線グループの現象の原因の特定が可能になると予想される．

例えば，図 2 の赤線グループの apache2 は空間的局所性の観点からは高いヒット率が見込まれるが，極端に時間的局所性が低い場合はその効果を打ち消してしまうはずである．この場合はすでに空間的局所性が高い状態なので，Gather 付きメモリの効果は望めず，そのままではキャッシュも効かない．ただし，オーダリングによって時間的局所性を高められる可能性は皆無ではないので，自動最適化においてはオーダリングの方向を模索すべきである．オーダリングを変えると時間的局所性や空間的局所性にも変化が出るので，表 1 の別のカテゴリに入るようになる可能性もある．逆に，図 2 の青線グループのように空間的局所性の観点からは高いヒット率が見込まれるが，平均ワーキングセットサイズがキャッシュ容量より十分に小さく，極端に時間的局所性が高い場合はその効果を打ち消してしまうはずである．その場合はキャッシュを用いても実行効率があまり悪くならないので，自動最適化においてはキャッシュを用いる方向を優先すべきである．

Vol.2012-ARC-202 No.28
Vol.2012-HPC-137 No.28
2012/12/14

また，図 2 の緑線グループの左下周辺の疎行列のように空間的局所性が低く，極端に時間的局所性が高くはない場合は，Gather 付きメモリの効果が確実に望めるので，自動最適化においてはそれを用いる方向を優先すべきである．

## 5. 評価

### 5.1 実験環境と評価行列

今回の実験に用いた計算機環境を表 2 に示す．また，評価に用いた行列を表 3 に示す．二つの局所性を組合せた考察を行なうため，表 3 の行列は空間的局所性指標を評価した際に用いたものと同じものとした．これらの行列は University of Florida Sparse Matrix Collection から抜粋したものである．University of Florida Sparse Matrix Collection とは，実際のアプリケーションでよく生じる疎行列を集めたものである．これらの疎行列は，疎行列アルゴリズムの開発と性能評価のための数値線形代数の研究者に多く使用されている. 先行研究[8][9]で使用した疎行列に新たに追加を行った．追加された行列は，行列形状図上で非零要素が不規則に散らばっているように見える(キャッシュ向けの最適化が効きにくいと考えられる)疎行列を選んだ. それらは構造解析，電子回路解析，web 解析，道路網解析のアプリケーションに由来する疎行列である．nd24k については作者不明(アプリ種:2D/3D 問題)な行列で，行列形状図上では他の行列との質的な差が判らないものである．

### 5.2 評価実験

4.1 節で述べた疎行列の特性指標の有効性を確認するため，CRS 形式と前処理後における提案指標と，前処理後の L1 ヒット率を測定した．本研究における前処理は，Fold 法[8][9]を用いた．この Fold 法という前処理では，CRS 形式からインデックス配列内部のアクセス順序を GPU 向けに転置を用いて変更している．そのため，キャッシュヒット率はその影響を受ける．つまり，CRS 形式のアクセス順序とはキャッシュへの相性が大幅に異なるため，前処理の影響が顕著に観測できるものと期待できる．

CRS 形式で格納されたインデックス配列を先頭から読み込むと，非零要素が行内では昇降順となったインデックス値列が読み込まれる．時間的局所性に関する提案指標値を計測する手順は以下の通りである．

(0) 配列 TIME を 0 で初期化する．

(1) 現在の時刻インデックス t で読み込んだインデックスの下位 5bit を右シフトで削除して line_ID とする．

(2) line_ID を最後にアクセスした際の時刻インデックスを配列 TIME から読み出し, 現在の時刻インデックス t との差(これを現在の参照間隔 x とする)を計算し，累算する．ただし読み出した TIME が 0 だった場合は初回のアクセスを意味する．この場合の x を 0 とし，平均の計算には加えないものとする．

(3)直前現在のインデックスを配列 TIME に記録する．回数をカウントする．不一致が生じるとその時のカウント値を出力し，1 からカウントしなおす．その出力されたカウント値の数列の平均を取る．

(4)インデックス配列の最後まで上記が終わったら累算値を N で割って，平均参照間隔を求める．それを 128 倍してラインサイズ 128 バイトの場合の平均ワーキングセットサイズを求める．

さらに，上記の手法で得られる参照間隔の発生頻度を記録したヒストグラムも生成した. 最大 N/32 の参照間隔が発生する可能性があるので，その大きさの配列 COUNT(初期値 0)の x の位置をインクリメントすることで参照間隔のヒストグラムが得られる．

表 4 に計測した各行列の平均参照間隔および平均ワーキングセットサイズを示す．図 3 に各行列の参照間隔のヒストグラムを示す．各ヒストグラムの一番右端の階級は L より大きいものと初期ミスの回数の合計である．

表 2 測定環境

Table 2 Experimental environment.

| | |
|---|---|
| CPU | Ｉｎｔｅｌ®Xeon®CPU X5670 @ 2.93GHz |
| GPU | Nvidia Tesla C2050 (コア数 448) |
| デバイスメモリ | メモリバンド幅 144GB/s,3GB |
| ホスト I/F | PCI express x16 Gen.2 (最大バンド幅 8GB/s) |
| OS | RedHat Enterprise Linux Client release5.5 |
| CUDA | Cuda3.2 |

表 3 評価に用いた行列

Table 3 Experimented matrices.

| 行列名 | 非零要素数 | 行数 |
|---|---|---|
| crankseg_2 | 7,106,348 | 63,838 |
| nd24k | 14,393,817 | 72,000 |
| thermal2 | 3,489,300 | 147,900 |
| hood | 5,494,489 | 220,542 |
| F1 | 13,590,452 | 343,791 |
| msdoor | 10,328,399 | 415,863 |
| rajat29 | 4,866,270 | 643,994 |
| ASIC_680ks | 12,329,176 | 682,712 |
| apache2 | 2,766,523 | 715,176 |
| ldoor | 23,737,339 | 952,203 |
| webbase-1M | 3,105,536 | 1,000,005 |
| delaunay_n20 | 2,097,124 | 1,048,576 |
| roadNET-TX | 1,281,106 | 1,393,383 |
| Hamrle3 | 5,514,242 | 1,447,360 |
| G3_circuit | 4,623,152 | 1,585,478 |
| roadNET-CA | 1,844,404 | 1,971,281 |

表 4　疎行列の平均参照間隔および平均ワーキングセットサイズ

Table 4　Average inter-reference interval and average working set size for sparse matrices.

| 行列名 | 平均参照間隔 | 平均ワーキングセットサイズ[B] |
|---|---|---|
| crankseg_2 | 41.8 | 5,344 |
| nd24k | 73.6 | 9,418 |
| thermal2 | 103.2 | 13,206 |
| hood | 335.9 | 42,989 |
| F1 | 135.7 | 17,370 |
| msdoor | 135.0 | 17,281 |
| rajat29 | 463.3 | 59,306 |
| ASIC_680ks | 442.6 | 56,647 |
| apache2 | 1,088.5 | 139,323 |
| ldoor | 97.3 | 12,448 |
| webbase-1M | 360.5 | 46,141 |
| delaunay_n20 | 403.8 | 51,681 |
| roadNET-TX | 130.7 | 16,730 |
| Hamrle3 | 345.9 | 44,277 |
| G3_circuit | 499.1 | 63,886 |
| roadNET-CA | 652.5 | 83,526 |

ここで今回の測定では現時点で最大クラスの 8MB の L2 キャッシュのサイズに相当するウィンドーL の大きさを設定して平均距離を計測した．

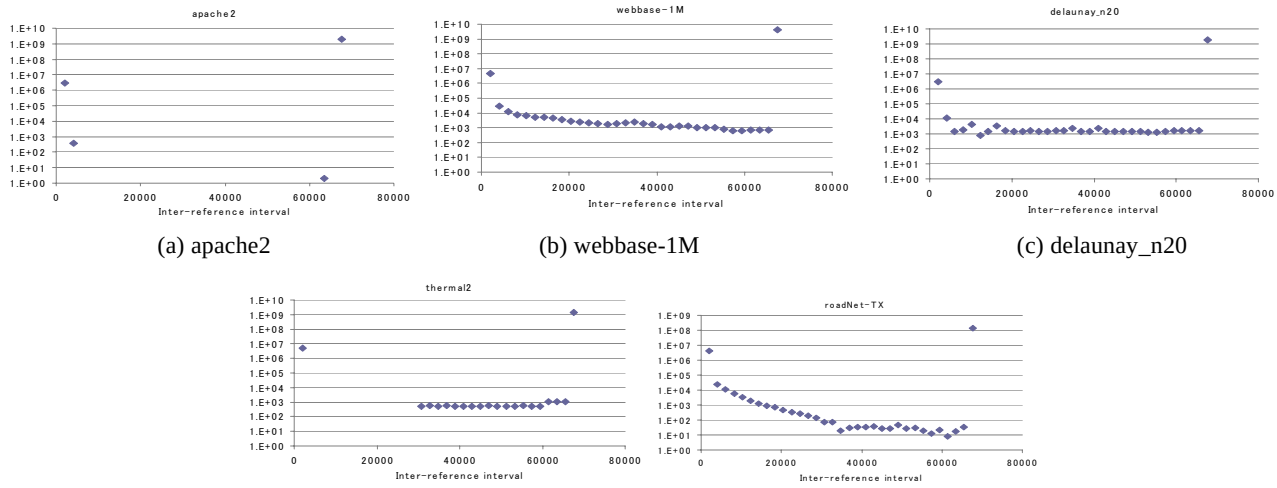
### 5.3 考察

thermal2 および roadNET-TX は平均ワーキングセットサイズが明らかに小さく，GPU の L1 キャッシュサイズ(Fermi では 16KB)と同等かそれ以下である．先行研究で空間的局所性が低かったにもかかわらずヒット率が比較的高く出たことは，これが原因であると考えられる．これは時間的局所性を併用することで明瞭になった効果である．したがって，自動最適化の観点から thermal2 および roadNET-TX のケースを考察すると，空間的局所性は低く，かつ時間的局所性が高く，その程度は平均ワーキングセットサイズが L1 キャッシュに多くの割合が入る範囲なのでキャッシュを優先して選択するという戦略をとるのが妥当と考えられる．ただし，行列のデータ部と index 部もキャッシュを通過する実装の場合は L1 キャッシュが手狭になるため，メモリ側 Gather も検討すべきである．

一方，apache2 は時間的局所性の観点から青グループの 4 つとは桁違いに平均ワーキングセットサイズが大きい．これが L1 キャッシュサイズを大幅に上回ることから，空間的局所性が高いにもかかわらず apache2 において L1 ヒット率が低く観測されたことが説明できる．これも時間的局所性を併用することで明瞭になった効果である．

図 3 の疎行列の参照間隔のヒストグラムによれば疎行列のキャッシュ適合性に関連した個性が見えてくる．例えばキャッシュのヒット率が低かった apache2 の場合は左右の両端にしかサンプル点が存在せず，中途半端な大きさのキャッシュでは右端のミスしている階級を全く救済することができない．この状態での空間的局所性は極めて高いのでメモリ側 Gather の効果も期待できない．そのため，キャッシュかメモリ側 Gather のどちらかで救済できるかオーダリングを探す以外に高速化が困難である．

他の行列も両端に分かれているが，中間部に左端の数百分の 1 以下のサンプルが存在する．これらは平均ワーキングセットサイズを大幅に上回るメガバイトクラスにキャッシュを増量すれば救済できるものの，頻度が両端と比べると極端に低いので投資効果が薄いことがわかる．




(a) apache2　(b) webbase-1M　(c) delaunay_n20

(d) thermal2　(e) roadNET-TX

図 3　疎行列の参照間隔のヒストグラム

Figure 3　The histogram of inter-reference interval for sparse matrices

Vol.2012-ARC-202 No.28
Vol.2012-HPC-137 No.28
2012/12/14

次に今回測定した平均参照距離と先行研究で測定したGPU(C2050)の L1 キャッシュヒット率の関係を表したものが図 4 である．相関係数は-0.682 であり，ややばらけた分散をしているものの先行研究の空間的局所性における同様の図のような大きく外れる異常サンプルは見られない．線形近似した場合の近似式は hit＝-0.018d+41.267 となる．

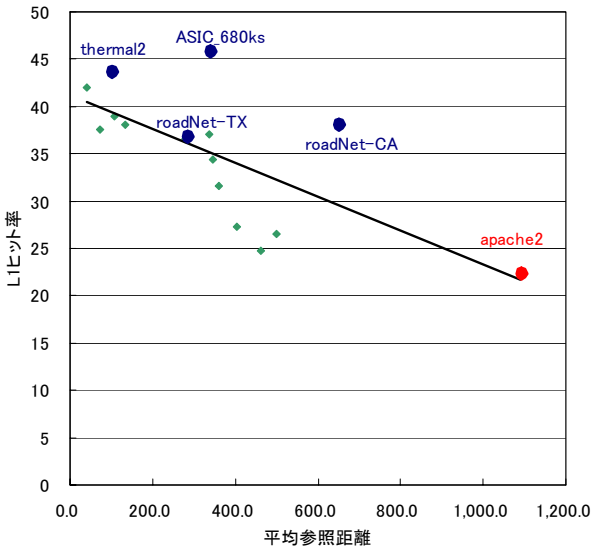



図 4 疎行列ベクトル積における平均参照間隔とGPU(C2050)の L1 キャッシュヒット率の関係

一方，キャッシュのヒット率は参照間隔を計測する前述のプログラムにおいて，参照間隔がキャッシュのライン数以下になるアクセス数をカウントし，これを全アクセス数で割った値を計算することで得られることが，田中の論文[18]に記載されている式から裏付けられる．なお，このヒット率の前提としてリプレースを取り込んだ時期が最も古いラインとする方式なので，リプレースアルゴリズムやウェイ数などの細かいキャッシュ構成の違いが強く影響するようなケースでは誤差が生じる可能性もある．

上記の 2 つの手法で得られる予測ヒット率と先行研究で提案した空間的局所性指標値を用いて，キャッシュを用いる場合の疎行列 A とベクトル x の積(疎行列ベクトル積)の性能モデルを立てることができる．これとメモリ側 Gather の Gather スループットで決まる同様の性能モデルとの間での大小関係から，客観性のあるメモリアクセス機構選択か可能になると考えられる．

1FLOPS あたりの必要バンド幅[B/FLOP]の式を作るにあたり，いくつかの値を以下のように定義する．

hit：列ベクトル x のヒット率
S：空間的局所性指標(ライン内 32 個中の有効データ数)

1FLOPS あたりの必要バンド幅は以下の三つの値の和で表される．

$BPF_{cache}＝\alpha＋\beta＋\gamma＝6＋(1\text{-}hit)*128/S$ [B/FLOP]

$\alpha$ ：index に必要なバンド幅：4[B/FLOP] (連続アクセス＆再利用性なし)
$\beta$ ：A に必要なバンド幅：2[B/FLOP] (連続アクセス＆再利用性なし)
$\gamma$ ：x に必要なバンド幅：有効データのみで 2B/s を出すのに必要なバンド幅

ここで，index は大きな場合を想定し，8 バイト符号なし整数，A は精度改良を取り入れた混合精度を用いる場合の性能を決める単精度浮動小数（4 バイト）を仮定した．また，x はキャッシュ容量より十分に大きく，キャッシュミスのリプレース動作によりキャッシュに取り込まれるものとしている．

上記の $\gamma$ は 0.5 個の x(2B)をリプレースで持ってくる時に消費されるバンド幅に対応する。ミス 1 回で S 個の x(4B)をリプレースで持ってくるのに 128B を 2 回移動する。ミス率 1 の時 0.5 個の x(2B)をリプレースで持ってくるのに 0.5*256/S[B]を移動する。これはミス率 1 の時 1FLOP あたりに 128/S[B]を移動と同じである。ミス率(1-hit)の時 1FLOP あたりに $\gamma$＝(1-hit)*128/S[B] を移動することになる．よって，主記憶(GPU の場合はデバイスメモリ)のバンド幅 $W_{cache}$[B/s]によって得られる処理速度は以下のようになる．

$F_{cache}＝W_{cache}/(6＋(1\text{-}hit)*128/S)$ [FLOPS]

一方，メモリ側 Gather では index はメモリ側から動かないので $\alpha$ に相当するバンド幅消費は Gather スループット $W_{gather}$[B/s]の中に組込まれる．$\beta$ はキャッシュの場合同様 2[B/s]であり， x は Gather によって連続化されるので $\gamma$ は $\beta$ 同様に 2[B/s]であり，全体として以下の式で 1FLOPS あたりの必要なメモリバンド幅が表される．

$BPF_{gather}＝4$ [B/FLOP]

Gather スループット $W_{gather}$[B/s]の場合の得られる処理速度は以下のようになる．

$F_{gather}＝W_{gather}/BPF_{gather}＝W_{gather}/4$[FLOPS]

$F_{cache}$ と $F_{gather}$ の大小関係を規定する両者が等しいとした方程式がメモリアクセス機構選択を行なう境界線を規定する．前述のようにヒット率と平均参照距離の間には相関関係があり，hit に参照距離 d による近似式を代入することで上記の境界線は d と S の二つの指標値で形成されるグラフ

Vol.2012-ARC-202 No.28
Vol.2012-HPC-137 No.28
2012/12/14

の領域を分割する境界線を形成する．$W_{cache}$ や $W_{gather}$ のハードウェアパラメータによってこの境界線は移動する．以上のメモリアクセス機構選択最適化手法の概念を図 5 に示す．2 つの指標値を測定し，理論的に導出された境界線方程式にハードウェアパラメータとともに代入することによってそれらの値がどちらの領域に属するかを判定することで客観性のあるメモリアクセス機構選択か可能になると考えられる．

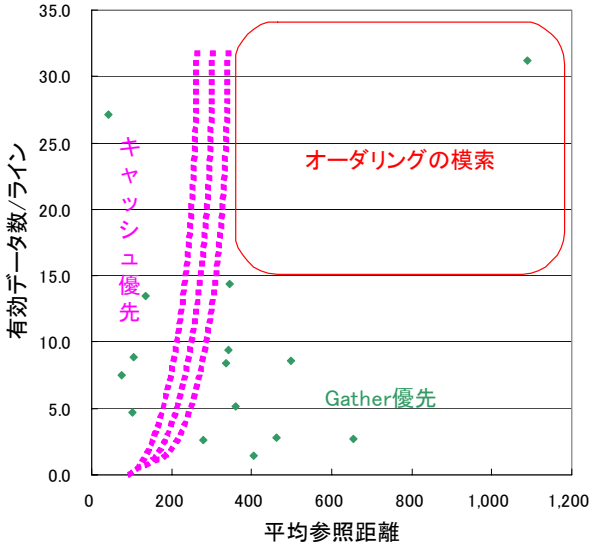



図 5 空間的局所性と時間的局所性の 2 つの指標を併用した境界線方程式を用いたメモリアクセス機構選択最適化手法の概念

## 6. 関連研究

京の上では，疎行列計算を主体にした 2 つのアプリケーション上で，アプリケーション固有の性質を利用した最適化を人手で行っていることが報告されている．例えば 1 行の最大非零要素数を 27 に固定する制約を課した最適化などが行われている[10]．汎用志向のアプローチでは複数のソフト実装の中から実行時に自動で試して選択する自動チューニングの研究も行われている．例えば CPU 上での処理アルゴリズムの選択[11]，GPU 上での行列格納方式の選択[12]基づく研究が報告されている．筆者らの先行研究では行列サイズとキャッシュ容量の関係が疎行列ベクトル積性能に重要な影響があることが示されている[6]．さらに筆者らの先行研究では「列インデックス列の空間的局所性」を疎行列の特性指標として提案している[5]．

ただし，筆者らの知る限りでは「列インデックス列の空間的局所性」と「列インデックス列の時間的局所性(平均ワーキングセットサイズ)」の両方を疎行列や疎行列向け前処理アルゴリズムの特性として事前に測定し，疎行列処理の自動チューニングに用いている前例は無い．

疎行列と局所性に着目した点で類似した研究としては，Heras の疎行列ベクトル積における局所性の研究[13]がある．3 種類の距離関数を指標として提案している．ただしそこで用いる最適なウィンドウサイズや指標の種類をキャッシュシミュレータベースの試行錯誤から得るため，自動チューニングには向かないと考えられる．Perarnau による研究[14]も疎行列と局所性に着目した点では共通だが，AMG 法を対象としており，疎行列ベクトル積を対象としている本研究とは異なる．これは実機からトレースを得る方式であり，疎行列の非零要素配置(列インデックス列)を入力として用いている本研究とはこの点でも異なる．

時間的局所性に関しては Denning によるワーキングセットに関する研究[15][16]が古くから存在し，本研究においてもそれを基礎においた参照距離(inter-reference interval)に基づいた指標の測定を行なっている．Matson による Reuse distance(Stack distance)[17]を用いている先行研究も多数存在する．ただし，こちらは解析時間が上記より多くかかると考えられ，自動チューニングで用いる場合には不利になると考えられる．さらに田中[18][19]は Denning の時間的局所性にスケールパラメータを追加し，空間的局所性について拡張した理論を構築している．本研究ではラインサイズが 128 であることが殆どである現状に鑑み，スケールパラメータは 32(1 ラインに 32 個の単精度浮動小数が格納される状態)に固定して議論を進めており，時間的局所性については基本的には Denning と同等である．田中のように両方の局所性が合成された状態ではなく，本研究はライン内の空間的局所性とライン間の時間的局所性を分離した評価指標で個別に特性を評価している点で異なる．ライン内の空間的局所性を分離して評価することにより，Gather 付きメモリを利用すべき疎行列の分別が可能になる．

## 7. おわりに

エクサスケールマシンは複雑なメモリシステムとなることが予想されている．同マシンへの適用を視野に入れた疎行列ライブラリの実現に向け，本報告では疎行列のキャッシュへの適合性分類に資するライン内の空間的局所性とライン間の時間的局所性を分離した評価指標で個別に特性を評価することを提案した．ライン間の時間的局所性の指標には平均参照間隔と平均ワーキングセットサイズを用いた．さらに，入力疎行列および Fold 法前処理後の提案指標の値をフロリダ大学の疎行列コレクションを用いて提案手法を評価した．その結果， 空間的局所性の観点だけからはキャッシュ挙動が説明困難だったケースが，時間的局所性の観点からは説明可能になった．これにより，より多くの疎行列に対して疎行列の非零要素配置のみから適切な最適化戦略を導出するための基準を強化することができた．さらに性能のモデル化を行い，それを基にした境界線方程式を導出した．2 つの指標値を測定し，理論的に導出された境界線方程式にハードウェアパラメータとともに代入することによってそれらの値がどちらの領域に属するかを判定する

ことで客観性のあるメモリアクセス機構選択最適化手法の一つを示すことができた.

今後の課題は，提案指標を用いた疎行列の網羅的調査，提案指標を用いた自動チューニング機構つきの疎行列ライブラリの構築などである.

**謝辞** 本研究の一部は総務省戦略的情報通信研究開発推進制度(SCOPE)の一環として行われたものである.

## 参考文献

1) 平木 : "[招待講演]将来の HPC アーキテクチャ", 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 2012 (HPCS'12), pp.163-167, Jan.2012.
2) X. Yang, S. Parthasarathy, P. Sadayappan : "Fast sparse matrix-vector multiplication on GPUs: implications for graph mining", Proc. VLDB Endowment, Vol.4, No.4, pp.231-242, Jan. 2011.
3) Tim Davis : " The University of Florida Sparse Matrix Collection", http://www.cise.ufl.edu/research/sparse/matrices/.
4) 冨森，田邊，小郷，高田，城 :“大規模疎行列やメモリシステムの特性を考慮した高速な汎用疎行列ライブラリ実現に向けて”, 先進的計算基盤システムシンポジウム(SACSIS’12)ポスター, pp.65-66, May 2012
5) 冨森，田邊，小郷，高田，城 :“疎行列のキャッシュへの適合性分類に関する予備評価”, 情報処理学会研究報告 2012-HPC-135, Aug. 2012
6) 田邊，Nuttapon，中條，小郷，高田，城 :“不規則型応用を加速するメモリアクセラレータ ー Exa FLOPS マシンの文脈から”, 情報処理学会研究報告 2011-HPC-132，Nov. 2011.
7) N. Tanabe， Y. Ogawa， M. Takata， K. Joe : " Scaleable Sparse Matrix-Vector Multiplication with Functional Memory and GPUs", Euromicro PDP2011, Feb.2011
8) 田邊，小郷，小川，高田，城 : "Gather 機能を有するメモリアクセラレータの疎行列計算への応用", 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 2012 (HPCS'12), pp.32-41, Jan.2012.
9) 田邊，堀，Nuttapon，中條 : "Gather 機能を有する Hybrid Memory Cube の FPGA を用いた予備評価", 情報処理学会 HPC 研究会, Vol.2010-HPC-133, Mar. 2012.
10) 南，井上，堤，前田，長谷川，黒田，寺井，横川 :"「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 2012 (HPCS'12), pp.32-41, Jan.2012.
11) 櫻井，直野，片桐，中島，黒田 : "OpenATLib：数値計算ライブラリ向け自動チューニングインターフェース", 情報処理学会論文誌コンピューティングシステム, Vol.3, No.2, pp.39-47，2010.
12) 久保田，高橋 : "GPU における格納形式自動選択による疎行列ベクトル積の高速化", 情報処理学会 HPC 研究会, Vol.2010-HPC-128, Dec. 2010.
13) D.B.Heras，J.C.Cabaleiro，F.F.Rivera，”Modelng data locality for the sparse matrix-vector product using deistance measures”， Parallel computing ， Vol.27,pp.897-912，2001
14) Perarnau：”Toward Automated Cache Partitioning for the K Computer”，HPC 研究会 Oct. 2012
15) P.J.Denning，: The Working Set Model for Program Behavior, Comm. ACM, Vol.11,No.5，pp323-333，1968
16) P.J.Denning，S.C.Schwartz : Properties of the Working-Set Model, Comm. ACM，Vol.15,No.3，pp.191-198，1972
17) Matson, 1970
18) 田中:“メモリ参照の局所に着目したソフトウェア性能評価手法”，コンピュータソフトウェア，Vol.16,No.2(1999),pp.32-46.
19) 田中：“メモリ参照の局所性に関する定量的な評価”，HPC 研究会，Aug.1996
20) K. Beyls and E. D’Hollander. Reuse distance as a metric for cache behavior. In Proceedings of the IASTED Conference on Parallel and Distributed Computing and Systems, 2001
21) Ding, Zhong : ”Predicitind Whole-Program Locality through Reuse Distance Analysis”，PLDI’03 pp.245-257，June.2003
22) P. Petoumenos, G. Keramidas, H. Zeffer, S. Kaxiras, and E.Hagersten. Modeling Cache Sharing on Chip Multiprocessor Architectures. Proc. of the International Symposium on Workload Characterization, 2006.