

DVFS を用いた EDF スケジューリングアルゴリズム向け省電力化手法

林 和宏^{†1} 並木美太郎^{†1}東京農工大学^{†1}

1 緒言

計算機システムの様々な分野で省電力化技術の必要性が高まる中、代表的な省電力化技術の一つとして CPU の DVFS 機能が注目されている。しかし、DVFS による省電力化は同時にプログラム性能の低下を招くため、性能制約の厳しいリアルタイム環境では特に慎重な制御が求められる。本論文では、EDF スケジューリングを対象とした OS による DVFS 制御手法を提案する。

過去の研究では、システムの余裕時間を計算し、これを使い切るように CPU 電圧・周波数を低下させる種々の手法 [1][2] が提案されている。しかし、いずれの手法も、周波数変更時の性能予測においてタスクごとの動作特性を無視している。実際の性能はタスクごとに異なる場合が多く、従来手法では予測精度に問題がある。提案手法では、タスク単位の動作特性に基づいた性能予測を行うことで、従来予測よりも高い省電力性とリアルタイム性を実現する。

2 設計

2.1 システムモデルと DVFS 制御の基本方針

提案手法では、シングルコア上で n 個の周期タスク τ_i が EDF スケジューリングされるシステムを想定する。タスク τ_i は周期 p_i 、デッドライン時刻 d_i 、最悪実行時間 w_i を持ち、常にプリエンプト可能とする。また、タスク間は独立であるとする。現在の周期の実行を完了したタスクはウェイトキュー T_W に、未完了のタスクはレディキュー T_R に格納されているものとする。

OS による電圧・周波数制御は、タスク単位制御の最大粒度であるコンテキストスイッチごとに行う。このとき、カレントタスク τ_{cur} の動作周波数を優先的に下げるように DVFS 制御を行う。本アプローチは、 τ_i の実際の実行時間が w_i より小さい場合に、他のアプローチよりも省エネルギーであり、また、タスク単位の性能予測が行い易いという利点がある。 τ_{cur} の周波

数を極力下げるためには、(1) τ_{cur} が利用可能な実行時間 c_{ava} を極力大きく見積り、(2) 周波数低下に伴う性能劣化をより正確に予測する必要がある。以下では、上記 2 点の具体的な手法について述べる。

2.2 c_{ava} の計算

c_{ava} は、 τ_{cur} 以外のタスクが最低限必要とする実行時間を保証したうえで極力大きく見積もる必要がある。しかし、この厳密な計算には、Hyper-period (全タスク周期の最小公倍数期間) 内の全タスク実体数を N ($N \gg n$) として $O(N)$ の計算量を必要とするため [2]、現実的ではない。提案手法では c_{ava} の値を近似的に計算する。

まず、現在時刻 t において $d_i < d_{cur}$ ($\tau_i \in T'_W \subseteq T_W$) なるタスクが存在する場合、これらのタスクに対し、区間 $[t, d_{cur}]$ 内で下記の実行時間を保証する。

$$r_W = \sum_{\tau_i \in T'_W} (d_{cur} - d_i) \cdot w_i / p_i \quad (1)$$

次に、 $T_R \neq \phi$ となる場合、タスク $\tau_i \in T_R$ に対し、その残り実行時間を c_i^{left} として、区間 $[t, d_{cur}]$ 内で

$$r_i = c_i^{left} - (d_i - d_{cur}) \cdot w_i / p_i \quad (r_i < 0 \Rightarrow r_i = 0) \quad (2)$$

$$r_R = \sum_{\tau_i \in T_R} r_i \quad (3)$$

の実行時間を保証する。最終的に c_{ava} は下記となる。

$$c_{ava} = (d_{cur} - t) - r_W - r_R \quad (4)$$

2.3 周波数変更時の性能予測

今、周波数状態 f における、最高周波数 f_{max} 時に対するタスク τ_i の性能比を $y(f, \tau_i)$ とすると、

$$c_{cur}^{left} / y(f, \tau_i) \leq c_{ava} \quad (5)$$

を満たす f のうち、電圧が最小になるものを τ_i の動作周波数として決定できる。従来研究の多くが $y(f, \tau_i)$ を単純な CPU 周波数比としているのに対し、提案手法では、タスク特性 x_i を用いて次のように定義する。

$$y(f, \tau_i) = b_0(f) + b_1(f) \cdot x_i \quad (6)$$

ここで、キャッシュミス回数がタスク性能に支配的であることに着目し、 x_i には 1 命令あたりのキャッシュミス率を用いる。係数 b_0 、 b_1 は回帰分析により取得する。

A Power-Saving Task Scheduler with DVFS for EDF Scheduling Algorithm
Kazuhiro Hayashi^{†1}, Mitaro Namiki^{†1}
Tokyo University of Agriculture and Technology^{†1}

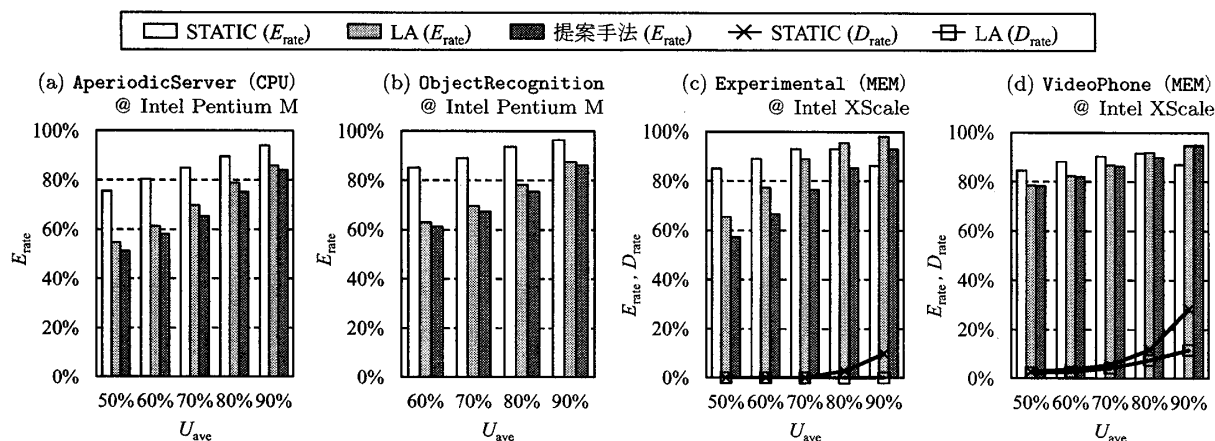


図 1: CPU エネルギー消費率 E_{rate} とデッドラインミス率 D_{rate}

3 実装と評価

まず, Linux のリアルタイム拡張である RTAI を用いて, 2 章のアルゴリズムに従って DVFS 制御を行うシステムを実装した. 加えて, 入力タスクセットを EDF スケジューリングしたときの CPU エネルギー消費量とデッドラインミス率を計算するシミュレータを作成し, これに対しても同様の DVFS 制御機構を実装した. ハードウェアとして, RTAI では Intel Pentium M, シミュレータでは Intel XScale をそれぞれ利用した.

提案手法の他にも, 関連研究 [1] が示す DVFS 制御手法のうち, 最も単純な静的制御 (STATIC) と, 特に省電力効果の高い Look-Ahead (LA) の二つを実装し, 各手法に対し CPU エネルギー消費率 E_{rate} (対常時最高周波数動作時) とデッドラインミス率 D_{rate} の評価を行った. 結果を図 1 に示す. 入力タスクセットとして, テレビ電話 (VideoPhone) や非周期サーバ (AperiodicServer) を模擬したものや, 物体認識 (ObjectRecognition), 試験的なタスクセット (Experimental) を用いた. ObjectRecognition 以外については, タスク処理が CPU バウンド (CPU), メモリバウンド (MEM) な場合の各々に対して評価した. また, タスクの実際の実行時間は, 合計 CPU 使用率の平均 U_{ave} が 50%~90%となるように周期ごとにランダムに変化させた.

図 1 の (a) や (b) では, すべての場合で提案手法が最も低エネルギーとなった (対 LA で最大 4.4%削減). なお, Pentium M では全手法でデッドラインミスは発生していない. (c) では, 提案手法はデッドラインミスを起こさず, LA に比べ最大 13.4%のエネルギー削減となった. また (d) では, 提案手法は従来手法と同等のエネルギーを維持しつつ, 対 LA で最大 11.6%のデッ

ドラインミス率削減となった. 提案手法では, キャッシュミス率に基づく性能予測を行うため, 特に (c) や (d) などのメモリバウンドなタスクセットに対して従来よりも予測精度を改善でき, 結果としてエネルギー効率やデッドラインミス率を改善できた. 本予測手法は, DVFS 制御時に CPU 周波数だけでなくバス周波数も変化する XScale のような環境において, 特に効果的な結果が得られることがわかる.

4 結言

本論文では, EDF スケジューリングを対象として, タスク単位で性能予測を行う DVFS 制御手法を提案し, 特にメモリバウンドなタスクに対し, 従来手法よりも高い省電力性とリアルタイム性を実現した.

今後の課題としては, 2.2 節の見積り手法の改善や, 重回帰や非線形定義への発展などによる 2.3 節の性能予測モデルの改善, XScale を用いた実システムにおける評価などが挙げられる.

参考文献

- [1] Padmanabhan Pillai, Kang G. Shin: Real-Time Dynamic Voltage Scaling for Low-Power Embedded Operating Systems, Proc. 18th ACM Symposium on Operating Systems Principles, pp.89-102 (2001).
- [2] Woonseok Kim, Jihong Kim, Sang Lyul Min: A Dynamic Voltage Scaling Algorithm for Dynamic-Priority Hard Real-Time Systems Using Slack Time Analysis, Proc. Design Automation and Test in Europe, pp.788-794 (2002).