大規模グラフ処理ベンチマーク Graph500 の概要と成果報告

鈴村豊太郎 [1,2]，上野晃司 [1]

東京工業大学大学院・情報理工学研究科 [1]　　IBM 東京基礎研究所 [2]

Graph 500 とは，スーパーコンピュータのグラフ処理性能を測定する新しいベンチマークである. スーパーコンピュータのベンチマークでは, 数値計算性能を測る Linpack による Top 500 が有名だが，近年，大規模グラフ処理が，重要性を増しており，Graph500 ベンチマークが広がりを見せている．Graph500 のリファレンス実装は，使用されているアルゴリズムの問題により，分散メモリ環境で大規模にスケールさせることが困難である．そこで，大規模にスケール可能な 2 次元分割に注目した．本稿では Graph500 の概要を述べると共に，我々が Oakleaf-FX スーパーコンピュータ上に実装した、分散メモリ環境において大規模にスケールするための 2 次元分割手法の概要について述べる．また、この手法を基盤として通信圧縮などの最適化を施した結果、 4096 ノード（65536 コア） を用いて頂点数は 2 の 38 乗（2748 億頂点），エッジ数は 2 の 42 乗（4.4 兆辺）のグラフ（Graph500 の Scale 38）に対する BFS(幅優先探索)の計算を 12.56 秒で完了した．TEPS 値は 358.10 GE/s であり，2012 年 6 月に発表されたランキングでは世界 3 位を獲得した

写真 1：東大センターブースにて，筆者は後列向かって左から 1 人目（鈴村），2 人目（上野）

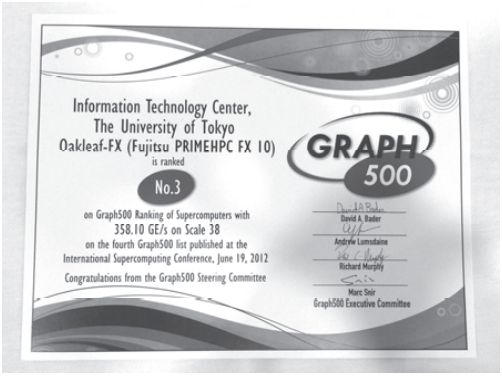

写真 2：Graph500 3 位の賞状

2.　Graph500 ベンチマークの概要と最適化手法

本章では，Graph500 ベンチマークの概要と，分散メモリ環境で動く基本的な分散 BFS アルゴリズム、そして大規模環境において性能をスケールさせるための最適化手法および性能結果について述べる．

2.1.　Graph500 ベンチマークの概要

Graph500 は, 大規模なグラフに対して BFS(幅優先探索)による探索を実行するベンチマークである．単位時間に処理できたエッジ数と，扱える最大問題サイズが評価指標となる．計算インテンシブな Top500 ベンチマークと違い，Graph500 ベンチマークは，データインテンシブなベンチマークである．扱える問題サイズは，グラフの頂点数=(2 の SCALE 乗) であるような SCALE 値で表す．単位時間に処理できたエッジ数は，TEPS (Traversed Edges Per Second) 値で表す．例えば，100 万 TEPS とは，100 万個の枝を持つ連結グラフの BFS が 1 秒で完了した場

合の性能である．

ベンチマークを実行するプログラムは，(a)グラフデータの生成，(b) 計算するのに最適なデータ構造への変換，(c) BFS による探索，(d) 計算結果の検証の４つの部分から成る．ベンチマークの実行順は次のようになっている．最初に (a), (b)によりグラフデータを構築し，グラフから始点を 64 個選ぶ．次に，64 個の始点それぞれに対して順番に，(c) BFS による探索と，(d) 計算結果の検証を行う．複数の始点からの探索を同時に行うことはできない．時間を計測しベンチマークとする部分は，(b)のグラフデータ構造への変換（Kernel 1）と，(c)の BFS による探索（Kernel 2）のみである．(a)では，枝数が頂点数の 16 倍となるようなクロネッカーグラフ[5]を生成する．枝はすべて重みなし，無向辺である．ここで生成されるデータは規則性のない順番で並んだ，枝のリストである．(b)では(a)で生成された枝リストから，隣接行列の CSR (Compressed Sparse Row)や，CSC (Compressed Sparse Column)などのグラフデータ構造に変換する．(c)の BFS は，BFS で辿った頂点の軌跡である BFS 木を出力する．(d)では，この BFS 木が正しいかどうかチェックする．このチェックでは，BFS 木にループがないこと，枝の張っている頂点同士の深さの差が 1 以下であること，などの 5 つのルールを満たしていることをチェックする．

3. 最適化手法- 2 次元分割によるスケーラブルな実装と Oakleaf-FX 上での性能

リファレンス実装は全て 1 次元分割を使っているが，我々の先行研究[2] によって大規模環境では 1 次元分割はスケールさせることが困難であるという知見を得た．そこで，隣接行列を 2 次元に分割するアルゴリズム（2 次元分割）[3] を実装した． まず, プロセッサを P = R x C の 2 次元メッシュ(mesh)に配置する．このメッシュの行を「プロセッサ行」，列を「プロセッサ列」と呼ぶことにする．隣接行列を R*C 個の行と C 個の列に分割し，プロセッサ(i, j)は，隣接行列の A_(i,j)^((1))~A_(i,j)^((C))の C ブロックを担当する．頂点は，R x C 個のブロックに分割し，プロセッサ(I, j)は，j*R+I 番目のブロックを担当する． 1 レベルにつき，Expand と Fold の２段階の通信を行う．各プロセッサは自分の担当する頂点ブロックの CQ(Current Queue)を同じプロセッサ列の他のプロセッサに送信する．これを Expand という．Expand は 1 次元分割の縦分割と同じように，CQ をコピーする通信であるが，隣接行列は横にも C 個に分割されているので，通信は，同じプロセッサ列の他のプロセッサとだけ行う．次に，各プロセッサは CQ と各プロセッサが持っている部分隣接行列から，CQ の隣接頂点を探す．PRED (Predecessor)や NQ(Next Queue)を更新するため，CQ の隣接頂点を，その頂点の担当プロセッサに送信する．この通信を Fold という．PRED を更新するのに，親の頂点が必要なので，Fold では，CQ の隣接頂点と，親頂点（CQ の頂点）の組みを送信することになる．Fold は 1 次元分割の横分割と同じように，CQ の隣接頂点を担当プロセッサに送信する通信である．しかし，２次元分割では，隣接行列の分割方法から，Fold の通信を行う必要のある相手は，同じプロセッサ行の他のプロセッサのみとなる．

２次元分割の利点は，通信で絡むプロセッサ数が少ないことである． 1 次元分割では，２種類の分割方法のどちらも，全対全の通信が必要だったのに対し，２次元分割の場合，Expand では同じ列のノード(R-1)プロセッサと, Fold では同じ行のノード(C-1)プロセッサとしか通信を行わない．よって，通信するプロセッサ数を少なくすることができ，大規模に分散可能になる．この２次元分割の詳細に関しては筆者らの論文[3] を参照して頂きたい.

また、これまで述べた２次元分割による手法を基盤にし、通信圧縮、頂点の並び替えなどの最適化を施した実装 [3] を用いて, Oakleaf-FX スーパーコンピュータシステムにおける性能を測定した. 図 1, 図 2 は 1 ノードあたり Scale 23 による Weak Scaling による結果である. 4096 ノードまで TEPS 値が線形で向上することに成功している.

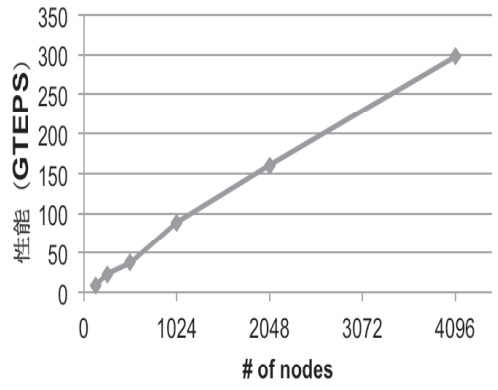

図 1　Oakleaf-FX 上での Graph500 の性能

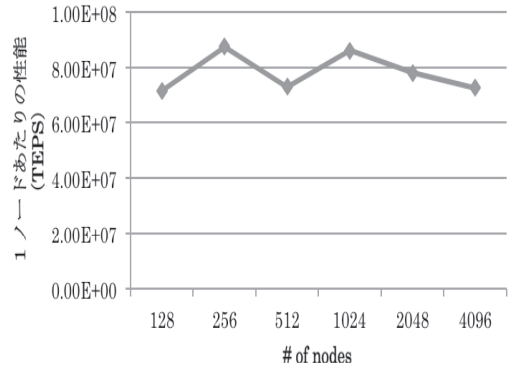

図 2　１ノードあたりの性能（TEPS）

4. まとめ

本稿では 大規模グラフ処理ベンチマーク Graph500 の概要, 大規模分散環境でスケールさせるための２次元分割による BFS の手法および性能について述べた. 2012 年 6 月に発表された Graph500 では, 本稿で述べた最適化手法を東京大学情報基盤センターのスーパーコンピュータ Oakleaf-FX 上で実装し, Scale 38 の大規模グラフの問題に対して 358.10 GTEPS のスコアを達成し, 世界 3 位を獲得した. 今後は更なるアルゴリズム・実装の最適化と、2012 年 11 月のランキングから採用される 2 番目のカーネル「最短経路問題」に関する実装, 性能評価を行う予定である. 本実装に関しては, 科学技術振興機構　CREST プログラムから助成されている研究プロジェクト「ポストペタスケールシステムにおける超大規模グラフ最適化基盤」における成果です. 今後も継続して成果が出ていきますように頑張っていく所存であります. 最後に東京大学情報基盤センターの方々および大規模 HPC チャレンジ制度の関係者の方々にこの場をお借りして感謝致します.

参考文献

[1] Graph500 : http://www.graph500.org/.

[2] Toyotaro Suzumura, Koji Ueno, Hitoshi Sato, Katsuki Fujisawa and Satoshi Matsuoka, "Performance Evaluation of Graph500 on Large-Scale Distributed Environment", IEEE IISWC 2011 (IEEE International Symposium on Workload Characterization) , 2011/11, Austin, TX, US

[3] Koji Ueno and Toyotaro Suzumura, Highly scalable graph search for the Graph500 benchmark, HPDC '12 Proceedings of the 21st international symposium on High-Performance Parallel and Distributed Computing, Pages 149-160, 2012/06, Delft, Netherland