Vol.2016-HPC-155 No.16
2016/8/9

CPUとDRAMへの電力バジェット配分を考慮したGraph500の性能評価

垣深 悠太[1,a)]　安井 雄一郎[1,b)]　小野 貴継[1,c)]　稲富 雄一[1,d)]　藤澤 克樹[1,e)]　井上 弘士[1,f)]

概要：大規模計算機クラスタの設計において，電力制約下で高い性能を得るためにオーバープロビジョニングと呼ばれるシステム構成法が提案されている．システムの理論ピーク消費電力（すなわち熱設計電力）が供給可能な最大電力を超過することを許容して大量のハードウェアを設置し，その上で，システム稼働時の実効消費電力（実際に消費する電力）が電力制約値を超えないよう制御する．このような電力制約型計算機システムにおいて高い性能を得るためには，アプリケーションの電力特性に基づき，各ハードウェアコンポーネントに対し適切に電力資源を配分する必要がある．一方，アプリケーションに目を向けると，ビッグデータ時代の到来を背景に大規模グラフ解析への関心が高まっている．そこで本稿では，大規模グラフ解析で主要な処理となる幅優先探索に着目し，実機を用いた電力特性解析を行う．そして，CPU とDRAM への電力資源配分が幅優先探索処理の実効性能へ与える影響を調査し，大規模グラフ処理向けの電力資源配分戦略に関して考察する．

1. はじめに

大規模計算機クラスタの開発において，消費電力は重要な設計制約の１つとなっている．一般に，計算機クラスタの理論ピーク消費電力は，搭載された全てのハードウェアコンポーネントにおける熱設計電力の総和として算出される．したがって，供給可能電力が制約として与えられた際には，この電力制約を超過しない範囲でハードウェアコンポーネントを選択しなければならない．そのため，搭載可能なハードウェア量が電力制約により大幅に制限される．この問題を解決する手段として，計算機クラスタ稼働時の最大消費電力（実効最大消費電力）が与えられた電力制約を超過することを許容し，大量のハードウェアコンポーネントを搭載した上で，システム稼働時の実効消費電力（実際に消費する電力）が電力制約値を超えないよう電力キャッピングを適用するオーバープロビジョニング設計法が提案された [8]．オーバープロビジョニング・システムにおいて高い性能を達成するには，アプリケーションの電力特性に応じて計算機が有する CPU や DRAM などのハードウェアコンポーネントに適切に電力資源を配分する必要がある．そのためには，電力制約下でのアプリケーション実行における電力性能特性を明らかにしなければならない．

本研究では，大規模計算機クラスタ上で実行するアプリケーションとして大規模グラフ解析に着目する．ワールドワイドウェブやソーシャルネットワークをはじめとして，神経網や電力網に至るまで点とそれを結ぶ線として表現可能なものは，グラフアルゴリズムを適用した解析が可能である．ビッグデータ時代の到来に伴い大規模グラフ解析の重要性が広く認知され，分散環境における高速化や実装コスト削減などを目的とした様々な研究が行われるようになった [6], [7]．しかしながら，電力性能特性に関する研究は少なく，特に，電力制約ならびに電力資源配分が大規模グラフ解析処理の性能に与える影響は明らかになっていない．

そこで本稿では，大規模グラフ解析において主要な処理となる幅優先探索を対象とした電力性能特性解析を行う．具体的には，CPU と DRAM への個別電力制御が可能な単一計算ノードを評価用プラットフォームとし，大規模グラフ解析ベンチマークとして知られる Graph500 [3] を用いた以下の解析を実施する．

- 問題サイズ依存性解析：電力制約が無い場合に，解くべき問題のサイズが電力性能特性に与える影響を調査．
- スレッド数依存性解析：電力制約が無い場合に，スレッド数が電力性能特性に与える影響を調査．

[1] 九州大学
Kyushu Uniersity
[a)] yuta.kakibuka@cpc.ait.kyushu-u.ac.jp
[b)] y-yasui@imi.kyushu-u.ac.jp
[c)] takatsugu.ono@cpc.ait.kyushu-u.ac.jp
[d)] yuichi.inadomi@cpc.ait.kyushu-u.ac.jp
[e)] fujisawa@imi.kyushu-u.ac.jp
[f)] inoue@ait.kyushu-u.ac.jp

- 消費電力変動性解析：電力制約が無い場合に，グラフアプリケーションの実行時に CPU および DRAM の消費電力がどのように変動するのかを調査.
- 電力資源配分依存性解析：電力制約下において，CPU および DRAM に対する電力資源配分が実効性能へ与える影響を様々な問題サイズにおいて調査.

本稿の構成は以下の通りである．まず，第 2 節ならびに第 3 節にて，本解析で対象としたベンチマークプログラムならびに実験環境の詳細を説明する．次に，第 4 節は非電力制約時，第 5 節は電力制約時を想定した各種解析結果を報告する．そして，第 6 節にて関連研究について述べ，最後に第 7 節で本稿をまとめる．

2. Graph500

Graph500 は，メモリインテンシブなアプリケーションを代表するベンチマークとして 2010 年に発表された [3]．処理内容としては，2 つのパラメータ edgefactor と scale で指定される頂点数 $|V(G)| = 2^{\text{scale}}$ ならびに辺数 $|E(G)| = \text{edgefactor} \times |V(G)|$ の無向グラフ $G = (V, E)$ において幅優先探索を行う．探索対象となるグラフは R-MAT [2] と類似のアルゴリズムにより Graph500 の内部にて生成され，Web ページの参照関係や友人関係などをモデル化したグラフと近い性質（各頂点の次数やクラスタ構造など）を有することが知られている [2]．なお，edgefactor の値を 16 として生成したグラフ上で探索を行った結果がベンチマーク性能値として登録できる．性能を表す指標としては，探索対象グラフのうち，幅優先探索の対象となった成分に含まれる辺の数を 1 回の幅優先探索に要した時間で割った TEPS（Traversed edge per second）が用いられる．

Graph500 の処理内容を Algorithm1 に示す．まず，探索の対象となるグラフを生成する．本グラフは端点の組で表現された辺のリストとして表現される．次に，生成した辺のリストを幅優先探索に適するデータ構造による表現に改める（例えば，代表的な表現として CSR: Compressed Sparse Row が挙げられる）．これらの処理部分はグラフ構築と呼ばれ，Graph500 に性能値を登録する際にはその実行時間をベンチマークの結果として報告する必要がある．グラフを構築した後，次数が 1 以上の頂点から無作為に 64 個の頂点を選択し，各頂点を始点として幅優先探索（グラフ探索と呼ぶ）と結果の検証を行う．幅優先探索の実行時間はベンチマークの結果として報告する必要があり，その結果に基づき TEPS 値が決まる．

3. 実験環境

実験には 表 **1** に示す計算機サーバを用いた．消費電力の測定や制御に関しては独自に開発した専用ライブラリ（RIC ライブラリと呼ぶ）を用いた．RIC ライブラリでは SandyBridge 以降のインテルプロセッサに搭載されている

Algorithm 1: Graph500 の処理内容

```
探索の対象となるグラフを，端点の組として表現された辺のリストとして生成 (グラフ生成)
辺のリストを幅優先探索に適したデータ構造で表現する (グラフ構築)
1 次以上の頂点から 64 個の頂点を選択
foreach 選択された頂点 v do
    v を始点とし幅優先探索を行う (グラフ探索)
    幅優先探索が正しく行われたことを検証
ベンチマーク結果を出力
```

表 **1** 計算機環境

CPU	Intel Xeon E5-2620 (6 コア) × 2
	TDP = 95[W]
メモリ	16GB × 8 (128GB)
OS	CentOS 6.4 64bit (kernel 2.6.32)
コンパイラ	Intel Compiler Version 14.0.0
BIOS 設定	Enhanced Intel SpeedStep Technology (EIST) = Enable
	Turbo Boost = Disable
	DRAM RAPL = mode 0
	Hyper-threading = Enable

Running Average Power Limit (RAPL) [9] を利用している．RAPL では CPU 全体とコア部分，CPU に直接接続された DRAM 消費エネルギーをそれぞれ測定することができる．また，一定時間（デフォルトでは約 1ms）ごとに平均消費電力の制約値を指定することが可能である．本研究では，このような電力キャッピング機能を用いて CPU や DRAM への電力制御を行う．

電力制約下で高い性能を得るためには，電力あたり性能が必要だと考えられる．そこで，本評価では電力あたり性能に優れた Graph500 の実装 [10] を用いる．この実装では，直径の小さなグラフに対して最適化された幅優先探索手法 [1] を採用している．また，探索対象グラフの頂点の探索順に関する最適化や NUMA アーキテクチャを考慮した各種最適化も施されている．なお，幅優先探索結果の検証を省略する機能が実装されており，消費電力測定ならびに制御を行う際にはこの機能を用いて探索結果の検証を対象外とした．

4. Graph500 の電力特性

オーバープロビジョニング・システム上で高い性能を得るためには，実行するアプリケーションの消費電力特性を把握し，それに応じた適切な電力資源配分を行う必要がある．そこで本節では，消費電力制約が設けられていない状況を前提とし，Graph500 実行時の消費電力と性能の相関を調査する．

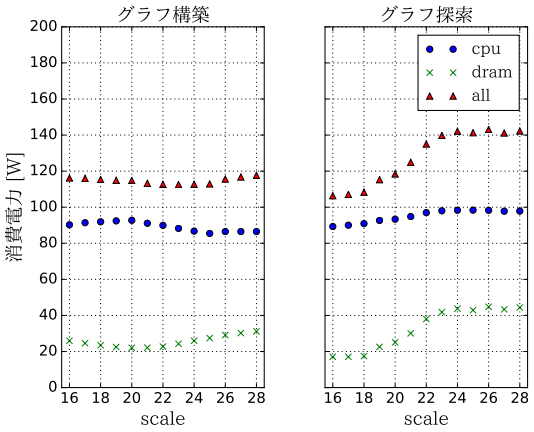

図 1　問題サイズ対全ソケットの消費電力

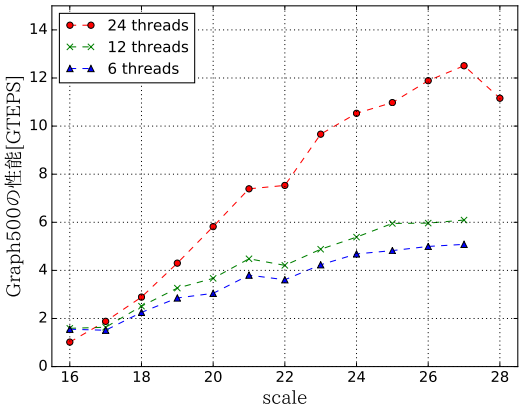

図 3　性能のスレッド数依存性

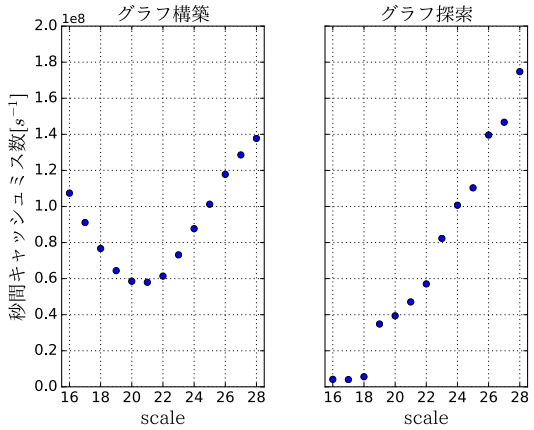

図 2　問題サイズ対コアあたり秒間キャッスミス数

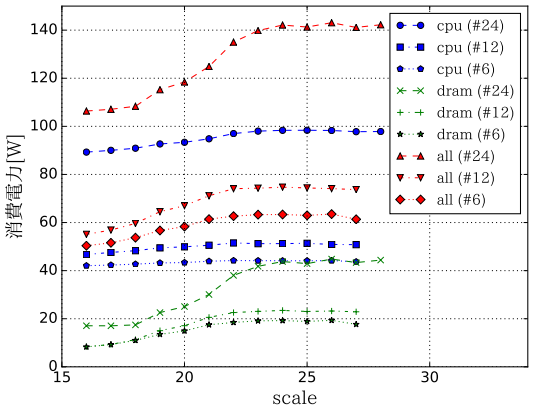

図 4　消費電力のスレッド数依存性

4.1　消費電力の問題サイズへの依存性

一般に，コンピュータシステムの性能と消費電力は，実行されるアプリケーションに与える入力の大きさに依存する．そこで，幅優先探索の対象となるグラフの大きさを $16 \leq \text{scale} \leq 28$ の範囲で変化させた場合の性能と消費電力を測定した．

グラフ構築部分と探索部分それぞれに関し測定した結果を図 **1** に示す．縦軸は CPU と DRAM の消費電力，横軸は scale を表す．青色の点が CPU の消費電力，緑色の点が DRAM の消費電力，赤色の点が CPU と DRAM の消費電力の合計である．グラフ構築において，CPU の平均消費電力は scale = 20 を境に増加から減少に転じている．そして，scale = 25 で最小となり，以降は問題サイズの増加と伴にわずかに消費電力も増加し，最終的にはほぼ一定となっている．これに対し，DRAM の平均消費電力は scale = 20 を境として，減少から増加に転じている．一方，グラフ探索では，CPU の平均消費電力は scale = 23 まで scale の増加とともに増加し，以降はほぼ一定となっている．また，DRAM の平均消費電力は $\text{scale} \leq 18$ まではほぼ一定，$18 \leq \text{scale} \leq 24$ では scale の増加とともに増加，$24 \leq \text{scale}$ ではほぼ一定という結果が得られた．

これらの結果を考察するため，ラストレベルキャッシュ・ミスの発生状況を調査した．その結果を図 **2** に示す．縦軸は一秒あたりのラストレベルキャッシュ・ミス発生回数，横軸は scale の値である．グラフ構築においては，単位時間当たりのキャッシュミス数が scale = 21 を境として減少から増加に転じていることが分かる．一方，グラフ探索では，秒間のキャッシュミス数は $\text{scale} \leq 18$ まではほぼ一定であり，以降は scale の増加に伴い増加している．DRAM の消費電力と秒間のキャッシュミス数の寡多はおおよそ一致しており，高い相関があることが分かる．なお，グラフ探索において，scale = 24 以降もキャッシュミスが増加し続けている（図 2）のに対し，消費電力はほぼ一定（図 1）であり，キャッシュミス発生頻度と消費電力の間に相関が見られない．この原因については現在調査中である．

4.2　消費電力のスレッド数への依存性

Graph500 の並列実行におけるスレッド数を 6，12，24 とし，スレッド数が消費電力と性能へ与える影響を調査した．なお，スレッド数 6 ならびに 12 の場合は 1 ソケット，スレッド数 24 の場合は 2 ソケットを用いた実行となる．

各スレッド数における Graph500 実行時の性能を図 **3** に

示す．縦軸は TEPS 値，横軸は scale 値である．scale が大きくなるにつれ性能が向上する傾向があり，scale が 23 を超えるほど大きな範囲では，スレッド数 24 での性能はスレッド数 12 での性能の 2 倍程度となっている．スレッド数 12 での性能が，スレッド数 6 での性能の 1.2 倍程度にしかならないのは，Graph500 がメモリバウンドな特性を有するためと考えられる．

次に消費電力に関して解析する．図 1 に，各スレッド数における消費電力の値を示す．縦軸が消費電力，横軸が scale 値である．ここで，青色のプロットは CPU の消費電力，緑色のプロットは DRAM の消費電力，赤色のプロットは CPU と DRAM の消費電力の合計を示したものである．また，破線，点鎖線，点線は，それぞれ，スレッド数が 24，12，6 の場合に対応する．マーカーとプロットの対応は図 1 の凡例に示したとおりである．凡例中の (# number) は，スレッド数 number で実行したことを意味する．スレッド数 24 のときの消費電力に関して，DRAM の消費電力が scale に対してある範囲で増加を続け，他の範囲では一定となるという傾向が見られる．同様の傾向はスレッド数が 12 ならびに 6 の場合にも観測されるが，scale に対する消費電力の増加量はスレッド数と伴に小さくなっている．また，DRAM の消費電力が増加を始める scale，ならびに，増加が終わり一定となる scale は若干異なる．実際，スレッド数 12，6 ともに，scale = 16 から DRAM の消費電力の増加が始まり，scale = 22 まで増加が続いている．

4.3 消費電力の時間変化

Graph500 実行時の消費電力の時間変化について調査を行った．scale = 24 における Graph500 実行時の消費電力を 10[ms] の間隔で測定し，前後 50[ms] の消費電力の移動平均とともにプロットした結果を図 **5** に示す．この結果より，Graph500 の消費電力にはいくつかのフェーズが存在することが確認できる．Graph500 は，18.2[s] まではグラフの生成，60.3[s] まではグラフの構築を行っている．グラフ構築の進行は以下の通りである．

(1) 18.2[s] から 21.6[s] まで：グラフの各頂点の出次数をカウント．
(2) 21.6[s] から 25.0[s] まで：頂点を出次数に基づきソート．
(3) 25.0[s] から 29.4[s] まで：辺の集合から，CSR 形式のインデックスを作成．
(4) 29.4[s] から 36.2[s] まで：Numa-aware な CSR 形式のグラフ表現を構築．
(5) 36.2[s] から 47.5[s] まで：多重辺の探索．
(6) 47.5[s] から 60.3[s] まで：辺をソート．

なお，以降の約 60 秒間はグラフ探索を実施している．

消費電力は，グラフ生成ならびに探索では全体を通して一つのフェーズを形成しているが，グラフ構築では複数の

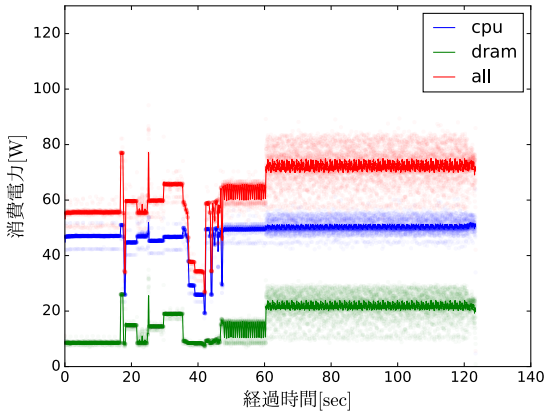

図 **5** 実行中の消費電力変化

フェーズが存在する．グラフ構築で見られる電力フェーズは処理内容の違いと一致しており，36.2[s] から 47.5[s] に消費する電力には更に前半と後半の二つのフェーズが読み取れる．実際，前半部分では頂点の次数のカウント，後半部分では辺のソートを行っており，処理内容が異なっている．

5. 電力制約時の性能評価

本節では，電力制約が与えられることを前提とし，制約内での CPU/DRAM 電力資源配分が Graph500 の性能に与える影響を評価する．

5.1 実験方法

満たすべき消費電力制約に基づき，CPU と DRAM の両方が稼働可能となる電力資源配分の候補を決定する．そして，それぞれの候補に関して，CPU と DRAM に電力キャッピングを適用し性能を評価する．なお，本実験で用いた計算機サーバは 2 つの CPU ソケットを搭載しているが，これらに対しては一律の電力制約を与えるものとした．すなわち，文献 [4] で指摘された半導体の製造ばらつきの影響は考慮しない．

5.2 実験結果

1 ソケット当たりの電力制約が 50[W]，60[W]，ならびに，70[W] のときの Graph500 の性能を 図 **6**，図 **7**，図 **8** に示す．横軸は問題サイズであり，凡例の cpu:x[W]-dram:y は，CPU へ x[W]，DRAM へ y[W] の電力資源を配分したことを意味する．

5.3 考察

まず，scale と電力配分が性能に与える影響について考察する．図 6，図 7 および図 8 において，$16 \leq \text{scale} \leq 18$ の範囲では CPU に多くの電力を配分するほど性能が高いことが分かる．この問題サイズの範囲においては，実質的に DRAM に対する電力制約が性能に影響しないことが主

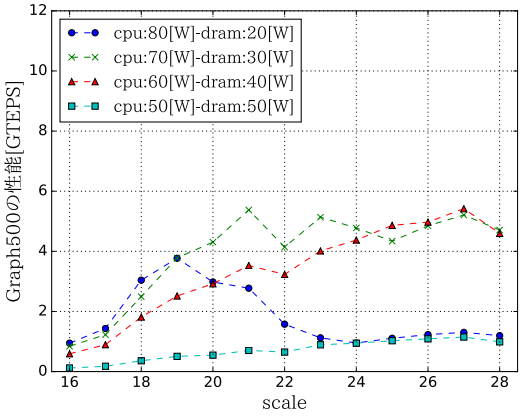

図 6　電力資源制約 100W の場合の問題サイズ対幅優先探索性能

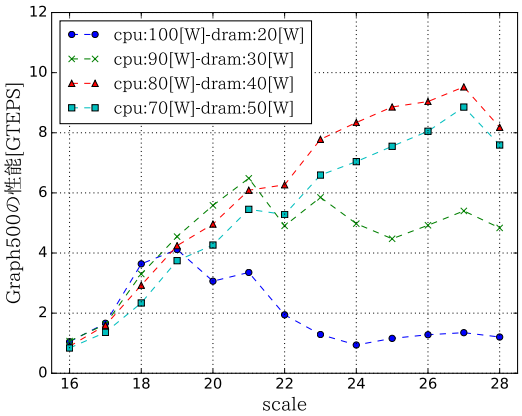

図 7　電力資源制約 120W の場合の問題サイズ対幅優先探索性能

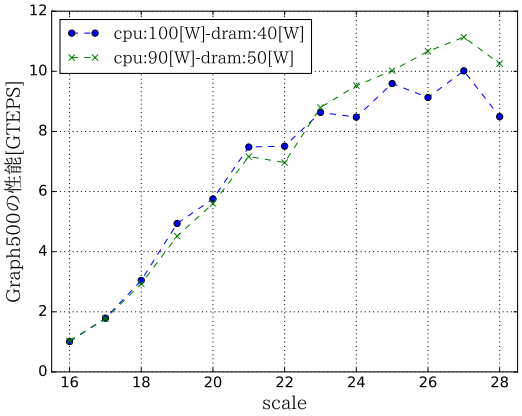

図 8　電力資源制約 140W の場合の問題サイズ対幅優先探索性能

な理由として考えられる．実際，第 4.1 節で示した図 1 から，電力非制約時の DRAM の消費電力は全体で 20[W] 弱，CPU の消費電力が全ソケットで 90[W] ほどである．つまり，問題サイズが $16 \leq \text{scale} \leq 18$ の範囲においてはそもそも DRAM の消費電力が小さく，本実験での DRAM 最低電力制約値である 20[W] でも十分な電力資源配分となる．その結果，全ての電力資源配分において実質的には CPU に対してのみ電力制約が与えられていることと等価になる．これに対し，図 6 における $25 \leq \text{scale} \leq 28$，図 7 における $22 \leq \text{scale} \leq 28$，そして 図 8 における $24 \leq \text{scale} \leq 28$ に関しては，DRAM に多くの電力を配分した方が高い性能となる．この範囲では，第 4.1 節で示した図 1 から，電力非制約時の CPU の消費電力は 100[W] 前後，DRAM の消費電力は 40[W] 前後となっており，本実験において DRAM に 50[W] の制約を与えている場合を除いて CPU と DRAM の双方に電力キャッピングを施している状況になる．Graph500 のようにメモリインテンシブなアプリケーションの場合には DRAM へ十分な電力資源を配分することが重要となる．

次に，DRAM に対する電力制約が性能に与える影響について議論する．Graph500 ではその問題サイズである sacle が大きくなるほどメモリアクセスが増加する．したがって，DRAM への電力配分が少ない場合，scale が大きくなるほど性能が低下すると考えられる．実際に，図 7 の cpu:100[W]-dram:20[W] では，scale が 19 以上になると他の電力配分の場合と比較して低い性能となっている．ここで，図 1 より，電力制約を与えない場合の DRAM 消費電力は，scale=19 以上において 20[W] を超過していることが分かる．このような傾向は図 8 の cpu:100[W]-dram:40[W] に関しても観測することができる．したがって，これらの結果から，DRAM への電力配分が電力制約がない場合の DRAM 消費電力よりも小さい場合は性能が大きく低下すると結論付けられる．

しかしながら，図 6 の cpu:50[W]-dram:50[W] では，DRAM の電力配分が多いにも関わらず，他のどの電力配分の結果よりも性能が低い．cpu:50[W]-dram:50[W] の制約を与えた際の CPU と DRAM の実際の消費電力を調査したところ，CPU 消費電力値は約 50[W]，DRAM 消費電力値は約 17.9[W] であることが分かった．実際の DRAM 消費電力は電力制約値の 50[W] よりも小さく，DRAM 稼働率が低くなっていると予想される．この理由として，CPU に対する電力制約が大きくその動作周波数が低下し，単位時間当たりに発行されるメモリアクセス数が減少したことで，DRAM 消費電力が低下したこと考える．実際，単位時間当たりのラストレベルキャッシュミス数を調査したところ，電力資源制約 100[W] で cpu:50[W]-dram:50[W] では，$16 \leq \text{scale} \leq 28$ の範囲で一貫して他の電力配分よりも単位時間あたりのラストレベルキャッシュミス数が少ない状況であった．

6. 関連研究

スーパーコンピュータの消費電力問題を解決する手段として，オーバープロビジョニング・システムに関する研究が注目を集めている [8]．これまでに，幾つかのアプリケーションを対象とし最適な電力資源配分を解析する研究 [12]

や，プロセッサ毎の電力ばらつきを考慮した電力制約下での高性能化手法に関する研究が行われてきた [4].

Graph500 を対象に，性能および電力に関する研究が実施されている [5], [11]．文献 [11] では，Graph500 実行時における電力あたりの性能に関して解析している．また，Intel 社のシングルチップ・クラウド・コンピュータを対象とし，DVFS（Dynamic Voltage and Frequency Scaling）の設定や実行に用いるコア数を変更した際の Graph500 の性能および消費電力の解析も行われている [5]．しかしながら，著者らが知る限り，大規模グラフ解析を対象とした電力制約下での性能評価は未だ行われていない．

7. まとめ

本稿では，2 ソケットの CPU を搭載した計算機サーバを用い，Graph500 を対象とした電力性能特性解析ならびに電力制約下での性能評価を実施した．電力制約が無い状況での解析の結果，Graph500 の性能を決める幅優先探索では，1) DRAM の消費電力は入力となる問題サイズが一定の範囲で増加すること，2) 実行に要するスレッド数により DRAM で消費する電力が大きく異なること，3) グラフ構築処理においては電力消費の振る舞いに関して幾つかのフェーズが存在すること，4) グラフ探索では全体を通して一つのフェーズとなっていること，などが分かった．また，電力制約下での Graph500 の性能について解析を行った結果，5) 最大の性能となる CPU/DRAM への電力資源配分は探索対象グラフのサイズにより異なり，グラフのサイズが小さい場合には CPU へ，グラフのサイズが大きい場合には DRAM へより多くの電力資源を配分する必要があることを示した．今後の課題としては，電力性能を改善するための電力資源配分法の考案やマルチノードを用いたより大規模なシステムでの評価が挙げられる．

謝辞　本研究は，一部，JST CREST「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出」ならびに文部科学省科学研究費補助金 16H02796 の補助による．

参考文献

[1] 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*, pp. 12:1–12:10 (2012).

[2] Chakrabarti, D., Zhan, Y. and Faloutsos, C.: R-MAT: A Recursive Model for Graph Mining, *Proceedings of the 2004 SIAM International Conference on Data Mining*, Society for Industrial and Applied Mathematics, Philadelphia, PA, pp. 442–446 (2013).

[3] Committee, G. . S.: Graph 500 Benchmark 1 (”Search”), Graph 500 Steering Committee (online), available from ⟨www.graph500.org/specifications⟩ (accessed 2016-1-19).

[4] Inadomi, Y., Patki, T., Inoue, K., Aoyagi, M., Rountree, B., Schulz, M., Lowenthal, D. K., Wada, Y., Fukazawa, K., Ueda, M., Kondo, M. and Miyoshi, I.: Analyzing and mitigating the impact of manufacturing variability in power-constrained supercomputing., *SC :1-78:12*, pp. 78–12 (2015).

[5] Lai, Z., Lam, K. T., Wang, C.-L. and Su, J.: Power and performance analysis of the Graph 500 benchmark on the Single-chip Cloud Computer, *Proceedings of 2014 International Conference on Cloud Computing and Internet of Things*, pp. 9–13 (2014).

[6] Low, Y., Bickson, D., Gonzalez, J., Guestrin, C., Kyrola, A. and Hellerstein, J. M.: Distributed GraphLab: A Framework for Machine Learning and Data Mining in the Cloud, *Proc. VLDB Endow.*, Vol. 5, No. 8, pp. 716–727 (2012).

[7] Malewicz, G., Austern, M. H., Bik, A. J., Dehnert, J. C., Horn, I., Leiser, N. and Czajkowski, G.: Pregel: A System for Large-scale Graph Processing, *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, pp. 135–146 (2010).

[8] Patki, T., Lowenthal, D. K., Rountree, B., Schulz, M. and de Supinski, B. R.: Exploring Hardware Overprovisioning in Power-constrained, High Performance Computing, *Proceedings of the 27th International ACM Conference on International Conference on Supercomputing*, pp. 173–182 (2013).

[9] Rotem, E., Naveh, A., Ananthakrishnan, A., Weissmann, E. and Rajwan, D.: Power-Management Architecture of the Intel Microarchitecture Code-Named Sandy Bridge, *IEEE Micro*, Vol. 32, No. 2, pp. 20–27 (2012).

[10] Yasui, Y., Fujisawa, K., Goh, E. L., Baron, J., Sugiura, A. and Uchiyama, T.: NUMA-aware Scalable Graph Traversal on SGI UV Systems, *Proceedings of the ACM Workshop on High Performance Graph Processing*, pp. 19–26 (2016).

[11] Yasui, Y., Fujisawa, K. and Sato, Y.: Fast and Energy-efficient Breadth-First Search on a Single NUMA System, *Proceedings of the 29th International Conference on Supercomputing*, pp. 365–381 (2014).

[12] 吉田匡兵，佐々木広，深沢圭一郎，稲富雄一，上田将嗣，井上弘士，　青柳睦：CPU と主記憶への電力バジェット配分を考慮した HPC アプリケーションの性能評価，情報処理学会研究報告, 2013-HPC-141(21)，pp. 1–8 (2013).