Vol.2013-HPC-141 No.20
2013/10/1

RAPLインタフェースを用いたHPCシステムの消費電力モデリングと電力評価

カオ タン[1,2]　和田 康孝[1]　近藤 正章[1,2]　本多 弘樹[1]

概要：将来の HPC システムでは，消費電力がシステム設計や実効性能を制約する最大の要因の一つになると考えられている．運用時のピーク消費電力が電力制約を超えないことを保証する従来の設計思想では，アプリケーションを今後の大規模システムに対してスケールさせることは難しいとの認識のもと，我々は，ピーク消費電力が制約を超過することを積極的に許容し，適切に電力性能ノブを調整しつつ限られた電力資源を有効に使用して高い実効性能を得る電力制約適応型システムと，その実現に必要となる電力マネージメントフレームワークの研究開発を実施している．このような電力制約適応型システムにおいては，アプリケーション実行時の電力消費状況を観測し，また柔軟に電力制御を行える環境が必須となる．近年の Intel 社のプロセッサには RAPL (Running Average Power Limit) と呼ばれるプロセッサと DRAM の消費電力を観測・制御するインタフェースが備えられている．本稿ではこの RAPL を用い，アプリケーションを実行させた際の消費電力計測と制御を行い，HPC システムに用いられる計算機の電力計測特性について調査する．また，ノード全体の電力の柔軟な計測を可能とするべく，RAPL の計測値を用いてノード全体の電力のモデリングを行う．実験の結果，RAPL により高い精度でプロセッサや DRAM，また　ノードの消費電力を観測できることがわかった．

1. はじめに

将来の HPC システムでは，消費電力がシステム設計や実効性能を制約する最大の要因の一つになると考えられている．例えば，現時点で世界最高性能を誇る Tianhe-2 は 33 ペタフロップス超の性能を 18MW 近い消費電力で達成している．地球規模の省エネ要求や現在の大型計算機センターの電力設備状況を鑑みると，将来的にも 100MW 級の電力供給能力を持つセンターを配することは不可能であり，2020 年あたりに実現されるエクサスケール級のシステムは 20〜30MW とほぼ同程度の電力で現在の世界トップクラスのスーパーコンピュータの 30〜50 倍近い性能を達成することが求められる．さらに，環境負荷低減の重要性が叫ばれる中，高性能計算システムでも太陽光発電などの再生可能エネルギー利用が拡大し，電力供給が時々変化するといった運用環境の変化が訪れることも予想される．

このような背景のもと，供給電力，あるいは熱設計消費電力制約の中でハードウェア資源を投入し，運用時のピーク消費電力が制約を超えないことを保証する従来の設計思想では，アプリケーションを今後の大規模システムに対してスケールさせることは難しいと考えられる．そこで，我々はピーク消費電力が制約を超過することを積極的に許容し，ハードウェアが持つ電力性能ノブを調整することで限られた電力資源を計算・記憶・通信という各要素に適応的に配分し，実効電力を制約以下に制御しつつ高い実効性能を得る電力制約適応型システムがポストペタスケール HPC システムのあるべき姿との認識に立ち，その実現に必要となる電力マネージメントフレームワークの研究開発を実施している [1].

このような電力制約適応型システムでは，アプリケーションの特徴やシステムの運用状況等に合わせた電力制御・電力管理が電力マネージメントフレームワークの最も重要な役割の一つとなる．さらに，適切な電力制御のためには，まずアプリケーション実行時の電力消費状況を観測することが必須となる．実際に TSUBAME2.0 は各計算ノード，ラック，及び計算機室の消費電力情報を監視するシステムを備えている [2].

将来的な電力制約適応型システムの実現には，各構成要素の電力計測と制御を細粒度に行うことができ，かつ大規模システムでも効率的に電力消費状況の観測ができる柔軟さを持つことが重要になる．大規模 HPC システムでは，空調や電力供給系を含め電力消費には様々な要因があるが，プロセッサチップとメインメモリ (DRAM) の消費電力は依然としシステム全体の中で大きな割合を占めている．そ

[1] 電気通信大学大学院情報システム学研究科
[2] 独立行政法人科学技術振興機構，CREST

のプロセッサと DRAM の消費電力を観測・制御する手段として，近年の Intel 社のプロセッサには RAPL (Running Average Power Limit)[3], [4] と呼ばれるインタフェースが備えられている．RAPL はプロセッサチップと DRAM の電力計測，および電力制御を可能とするインタフェースであり，ソフトウェアから簡便に，かつ時間的に細粒度に電力計測を行うことができるという特徴を持つ．

本稿では，この RAPL インタフェースを用いた HPC システムの電力計測と制御を行い，電力計測器によるノード全体の消費電力と比較しつつ，HPC システムに用いられる計算機の電力計測特性について調査する．また，電力制約適応型システムにはプロセッサと DRAM の消費電力のみならず，ノード全体の電力も細粒度に観測する必要があることから，RAPL の情報をもとにノード電力モデリングを行い，ノード全体電力の推定に関する考察を行う．将来の HPC システムでは電力効率が重要となるのは周知の事実であり，Intel 社のプロセッサのみならず，今後多くのシステムにおいても同様の機能を持つシステムが登場すると予想される．ここで，そのようなインタフェースを用いて HPC システムの電力消費状況を調査・検討することは重要であると考えられる．

2. RAPL インタフェースと関連研究

2.1 RAPL インタフェース

RAPL（Running Average Power Limit）インタフェースは Intel 製プロセッサにおいて Sandy Bridge マイクロアーキテクチャより搭載された機能であり，この機能を介して，プロセッサおよび DRAM の消費電力に関する情報を取得したり，消費電力の上限を設定することができる．プロセッサは，パフォーマンスカウンタや温度などの情報を基に消費電力を見積り，与えられた消費電力の上限を超えないように制御を行う [3], [4].

RAPL では，図 1 に示すように，消費電力を観測・制御する単位が 3 種類定義されており，サーバ環境では，チップ全体（Package，PKG），チップ上のコア部分（Power Plane 0，PP0），およびメモリ（DRAM）がそれにあたる．ユーザは MSR（Model Specific Register）を介することで，消費電力の取得，消費電力の上限設定などの操作を上記の各ドメイン毎に適用することができる [5].

2.2 関連研究

従来から，パフォーマンスカウンタ等を用いた電力観測技術や DVFS に代表される電力制御技術が実装されてきた．

実際のコンピュータシステム上で DVFS 機能を利用する環境としては，例えば Linux に搭載された CPUFreq が広く用いられており，OS がシステムの負荷状況に応じてに自動的に DVFS を適用したり，Sysfs などの仮想ファイルシステム上のインタフェースを介してユーザが動的に DVFS

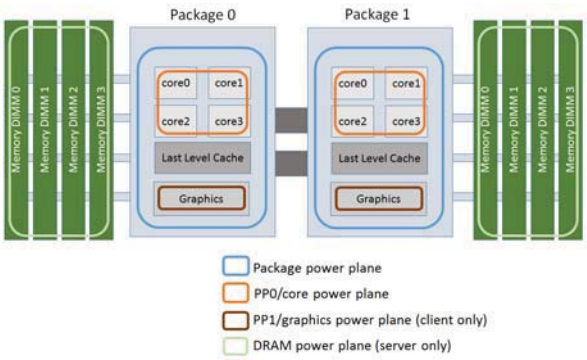

図 1 RAPL における消費電力観測・制御の単位 [6]

を適用することを可能としている．このような DVFS が利用可能な環境において消費電力を制御・削減する手法としては，動作周波数と電圧の変更によるアプリケーション実行時間への影響を MIPS 値から推定し，性能への影響を最小限にしつつ消費電力を削減する手法 [7] や，MPI プログラム実行に用いるノード数やプロセッサの動作周波数を最適化し，与えられた消費電力の上限を超えない範囲でプログラムの実行時間を最小化する手法 [8]，MPI プログラム内の各タスク実行毎に情報を取得し，次回以降のタスク実行におけるプロセッサの動作周波数を動的に決定する手法 [9]，実際の HPC アプリケーションのプロファイル結果を基に，性能への影響が無い範囲で HPC システムの消費電力を削減するアルゴリズム [10] などが提案されている．

さらに，上記のような低消費電力制御技術を効率よく適用するためには，対象システムの消費電力特性を考慮する必要があり，そのために様々な消費電力観測・推定技術が研究されている．例えば，パフォーマンスカウンタの情報と線形回帰によって HPC 向けアプリケーション実行時の消費電力を見積る手法 [11] や，実際に様々なアプリケーションを実行した結果から導出したモデルとパフォーマンスカウンタの値を用いてシステム内の各要素（プロセッサやメモリ，ディスク等）の消費電力を推定する手法 [12]，GPU の消費電力をモデル化し，パフォーマンスカウンタの値からカーネル実行時の消費電力を見積る手法 [13] などが提案されている．

また，近年の HPC システムにおいては，実際にラック単位，あるいはより細かい単位で消費電力を監視する機構を備えているものもある．TSUBAME2.0 は各計算ノード，ラック，及び計算機室の温度情報・消費電力等を監視するシステムを備えている他，IBM の Blue Gene/P や Blue Gene/Q はラックの AC/DC コンバータや各ノードボード，リンクカード等消費電力を一定間隔で取得・監視する機能を備えている [14].

3. RAPL を用いた電力計測

本章では，RAPL を用いて HPC システムに利用される

表 1　実験システムの仕様

Processor: Intel Xeon E5-2690	
Num. of Cores	8
Primary Cache	32KB I + 32KB D cache per core
Secondary Cache	256KB per core
L3 Cache	20MB per chip
Motherboard: Asus Z9PE-D8WS	
Num. of CPU Socket	2
Num.of DIMM Slot	8
Num. of Memory Channels	Quad Channels
Chipset	Intel C602
LAN Controller	Intel 82574L, 2 x Gigabit
DIMM: DDR3-1600 TED316G1600C11DC	
Size	8GB x 8
Latency	11-11-11-28

表 2　電力設定パラメータ

PKG Thermal Spec Power	135W
PKG Min Power	51W
PKG Max Power	215W
DRAM Thermal Spec Power	35W
DRAM Min Power	15W
DRAM Max Power	75W

サーバ計算機の消費電力を計測し，各計測ドメインの電力消費の傾向を調査するとともに，AC 電源に接続した外部電力測定器と計測値を比較することで，RAPL による電力計測の特性について議論する．

3.1　評価環境

RAPL が利用できるサーバ計算機として**表 1** に示すシステムを用いた．プロセッサには Intel Xeon E5-2690（2.90GHz）を使用し，また今回利用するマザーボード (Asus Z9PE-D8WS) には，プロセッサを 2 ソケット，DDR3 の DIMM モジュールを 8 枚搭載可能である．また，外部電力メータとしては，ThinkTank Energy Products Inc. の Watts up? PRO[15]（以降 WattsUp と表記）を用いた．WattsUp は最小で 1 秒間隔で AC 電源の電力を計測し，USB インタフェースよりログを取得可能である．

参考までに，Xeon E5-2690 プロセッサから取得した電力設定のパラメータを**表 2** に示す．これによると，当該システムのパッケージ (ソケット) の電力制約値として最大 215W および最小 51W，また DRAM の電力制約値として，最大 75W および最小 15W を設定できることがわかる．

電力測定に用いるベンチマークは，2 つの配列に連続してアクセスしつつ，各要素に乗算を行うストリームアクセスプログラム，および NPB から EP，FT，IS，MG，MG の各カーネル (クラス D) である．

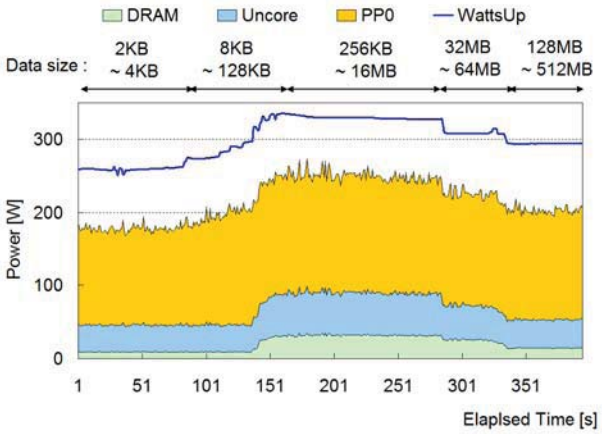

図 2　ストリームアクセスプログラムの電力計測結果

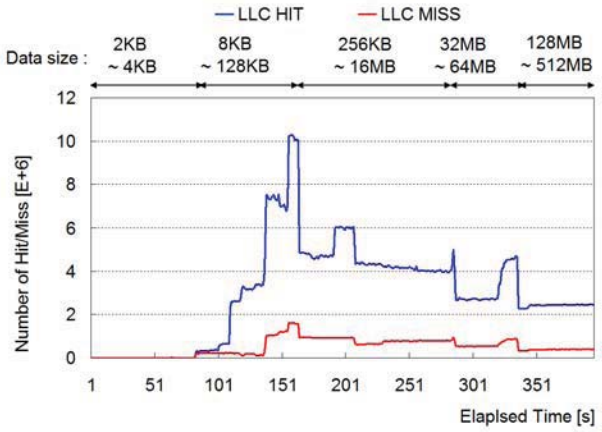

図 3　LLC ヒット・ミス回数 (図 2 に対応)

3.2　電力計測結果

まず，キャッシュや DRAM アクセス頻度の違いによる電力消費の変化を観測するために，ストリームアクセスプログラムにおいて，アクセスする配列のサイズを 2KB から 2GB まで段階的に変化させて，RAPL による電力計測を行った．**図 2** に 2 ソケット合計の計測結果を示す．なお，本評価は MPI を利用したプロセス並列により全 16 コアを用いている． RAPL による電力値は 500 ミリ秒間隔で取得した．

図中，RAPL で取得された各ドメインの電力は積み上げグラフとして示しており，WattsUp の電力はノード全体の電力である．また “Uncore” は PKG ドメインから PP0 ドメインを差し引いたプロセッサ・コア以外で消費される電力を意味している．上部の “Data size” はリードアクセスをする配列のサイズ (2 つの配列の合計) を示している．図より，アクセスする配列サイズが小さい場合は DRAM および Uncore の電力が小さいが，配列サイズが L2 キャッシュサイズである 256KB 程度以上になるとそれらの電力が増加し，全体の電力も増加することがわかる．これは，アクセスする配列サイズが小さい場合はコア内にある L1 あるいは L2 キャッシュでヒットするため L3 キャッシュへのアク

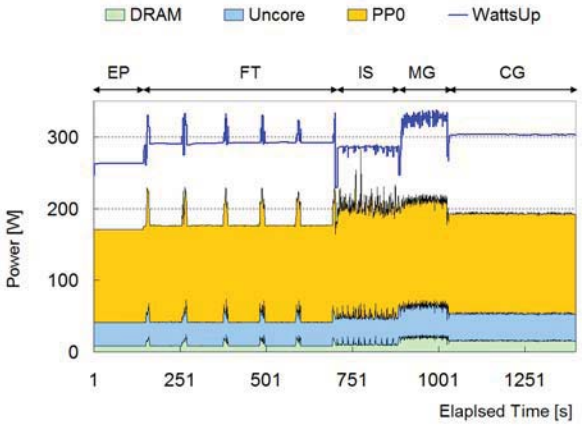

図 4 NPB の電力計測結果

セスが生じないが，配列サイズが大きくなると L3 キャッシュ，またプリフェッチも含めた DRAM へのアクセスが発生し，DRAM および Uncore の電力が増加するためと考えられる．逆に配列サイズが LLC サイズである 20MB あたりを超えると，再び DRAM および Uncore の電力が減少している．これは，配列アクセスのほとんどが遅延の大きな DRAM アクセスになると，単位時間のアクセス発行が減少するためであると考えられる．**図 3** は，本ベンチマークにおける単位時間あたりのラストレベルキャッシュである L3 キャッシュのヒットとミス回数を示している．これからも，配列サイズが 256KB 前後の L3 ヒット・ミス回数が多く，電力消費の増大に繋がっていることがわかる．

図 4 は NPB の計測結果であるる．EP，MG，CG は各カーネル内での消費電力変化は小さいが，FT や IS ではカーネル内でも電力値に変化がある．また，特に DRAM の消費電力はカーネル毎に大きく異なることがわかる．

上記計測結果において，RAPL と WattsUp での電力を比較すると，WattsUp の計測電力はマザーボードやファンなど，ノード全ての電力が含まれるため，RAPL で計測した PP0，Uncore，DRAM の合計電力に比べて値が大きい．ただし，電力消費の傾向は WattsUp の計測結果と非常に似通っており，RAPL により高い精度での電力計測が行えると考えられる．

3.3 DRAM 構成を変化させた場合の消費電力計測結果

RAPL では，主にプロセッサ内部のイベントカウンタの情報を基に電力値を推定しているため，プロセッサの電力を正確に見積もることが可能であったと考えられる．しかし，DRAM の電力計測の精度に関しては不明な点も多い．特に DRAM 構成の違いに応じて電力値がどう変化するかは興味深い事項である．そこで，DRAM 構成を変化させ消費電力の計測を行った．具体的には，もともと 8GB の DIMM モジュールを 8 枚接続している構成から，数枚のモジュールを抜くことで 48GB および 32GB の構成に変化さ

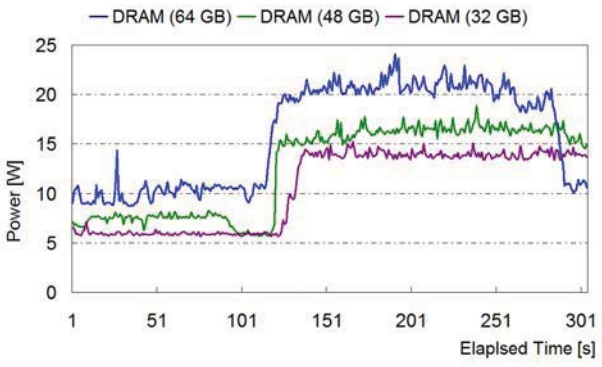

図 5 DRAM 構成を変化させた際の電力計測結果

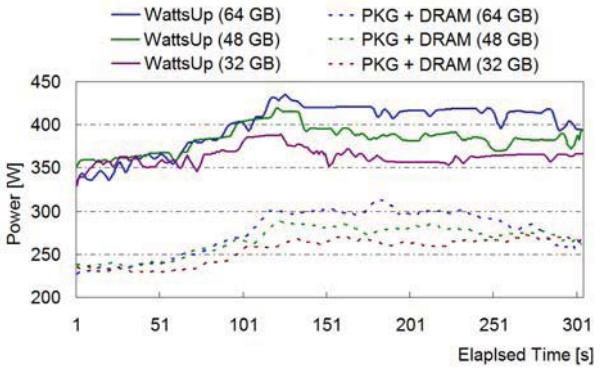

図 6 DRAM 構成を変化させた際のノード電力

せ評価を行う．

図 5 に，ストリームアクセスプログラムを実行した場合の各構成での DRAM ドメインの電力を，また**図 6** に WattsUp の計測値と RAPL による PKG と DRAM ドメインの合計電力値を示す．図 5 では，DRAM モジュールの枚数に応じて DRAM ドメインの電力値が異なっている．また，DRAM アクセスが多い中程の電力に着目すると，WattsUp のノード全体電力を見た場合の各構成の電力差は 18W 程度であるが，RAPL による PKG+DRAM ドメインの電力の差は 6W から 16W 程度であり，DRAM モジュールあたりの電力が実際よりもやや小さく見積もられていることがわかる．

3.4 電力制約を設定した際の消費電力計測結果

前述のように，RAPL インタフェースはパッケージと DRAM の電力制約を設定することが可能である．ここでは電力制約を設定した際の消費電力の傾向と性能への影響を調査する．なお，当該マザーボードでは DRAM ドメインの電力制約を設定することができないため，本評価では PKG ドメインのみ電力制約を設定して評価を行った．**図 7** に電力測定結果を示す．ここでは，配列アクセスのベンチマークを 16 コアで実行し，各ソケットの PKG ドメインの電力制約を最大電力の 75% (101W)，50% (68W)，設定可能な最小電力 (51W) の 3 通りに設定して評価を行った．

図より，電力制約を設定すると，消費電力が実際に低下していることがわかる．2 ソケット分の電力であることを

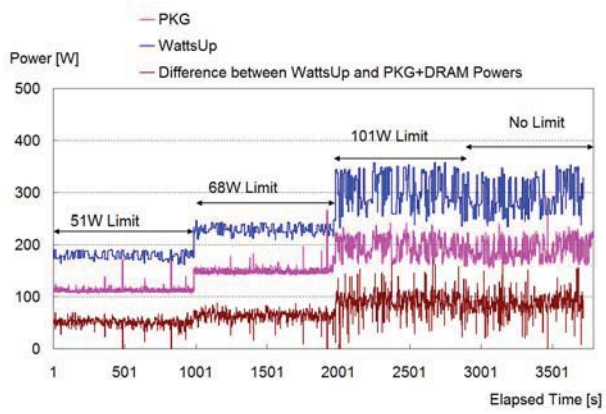

図 7 電力制約を設定した際の消費電力計測結果

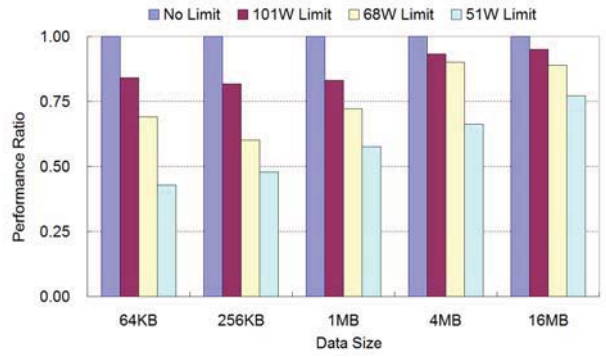

図 8 電力制約を設定した際の性能変化

考慮すると，制約対象である各ソケットの PKG ドメインの消費電力は，ほぼ設定した制約値と同程度以下に抑えられていることがわかる．このことから，RAPL により高い精度で電力制約の設定が行えると考えられる．なお，図には WattsUp の電力と RAPL で計測した PKG+DRAM ドメインの合計電力との差も示しているが，制約を設定しても差が一定にはならず，制約値が低いほど差分も小さくなる傾向がある．そのため，ノード合計の電力に注目した場合，その電力制約を適切に設定するためには RAPL の計測結果をモデリングし，ノード電力を正確に推定することが必要になると考えられる．

図 8 は電力制約を設定した場合の相対性能を，ストリームアクセスプログラムの 5 種類の配列サイズについて示したものである．図より，電力制約を設定すると，制約値が厳しくなるに従って性能が低下している．特に配列サイズが小さくキャッシュ上のデータで演算が行える，すなわち演算バウンドである場合に性能低下が大きい．一方で，配列サイズが大きい場合，DRAM アクセスがボトルネックとなることで，PKG ドメインの性能を制約する影響は相対的に小さくなるため，性能低下も小さいという結果になった．

4. ノード電力のモデリング

ポストペタスケール HPC システム時代において，電力制約適応型システムを実現するためには，ノード全体の電力を簡便に，かつ柔軟にリアルタイムで推定できる必要がある．RAPL は非常に簡便かつ柔軟に電力計測が行えるインタフェースであるが，プロセッサソケットと DRAM の電力のみが計測対象であり，ノード全体の電力を計測することはできない．しかし，3 章の計測結果を見ると，WattsUp で測定したノード全体の電力と RAPL の電力は高い相関があり，RAPL の測定結果からノード電力も高い精度で推定が可能であると考えられる．ただし，RAPL の計測値に一定のベース電力値を加算するだけでは十分に正確ではなく，電力制約を設定した場合や MPI 通信の負荷が高い場合など，多少 RAPL と WattsUp の電力消費の傾向が異なる場合も見受けられる．そこで，本章では RAPL 計測値を用いてノード全体の電力をモデリングすることで，高い精度でノード全体の電力を推定することを考える．

4.1 データの取得

消費電力のモデリングを行うためには，計算処理やメモリアクセスなどに関して，種々の条件で電力測定をする必要がある．本稿では，3.1 節で述べたストリームアクセスベンチマーク，NPB の他に HPC Challenge (HPCC) ベンチマークの中から 6 種類 (DGEMM, STREAM, PTRANS, RandomAccess, FFT, Latency/Bandwidth) のベンチマークも用いる．なお，HPCC の問題サイズは 5,000 から 20,000 まで変化させた．また，電力制約を設定した場合にも適切にノード消費電力の見積りができるよう，様々な電力制約を与え電力計測を行った．さらに，複数ノードを用いた際の傾向も電力に影響を与える可能性があるため，ランク数も変化させてデータを取得した．RAPL と WattsUp による電力計測では，それぞれ 1 秒間隔の電力データを取得することにし，電力計測値と同時に 216 個のパフォーマンスカウンタ値も取得した．なお，異なる環境におけるモデリングの精度を議論するために，本章では 3 章で使用した Asus Z9PED8WS の他に，SuperMicro MBD-X9DRL-IF-O マザーボードを用いて実験を行う．

4.2 モデリング手法

計算ノード全体の電力と，RAPL で計測した各ドメインの電力値やパフォーマンスカウンタの値は基本的に線形関係にあると考えられるため，本稿では線形回帰モデルを利用してノード全体電力のモデリングを行うことにした．RAPL による各ドメインの電力計測値および各種カウンタ値と WattUp で求めた実際のノード電力との相関を調べたところ，RAPL の各ドメインの計測値が最も相関が高いことがわかった．さらに，時刻 t のノード電力は時刻 t の RAPL 計測値だけでなく，時刻 $t-1, t-2, t-3$ の RAPL 計測値にも大きく依存することがわかった．これは，AC 部で測定している計算ノードの電力は，電源部分やマザーボードに搭載されているキャパシタ等の影響により，チッ

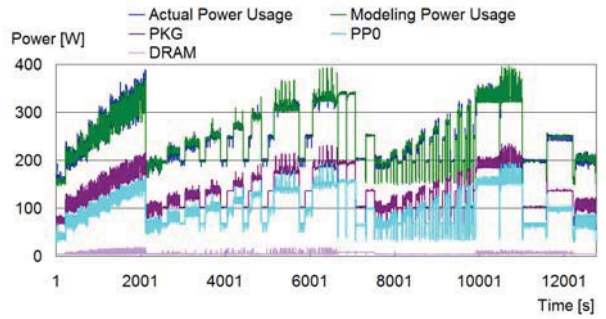

図 9 モデリングによるノード電力の推定

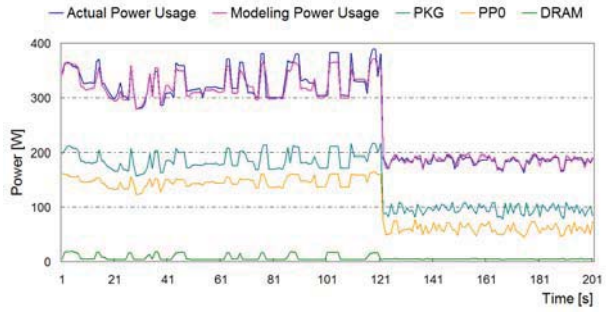

図 10 モデリングによるノード電力の推定の詳細

プ内部の電力変化に比べて急な電力変化が抑制されるためと考えられる．

以上の結果を踏まえ，本稿ではある時刻 t の電力を得るために，線形関数 f を用い，以下の式によりモデリングすることとした．

$$NodePower_t = f(PKG_{t-i}, PP0_{t-i}, DRAM_{t-i} | i = [0,3]) \quad (1)$$

式 (1) によりノード電力をモデリングする上で，取得した電力値のうち 70%のデータを学習に，残り 30%を検証に用いることとした．具体的には，29,777 データポイントを学習に，12,761 データポイントを検証に用いることになる．**図 9** に Asus Z9PE-D8WS マザーボードにおける Wattup により測定した実際のノード消費電力 (Actual)，モデリングにより見積もられた消費電力 (Modeling)，および RAPL により計測された消費電力を示す．また，**図 10** は，図 9 のある区間を拡大したものである．

図より，RAPL の計測値を用いることで，非常に正確にノード全体の電力を推定できていることがわかる．これは，ノードの中でプロセッサチップと DRAM の消費電力が大きな割合を占めていること，またその他の構成部品の電力は実行するプログラムの特徴によらず，比較的一定であり，線形回帰で十分にモデリングが行えることが理由であると考えられる．**表 3** に 2 種類のマザーボードにおける，モデリングにより見積もられた電力と実際の電力の誤差の内訳を示す．例えば，Asus Z9PE-D8WS マザーボードでは，88.33%のデータポイントが誤差 2.5% (10W に相当) 以内に，9.13%が誤差 2.5%から 5.0%の範囲に収まっている．これより，ほとんどの場合で，誤差は 5%以下と非常に

表 3 モデリング精度

Modeling Error	Z9PE-D8WS	MBDX9DRL-IF-O
Less than 2.5%	88.33%	79.97%
2.5% to 5.0%	9.13%	19.67%
5.0% to 10.0%	1.71%	0.32%
Larger than 10.0%	0.83%	0.04%

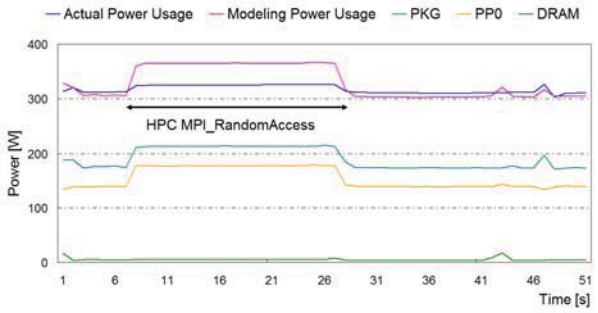

図 11 モデリングによる電力推定誤差の大きな部分の拡大

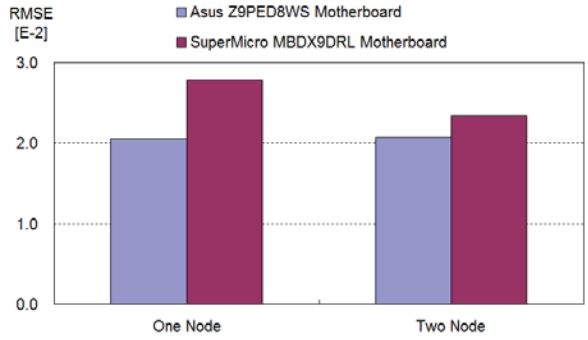

図 12 モデリング結果の平均二乗誤差

小さく，RAPL のデータを利用することで高い精度でノード全体の電力を推定可能と言える．

図 11 は 10%以上の誤差が生じた部分を抜き出して，Wattup により測定した電力とモデリング電力を示したものである．誤差の大きな区間は，HPCC の Random Access ベンチマークの中で MPI の all to all 通信が行われている部分であり，転送待ちのランクが多いことが特徴である．

次にモデリング精度をより定量的に評価するため，平均二乗誤差 (RMSE: Root Mean of Squared Errors) を用いて評価する．$RMSE$ は以下の式により求めることができる．

$$RMSE = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(y^j_{modeling} - y^j_{actual}\right)^2} \quad (2)$$

なお，$y_{modeling}$，y_{actual} はそれぞれモデリング結果と WattsUp による実際のノード電力値であり，N は評価したデータポイントの数である．

図 12 に，2 つのマザーボードにおける平均二乗誤差を示す．なお，ここでは通信の影響も評価するため，2 ノードによるモデリングの精度も示している．評価結果より，平均二乗誤差は最大でも 0.03 以下であり，十分に高い精度で

ノード消費電力を推定することが可能であると結論付けることができる．

4.3 AdaBoost によるモデリング誤差の解析

前節の結果より，モデリングによりノード消費電力が高い精度で推定できることがわかったが，いくつかの部分で誤差が 10%以上と高くなる部分があり，その解析を行うことは重要である．そこで，AdaBoost アルゴリズム [16] を用い，どのパフォーマンスカウンタ (PMC) が誤差に最も強く影響を及ぼしているかを調査した．

AdaBoost は複数の弱い識別器 h_t の線形結合を用いて，強い識別器 H を作成する機械学習アルゴリズムであり，以下の式で表される．

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right) \tag{3}$$

ここで，α_t は識別器 h_t の重みである．

モデリング誤差に影響する PMC を調べるために，1 ノードのモデリング用に取得したデータを利用し，誤差が 5%以上かあるいは未満かにより 2 つのグループに分割し，それらを AdaBoost により識別を行った．AdaBoost の入力としては，216 個の単位時間あたりの正規化した PMC 値である．本学習結果として，重みの値を見ることで誤差 5%以上か未満かに分割する上で重要となる PMC を知ることができる．

以下に学習の結果判明した，5%以上の誤差へ影響を与える PMC を 5 個，影響度の強い順に示す．

- OTHER ASSISTS SSE TO AVX: Number of transitions from SSE to AVX 256 when penalty applicable.
- OFFCORE REQUESTS OUTSTANDING DEMAND RFO: Offcore outstanding RFO store transactions in SQ to uncore RFO transactions are performed when store operations miss the L2 cache.
- MEM LOAD UOPS RETIRED HIT LFB: Retired load uops which data sources were load uops missed L1 but hit FB due to preceding miss to the same cache line with data not ready.
- L2 RQSTS ALL PF: Any requests from L2 Hardware prefetcher.
- L2 RQSTS RFO MISS: Counts the number of store RFO requests that miss the L2 cache.

これによると，SSE 命令に関する PMC とキャッシュに関連するイベントの影響が大きいことがわかる．これらの PMC を用いることで，モデリングの精度を向上することができると考えられる．

5. まとめと今後の課題

将来のポストペタスケール HPC システムでは消費電力を意識したデザイン，およびアプリケーション最適化が重要であるとの認識のもと，本稿では最近の Intel 社プロセッサに備えられている，プロセッサおよび DRAM の消費電力を計測・制御可能な RAPL インタフェースを用い，電力メータと比較しつつ，アプリケーションを実行させた際の消費電力計測と制御を行った．また，ノード全体の電力の柔軟な計測を可能とすべく，RAPL の計測値を用い，ノード全体の電力のモデリングを行った．

電力計測・制御実験から，RAPL により高い精度で電力を測定，また制御が行えることを確認した．また，モデリングにより，ノード全体電力も高い精度で推定できることがわかった．これらより，RAPL を利用することで，HPC システムの電力制御や電力性能の最適化が可能になると考えられる．

今後は，電力制約適応型システムの実現に向け，より大規模なシステムで，また本稿で実施した実験よりも細粒度な時間間隔で電力計測を行い，アプリケーション毎の電力消費傾向を調査することや，パフォーマンスカウンタ値も利用することで，ノード電力の推定精度を向上させることなどが課題である．

謝辞 本研究は JST CREST の研究課題「ポストペタスケールシステムのための電力マネージメントフレームワークの開発」の一部として行われたものである．

参考文献

[1] `http://www.postpeta.jst.go.jp/reserchers/kondo24.html`.
[2] 松岡　聡：グリーンなスパコンはエクサスケールの夢を見るか - TSUBAME2.0 を例にして，第 10 回 PC クラスタシンポジウム招待講演 (2010).
[3] Rotem, E., Naveh, A., Rajwan, D., Ananthakrishnan, A. and Weissmann, E.: Power-Management Architecture of the Intel Microarchitecture Code-Named Sandy Bridge, *IEEE Micro*, Vol. 32, No. 2, pp. 20–27 (2012).
[4] David, H., Gorbatov, E., Hanebutte, U. R., Khanna, R. and Le, C.: RAPL: Memory Power Estimation and Capping, *Proceedings of the 16th ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED)*, pp. 189–194 (2010).
[5] Intel Corporation: *Intel 64 and IA-32 Architectures Software Developer's Manual* (2013).
[6] Dimitrov, M., Strickland, C., Kim, S., Kumar, K. and Doshi, K.: Intel Power Governor, `http://software.intel.com/en-us/articles/intel-power-governor/`.
[7] Hsu, C. and Feng, W.: A Power-Aware Run-Time System for High-Performance Computing, *Proceedings of the 2005 ACM/IEEE Conference on Supercomputing (SC'05)*, pp. 1– (2005).
[8] Springer, R., Lowenthal, D. K. and Rountree, B.: Minimizing Execution Time in MPI Programs on an Energy-Constrained, Power-Scalable Cluster, *Proceedings of the 11th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP '06)*, pp. 230–238 (2006).
[9] Rountree, B., Lowenthal, D. K. and Supinski, B. R.: Adagio: Making DVS Practical for Complex HPC Applications, *Proceedings of the 23rd International Conference on Super-*

computing (ICS '09), pp. 460–469 (2009).

[10] Rodero, I., Chandra, S., Parashar, M., Muralidhar, R., Seshadri, H. and Poole, S.: Investigating the Potential of Application-Centric Aggressive Power Management for HPC workloads, *Proceedings of the 2010 International Conference on High Performance Computing (HiPC)*, pp. 1–10 (2010).

[11] Witkowski, M., Oleksiak, A., Piontek, T. and Weglarz, J.: Practical Power Consumption Estimation for Real Life HPC Applications, *Future Generation Computer Systems*, Vol. 29, No. 1, pp. 208–217 (2013).

[12] Bircher, W. L. and John, L. K.: Complete System Power Estimation Using Processor Performance Events, Vol. 61, No. 4, pp. 563–577 (2012).

[13] Nagasaka, H., Maruyama, N., Nukada, A., Endo, T. and Matsuoka, S.: Statistical Power Modeling of GPU Kernels using Performance Counters, *Proceedings of the 2010 International Green Computing Conference*, pp. 115–122 (2010).

[14] Yoshii, K., Iskra, K., Gupta, R., Beckman, P., Vishwanath, V., Yu, C. and Coghlan, S.: Evaluating Power-Monitoring Capabilities on IBM Blue Gene/P and Blue Gene/Q, *Proceedings of the 2012 IEEE International Conference on Cluster Computing (CLUSTER 2012)*, pp. 36–44 (2012).

[15] ThinkTank Energy Products Inc.: Watts Up? Plug Load Meters, `https://www.wattsupmeters.com/`.

[16] Freund, Y. and Schapire, R. E.: A Brief Introduction to Boosting, *Proceedings of the 16th International Joint Conference on Artificial Intelligence - Volume 2*, pp. 1401–1406 (1999).