Vol.2013-HPC-140 No.27
2013/8/1

Graph500 の Hybrid 解法に内在する局所性

田邊昇[†1]　冨森苑子[†2]　高田雅美[†2]　城和貴[†2]

Graph500 はビッグデータ解析や疎行列処理のベンチマークとも言われており，近年注目を集めている．ビットマップ式の訪問管理方式や，疎行列の一部だけアクセスする Beamer の Hybrid BFS アルゴリズムの公開により，今後はこれらを用いることがデファクトになる流れにある．本研究ではこれら 2 つを取り入れた最新の Graph500 の OpenMP 参照実装 Pre release 版のメモリアクセス特性を，局所性の観点から解析した．その結果，従来法よりもメモリアクセス回数，ミス時のダメージの大きさを示す列インデックス列の空間的局所性指標，ミスの起きにくさを示す時間的局所性(平均ワーキングセットサイズ)が劇的に変化してしまっており，メモリアクセス回数削減率や時間的局所性がグラフサイズに比例して楽な状況になる傾向を確認した．つまり，従来版 Graph500 が従来の計算機では苦手な処理である疎行列ベクトル積系応用全般を代表し，計算機の発展を促進する意義のあるベンチマークであったのに対し，Hybrid 解法を入れた Graph500 は BFS の最適実装のみの特性を測るベンチマークに変質することが判った．さらに処理時間のホットスポットから BFS を行う関数が消え去り，実質的に検索の速度を測るベンチマークではなくなることが判った．ベンチマークの本来の意義を考慮して，それを損なわない適切な実施ルール作りを行なうべきことを提唱する．

Access Locality in Hybrid Solution of Graph500

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

Graph500 is a benchmark suite for big data analysis and sparse matrix processing which receives attention in these years. Since the use of a bitmap to mark the visit of vertices and Beamer's Hybrid BFS algorithm to make a part of a sparse matrix be accessed are reported, we believe that such methods are to be a de facto standard. In this paper, from the viewpoint of reference locality, we analyze the memory access characteristics of pre-release version implemented with OpenMP and the two methods for the latest Graph500 reference code. As the results, we confirm that the number of memory accesses, a spatial locality metric of row index that shows performance degradation in the case of cache misses, and temporal locality (average working set sizes) that shows the resistance to cache misses, change drastically. We confirmed that the reduction rate of the number of memory accesses and temporal locality are accelerated in proportion to the target graph sizes. It means that the Graph500 with Hybrid method is another benchmark suite to measure the characteristics of optimized BFS implementation, while the conventional Graph500 represents most applications with sparse matrix vector multiplications that are not efficiently performed on conventional computer systems to require the improvement of computer systems. Furthermore, the hot spots against processing times do not include the functions to perform BFS, and we find it is no longer the benchmark for search speed practically. To investigate the essential meaning of the benchmarks, we propose appropriate and practical rules that do not damage it.

1. はじめに

スーパーコンピュータの性能評価指標としては永年 Top500[1]ベンチマーク(密行列の LU 分解)が用いられてきたが，その指標が疎行列の反復解法を主体とする多くの技術計算の実状からかけ離れているため，問題視されることが多くなってきた．特に，Top500 が近年のコンピュータ製造上のネックになっていない演算性能に偏った指標になっていることが問題である．その結果，グラフ処理に対する社会的ニーズの高まりとあいまって， Graph500[2]ベンチマークが Top500 を補う評価指標として登場してきた．Graph500 はビッグデータ解析や疎行列処理のベンチマークとも言われており，近年注目を集めている．

Graph500 は Top500 と同様に毎年 2 回のランキングが発表されるため，HPC コミュニティにおいても Graph500 ランキングの国際競争が激しさを増してきている．その中核部分である BFS（幅優先探索）ループの電力効率を競う Green Graph500[3]も開始されている．

歴史のある Top500 のワークロードについては様々な研究の蓄積がなされており，最適化手法は十分に確立されている．一方，歴史の浅い Graph500 のワークロードの解析の研究はまだ数少ない．従来は主に並列化や通信を中心にした最適化の研究がなされてきた．グラフ解析ワークロードと従来型のコンピュータとの不適合は様々な観点で存在するが，その一つとして，ランダムアクセスに伴うキャッシュメモリの効率低下問題がある．この問題は Gather 機能付きメモリシステム[4]-[10]を軸に田邊らが取り組んできた課題であるが，田邊らはこの観点から Graph500 に関して行った研究[11][12]を発表した．それらでは空間的局所性の観点から Graph500 が解析されている．ただし，解析対象は古典的なアルゴリズムを採用している当時の参照実装のアクセスパターンに対するものであった．

一方，Graph500 のアルゴリズムは現在も発展途上にある．このためベンチマーク結果(ランキング)がアルゴリズムの

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

ⓒ 2013 Information Processing Society of Japan

差なのか, ハードウェアの差なのかが不明瞭な状態である．Top500 が DGEMM(倍精度行列積)の実効性能(ハードウェア的には演算器とキャッシュの性能)を主に測るベンチマークであるのに対し, 現状の Graph500 は何を測っているのかわからない状態にある．この状態は，過渡的なものであり，やがてアルゴリズム上の改良は優れたものに収束し，年々進化を遂げるハードウェアの差を測るものになっていくと思われる．現状で優れたアルゴリズムとして広く認識されているのは Beamer らが SC12 で発表した Hybrid BFS[13]である．本稿執筆時点で Graph500 公式 web サイトの参照実装配布ページ上からもリンクされている別サイト(Graph 500 project on Gitorious[14])で Pre-release 版が公開されている．この OpenMP 向けソースコードには Beamer の Hybrid アルゴリズムが適用されている．既に Pre-release 版が推奨版であることがアナウンスされており, 今後, Hybrid アルゴリズムが主流として用いられていく流れにあると考えられる．

本研究では，上記の Beamer の Hybrid アルゴリズムが適用された参照実装の OpenMP ソースコードをもとに，そのアクセスパターンについて解析を行なった．

本論文の構成は以下の通りである．第 2 章ではターゲットとなるアプリケーションである Graph500 とその Hybrid アルゴリズムについて紹介する．第 3 章では上記においても問題となる不連続アクセスに伴うキャッシュの問題について述べる．第 4 章では上記の問題の度合いを定量的に表す局所性に関する指標について紹介しつつ，Hybrid アルゴリズム向けの修正を提案する．第 5 章では提案指標を基にした評価について述べる．第 6 章で関連研究を紹介し，第 7 章でまとめる．

2. Graph500 とその Hybrid アルゴリズム

2.1 Graph500 ベンチマーク

近年，社会的ニーズとしては Web，ソーシャルネットワークサービス(SNS)の爆発的普及に伴い, グラフ解析のニーズが高まってきている．さらに，嗜好分析やリコメンデーション，車・電話・機械についた GPS やスマートメータなどの大量かつ多様なセンサからの時系列データの解析に基づくリアルタイム制御など，ビッグデータ解析のニーズが高まりを見せている．それらの多くはグラフ解析にカテゴライズされる．

グラフ解析のワークロードの特徴は，例えばある web サイトがどの web サイトをリンクしているという参照関係のような，大規模で不規則的なネットワーク構造を反映した大規模なランダム疎行列の処理に帰着される．これは Irregular processing にカテゴライズされる処理であり, 従来型のコンピュータでは効率的に処理することが困難とされてきた．しかし，この種のアプリケーションを効率よく実行できることは将来のコンピュータシステムにおける重要な要求となってきている．

Graph500[2]ベンチマークは上記のようなグラフ処理の特徴への適合性を測るベンチマークであるとともに，Top500[1]を補う評価指標として 2010 年 6 月の ISC'10 にアナウンスされ，同年 11 月の SC10 から継続的に実施されてきた．毎年 2 回（ISC および SC において）ランキングが発表されるため，HPC コミュニティにおいても Graph500 ランキングの国際競争が激しさを増してきている．

Graph500 ベンチマークのワークロードは Level synchronized BFS(レベルごとに同期した幅優先探索)である．Level synchronized BFS はグラフの隣接行列と列ベクトル(CQ に相当)の疎行列ベクトル積(SpMV)によって NQ を求め，次のレベルではそれを CQ として同様の処理を繰り返す処理ととらえることができることが明らかになっている．[22] ここで CQ は Current Queue，NQ は Next Queue の略である．CQ の初期値となる非零要素は重複せずランダムに与えられる 64 個の節点が入る. レベルが進むごとにその個数は一旦増加後，探索終了した節点の増加によって減少して，完全に無くなったときが探索の終了を意味する．各節点の探索済か否かを記録する配列への読み書きが，ランダムな不連続アクセスとなる．

グラフの隣接行列はクロネッカーグラフ[20]の隣接行列であり，非零要素の配置にランダム性が強いとともに，行内の非零要素数の変動幅が大きい．このため，0 パディングによって行内の非零要素数を一定とみなしてソフトウェアパイプライニングを行う最適化[21]はうまく動作しない．

並列処理における最適化手法としては隣接行列の二次元分割によって通信相手を行方向の分割数や列方向の分割数に減少する方式を多数のノードを用いている実装では殆どのグループが用いているものと考えられる．

Graph500 の中核部分である BFS ループの電力効率を競う Green Graph500[3]も 2012 年 11 月より開始される．このベンチマークでは Graph500 の前処理や後処理を除いた中心部分の処理速度と消費電力のバランスが求められる．このため, 電力や逐次処理部分の最適化をおろそかにしたまま, 単純に PC のノード数を増やしてスケールアウトさせるという手法では Graph500 以上に Green Graph500 では上位にランクされるのは難しいと考えられる．BlueGene/Q など，徹底的に電力あたりの処理性能の高さを追求したシステムが有利である．

2.2 近年主流のアルゴリズム

近年，ノード内の処理に関するアルゴリズムとして，Beamer らが SC12 で発表した Hybrid BFS [BeamerSC12]が主流になってきている．本稿執筆時点で Graph500 公式 web サイトの参照実装配布ページ上からもリンクされている Pre-release 版が別サイト(Graph 500 project on Gitorious[14])で，この Beamer の Hybrid アルゴリズムが適用された参照実装の OpenMP ソースコードが公開されている．既に

Pre-release 版が推奨版であることがアナウンスされており，今後，Hybrid アルゴリズムが主流として用いられていく流れにあると考えられる．

このアルゴリズムの特徴は前節で説明した従来手法（Top down アプローチ）と，その逆方向に探索を行なう Bottom up アプローチを切り換えながら処理を進める点にある．Bottom up アプローチは訪問済みの節点が検出されるとそこで Break し，それ以上その節点と共通の節点と接続している他のエッジの先にある節点の探索を省略する．この動きによってアクセスされるエッジが特に次数が大きな節点において多く省略され，結果としてアクセス回数が減少し，局所性が高まり，高速化が達成される．序盤のステップと終盤のステップでは Top down アプローチで動作するが，多くの実行時間を消費する中盤のステップでは Bottom up アプローチで動作する．よって，全体の高速化の観点からは Bottom up アプローチの高速化が重要である．

なお，国内でも安井らが Hybrid BFS の改良版を開発し，更なる高速化を達成し，2013 年 6 月の Green Graph500[3] の小規模カテゴリで上位をほぼ独占している．しかし，筆者が知る限りにおいてはソースコードが未公開であり，Beamer のように詳細が論文化されていないため現時点では主流にはなっておらず，それゆえ汎用 CPU1 ノードの上位ランキングを独占している状態にある．いずれは他の実装者が追いつき，公開され主流になる可能性が高いと考えられる．

安井が共著になっている岩渕らの論文[16]には安井の手法が若干記載されている．そこには，グラフデータを Top down アプローチ用と Bottom up アプローチ用に別々に CSR 形式で持った上で，分割方法を工夫することで，並列実行時の訪問済フラグ書込み時のアトミックオペレーションを排除しているとされる．平櫛らの GPU クラスタへの実装[17]においても，連続した頂点を同じスレッドが処理するような分割によりアトミックオペレーションを排除している．

筆者らは，Graph500 の上位にランクされる実装，もしくは将来の主流アルゴリズムではアトミックオペレーションが排除されると考える．そのような展望を持ちつつも，ソースコードの入手性が良く，現時点でのデファクトに一番近いコードという観点から，本研究ではアトミックオペレーションを使用している Hybrid BFS を用いた参照実装 OpenMP 版を評価対象として取り上げることとした．

3. Graph500 のメモリアクセスとキャッシュ

本章では Graph500 におけるメモリアクセスにおけるキャッシュの問題について述べる．

3.1 不連続アクセスによるキャッシュの問題

キャッシュアーキテクチャでは間接参照などの不連続アクセス時にキャッシュラインの利用効率が低下する．例えば，Graph500 ベンチマークの参照実装における疎行列の保持方法には CSR 方式（CRS 方式と呼ばれることも多く，これと同等の方式）と CSC 方式の二つのデータ構造による実装が示されている．これらのデータ構造で格納された疎行列とベクトルの積を計算する際には，ベクトルを格納した配列に対するインデックス配列を引数とした間接参照が発生する．グラフ解析アプリケーションにおいては疎行列のランダム性が高い．このため，従来のキャッシュベースな CPU・GPU や従来のメモリを用いる場合は，上記の間接参照における図 1 に示すような問題の発生が顕著になると考えられる．

(1)インデックスはパッケージ間配線を通過するため，メモリバンド幅を消費するとともに，大きな電力も消費する．
(2)ライン単位の転送により，有効データが少ないゆえのパッケージ間配線バンド幅と電力の浪費が発生する．
(3)有効データが少ないゆえのキャッシュエントリの浪費と，それに伴うリプレースの多発やヒット率の低下が発生する．
(4)有効データが少ないゆえの TLB エントリの浪費とヒット率の低下が発生する．

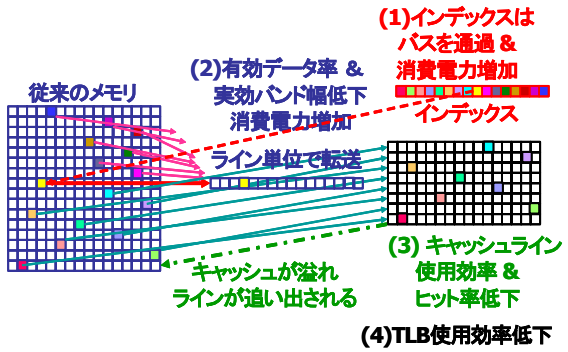

図 1　間接参照におけるキャッシュの問題
Figure 1　Problems caused by indirect accesses on conventional systems.

3.2 Hybrid アルゴリズムにおける不連続アクセス

Bottom up アプローチにおいて検索の方向が逆方向になり，メモリアクセス列への動的な中断が入るようになる．しかし，Graph500 において検索するグラフの構造が不規則であるため，不規則なアクセスパターンが完全に規則的になるわけではない．後述するように定量的な評価が必要であるが，前節で述べたミスが起きる状況でのキャッシュの問題は根本的には解決されていないと考えられる．

graph500 参照実装の Pre release 版では，検索方向を逆にしただけでなく，列インデックスによる間接参照に相当する部分でアクセスする対象として訪問管理ビットマップ[15]を用いる改良が入っている．この技法はキャッシュの観点からは定性的には一長一短である．訪問記録として使用するメモリ量が減るという観点からはキャッシュの効率

を上げる方向に働く．一方，アクセスのランダム性が高いとライン内の 1 データの占有率の低さがキャッシュの効率を下げる方向に働く．例えば，ビットマップのサイズがオンチップのキャッシュには載りきらないような SCALE での実行ではキャッシュミスが多発し，広大なメモリ空間への完全なランダムアクセスに近い状況だと 128 バイト(1024bit)の典型的なライン内で 1 ビットしか利用しないという極めて効率の悪いメモリアクセスが発生すると考えられる．マイナス方向の現象はグラフのサイズが大きくなるに従って顕著になると考えられるが，グラフ構造のランダム性(アクセスの局所性)およびビットマップサイズとキャッシュ容量の関係性が合成された結果としてメリット・デメリットが決まるため，定量的な評価が必要である．

4. 疎行列の局所性に関する指標

4.1 Top Down アプローチ向けの空間的局所性指標

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第一の指標として「列インデックス列の空間的局所性」を提案[18]した．本指標はキャッシュミス時のダメージの大きさを表すものである．ミス率が高い状況下ではこの指標が実行性能を支配する．

図 2 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 CSR 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．」

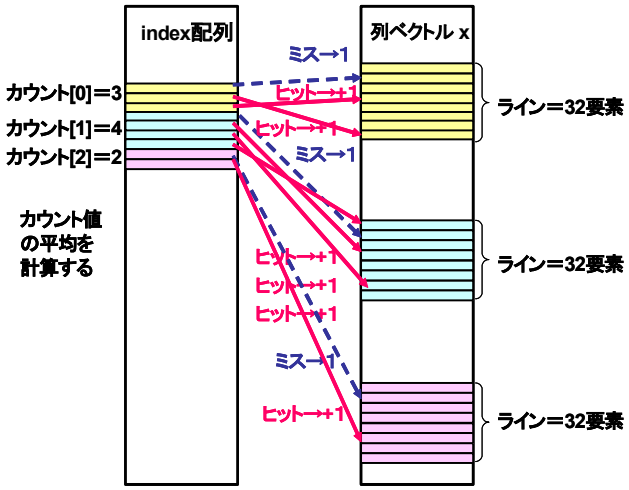

図 2　列インデックス列の空間的局所性[18]
Figure 2　Spatial locality of column index sequence

上記の値は，1 本のラインしかないキャッシュに対して疎行列ベクトル積を CSR 形式の疎行列に対して行なう際の列ベクトルをアクセスする際のキャッシュラインあたりにいくつの有効データが存在するのかという平均値を与える．その逆数は上記のアクセスに対するメモリバンド幅が，どれ位薄まってしまうのかを意味する．ただし，実際のキャッシュは多数のラインを用意することで時間的局所性を引き出すので，疎行列とキャッシュの相互作用の一面しか見ていないことから，上記の平均値は実機特性を厳密に表すものではなく，近似値である．

なお、上記の定義で下位 5bit としているのは 128 バイト(GPU 等の一般的なキャッシュラインサイズ)のラインの中に存在する 32(2^5)個の 4 バイトデータのいずれかへのアクセスは，キャッシュがヒットすることに対応している．

4.2 Bottom up アプローチ向けの空間的局所性指標

Bottom up アプローチで動く部分のメモリアクセスパターンは疎行列ベクトル積とは異なる．最大の違いはカーネルループの中で分岐があることにある．具体的にはビットマップ上で 1 がセットされている（つまり，訪問済みの）節点が検出されるとそこで Break し，それ以上その節点と共通の節点と接続している他のエッジの先にある節点を見に行かなくなる．ゆえに，前節で述べた CRS 形式の列インデックスをトレースの代用とすることは誤差が大きくなるため，列インデックスのトレースを取ることで，アクセスパターンの挙動を正確に把握すべきである．

なお, Beamer のアルゴリズムを反映した graph500 参照実装の Pre release 版では，列インデックス(8 バイト)による間接参照に相当する部分でアクセスする対象としてビットマップデータを用いている．アクセスされるデータの型は int（4 バイト）ではなく bit(1 ビット)である．つまり，128 バイトのキャッシュラインには 1024(2 の 10 乗)個のビットデータが存在する．よって，前節の定義における下位 5bit という部分を下位 10bit に置き換えたものを用いて，空間的局所性を判定すべきである．

4.3 時間的局所性の指標

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第二の指標として「列インデックス列の時間的局所性」の併用を提案[19]した．本指標は疎行列ベクトル積を実行する際の間接参照の際のキャッシュミスのしやすさを表すものである．

具体的に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．ここでいう参照間隔とは，あるデータを有するキャッシュラインをアクセスし，他のキャッシュラインを何回アクセスした後に，注目したデータを有するキャッシュラインを再びアクセスするかということを意味する．ワーキングセットサイズはどれくらいのキャッシュ容量を準備すれば再利用性を抽出できるかという目安を与える．

ⓒ 2013 Information Processing Society of Japan

5. 評価

本章では，Beamer のアルゴリズムを反映した graph500 参照実装の Pre release 版のメモリアクセス列のキャッシュメモリへの適合性を，局所性の観点から評価を行う．評価環境を表 1 に示す．

表 1　評価環境

Table 1　Evaluation environment.

CPU	Intel®Xeon®*CPU　W3565 @ 3.2GHz (コア数 4，スレッド数 8)
キャッシュ	L1 = 32KB, L2 = 256KB, L3 = 8MB
主記憶	DDR3×3 チャネル, 25.6GB/s, 6GB
OS distribution	Fedra17
OS カーネル	Linux カーネル 3.3.4
コンパイラ	gcc4.4
プロファイラ	Intel® Vtune Amplifiler XE 2013 update9
Graph500	Pre release 版(tar ボール生成日 6/20)

本評価において，ベンチマークを実行するプログラムによって，グラフ生成において枝数が頂点数の 16 倍となるようなクロネッカーグラフを生成する．graph500 ではこの生成過程の時間は対象外になる．次に，生成された枝リストからグラフデータ構造に変換する．その際，生成された疎行列を表 2 に示す．測定環境の主記憶容量で実行可能な SCALE21 までの測定を行なった．

表 2　評価に用いた疎行列

Table 2　Experimented matrices

SCALE 値	*非零要素数*	*行数*
10	20,998	1,024
11	45,536	2,048
12	97,010	4,096
13	203,826	8,192
14	426,578	16,384
15	883,126	32,768
16	1,818,824	65,536
17	3,730,586	131,072
18	7,609,740	262,144
19	15,481,872	524,287
20	31,398,208	1,048,576
21	63,538,102	2,097,152

5.1 GTEPS 値

Graph500 は実行させると最後にいくつかの数値を出力するようになっており，その中に GTEPS 値がある．ランキングはこの GTEPS 値を競うものである．Graph500 で扱う問題のサイズはグラフの頂点数＝2^{SCALE} であるような SCALE 値を用いて表す．測定環境で OpenMP 版 graph500 を逐次実行時および並列実行時の SCALE 値と GTEPS 値の関係を図 3 に示す．

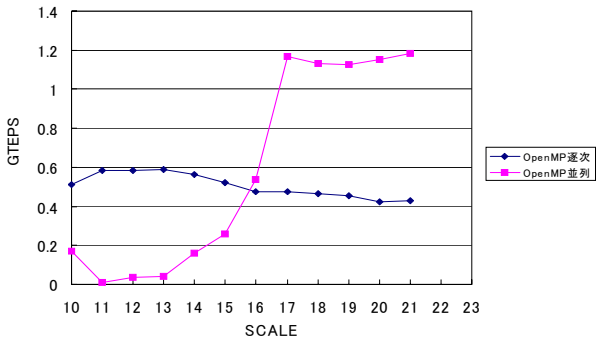

図 3　SCALE と GTEPS 値の関係（Graph500 OpenMP 版）

Figure 3　Relation between SCALE and GTEPS (graph500 OpenMP version).

結果として，SCALE が小さい時は逐次実行の方が高速であるが SCALE16 を境に並列実行の GTEPS 値が上回るようになる．ただし SCALE17 付近で並列実行の伸びが飽和する．これは L1 キャッシュからの溢れ，または L2 キャッシュからの溢れがこの SCALE から始まり，アクセスするようになった階層のキャッシュのスループットがボトルネックになっていると考えられる．この測定だけからはどこが全体の性能を決めているかというところまではわからない．

8 スレッドで実行されている割にはあまり高速化しない．これは graph500 がメモリネックのベンチマークであり，ソケットが増えない場合は外部メモリバンド幅または L3 キャッシュバンド幅が変わらないため，演算のみ高速化しても高速化は鈍る．

5.2 Hybrid 化に伴うホットスポットの変化

Intel Vtune Amplifiler XE 2013 によると実行時間の長い関数(Hot spot)を簡単に見つけることができる．Top down アプローチおよび Hybrid による実行の際の Graph500 計時対象部の Hot spot 上位 5 位を表 3 に示す．従来法（常時 Top down アプローチが選択されるようにしたもの）では BFS を行なう部分に属する関数 *int64_fetch_add* が Hot spot の 4 位に入っている．しかし Hybrid ではそれを含む BFS 関連の関数は Hot spot 上位 20 位から消えてしまった．

表 3　ホットスポットの関数名

Table 3　Function names of hot spots

順位	Top down 固定	Hybrid
1	mod_mac	pack_vtx_edges
2	pack_vtx_edges	scatter_edge
3	mod_mul	setup_deg_off
4	*int64_fetch_add*	mod_mac
5	scatter_edge	mod_mul

* Intel，Xeon は，米国およびその他の国における Intel Corporation の商標です．

ⓒ 2013 Information Processing Society of Japan

5.3 ビットマップのサイズ

SCALE 値とビットマップのサイズの関係を図 7 に示す．測定環境の L3 キャッシュ容量は 8MB(64Mbit)である．測定の範囲ではビットマップ全体が L3 キャッシュに収まってしまうと考えられる．つまり，訪問先の管理をビットマップで管理する方式を導入してしまうと，他にキャッシュ容量を消費する部分がないならば, SCALE28 程度までは 8 MB のキャッシュを搭載している汎用 CPU を用いて，ビットマップ全体がほぼ常時 L3 キャッシュの中に入ってしまうことになると考えられる．CPU の処理能力とのバランスの上で L3 キャッシュのバンド幅が十分にある場合には，その際に測定している部分はアプリケーション本体部分(この場合 BFS 検索時)のメモリ全体へのランダムアクセスとは異なり，この場合，エッジリストからグラフデータ構造の作成などのベンチマークになることを意味する．

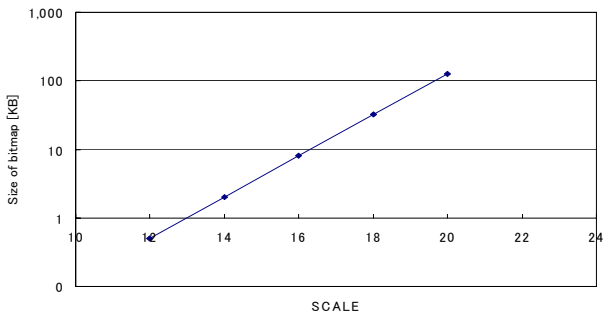

図 4　SCALE とビットマップのサイズの関係（Graph500 OpenMP 版）

Figure 4　Relation between SCALE and size of bitmap (graph500 OpenMP version).

5.4 Hybrid 化に伴うアクセス回数の変化

SCALE 値と Top down アプローチおよび Bottom up アプローチによるビットマップアクセス回数比の関係を図 5 に示す．2 つのアプローチの切り替え条件を常に片方になるように固定することによって，それぞれのアプローチのビットマップへのアクセス回数を測定した．Bottom up アプローチでは Frontier を発見するとアクセスループを中断して抜けることにより，それ以降に予定されていたアクセスを省略する．このため，従来の Top down アプローチと比較して Bottom up アプローチではメモリアクセス回数が約 1/8 に減少している．

この変化は「BFS を実行する」という問題設定(現状のルール)からは逸脱していないものの, 疎行列ベクトル積と同等だったメモリアクセスに関する負荷を，それとは全く異なる軽い負荷の問題に置き換えてしまったという見方もできる．graph500 が PageRank などの疎行列ベクトル積と同等なメモリアクセス要求を有するアプリケーションや，Top500 で拾い切れなかった数値処理をも統合するベンチマークを志向するものであったり，そのような従来の計算機には苦手な処理を抽象化して計算機アーキテクチャの発展を促進することが目的であったならば，この種の質的変化のベンチマークへの導入は好ましいものではないと考える．

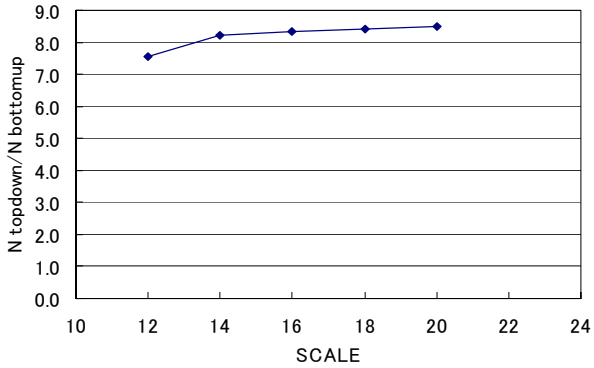

図 5　SCALE とビットマップアクセス回数比の関係（Graph500 OpenMP 版）

Figure 5　Relation between SCALE and ratio of the number of bitmap accesses (graph500 OpenMP version).

5.5 Hybrid 化に伴う空間的局所性の変化

SCALE 値と Top down アプローチおよび Bottom up アプローチによるビットマップアクセスの空間的局所性指標値(ライン内の有効データ数)の関係を図 9 に示す．Top down アプローチでは SCALE の増加とともに空間的局所性が単調減少し，絶対値的にもライン内に 1 個程度まで減少していた．それに対して Bottom up アプローチでは有効データ率がさらに低いことが判明した．

ただし，この項目が性能を左右するのであれば Bottom up アプローチの方が若干低速になるが, L3 キャッシュに全部載ってしまうビットマップの導入により，測定範囲の SCALE ではビットマップへの外部メモリアクセスがほとんど排除されているためにビットマップ化によって両者の差は顕在化していないと考えられる．

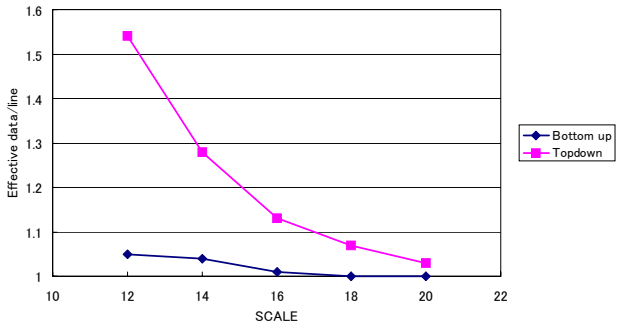

図 6　SCALE とライン内の有効データ数の関係（Graph500 OpenMP 版）

Figure 6　Relation between SCALE and average number of effective data per line (graph500 OpenMP version).

ⓒ 2013 Information Processing Society of Japan

5.6 Hybrid 化に伴う時間的局所性の変化

SCALE 値と Top down アプローチおよび Bottom up アプローチによる時間的局所性指標値(平均ワーキングセットサイズ)の関係を図 6 に示す．SCALE20 において平均ワーキングセットサイズは 278B に過ぎない．その意味は 1 度ミスをしたらその後は 2 つ後までの 3 本のキャッシュラインをアクセスするのでプリフェッチが効き、ほぼ常にキャッシュはヒットする状態になってしまったと言い換えられる．これは本来キャッシュが効きにくかった問題が，全く別種の問題に置き換わってしまったということを意味する．

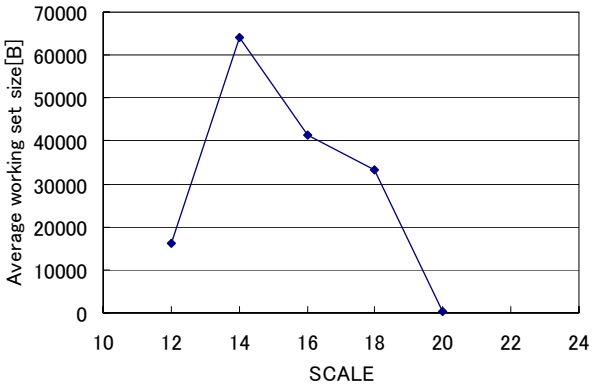

図 7　SCALE と平均ワーキングセットサイズの関係
（Graph500 Hybrid 解法）

Figure 7　Relation between SCALE and average working set size (Graph500 Hybrid version).

6. 関連研究

近年，ノード内の処理に関するアルゴリズムとして，Agrawal らが SC10 で発表したビットマップ式の訪問管理方式[15]や，Beamer らが SC12 で発表した Hybrid BFS [13]が主流になってきている．本稿執筆時点で Graph500 公式 web サイトの参照実装配布ページ上からもリンクされている Pre-release 版が別サイト(Graph 500 project on Gitorious)で，これらの 2 つのアルゴリズムが適用された参照実装の OpenMP ソースコードが公開されている．既に Pre-release 版が推奨版であることがアナウンスされており，今後，Hybrid アルゴリズムが主流として用いられていく流れにあると考えられる．

国内では安井らが Hybrid BFS の改良版を開発し，更なる高速化を達成し，2013 年 6 月の Green Graph500 リスト[3]の小規模カテゴリで上位をほぼ独占している．しかし，筆者が知る限りにおいてはソースコードが未公開であり，Beamer のように詳細が論文化されていないため現時点では主流にはなっていない．安井が共著になっている岩渕らの論文[16]には安井の手法が若干記載されている．そこには，グラフデータを Top down アプローチ用と Bottom up アプローチ用に別々に CSR 形式で持った上で，分割方法を工夫することで，並列実行時の訪問済フラグ書込み時のアトミックオペレーションを排除しているとされる．平櫛らの GPU クラスタへの実装[17]においても，連続した頂点を同じスレッドが処理するような分割によりアトミックオペレーションを排除している．

Suzumura[23]らは Graph500 課題の参照実装の解析を様々な観点から行っている．古い参照実装に関するものであるとともに，キャッシュに関する解析は行なわれていない．上野[22]らはこれを踏まえて，Graph500 課題の隣接行列の 2 次元分割の実装について報告している．Vertex sorting というキャッシュに関する最適化を提案評価しているが効果は 10%程度に過ぎない．これは Hybrid アルゴリズムでもなく，かつ，時間的局所性に関する改善であり，空間的局所性の改善に着目したものではない．田邊[11][12]らは Graph500 の課題疎行列の空間的局所性に関して，上記の Vertex sorting も含めた評価を行っている．ただし，評価対象が従来の参照実装に基づいている．

7. おわりに

本研究では SC12 で Beamer が発表した Hybrid BFS アルゴリズムが適用された Graph500 コードを用い，Graph500 におけるメモリアクセス列のキャッシュメモリへの適合性を，空間的局所性の観点から解析した．

その結果，従来法よりもメモリアクセス回数，ミス時のダメージの大きさを示す列インデックス列の空間的局所性指標，ミスの起きにくさを示す時間的局所性(平均ワーキングセットサイズ)が劇的に変化してしまっており，メモリアクセス回数削減率や時間的局所性がグラフサイズに比例して楽な状況になる傾向を確認した．さらに処理時間のホットスポットから BFS を行う関数が消え去り，実質的に検索の速度を測るベンチマークではなくなることが判った．

一方，Hybrid アルゴリズムの導入は「BFS を実行する」という問題設定(現状のルール)からは逸脱していないものの，疎行列ベクトル積と同等だったメモリアクセスに関する負荷を，それとは全く異なる軽い負荷の問題に置き換えてしまったという見方もできる．メモリアクセス回数が大幅に減少した上に，時間的局所性の傾向も大きい問題ほど楽になるという逆の方向に変わってしまったことが明らかになった．graph500 の存在目的が BFS のみならず，PageRank などの疎行列ベクトル積と同等なメモリアクセス要求を有するアプリケーションや，Top500 で拾い切れなかった数値処理をも統合するベンチマークを志向するものであったり，そのような従来の計算機には苦手な処理を抽象化して計算機アーキテクチャの発展を促進することが目的であったならば，この種の大幅な質的変化のベンチマークへの正式導入は慎重であるべきと考える．

今後の課題としては，キャッシュヒット率関連の評価，アトミックオペレーションを排除可能な割り当て方式を導

ⓒ 2013 Information Processing Society of Japan

入した場合の評価，より大きな SCALE に対する評価，メモリ側の Gather 機能を有する環境での Graph500 の実装と，それに基づく TEPS 値における加速率の評価が挙げられる．

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) Top500 : http://www.top500.org/.
2) Graph500 : http://www.graph500.org/.
3) Green Graph500 : http://green.graph500.org/.
4) N. Tanabe, M. Nakatake, H. Hakozaki, Y. Dohi, H. Nakajo, H. Amano : "A New Memory Module for COTS-Based Personal Supercomputing", Innovative Architecture for Future Generation High-Performance Processors and Systems, pp.40-48 (2004).
5) N. Tanabe, H. Hakozaki, Y. Dohi, Z. Luo, H. Nakajo : " An enhancer of memory and network for applications with large-capacity data and non-continuous data accessing", The Journal of Supercomputing, Vol. 51, No. 3, pp. 279-309 (2010).
6) N. Tanabe, Y. Ogawa, M. Takata, K. Joe : "Scaleable Sparse Matrix-Vector Multiplication with Functional Memory and GPUs", Euromicro PDP2011 (2011).
7) N. Tanabe, B. Nuttapon, H. Nakajo, Y. Ogawa, J. Kogou, M. Takata, K. Joe : "A memory accelerator with gather functions for bandwidth-bound irregular applications", Proceedings of the first workshop on Irregular applications: architectures and algorithm (IAAA'11) in conjunction with SC11，pp.35-42 (2011).
8) 田邊，Nuttapon，中條 : " Gather 機能付き拡張メモリのアクセス性能の評価"，情報処理学会 ARC 研究会, Vol.2010-ARC-192 (2010).
9) 田邊，Nuttapon，中條，小川，高田，城 : " GPU と拡張メモリによる疎行列ベクトル積性能の行列形状依存性軽減"，情報処理学会 HPC 研究会, Vol.2011-HPC-129 (2011).
10) 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.2012-HPC-133 (2012).
11) 田邊，冨森，高田，城 :“ 疎行列のキャッシュ適合性に基づく Graph500 ベンチマークの特性解析”，情報処理学会 HPC 研究会, Vol.2013-HPC-138 (2013).
12) 田邊，冨森，高田，城 :“疎行列のキャッシュ適合性に基づく Graph500 ベンチマークの特性解析”，先進的計算基盤システムシンポジウム（SACSIS 2013）(2013).
13) S. Beamer, K. Asanovic, D. Patterson : “Direction-Optimizing Breadth-First Search”, SC12 (2012).
14) “Gitorious : Working space for the graph500 reference implementation”, https://www.gitorious.org/graph500
15) V. Agarwal, F. Petrini, D. Pasetto, D. Bader : "Scalable Graph Exploration on Multicore Processors", SC10 (2010).
16) 岩渕，佐藤，安井，藤澤，松岡 :“不揮発性メモリを用いた Graph500 ベンチマークの大規模実行へ向けた予備評価”，情報処理学会 HPC 研究会, Vol.2013-HPC-138 (2013).
17) 平櫛，高橋 :“GPU クラスタにおける幅優先探索の高速化”，情報処理学会 HPC 研究会, Vol.2013-HPC-139 (2013).
18) 冨森，田邊，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 HPC 研究会, Vol.2012-HPC-135 (2012).
19) 冨森，田邊，高田，城 :“時間的局所性を考慮した疎行列のキャッシュ適合性”，情報処理学会 HPC 研究会, Vol.2012-HPC-137 (2012).
20) J. Leskovec, D. Chakrabarti, J. Kleinberg, and C. Faloutsos, "Realistic, mathematically tractable graph generation and evolution, using kronecker multiplication", in Conf. on Principles and Practice of Knowledge Discovery in Databases (2005).
21) 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 (2012).
22) K. Ueno and T. Suzumura "Highly Scalable Graph Search for the Graph500 Benchmark", 21st International ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC'12), pp.149-160 (2012).
23) Toyotaro Suzumura, Koji Ueno, Hitoshi Sato, Katsuki Fujisawa and Satoshi Matsuoka, "Performance Evaluation of Graph500 on Large-Scale Distributed Environment", IEEE IISWC 2011 (2011).

ⓒ 2013 Information Processing Society of Japan