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

岩渕 圭太[†,††] 佐藤 仁[†,††] 安井 雄一郎[†††]
藤澤 克樹[†††,††] 松岡 聡[†]

1. はじめに

近年，SNS 解析，道路ネットワークの経路探索，創薬等の応用で，超大規模なグラフに対する高速処理が求められている．単一の計算ノードでは保持できない規模のサイズのグラフも登場しはじめている一方で，一般に，グラフ処理は参照の局所性が低く，また，メモリアクセスはランダムなものとなるため，全てのデータを DRAM へロードして実行しなければ，妥当な性能で実行することは難しい．しかし，大容量の DRAM の導入は，非常に価格の面でのコストが高く，また，消費電力も増大してしまうため望ましくない．

一方で，近年，フラッシュデバイスなどのような，DRAM と比較するとアクセスレイテンシやスループットなどの性能面で劣るものの，容量あたりの価格，消費電力の点で優れたデバイスが登場し，普及している．このようなフラッシュデバイスを DRAM に対して補助的に利用すれば，単一の計算ノード上の DRAM には収まらない容量のグラフを実行できる可能性があるものの，その具体的手法やどの程度の性能低下が起きるのかなどの定量的な指標は明らかではない．

そこで，DRAM 容量を超えるグラフを扱う手法を提案し，不揮発性メモリと DRAM の容量の比率が性能にどのような影響を及ぼすのかについて予備評価を行った．その結果，一部データを SSD に退避することで DRAM 使用量を 49.8%まで削減でき，性能は 26.6%まで低下した．また，参照され難いエッジデータを退避することで性能の低下を抑えながらさらに DRAM 使用量を削減可能なことを確認し，本手法が有効である事が示唆された．

† 東京工業大学
Tokyo Institute of Technology
†† JST CREST
JST CREST
††† 中央大学
Chuo University

2. 提案手法

まず，Graph 500[1] とは，実グラフと似た性質を持つグラフに対して幅優先探索 (BFS) を行うベンチマークである．その際，グラフサイズは頂点数が 2^{SCALE}，エッジ数は頂点数*$Edgefacter$ で表され，性能は単位時間に処理されたエッジ数である TEPS（Traversed Edges Per Second）として表される．

提案手法は，Hybrid-BFS[2] アルゴリズムを対象とする．Hybrid-BFS アルゴリズムは，各レベル毎にその時訪問している頂点の集合を frontier とし，隣接する未訪問の頂点を次回の frontier として探索を進める Top-down アプローチと，未訪問の頂点側から frontier へのエッジを探し，その頂点を次回の frontier として探索を進める Bottom-up アプローチをハイブリッドに切り替えながら BFS を行う．具体的には探索方向切を切替えるためのパラメータ k を用いて，$frontier \geq \frac{k}{\text{全頂点}}$ の場合に Top-down から Bottom-up に切替え，逆に，$frontier \leq \frac{10k}{\text{全頂点}}$ の時に Top-down に戻る．使用するグラフデータは NUMA ノードを意識し，Top-down アプローチ用に最適化した forward graph と Bottom-up アプローチ用に最適化した backward graph，さらに，探索中に使用するキューやビットマップである BFS データである．

既存手法として，探索の性能に大きな影響を及ぼす BFS データのみを DRAM に乗せ，グラフデータは不揮発性メモリに乗せる手法があるが，この手法では性能が大きく低下してしまう[3]．

そこで我々の提案手法は，全てのグラフデータを退避するのではなく，まず，forward graph を不揮発性メモリに退避し，逐一 DRAM に読み込みながら BFS を行う．さらに，backward graph についても，アルゴリズムの特性上参照され難いエッジを退避し逐一読み込むことで，性能の低下を抑えながら DRAM の使

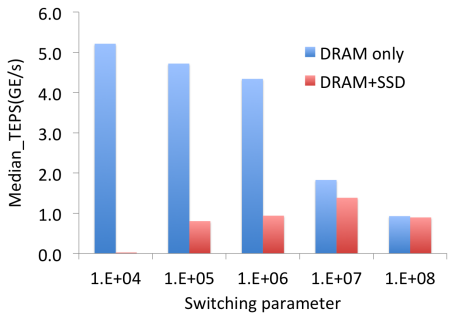

図 1 切替の閾値を変化させた場合の平均 TEPS(GE/s) 値

用量を削減する．

3. 評　　価

評価環境は CPU が AMD Opteron（tm）6172（6 コア，8 ソケット），DRAM 容量が 64GB と 128GB の 2 種類を用意した．DRAM 容量が少ないノードでは SSD（600GB) を搭載している．

まず，DRAM と不揮発性メモリ使用量の比率の変化に伴う性能の変化について評価を行うために，全グラフデータを DRAM に乗せた場合 (DRAM only) と，forward graph を SSD に退避した場合 (DRAM+SSD) での性能比較を行った．なお，実装は forward graph を SSD に保存し，POSIX 準拠の fread() を使用し 1 エッジずつ読み込むようになっている．グラフのサイズは Scale 27，Edge factor 16 であり，データサイズは forward graph は 40.1GB，backward graph は 33.1GB，BFS data は 15.1GB である．

評価を行った結果，DRAM only では最大で 5.2GTEPS であった**図 1**．一方，DRAM+SSD ではパラメータを調整することで 1.E+07 のとき最大で 1.38GTEPS となり，DRAM only の最大値と比較し 26.6%まで性能の低下を抑えることができた．しかし，スレッド間でのロードインバランスが発生しており，また SSD からの読込みも SSD のメモリバンド幅を有効活用できておらず，今後の課題が明らかになった．

次に，DRAM に載せるグラフデータ容量をさらに削減する手法として，backward graph の一部を退避する手法について予備評価を行った．サイズが Scale 16，Edge factor 16 のグラフに対して，次数が同じ頂点毎に参照されたエッジ数の最大値を集計した．その結果，次数が 2,000 を超えると参照されるエッジ数は一定の値以上は増加せず，最大でも次数の 15%程度で参照されている．

表 1 は SCALE 16， Edge factor 16 のグラフに対

表 1 DRAM に残す最大エッジ数を変更した場合の DRAM と SSD の使用量（backward graph）

最大エッジ数	削減後のサイズ	退避したエッジへのアクセス率
100	47%	1.740%
500	73%	0.003%
1000	87%	0.0%

して，DRAM 上に各頂点が DRAM に乗せるエッジ数の上限を調整した場合の backward graph のデータサイズである．例えば，各頂点に対して上限エッジ数を 500 とすれば，それを超えた頂点は不揮発性メモリに退避することでデータサイズは 73 %まで削減できる．一方で，SSD へのアクセスは 0.003%程度しか行われないことがわかり，性能低下を抑えながら DRAM 容量を削減できる見込みがあることを確認した．

4. まとめ・今後の方針

不揮発性メモリを利用することで性能低下を抑えながらより容量の大きなグラフデータを扱うことを目指しており，そこで本稿では DRAM 容量を超えるグラフを扱う手法を提案し，不揮発性メモリと DRAM の容量の比率が性能にどのような影響を及ぼすのかについて予備評価を行った結果を示した．その結果，forward graph を SSD に退避することで DRAM 使用量を 49.8%まで削減でき，性能は 26.6%まで低下した．また，参照され難いエッジデータを SSD に退避することで性能の低下を抑えながら backward graph の一部が退避可能なことを確認し，本手法が有効である事が示唆された．

今後の方針としては,backward graph の一部を実際に 退避した場合の性能評価, その他の NAND Flash デバイスを使用しての性能評価がある．

参　考　文　献

1) Graph500: `http://www.graph500.org/`.
2) Beamer, S., Asanović, K. and Patterson, D.: Direction-optimizing breadth-first search, *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC'12)*, Salt Lake City, USA, IEEE Computer Society Press, pp.12:1—-12:10 (2012).
3) Pearce, R., Gokhale, M. and Amato, N. M.: Multithreaded Asynchronous Graph Traversal for In-Memory and Semi-External Memory, *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC10)*, IEEE, pp. 1–11 (2010).