2015 年 6 月 15 日 情報システム基盤学合同輪講発表資料

Time Series based Dynamic Frequency Scaling Solution for Optimizing the CPU Energy Consumption

著者： *Cioara, T., Anghel, I., Salomie, I., Copil, G., Moldovan, D., & Grindean, M.*
出典： *ICCP, 2011 IEEE International Conference on, pp. 477-483, 2011*
発表者： 高性能コンピューティング学講座 本多・三輪研究室　1553007 澁谷俊憲

1　はじめに

1.1　背景

近年、データセンタにおける高消費電力化が問題となっている。消費電力は接続されたサーバ数、及びワークロードの量に強く依存している。サーバを高い性能状態で動作させ続けることは、消費電力が高くなる原因の一つである。したがって、ワークロード量が増加した際に処理能力を高め、そうでない時に処理能力を下げることは、消費電力を抑制することにつながる。

1.2　消費電力の抑制

消費電力を抑えるために、これまで様々なアプローチがなされてきた。ハードウェア側からの観点においては、始めから高い電力効率を備えた部品の設計がされ [1]、ソフトウェア側からのアプローチでは、CPU など構成部品が長時間使用されない場合に、低消費電力の電源状態 (P 状態) に遷移させることで、消費電力の抑制が行われた (Dynamic Power Management:DPM)[2]。P 状態には段階があり、それぞれにおいて動作周波数、消費電力が与えられており、P_0 が最高状態である。DPM 技術としては、予測手法、ヒューリスティック手法、QoS/Energy のトレードオフを用いた手法が知られている [3]。

1.3　Dynamic Frequency Scailing

サーバにおいて、電力消費の主たる原因は、プロセッサとある程度の内部メモリである [4]。省電力化の一手法である動的周波数スケーリング (Dynamic Frequency Scailing:DFS) による消費電力削減は、主に消費電力 P が、動作電圧 V の 2 乗、および周波数 f に依存することに基づいている (式 1)。

$$P \propto V^2 \times f \tag{1}$$

本論文では、ワークロードの変化に応じて、動的に CPU の電源状態を変化させることができる CPU DFS アルゴリズムの提案を行い、データセンタの高消費電力化の問題への対処を目指す。

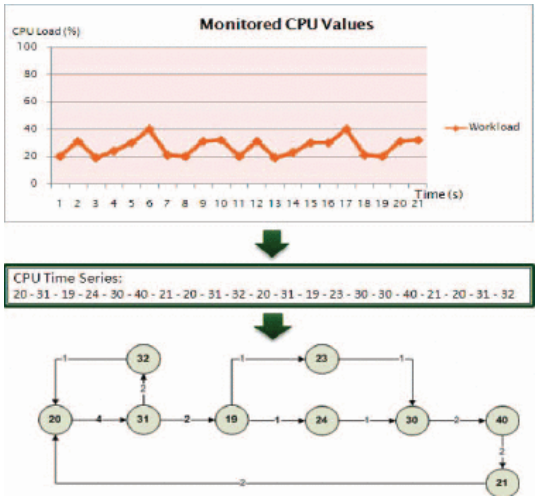

図 1: 状態遷移図の作成

2　提案手法

本方式は、準備段階、実行段階の 2 段階で行われる。

2.1　準備段階

準備段階は収集段階、予測段階に分けられ、ワークロードの変動に対してどのような DFS を適用するかを定める。

- 収集段階

この段階ではワークロードの時間変動パターンを収集する。ワークロードの大きさには CPU 使用率を用いる。一定時間間隔で CPU 使用率を監視し、それを適当な長さで時系列データとして記録する。それらの時系列データから重み付き状態遷移図を作り、各 CPU 使用率間の遷移確率を導出する (図 1)。ただし、あまりに時系列データが短い場合は、有意な効果を得られないと判断し除外する。

- 予測段階

収集した各パターンに対し、適切な DFS を当てはめる。まず、各パターンをサブシークエンスに分割する。各サブシークエンスを線形近似することで、その時間の CPU 使用率が増加傾向、減少傾向にあるか判別できる。

本アルゴリズムは、時間的に変化するワークロードに応じて CPU の P 状態、特に動作周波数を遷移させるものである。DFS の適用により、可能な限り低電力状態に

表 1: 遷移表

現在＼次回	p^+	p^0	p^-
p^+	上位の P_i へ	上位の P_iへ	※
p^0	遷移しない	遷移しない	遷移しない
p^+	※	下位の P_i へ	下位の P_i へ

CPU を維持させることを目指す。しかし、DFS を実行すること自体にも消費電力、性能に対するペナルティなど、オーバーヘッドがあることにも考慮する必要がある。

分割したサブシークエンスの CPU 使用率時系列に対し、電源状態 P_i との相関関数 (式 2) を定義する。

$$c(g(t), P_i) = \begin{cases} p^+ & g(t) \geq T_{P_i}^{max} \\ p^0 & T_{P_i}^{min} \leq g(t) \leq T_{P_i}^{max} \\ p^- & g(t) \leq T_{P_i}^{min} \end{cases} \quad (2)$$

$$g(t) = g(x_1, x_2, \ldots, x_t) = x_1 + \sum_{t=2}^{N}(x_t - x_{t-1}) \quad (3)$$

式 3 の x_1 はサブシークエンスの初期時刻における CPU 使用率であり、$x_2, \ldots, x_t$ はサブシークエンス内の各観測値である。したがって $g(t)$ は各サブシークエンスの境界時刻の CPU 使用率に相当する。また、$T_{P_i}^{min}$ と $T_{P_i}^{max}$ は、各電源状態 P_i に対し定義される CPU 使用率の最適範囲の最小値、最大値である。境界時刻における CPU 使用率と現在の電源状態で、$C(g(t), P_i)$ の出力値は決定される。現在の $C(g(t), P_i)$ の値と、次のサブシークエンスの $C(g(t), P_i)$ の値によって DFS の適用法を決定する (表 1)。

表 1 ※はサブシークエンス間での変化が急峻である場合を示す。サブシークエンスが短い場合はスパイクである可能性が高い。そのため、ある程度の長さより短いサブシークエンスでは遷移しないように定め、長い時には、$p^+ \rightarrow p^-$ の場合、上位の P_i 状態、$p^- \rightarrow p^+$ の場合、下位の P_i 状態に遷移する。これを各パターンについて行う。

2.2 実行段階

実行段階開始後は、現在の CPU 使用率変動状況パターンと準備段階で収集したパターンの比較を行う。現在のパターンが収集パターンと同様と判断した場合は、準備段階で定めた DFS を適用する。同様ではないと判断した際は、未知のパターンが発見されたとして、準備段階と同様の分析を行う。収集にはスライディングウィンドウ方式を用いる。

3 実験

実験環境として、最大動作周波数 3GHz の intel i7 プロセッサと 6GB のメモリを持つサーバを用いた。intel i7 プロセッサは 14 種の P 状態を持つ。ホスト OS として CentOS5.5 がインストールされている。実験用ワークロードは CPU 使用率が 40〜100%の間でランダムに頻繁に変化するものを作成した。消費電力測定はサーバ全体

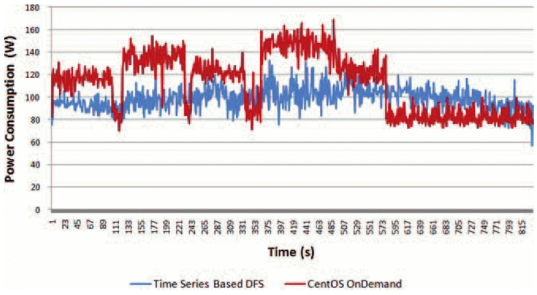

図 2: サーバ消費電力-時間特性

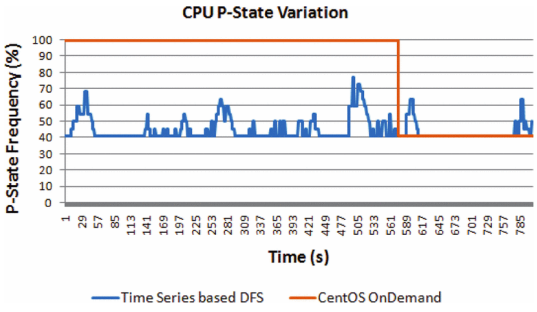

図 3: P 状態遷移-時間特性

の値で観測し、CentOS の CPU governor の標準である OnDemand モードで動作させた時のものと比較する。

815 秒間の消費電力測定結果を図 2 に示す。消費電力は、本方式で平均約 99.9W となり、OnDemand は約 113W であった。エネルギーに換算するとは本方式は約 81400J、OnDemand は約 91800J となり、エネルギー効率は 12.8% 向上した。P 状態の遷移としては、OnDemand が常に P_0 にあったのに対し、本方式は P_5〜P_{13} の間で遷移した。しかし、処理速度は 25% 低下した (図 3)。

4 まとめ

サーバ CPU の消費電力を抑制する方式として、CPU 使用率変化パターンを収集し、適切な DFS を関連付けし、実行するアルゴリズムを提案、実装した。本方式は CentOS の標準のものと比較して 12.8% エネルギー効率が向上したが、処理速度は 25% 悪化した。

参考文献

[1] V. L. Prabha, E. Monie, ”Hardware architecture of reinforcement learning scheme for dynamic power management in embedded systems”, EURASIP Journal on Embedded Systems, 2007

[2] Y.-H. Lu, T. Simunic, G. De Micheli, ”Software controlled power management”, 1999

[3] B. Khargharia, ”Adaptive Power and Performance Management of Computing Systems”, PhD Thesys, 2008

[4] C. Lefurgy, X. Wang, M. Ware, ”Server-level Power Control”, 2007