Vol.2016-OS-138 No.3
2016/8/8

仮想化環境における指数平滑化法を用いたVMの性能予測に基づくDVFS制御のKVM上での評価と分析

小柴 篤史[1,a)]　佐藤 未来子[1]　並木 美太郎[1]

概要：筆者らは，近年要求性能が増大しているクラウド環境の省電力化を目標として，仮想化環境向けのDVFS 制御の研究を行っている．先行研究の DVFS 制御手法は，ディスク I/O やネットワーク I/O 性能を考慮しないため省電力効果が十分に得られない，事前に仮想マシン (VM) の性能解析を必要とするため手間がかかる等の課題があった．そこで本研究では，VM の挙動に関する統計情報から指数平滑法を用いて VM の性能を予測し，最適な電圧と周波数を適用する DVFS 制御手法を提案する．本研究ではこれまでの筆者らの研究で行った評価に加えて，既存の DVFS 制御手法のエネルギー削減効果の評価と提案手法との比較，提案手法の VM の挙動予測の精度をより詳しく分析する．提案手法を KVM に実装し，DVFS 機能を搭載した AMD のマルチコア CPU 上で評価した結果，提案手法は最大で 33.4%の消費エネルギーを削減した．また，既存の DVFS 制御手法である cpufreq の ondemand governor と比較して，提案手法は CPU バウンドなベンチマークに対しては 10%以上多くのエネルギーを削減した．

キーワード：省電力制御，指数平滑法，DVFS, KVM 仮想化環境

1. はじめに

近年，クラウドを利用するデータセンタにおいて技術革新が進んでいる．クラウドでは，サーバの集約やマシン構成の柔軟な変更，低価格化などのメリットから仮想化技術が多く利用されている．仮想化技術を利用すると，一つの物理デバイスを複数の独立した仮想化環境に分割し，未利用の資源を効率的に活用できる．仮想化技術は計算機の実行性能の向上に有効だが，近年のプロセッサやメモリの処理性能の向上に伴う消費電力の増大が問題となっている．2010 年のデータセンタにおける電力消費量は，年間 2034〜2718 億 kWh で，これは全世界の電力消費量の 1.1%から 1.5%にものぼる [1]．このように，データセンタの省電力化は現代において大きな課題となっており，データセンタの電力効率を向上するための研究が進められている [2], [3]．

プロセッサは計算機システム全体の消費電力のうち，多くの割合を占める場合が多いことから，プロセッサに関する省電力化の研究が盛んに行われている．特にプロセッサの電力削減に有効な手法として，Dynamic Voltage and Frequency Scaling (DVFS) の研究が多く行われている．DVFS はプロセッサの電源電圧と動作周波数を動的に変化させる技術で，プロセッサの要求性能が低い場合に電圧と周波数を下げることで，効果的に電力を削減できる．しかし，プロセッサの要求性能が高いときに周波数を下げると性能が大きく低下するため，周波数を下げることができるタイミングを正確に予測することが重要となる．実行時のプロセッサの性能に基づく DVFS 制御手法として，CPU 利用率に応じて動作周波数を切り替える cpufreq[4] のほか，様々な研究が行われている [5], [6], [7].

既存の DVFS 制御手法の問題点として，ディスク，ネットワーク I/O などの情報を考慮しない点と，データセンタなどのクラウド環境向けの手法が少ない点が挙げられる．一般的な仮想化環境においては，1 台の計算機上で複数の仮想マシン (VM) が動作する．各 VM は VM の挙動によって最適な DVFS 戦略が異なるため，おのおのの挙動に適した制御を行う必要がある．しかし，既存の手法 [8] はディスク I/O やネットワーク I/O を考慮しないため，VM の挙動を正確に予測することは難しく，I/O バウンドなアプリケーションに対しては性能が低下する恐れがある．また，仮想化環境であることを考慮しないため，VM ごとに最適な制御を行うことが難しい．そこで筆者らは，CPU，ディスク I/O，ネットワーク I/O などの VM の様々な挙動を考慮した，仮想化環境向けの DVFS 制御の研究を行っている．筆者らの先行研究 [9] において，仮想マシンモニタ (VMM) が事前学習に基づき VM の挙動から消費エネル

[1] 東京農工大学
[a)] koshiba@namikilab.tuat.ac.jp

ギーと処理性能を予測して DVFS 制御を行う手法を提案し，更に [10] では事前学習なしでより高精度な DVFS 制御を行う手法を提案した．しかし，先行研究で行われた評価は不十分であり，既存の DVFS 制御手法に対する優位性を示すためには，より詳細な評価実験が必要である．

本研究では，筆者らの先行研究 [10] で提案した，指数平滑法を用いた仮想化環境向けの DVFS 制御手法について，KVM 仮想化環境および AMD のマルチコアプロセッサ上でのより詳細な評価と検証を行う．提案手法では VMM が実行時の VM の挙動を周期的に監視し，得られた情報に基づき指数平滑法を用いて次の周期の VM の処理性能や消費電力を予測する．予測された VM 性能から，VM の処理性能を保ちつつ消費エネルギーが最小となる最適な周波数を決定し適用する．本研究では，関連する筆者らの既存の発表 [10] で得られた実験結果を更に詳しく解析するため，代表的な DVFS 制御手法である cpufreq の評価と提案手法との比較を行い，また提案手法の VM の挙動予測の精度を分析する．これらの多角的な検証によって，既存の発表 [10] では不明瞭であった提案手法の有用性を明らかにする．

2. 関連研究と課題

消費電力の増加が顕著であるデータセンタの省電力化に向けて様々な研究が進められている．Nathuji ら [11] が提案している VirtualPower は，異なる仮想化環境のプラットフォーム間や同一プラットフォーム間で，VM ごとに省電力制御を行い，システム全体の省電力化を実現する．Beloglazov ら [12] は，物理リソースの利用率，消費電力，演算性能などを考慮した VM の動的再分配手法を提案している．資源利用率が低い VM をアイドルマシンへマイグレーションし，アイドル物理マシンをシャットダウンすることで電力を削減する．このように，データセンタの省電力化は重要な課題となっている．

データセンタの電力消費量の割合の多くを占める計算機システムの省電力化手法として，DVFS が注目されている．プロセッサ向けの DVFS 機能として，Intel の Enhanced Intel SpeedStep Technology (EIST)[13]，AMD の PowerNow!テクノロジ [14] などがある．これらの機能は，マルチコアプロセッサの各 CPU コアの動作モードを表す P-State を各コアに設定することで，動的にプロセッサの周波数と電圧を変更できる．システムの消費電力は動作周波数および電圧の二乗に比例するため，DVFS は高い電力削減効果を得られる．しかし，DVFS 機能によって周波数を低くすると，演算性能またはスループットも低下し，アプリケーションの実行時間が延びる．そのため，必ずしも周波数を最低にすれば消費エネルギーを最も削減できるとは限らない．これらの DVFS 機能を活用するためには，実行時のシステムの負荷や要求に応じて適切な電圧と周波数を設定することが重要である．

代表的な DVFS 制御手法として，Linux システムに搭載された cpufreq モジュールと各種 governor システム [4] が用いられている．cpufreq は，CPU コアごとに設定された governor の戦略に応じて DVFS 制御を行う手法で，特に ondemand governor は周期的に計測したシステム稼働時の cpu 使用率と，あらかじめ設定された閾値に応じて，周波数と電圧を切り替える．これにより，実行時の CPU 負荷に適した動的な DVFS 制御を可能にしている．この他にも，プロセッサの性能予測に基づく DVFS 制御手法が多く提案されている [5], [6], [7].

cpufreq などの従来の DVFS 制御は，クラウド環境においては制御方法や精度の点で課題がある．まず，クラウド環境においては複数の VM やゲスト OS が存在し，各 VM は直接ハードウェアを制御できない．既存手法は主にアプリケーションや OS が直接 CPU を制御するため，従来の手法をクラウド環境にそのまま適用することは難しい．また，昨今のデータセンタでは，Web サーバや DB サーバなど，様々な種類の VM が稼働している．これらのシステムではディスク I/O のボトルネックやネットワーク I/O のスループットがシステム性能に大きな影響を及ぼすが，cpufreq をはじめとする従来手法 [7] では CPU のみの情報を元に性能予測を行うため，正確に VM の挙動を予測することが難しい．

これらの課題に対応するため，サーバやクラウド環境向けの DVFS 制御手法がいくつか提案されている．Snowdon ら [15] が提案している Koala は，CPU だけでなく，メモリとメモリパスを考慮し，演算性能と消費エネルギーを予測し，DVFS による電力制御を提案している．また，Deng ら [16] が提案している CoScale は，実行時に各 CPU コアとメモリの処理性能を予測し，CPU コアとメモリの DVFS を行う手法を提案している．しかし，これらの手法は CPU とメモリの挙動を考慮した DVFS 制御を実現しているものの，ディスク I/O やネットワーク I/O のスループットは考慮しておらず，クラウド環境に最適な DVFS 制御手法はまだ検討の余地がある．

このように，従来の DVFS 制御手法はディスク I/O，ネットワーク I/O に関する情報を考慮できず，個々の VM の挙動に適した制御が難しい点が課題となる．そこで，筆者らはこの既存手法の課題を解決するため，CPU やメモリだけでなく，ディスク I/O，ネットワーク I/O のスループットを考慮した DVFS 制御の研究を進めている [9][10]. 本研究では，筆者らの先行研究 [10] で提案した，指数平滑法を用いた DVFS 制御手法について，先行研究で不十分であった提案手法の評価と結果の解析を詳しく行う．これにより，提案手法の有用性を明らかにし，クラウド環境における VM の挙動に適した DVFS 制御が可能であることを示す．

3. 本研究の目標

本研究では，VM 稼働中の CPU，メモリ，ディスク I/O，ネットワーク I/O の各挙動に適した仮想化環境向けの DVFS 制御の提案，および提案手法による高効率なプロセッサの電力削減を目的とする．目的達成のための本研究の目標を示す．

仮想化環境に適した DVFS 制御の実現:

本研究では，CPU だけでなく，ディスク I/O やネットワーク I/O の情報を考慮した VM の DVFS 制御手法を提案する．提案手法は，時系列分析手法の一つである指数平滑法を用いて実行時の VM の挙動を予測することで，より正確な VM の性能解析を実現する．これにより，仮想化環境における実行時の VM の要求性能に適した DVFS 制御を可能にする．

KVM 仮想化環境における評価と結果の分析:

筆者らの先行研究 [10] で提案手法のプロトタイプの実装と評価を行ったが，その効果の検証が十分に行われていなかった．そこで本研究では，提案手法の省エネルギー効果をより具体的に示すため，(1) 既存の DVFS 手法である cpufreq との比較，(2) 提案手法の VM 挙動予測の精度の検証を行う．これにより，既存手法に対する提案手法の優位性を定量的に明らかにする．

4. 指数平滑法を用いた DVFS 制御

本研究では，マルチコアプロセッサ上で稼働する仮想化環境を対象とした DVFS 制御手法を提案する．本研究の対象システムとして，マルチコアプロセッサを持つ PC 上で VMM が稼働し，更に VM 上で複数台の VM が稼働している状況を想定する．システムで稼働する個々の VM の挙動に適した DVFS 制御を行うため，提案手法は仮想化環境の仮想マシンモニタ (VMM) に適用される．VM 単位に異なる性能条件を設定可能にすることで，動的に変化する VM の稼働状況に対応する．

提案手法において VMM は，各 VM を稼働するのに最適な CPU の動作周波数を決定するため，それぞれの周波数で VM を稼働したときの性能を指数平滑法を用いて予測し，システムの性能要件を満たしつつ最も消費エネルギーが小さくなると予測した周波数を適用する．本章では，指数平滑法を用いた VM の性能予測に基づく DVFS 制御手法の詳細について述べる．

4.1 指数平滑法に基づく VM の性能予測

本論文では，行列は太字の大文字，ベクトルは太字のイタリック体小文字で表記する．本研究では，動作周波数によって変化する各 VM の処理性能を考慮した DVFS 制御を行うため，指数平滑法を用いて各動作周波数における VM 性能を予測する．この VM の性能予測をシステムで CPU コアに設定可能な全ての動作周波数について行い，最適な動作周波数を決定する．本論文では，VM の演算性能，スループット，消費電力の 3 種類の値を VM の性能指標として用いる．あるタイムスロット t において VM を動作周波数 f で稼働した時の VM 性能の予測値群をベクトル $\boldsymbol{p_{f,t}}$ で表し，式 (1) で定義する．

$$\boldsymbol{p_{f,t}} = \begin{pmatrix} p^f_{perf,t} \\ p^f_{thrp,t} \\ p^f_{pow,t} \end{pmatrix} \tag{1}$$

ここで，$p^f_{perf,t}$ は VM の演算性能の予測値で，タイムスロット t における IPS(Instruction Per Second) を表す．$p^f_{thrp,t}$ は VM の I/O スループットの予測値で，単位時間あたりのネットワークのデータ転送量を表す．$p^f_{pow,t}$ は VM の消費電力の予測値で，VM が稼働している CPU コアの消費電力を表す．

提案手法では時系列分析手法の一つである指数平滑法を用いて，次のタイムスロット $t+1$ における VM 性能 $\boldsymbol{p_{f,t+1}}$ を予測する．指数平滑法は，周期的に計測した過去の長期間のデータから次の周期のデータを予測する手法で，VM のタイムスライス単位での周期的な性能予測を行う提案手法に適している．次のタイムスロット $t+1$ における VM 性能の予測値 $\boldsymbol{p_{f,t+1}}$ は，指数平滑法を用いて式 (2) で表される．

$$\boldsymbol{p_{f,t+1}} = \mathbf{A_{f,t}}\boldsymbol{x_{f,t}} + (\mathbf{E} - \mathbf{A_{f,t}})\boldsymbol{p_{f,t}} \tag{2}$$

ここで，$\mathbf{E}$ は 3 行 3 列の単位行列を表す．また，行列 $\mathbf{A_{f,t}}$ は平滑化係数，ベクトル $\boldsymbol{x_{f,t}}$ は $\boldsymbol{p_{f,t}}$ の実測値で，それぞれ式 (3)，式 (4) で表される．

$$\mathbf{A_{f,t}} = \begin{pmatrix} a^f_{perf,t} & 0 & 0 \\ 0 & a^f_{thrp,t} & 0 \\ 0 & 0 & a^f_{pow,t} \end{pmatrix} \tag{3}$$

$$\boldsymbol{x_{f,t}} = \begin{pmatrix} x^f_{perf,t} \\ x^f_{thrp,t} \\ x^f_{pow,t} \end{pmatrix} \tag{4}$$

ここで，$\mathbf{A_{f,t}}$ の要素 $a^f_{perf,t}$ は平滑化係数，$\boldsymbol{x_{f,t}}$ の要素 $x^f_{perf,t}$ は統計情報 $p^f_{perf,t}$ の実測値を表す．平滑化係数 $a^f_{perf,t}$ は 0 から 1 の範囲で任意の値をとることができるが，本手法では VM の挙動が急に変化した場合も正確な予測を行うため，$a^f_{perf,t}$ を式 (5) で定め，実行時に VMM が算出する．

$$a^f_{perf,t} = \frac{|x^f_{perf,t} - p^f_{perf,t}|}{x^f_{perf,t} + p^f_{perf,t}} \tag{5}$$

$a^f_{thrp,t}, a^f_{pow,t}$ についても式 (5) と同様の方法で算出する．

VM の挙動の変化が大きい場合は実測値と予測値の差が大きくなるため，並列化係数が 1 に近くなり，実測値により重みを付けた予測を行う．これにより，VM の挙動が急激に変動した場合でも，変化に追従する予測を可能にする．

式 (2) から $\boldsymbol{p_{f,t+1}}$ を各動作周波数について予測するには，システムで設定可能な動作周波数それぞれについて実測値 $\boldsymbol{x_{f,t}}$ を VMM が計測する必要がある．しかし，全ての値を実測することは難しい．まず，VM の演算性能とスループットの実測値 $x^f_{perf,t}, x^f_{thrp,t}$ については，計測時に VM が稼働していた CPU の動作周波数については求めることができるが，他の周波数で動作させた場合については実測できない．また，VM の消費電力 $x^f_{pow,t}$ は，プロセッサが自身の電力を計測する機構を備えておらず，いずれの動作周波数についても実測できない．そこで提案手法では，これらの実測が難しい VM 性能値を，回帰分析を用いて，システムで計測可能な他のパラメータから推定する．

提案手法では回帰分析を用いて，パフォーマンスカウンタ等から取得したプロセッサ，ディスク I/O，ネットワークに関する各種パラメータから，VM 性能の実測値 $\boldsymbol{x_{f,t}}$ を推定する．タイムスロット t において，VM を動作周波数 f の CPU 上で稼働したときの VM 性能の実測値 $\boldsymbol{x_{f,t}}$ は，回帰分析を用いて式 (6) で推定される．

$$\boldsymbol{x_{f,t}} = \mathbf{B_f}\boldsymbol{s_t} + \boldsymbol{c_f} \tag{6}$$

ここで，行列 $\mathbf{B_f}$，ベクトル $\boldsymbol{c_f}$ は，$\boldsymbol{x_{f,t}}$ の回帰係数，ベクトル $\boldsymbol{s_t}$ はパフォーマンスカウンタ等から取得した VM のパラメータ群を表し，それぞれ式 (7)，式 (8) で表される．

$$\mathbf{B_f}=\begin{pmatrix} b^f_{perf,1} & b^f_{perf,2} & \cdots & b^f_{perf,n} \\ b^f_{thrp,1} & b^f_{thrp,2} & \cdots & b^f_{thrp,n} \\ b^f_{pow,1} & b^f_{pow,2} & \cdots & b^f_{pow,n} \end{pmatrix}, \boldsymbol{c_f}=\begin{pmatrix} c^f_{perf} \\ c^f_{thrp} \\ c^f_{pow} \end{pmatrix} \tag{7}$$

$$\boldsymbol{s_t} = \begin{pmatrix} s_{1,t} \\ s_{2,t} \\ \vdots \\ s_{n,t} \end{pmatrix} \tag{8}$$

ここで，$\mathbf{B_f}$ の要素 $b^f_{perf,i}, b^f_{thrp,i}, b^f_{pow,i}$ $(i = 1, 2, ..., n)$ と $\boldsymbol{c_f}$ の要素 $c^f_{perf}, c^f_{thrp}, c^f_{pow}$ はそれぞれ $x^f_{perf,t}, x^f_{thrp,t}, x^f_{pow,t}$ の回帰係数を表す．また，$\boldsymbol{s_t}$ の要素 $s_{i,t}(i = 1, 2, ..., n)$ は，回帰分析で $\boldsymbol{x_{f,t}}$ の推定に用いる n 種類のパラメータを表す．本研究では，L2 キャッシュミス回数 $s_{1,t}$，命令あたりの L3 キャッシュミス回数 $s_{2,t}$，命令あたりの L1 キャッシュメモリアクセス回数 $s_{3,t}$，命令あたりのネットワークのパケット到着総数 $s_{4,t}$，ディスクの読込み回数 $s_{5,t}$，ディスクの読込みサイズ $s_{6,t}$，ディスクの書込み回数 $s_{7,t}$，ディスクの書込みサイズ $s_{8,t}$ の 8 種類のパラメータを用いる．

回帰係数 $\mathbf{B_f}$, $\boldsymbol{c_f}$ は，$\boldsymbol{x_{f,t}}, \boldsymbol{s_t}$ を各動作周波数について実測し，実測値から最小二乗法を用いて求める．提案手法では回帰係数 $\mathbf{B_f}$, $\boldsymbol{c_f}$ の算出はシステムのコンフィグレーション時に一度だけ行うものとする．このコンフィグレーション時に，システムで設定可能な全ての周波数について多変量回帰分析の予測式を構築する．システム稼働中は，計測した $\boldsymbol{s_t}$ から式 (6) を用いて $\boldsymbol{x_{f,t}}$ を推定し，式 (2) を用いて $\boldsymbol{p_{f,t+1}}$ を求める．これを VMM が各動作周波数について行い，動作周波数ごとの VM 性能予測を行う．

4.2 最適な動作周波数，電圧の決定と適用

VMM は，指数平滑法を用いて予測した各動作周波数の VM 性能 $\boldsymbol{p_{f,t+1}}$ から，VM を稼働するのに最適な CPU の動作周波数を決定する．ここで，DVFS 制御時の CPU の動作周波数の低減による VM の演算性能とスループットの大幅な低下を防ぐため，本研究ではそれぞれの限界値を設定する．タイムスロット t において実測した VM の演算性能 $x_{perf,t}$ とスループット $x_{thrp,t}$ の平均値をそれぞれの限界値として設定し，VM の性能が限界値を常に上回るように DVFS 制御を行うことで，VM の性能維持を図る．この平均値を算出する際，次回のシステム状況は直近の挙動に大きな影響を受ける可能性が高いと仮定し，影響因数 $g(0 < g < 1)$ を定義して直近の VM 性能に重みを付ける．本研究では，VM の演算性能の限界値を $Ave_{perf,t}$，スループットの限界値を $Ave_{thrp,t}$ で表し，式 (9) で定義する．

$$\begin{pmatrix} Ave_{perf,t} \\ Ave_{thrp,t} \end{pmatrix} = g\begin{pmatrix} x_{perf,t} \\ x_{thrp,t} \end{pmatrix} + (1-g)\begin{pmatrix} Ave_{perf,t-1} \\ Ave_{thrp,t-1} \end{pmatrix} \tag{9}$$

VMM はシステムで適用可能な各動作周波数について，VM の演算性能の予測値 $p^f_{perf,t+1}$ とスループットの予測値 $p^f_{thrp,t+1}$ を式 (9) で算出した限界値 $Ave_{perf,t}, Ave_{thrp,t}$ と比較し，それぞれの限界値を上回る動作周波数を探索する．ここで，条件を満たす動作周波数 f の集合 $\mathbb{F}$ は，式 (10) で表される．

$$\begin{aligned} \mathbb{F} = \{f \mid & (p^f_{perf,t+1} > Ave_{perf,t})\& \\ & (p^f_{thrp,t+1} > Ave_{thrp,t})\} \end{aligned} \tag{10}$$

そして，周波数集合 $\mathbb{F}$ のうち，消費電力の予測値 $p^f_{pow,t+1}$ が最も小さくなる f を次の周波数 f_{t+1} として式 (11) で定める．

$$f_{t+1} = \{f \mid min(p^f_{pow,t+1}), f \in \mathbb{F}\} \tag{11}$$

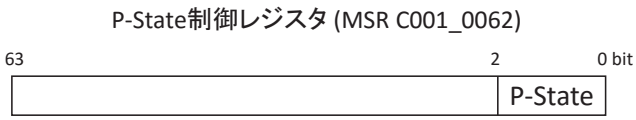
もしも全ての周波数，電圧でスループット及び演算性能が限界値より低く，条件を満たす周波数が存在しない場合は，周波数と電圧を最大値に設定する．以上の処理を VMM が各 VM に対して周期的に行い，実行時の VM の挙動に適した CPU コア単位での DVFS 制御を実現する．

5. 実装

提案手法が実システムに適用可能であることを示し，そ

表 1　評価環境

名称	製品名など
CPU	AMD FX-8370 (8 cores / 4.0 GHz)
メモリ	16 GB
仮想化技術	kvm-kmod 3.4, qemu-kvm 1.0
各 VM の VCPU	2
各 VM のメモリ	2GB



P-State	レジスタ値	動作周波数 [GHz]	電圧 [V]
P-0	0x0	4.0	1.35
P-1	0x1	3.4	1.2
P-2	0x2	2.8	1.1
P-3	0x3	2.1	0.9875
P-4	0x4	1.4	0.8625

図 1　P-State 制御レジスタと各 P-State の設定値

の省電力性能を検証するため，提案手法を KVM および DVFS 機能を持つ AMD プロセッサ向けに実装し，評価する．本研究では，表 1 に示す環境に提案手法を実装した．以下に提案手法の実装対象とした AMD プロセッサの FX-8370，および KVM に実装した DVFS 制御機構の詳細を示す．

5.1　AMD FX-8370 プロセッサ

前章までに述べた提案手法の有効性を評価するため，DVFS 機能を搭載する AMD の FX-8370 プロセッサを対象に提案手法を実装する．FX-8370 はコア数 8 のマルチコアプロセッサで，動作周波数は最大 4GHz，TDP が 125W，命令 256KB・データ 128KB の各コア占有 L1 キャッシュ，8MB の L2 キャッシュ，8MB の L3 キャッシュを持つ．各 CPU コアは固有の P-State 制御レジスタを持っており，レジスタの値を更新することで，5 段階の動作周波数と電圧を切り替えることができる．図 2 に FX-8370 の P-State 制御レジスタおよび各 P-State に設定したときの CPU コアの動作周波数と電圧を示す．本研究では，4 章で述べた DVFS 制御手法を用いて，制御対象の仮想マシンの VCPU が割り当てられたコアの P-State を適切に切り替えることで，プロセッサの高効率な省電力化を図る．

また，各 CPU コアはそれぞれパフォーマンスカウンタ (PMC) を持ち，一定期間におけるキャッシュのアクセス回数やミス回数，命令実行数などを計測することができる．本研究では，4 章で述べた VM の性能予測のための統計情報として，L1 キャッシュアクセス回数，L2 キャッシュミス回数，L3 キャッシュミス回数を PMC を用いて計測する．

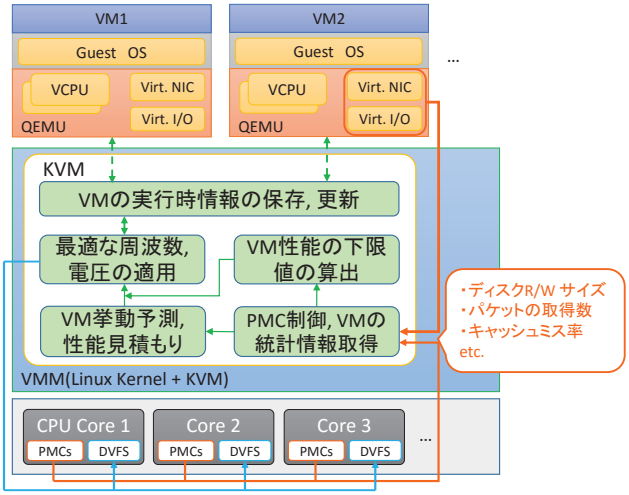


図 2　KVM における提案システムの全体構成

5.2　KVM における DVFS 制御機構

本研究では，ゲスト OS やアプリケーションを改変することなく高効率な DVFS 制御を実現するため，完全仮想化環境を提供する KVM の VMM レイヤに本手法を適用する．本研究で想定している仮想化環境では，VMM 上で複数の VM が稼働し，それぞれの VM が複数の仮想 CPU(VCPU) をもち，全ての VCPU は物理 CPU に対応付けられる．各 VM が持つ VCPU ごとに適切な DVFS 制御を行うため，本研究では提案手法を KVM の VCPU スケジューラに実装した．また，KVM に実装した DVFS 制御機構が VM のディスク I/O とネットワーク I/O の統計情報を計測するため，各 I/O をエミュレーションする QEMU に各値を定期的に計測，保持する機能を実装した．これにより，VCPU コンテキストスイッチごとに VM 稼働中のパラメータの計測および DVFS 制御を行う．

図 2 に KVM 仮想化環境における DVFS 制御機構の全体構成を示す．KVM は VCPU コンテキストスイッチの発生時，直前まで VCPU が稼働していた物理 CPU の PMC，および QEMU の仮想 I/O から VM の統計情報を取得する．取得した統計情報から 4 章に示した手法に従い指数平滑法による性能予測，VM 性能の限界値の算出をそれぞれ行う．予測した VM 性能と限界値から最適な動作周波数と電圧を決定し，対応する P-State を次に VCPU が割り当てられる物理 CPU に適用したのち，VCPU コンテキストスイッチを抜ける．

なお，本評価環境にはプロセッサが自身の消費電力を直接計測する機能がないため，VM の消費電力 $x^f_{pow,t}$ の回帰分析式は，外部から取り付けた電流計で実測したプロセッサの電力から手動でパラメータを学習させ，構築する．

6.　評価

前節の内容に基づき提案手法を実装した KVM および QEMU を，AMD の FX-8370 プロセッサを搭載した計算

機上で稼働し，電力評価を行った．本研究において，VM の消費電力の値は，VM 稼働中のプロセッサの消費電力を外部に接続した電流計を用いて実測する．ただし，マルチコアプロセッサにおいては，共有キャッシュなどのコア間の共有資源が存在するため，各コアの消費電力を直接計測することは困難である．そこで本研究では，プロセッサ全体の消費電力を電流計を用いて計測し，これをコア数で除算した値を各コアの消費電力として評価する．

6.1 提案手法と cpufreq のエネルギー削減率

システムの電力削減効果を評価するために，提案手法を適用した場合と適用しない場合 (常に周波数 MAX)，また既存の DVFS 手法である cpufreq の ondemand governor を適用した場合のそれぞれの消費エネルギーを計測し，比較した．提案手法について，VM 性能の限界値の算出に用いる影響因数 g は 0.01 に設定した．cpufreq について，周波数切り替えの閾値となる CPU 使用率はデフォルトの 95%，制御周期は 10ms とした．各ベンチマーク実行中の消費電力と実行時間を計測し，ベンチマーク実行によって消費されるエネルギー遅延積を算出した．異なる挙動を持つ VM に対する評価を行うため，評価には CPU・メモリバウンドなベンチマークである SPECCPU2006，ネットワークベンチマークである httperf，ディスクベンチマークである bonnie++を用いた．SPECCPU 2006 は，CPU バウンドな 444.namd, 450.soplex, メモリバウンドな 462.libquantum, 470.lbm, その中間の 456.hmmer の計 5 種類を用いた．また，Httperf を用いた評価は，提案手法を適用した KVM 上で Apache 2.2.22 を稼働し，他の計算機から Httperf を実行して Apache に負荷をかけ，そのときの消費電力を計測して行った．このとき，Apache に送る 1 秒あたりのリクエスト数を変更し，VM に掛ける負荷を調節しながら行った．Apache の MPM は prefork とし，子プロセスの稼働数は 200 とした．本研究では，1 秒あたりのリクエスト数 100, 200, 300, 400, 500, 1000, 1500 の 7 種類について評価した．

cpufreq，および提案手法について，常に周波数 MAX で動作した場合と比較した時との割合を算出した．図 1 に SPECCPU 2006 の消費電力の割合，図 2 に実行時間の割合，図 3 にエネルギー遅延積の割合を示す．また，図 4 に Httperf のエネルギー遅延積の割合を示す．CPU・メモリバウンドな SPECCPU 2006 の 5 種のベンチマークに対しては，提案手法は平均 11.1%, 最大 13.0%(lbm) のエネルギー遅延積を削減した一方，cpufreq は平均 0.6%, 最大 1.4%(soplex) のエネルギー削減となり，ほとんど効果が得られなかった．ディスク I/O バウンドな bonnie++に対しては，提案手法は 33.2%，cpufreq は 36.8%と，わずかに cpufreq のエネルギー削減率が高い結果となった．ネットワーク I/O バウンドな httperf に対しては，提案手法は平均 38.0%，最大 42.5%(秒間 300 リクエストの時) のエネルギー削減，cpufreq は平均 51.0%，最大 71.6%(秒間 400 リクエストの時) のエネルギー削減となった．

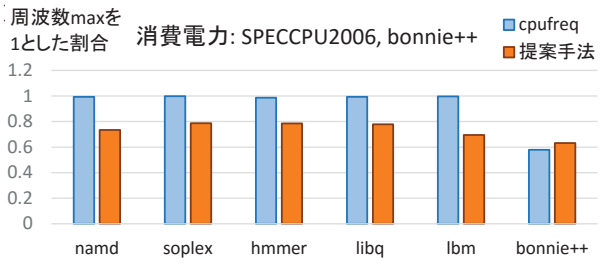


図 3 SPECCPU 2006 の消費電力割合

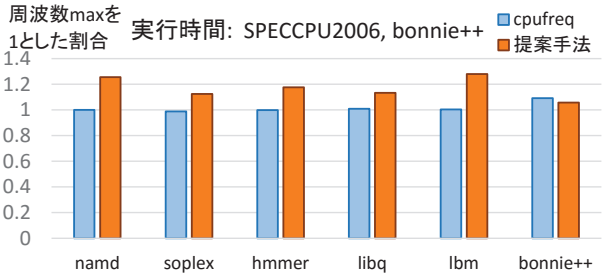


図 4 SPECCPU 2006 の実行時間割合

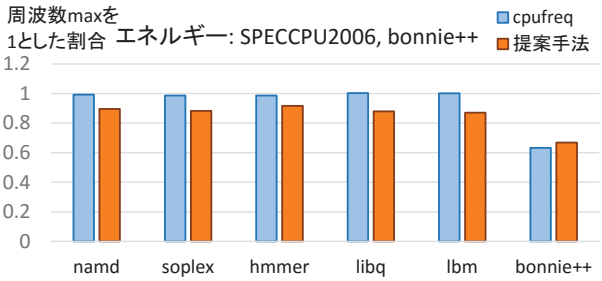


図 5 SPECCPU 2006 の消費エネルギー割合

SPECCPU 2006 の各種ベンチマークに対しては，cpufreq はほとんどエネルギー削減効果が得られていない一方で，提案手法は cpufreq よりも 10%以上多くのエネルギーを削減している．これは，プログラム実行中の CPU 利用率がほぼ 100%であるため，ondemand governor が常に最大の動作周波数を設定し，エネルギー削減効果が得られないためである．また，bonnie++に対しては，提案手法と cpufreq はほぼ同等のエネルギー削減率が得られている．

一方で，Httperf については，秒間のリクエスト数が 500 以下の時は cpufreq のエネルギー削減率が提案手法を大きく上回っているが，リクエスト数が 1000 以上のときは提案手法のエネルギー削減率が上回る．これは，リクエスト数を増やしていくと CPU 使用率が 100%になり，cpufreq は最大の動作周波数で VM を稼働するためである．

6.2 提案手法の VM 挙動予測の精度

前節で示した提案手法のエネルギー削減率について，提案手法の VM 性能の予測の精度がエネルギー削減率に与え

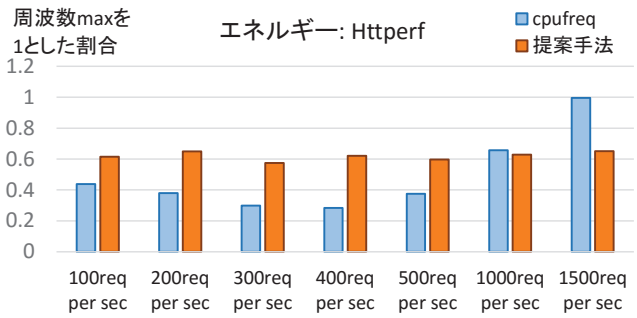


図 6　Httperf の消費エネルギー割合

る影響を検証するため，提案手法による VM 性能の予測値と実測値の誤差をベンチマークごとに求め，比較する．タイムスロット t における VM 性能の予測値を p_t，実測値を x_t として，予測の絶対誤差を ε，相対誤差を ε_R で表し，それぞれ式 (12)，式 (13) で算出する．

$$\varepsilon = p_t - x_t \tag{12}$$

$$\varepsilon_R = \frac{p_t - x_t}{x_t} \tag{13}$$

本実験では，提案手法によって予測した VM の演算性能 $p_{perf,t}$，スループット $p_{thrp,t}$，消費電力 $p_{pow,t}$ について，ベンチマーク実行中に得られた全サンプルの ε と ε_R を求め，それらの平均値と分散を算出する．更に，VM 性能の実測値の周期的な変動が予測の精度を低下させる可能性を考慮し，VM 性能の実測値そのものの平均値と分散を求める．

表 2, 3, 4 に，VM の演算性能，スループット，消費電力の実測値，絶対誤差，相対誤差の平均値と分散をそれぞれ示す．なお，表中の E は指数表記を表し，$4.0\text{E}+14 = 4.0 \times 10^{14}$ である．まず，表 2 の結果から，VM の演算性能の予測については，ほとんどのベンチマークで誤差が非常に大きく，精度が低いことが分かる．これは，VM の演算性能の実測値の分散が非常に大きいことから，VM の挙動が周期的に大きく変動しているために予測が外れていると考えられる．今後，このような VM の挙動が周期的に変動する場合における予測精度の向上方法について検討する．一方，表 3 に示した VM のスループットの予測結果から，httperf におけるスループット予測の相対誤差は最大で 2 割程度であることが分かる．これは，httperf は単位時間ごとに一定の割合でネットワーク通信を行うベンチマークのため，実測値の分散が比較的小さく，提案手法によって高精度な予測が可能となったと考えられる．今後，VM の演算性能の予測の精度をより向上する方法の検討，および提案手法のエネルギー削減率と予測精度の関連性の更なる解析と検証を進める．

7. おわりに

本論文では，仮想化環境が多く用いられるデータセンタの省電力化を目的とし，指数平滑法による仮想マシンの動作予測に基づいた省電力化を行なう VMM の設計，実装と評価について述べた．提案手法は，VMM が各 VM の稼働状態を管理し，CPU 演算性能，データ通信のスループット，消費電力に関する情報を計測する．次に，指数平滑法により各 VM を異なる動作周波数で動作させたときの CPU 演算性能とスループット，消費電力を予測する．そして，予測した VM 性能がシステム要求を満たし，かつ消費エネルギーが最小となるような周波数を，次に VM が稼働する CPU に設定する．本提案手法を完全仮想化を提供する VMM の一つである KVM に実装し，DVFS 機能を搭載する AMD のプロセッサ上で評価した．評価結果より，提案手法は CPU バウンドなベンチマークに対して最大 13%，ディスク I/O バウンドなベンチマークに対して最大 33.2%の消費エネルギーを削減した．更に提案手法は，既存の DVFS 手法である cpufreq と比較しても，CPU バウンドなベンチマークに対しては 10%以上の高いエネルギー削減率を達成した．今後の課題としては，VM の挙動の学習を十分に行った場合の提案手法の電力評価，複数の VM が稼働している場合の提案手法の評価，VM 性能の限界値についてのより詳細な検討があげられる．

参考文献

[1] Koomey, J. G.: Growth in data center electricity use 2005 to 2010, Analytics Press (online), available from ⟨http://www.analyticspress.com/datacenters.html⟩ (accessed 2016-07-06).

[2] Pegus, II, P., Varghese, B., Guo, T., Irwin, D., Shenoy, P., Mahanti, A., Culbert, J., Goodhue, J. and Hill, C.: Analyzing the Efficiency of a Green University Data Center, *Proceedings of the 7th ACM/SPEC on International Conference on Performance Engineering*, ICPE '16, New York, NY, USA, ACM, pp. 63–73 (online), DOI: 10.1145/2851553.2851557 (2016).

[3] Takouna, I., Dawoud, W., Sachs, K. and Meinel, C.: A Robust Optimization for Proactive Energy Management in Virtualized Data Centers, *Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering*, ICPE '13, New York, NY, USA, ACM, pp. 323–326 (online), DOI: 10.1145/2479871.2479917 (2013).

[4] Brodowski, D.: Linux CPUfreq CPUFreq Governors, Linux Foundation (online), available from ⟨https://www.kernel.org/doc/Documentation/cpufreq/governors.txt⟩ (accessed 2016-07-05).

[5] Weiser, M., Welch, B., Demers, A. and Shenker, S.: Scheduling for Reduced CPU Energy, *Proceedings of the 1st USENIX Conference on Operating Systems Design and Implementation*, OSDI '94, Berkeley, CA, USA, USENIX Association, (online), available from ⟨http://dl.acm.org/citation.cfm?id=1267638.1267640⟩ (1994).

[6] Wu, Q., Reddi, V. J., Wu, Y., Lee, J., Connors, D., Brooks, D., Martonosi, M. and Clark, D. W.: A dynamic compilation framework for controlling microprocessor energy and performance, *38th An-*

表 2　各ベンチマーク実行中の VM の演算性能の予測精度

Benchmark	エネルギー削減率	実測値 [IPS]		絶対誤差 ε		相対誤差 ε_R	
		平均値	分散	平均値	分散	平均値	分散
namd	10.4%	2469177.9	4.0E+14	1758702.2	6.8E+14	105.0	3756761.5
soplex	11.7%	880540.7	1.0E+13	15675296.9	1.6E+17	435.1	1.09E+9
hmmer	8.3%	309952.6	1.7E+13	2152191.2	8.4E+15	18.3	242895.3
libquantum	12.0%	5006376.6	2.4E+15	-1684301.4	7.0E+15	35.7	24272860.0
lbm	13.0%	2143010.1	8.7E+13	22667560.7	1.9E+16	274.5	1.5E+8
bonnie++	33.2%	572301.5	1.4E+14	1601721.1	1.9E+16	58.1	1.3E+8
httperf 100req	38.5%	309952.6	1.7E+13	2152191.2	8.4E+15	18.3	242895.3
httperf 200req	35.1%	1143519.5	3.3E+13	27955.2	1.2E+15	-0.4	8095.7
httperf 300req	42.6%	1722165.6	4.2E+13	1020893.5	7.3E+14	0.9	2281.1
httperf 400req	37.9%	2843109.6	1.4E+14	-1979190.0	1.4E+15	-1.0	5908.9
httperf 500req	40.3%	5517964.1	5.3E+14	-1.6E+8	2.4E+17	-108.2	859521.9
httperf 1000req	37.2%	38950901.8	6.5E+15	-796304.7	1.5E+16	6.5	18304.7
httperf 1500req	34.9%	42823414.6	1.5E+16	27177967.9	8.8E+16	16.0	2260631.6

表 3　各ベンチマーク実行中の VM のスループットの予測精度

Benchmark	エネルギー削減率	実測値 [kbps]		絶対誤差 ε		相対誤差 ε_R	
		平均値	分散	平均値	分散	平均値	分散
namd	10.4%	0.54	130.11	0.09	233.54	-0.92	0.04
soplex	11.7%	0.49	92.83	-0.44	71.70	-0.80	0.09
hmmer	8.3%	0.91	71.56	-0.60	183.90	-0.78	0.08
libquantum	12.0%	0.63	142.90	-0.61	126.10	-0.98	0.08
lbm	13.0%	0.18	113.35	-0.63	264.63	-1.01	0.00
bonnie++	33.2%	0.00	101.08	2.08	98.24	-0.80	0.07
httperf 100req	38.5%	830.9	47342.1	29.5	124421.5	0.03	0.28
httperf 200req	35.1%	1721.2	32946.0	-301.4	1496626.4	-0.21	0.80
httperf 300req	42.6%	2621.1	49915.0	-168.4	185582.4	-0.06	0.05
httperf 400req	37.9%	3533.0	107787.5	-123.7	2047747.2	-0.06	0.48
httperf 500req	40.3%	4432.1	134020.7	-465.0	949781.4	-0.10	0.08
httperf 1000req	37.2%	9090.2	4449791.9	-476.2	5040885.4	0.01	5.20
httperf 1500req	34.9%	13776.0	29290452.9	-3595.7	3.68E+8	-0.22	55.80

nual IEEE/ACM International Symposium on Microarchitecture (MICRO'05), pp. 12 pp.–282 (online), DOI: 10.1109/MICRO.2005.7 (2005).

[7] Yuan, W. and Nahrstedt, K.: Energy-efficient Soft Real-time CPU Scheduling for Mobile Multimedia Systems, *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, SOSP '03, New York, NY, USA, ACM, pp. 149–163 (online), DOI: 10.1145/945445.945460 (2003).

[8] Le Sueur, E. and Heiser, G.: Dynamic Voltage and Frequency Scaling: The Laws of Diminishing Returns, *Proceedings of the 2010 International Conference on Power Aware Computing and Systems*, HotPower'10, Berkeley, CA, USA, USENIX Association, pp. 1–8 (online), available from ⟨http://dl.acm.org/citation.cfm?id=1924920.1924921⟩ (2010).

[9] DouangchakSithixay，佐藤未来子，並木美太郎：KVM を用いた仮想化環境における省電力制御の研究，技術報告 8，東京農工大学, 東京農工大学, 東京農工大学 (2013).

[10] 林瞻　郭，未来子佐藤，美太郎並木：指数平滑化法による KVM 仮想化環境における VM 動作予測に基づいた省電力制御，技術報告 3，東京農工大学, 東京農工大学, 東京農工大学 (2015).

[11] Nathuji, R. and Schwan, K.: VirtualPower: Coordinated Power Management in Virtualized Enterprise Systems, *Proceedings of Twenty-first ACM SIGOPS Symposium on Operating Systems Principles*, SOSP '07, New York, NY, USA, ACM, pp. 265–278 (online), DOI: 10.1145/1294261.1294287 (2007).

[12] Beloglazov, A. and Buyya, R.: Energy Efficient Allocation of Virtual Machines in Cloud Data Centers, *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*, pp. 577–578 (online), DOI: 10.1109/CCGRID.2010.45 (2010).

[13] Pallipadi, V.: Enhanced Intel SpeedStepR Technology and Demand-Based Switching on Linux, Intel (online), available from ⟨https://software.intel.com/en-us/articles/enhanced-intel-speedstepr-technology-and-demand-based-switching-on-linux⟩ (accessed 2016-07-05).

[14] Advanced MicroDevices, Inc.: *AMD PowerNow! Technology Dynamically Manages Power and Performance* (2010).

表 4　各ベンチマーク実行中の VM の消費電力の予測精度

Benchmark	エネルギー削減率	実測値 [W]		絶対誤差 ε		相対誤差 ε_R	
		平均値	分散	平均値	分散	平均値	分散
namd	10.4%	36.0	164.4	-2.2	137.4	0.02	0.17
soplex	11.7%	20.0	101.2	-3.0	100.7	-0.05	0.17
hmmer	8.3%	23.1	176.1	-3.2	160.4	-0.03	0.24
libquantum	12.0%	41.1	55.7	-3.7	213.2	-0.07	0.14
lbm	13.0%	41.4	78.1	-10.6	321.2	-0.23	0.19
bonnie++	33.2%	11.6	35.6	12.7	511.6	1.30	8.78
httperf 100req	38.5%	9.8	10.5	11.1	61.7	1.17	0.54
httperf 200req	35.1%	24.2	39.0	22.8	44.7	0.99	0.09
httperf 300req	42.6%	29.2	40.5	18.0	48.6	0.65	0.07
httperf 400req	37.9%	32.0	41.1	15.4	56.4	0.51	0.07
httperf 500req	40.3%	33.1	38.4	14.6	52.2	0.47	0.05
httperf 1000req	37.2%	39.8	67.3	5.1	133.0	0.15	0.11
httperf 1500req	34.9%	24.6	295.1	19.7	392.7	2.36	6.40

[15] Snowdon, D. C., Le Sueur, E., Petters, S. M. and Heiser, G.: Koala: A Platform for OS-level Power Management, *Proceedings of the 4th ACM European Conference on Computer Systems*, EuroSys '09, New York, NY, USA, ACM, pp. 289–302 (online), DOI: 10.1145/1519065.1519097 (2009).

[16] Deng, Q., Meisner, D., Bhattacharjee, A., Wenisch, T. F. and Bianchini, R.: CoScale: Coordinating CPU and Memory System DVFS in Server Systems, *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-45, Washington, DC, USA, IEEE Computer Society, pp. 143–154 (online), DOI: 10.1109/MICRO.2012.22 (2012).

[17] Intel: *IntelR Virtualization Technology and IntelR Active Management Technology in Retail Infrastructure* (2006).

[18] Advanced MicroDevices, Inc.: *AMD-V Nested Paging* (2008).