Modeling Power Consumption for DVFS Policies

Fábio Diniz Rossi, Mauro Storch, Israel de Oliveira, César A. F. De Rose
Pontifical Catholic University of Rio Grande do Sul
Computer Science School – Porto Alegre – Brazil
fabio.diniz@acad.pucrs.br

***Abstract*—Power-aware management strategies are a trend towards achieving energy-efficient computing environments. One of the approaches behind those strategies is dynamic frequency and voltage scaling (DVFS). Since frequency adjustments may have a negative impact on system performance, users often have to experiment with these policies to find the optimal configuration for their application and energy reduction goals. While the performance impact can be easily measured by the total execution time of an application, power consumption measurements require additional logging and frequently external equipment. The following paper presents a mathematical model to help users estimate the power consumption of their application when using different DVFS policies. A preliminary evaluation shows that the model has 94% accuracy when compared against real-time measurements.**

***Keywords*—DVFS, modeling, prediction, power consumption**

I. Introduction

Smarter power management becomes essential in all computational systems to achieve better results in terms of green computing and also reducing costs on a given amount of computational work, especially in large data centers that host thousands of computers [1]. In doing so, a large variety of models have been developed to draw a threshold line between energy efficiency and power consumption. These models have focused on components that spend more power, such as CPU, memory and disk.

Specifically, when dealing with power consumption of the processor, there is an opportunity to save power by using the DVFS (Dynamic Voltage and Frequency Scaling) [2], which is a feature of the Linux kernel that allows changing the frequency of processor. Several studies show the benefits of DVFS at different operating policies using several different types of scenarios. DVFS is an interesting feature to save power. However, it is difficult to predict the consequences of DVFS or even verify its feasibility for power-saving, besides the difficulty deducing which parameters work better in each case. Mathematical models are options which allows an approximation to real results, without any actual measurements involved.

In this paper, we present a multiple linear regression model for power consumption estimation based on DVFS and CPU usage. This model enables profiling the resource and power consumed by a machine, under the main policies of DVFS. The regression model was fitted to a set of results based on the SPEC benchmark. In order to evaluate the proposed model, GROMACS [3], a molecular dynamics application, was used in high performance environments. During its executions, power consumption, CPU load, and frequency were collected and compared with estimated values from model's regression.

The main contributions of this research are: (i) an evaluation of DVFS policies under a mathematical method; (ii) a mathematical model for measuring power consumption of the nodes; and (iii) a model validation using an well-known industry standard.

The results have shown that the proposed model is able to estimate the energy consumption with 94% average accuracy, when compared to actual consumption measured using a externally connected power meter. This paper is organized as follows: Section II presents the DVFS concepts and related work; our preliminary experiments are shown in Section III; the mathematical model for power consumption is shown in Section IV; the testbed is explained and our results are discussed in Section V; finishing with our conclusions in Section VI.

II. Background

Due to current market needs, energy efficiency processors became increasingly relevant. In this context, both technical architecture and design are extensively employed in order to achieve the best results for each type of usage. Dynamic Voltage and Frequency Scaling (DVFS) is a technique that saves power through (i) changing frequencies in the range; (ii) reducing the power consumption of the CMOS (Complementary Metal-Oxide-Semiconductor) chip; reducing the frequency as

$$P = C.f.V^2 + P_{static} \quad (1)$$

where C is the transistor capacitance, f is the operating frequency and V is the voltage supplied. The voltage required for a stable operation is determined by the frequency at which the circuit is timed, and it can be reduced if the frequency is also reduced. It can produce a significant reduction in power consumption due to the relation V^2 shown above.

DVFS implements the management of P-states [4]. A P-state (Running States) is an operational state, which means that the core or processor can be doing useful work in any P-state. The most obvious example is when the laptop is using a low power profile by running on battery. The operating system will reduce the C0 (idle state) operating frequency and voltage, i.e. entering into a higher P-state. Reducing the operating frequency reduces the processor performance and the power consumption per second. Also, when reducing the voltage, the leakage current from the CPU's transistors decreases, making the processor most energy-efficient resulting in further gains. By tuning such variables, it results in a significant reduction in the power usage per second of the processor. P-states can be set by several policies such as:

- Performance: the frequency of the processor is always fixed at the maximum, even if the processor is under-utilized.
- Ondemand: the frequency of the processor as adjusted as the workload behavior within the range of frequencies allowed.
- Powersave: the frequency of the processor is always fixed at the minimum allowable frequency.

978-1-4799-8391-9/15/$31.00 ©2015 IEEE

Thus, processor frequency scaling is a technique that provides automatic adjustment with the intention of saving power. In order to make it happen, the processor must be able to perform in a range of frequencies, being adjusted due to processor usage. Since changing the processor frequency might decrease the number of instructions that can be executed, the performance is also reduced. Therefore, implementations that changes the frequency DFVS are not suitable for processes that are processor-intensive.

Based on these features, several studies present prediction models in order to assess the trade-off between power consumption and memory latency [5], [6]. The work [5] proposes a model that aims to improve several limitations in earlier proposals to model the power consumption of the memory system when using DVFS. The work [6] uses simulation to predict the impact of using DVFS operations on the cache memory. Other studies already point the overhead in changing the frequency as in [7]. With assessments of this overhead, this work presents an analytical model to regulate the voltage of the chip, in order to save more power in memory-bound applications. The paper [8] aims to present a unified formulation and an efficient solution to the problem. This solution considers dynamic frequency and voltage range, migration of active cooling lines and the ways to control the cores.

Although several studies have developed prediction models for DVFS, such studies do not consider any DVFS policies variation nor validate in a real environment.

III. Preliminary Experiments

In order to create an initial model, some data were collected. The power consumption is acquired by using a multimeter which is connected between the power source and the machine node. This device (EZ-735 digital multimeter) has a USB connection that allows external periodic reading and gives the values of power consumption as watts.

Once this device was connected, the experiments are performed on a node with 2 x Intel Xeon E5520 (16 cores total), 2.27GHz, 16 Gb RAM with Ubuntu Linux 12.04 - server version. In the server version, DVFS is set to *Performance*, and this leads us to believe that systems developed for large-scale or HPC environments are configured to keep the maximum possible performance, without concern for power saving.

The benchmark used in model fitting was SPECint (Standard Performance Evaluation Corporation) [9], a well-known industry standard. SPEC benchmarks are widely used in intensive computation in order to evaluate different architectures for high performance, enabling comparison between them. Specifically, SPEC is used to evaluate the processor performance, through a CPU-intensive application. The model fitting details are explained in the next section.

Figure 2 shows the behavior of the workloads which are controlled slices of processor usage between 10% and 100%, and their relative power consumption in watts-hour. Evaluations using different rates of processor usage are justifiable because it represents the behavior of most real world applications. By 10%, the tests show typical behavior of IO-bound applications which spend the most part of the time executing input/output. By 100%, the tests show the typical behavior of CPU-bound applications which spend more time using the processor. Figure 1(b) shows the relationship between the CPU usage for the three DVFS policies and the execution time in each of them.

IV. Power Modeling

With the analysis of such behaviors, we observed that there is a linear relationship between power consumption and CPU usage, and a relationship between the DVFS policy models and the execution time of jobs. This approach has been used in other studies as in [10]. By doing this, Table I (a) present the results of DVFS *ondemand* policy measurements as well as the frequency and execution time related to each rate of CPU usage. It is possible to infer that the use of *ondemand* policy causes a fluctuation between the frequencies of the processor, which provides an power consumption adjustment, impacting on the execution time. Table I (b) presents the measurements of power consumption and execution time of the rates of CPU usage when *performance* policy is used. The results show that the processor is always running at its greatest frequency, which reduces the execution time, but increases power consumption. The *powersave* policy is an alternative to reduce the power consumption. Table I (c) shows the results of this policy, which always uses the lowest frequency possible. Although it has lower power consumption per second, it has a significant impact in the execution time.

TABLE I. Measurement Results

(a) Ondemand policy

CPU %	Frequency	Time (sec.)	Watts
10	1.6	205200	0.041
20	1.6	190800	0.047
30	1.73	176400	0.052
40	1.73	165600	0.057
50	1.86	151200	0.061
60	1.86	136800	0.065
70	2	126000	0.068
80	2.13	111600	0.071
90	2.13	97200	0.073
100	2.26	86400	0.074

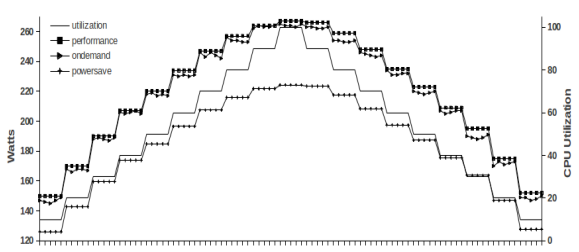
(b) Performance policy

CPU %	Frequency	Time (sec.)	Watts
10	2.26	203000	0.052
20	2.26	188500	0.054
30	2.26	174100	0.057
40	2.26	163100	0.060
50	2.26	149300	0.063
60	2.26	134700	0.067
70	2.26	124400	0.070
80	2.26	110800	0.072
90	2.26	96800	0.074
100	2.26	86400	0.074

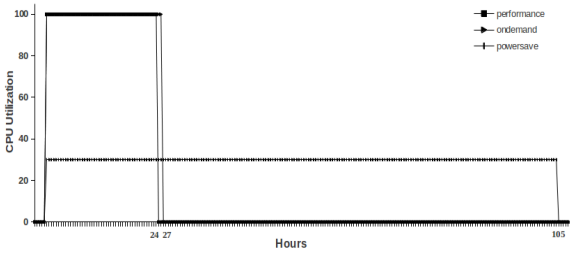
(c) Powersave policy

CPU %	Frequency	Time (sec.)	Watts
10	1.6	205200	0.041
20	1.6	206300	0.042
30	1.6	201500	0.043
40	1.6	194700	0.045
50	1.6	189100	0.046
60	1.6	185200	0.048
70	1.6	181600	0.051
80	1.6	176100	0.054
90	1.6	169900	0.057
100	1.6	165000	0.061

Such measurements confirm the expected trade-off between power savings and execution time. The model proposed in this paper could be used to infer these values, based on the



(a) Workload Behavior vs. Power Consumption



(b) CPU usage vs. Execution Time

Fig. 1. Primary Evaluations

usage rate of the processor. Due to the fact that there are independent variables (Table I (a), (b), and (c)) that maintain a relationship between itself, a multilinear regression model appears as a proposal to generate new values, predicting a dependent variable.

This regression is the base of the following model:

$$P_{(watts)} = B_0 + B_1 P_{(frequency)} + B_2 P_{(time)} + B_3 P_{(use)} \quad (2)$$

where $P_{(watts)}$ represents the total power consumed in the entire node. The parameter B_0 represents an initial state without power consumption. The parameters B_1, B_2 and B_3 are the weights assigned to each variable, in each DVFS policy. $P_{(frequency)}$, $P_{(time)}$ and $P_{(use)}$ are monitored values that were presented in Table I. For each set of data in these three tables is generated different weights for each variable for the model. An important validation aspect of a set of multiple linear regressions is the residue analysis, which shows the model significance and evaluates the contributions of regression variables. In the proposed model, it is possible to assert that all the points follow the behavior of the line, indicating that the errors are normally distributed. The accuracy of estimation of this model is grater than 94% when compared against the external measurement method using the multimeter.

V. Evaluations

This section presents the testbed used to validate the proposed model. It describes the set of experimental data followed by an analysis of the results. We also discuss the accuracy of the model, as well as the correlation between the model and the external measurement.

To evaluate the model, nine real traces for different CPU usage rates from molecular biology software were used. Molecular dynamics simulations are one of the main methods used in the theoretical study of biological molecules. This computational method calculates the behavior of a molecular system over time. Molecular Dynamics simulations have provided detailed information on the fluctuations and conformational changes of proteins and nucleic acids. GROMACS (GROningen MAchine for Chemical Simulations) [3] is a molecular dynamics simulation package and has been used in several studies.

The algorithm simulates each time step by calculating the atom force fields and solving motion equations. These motion equations are based on the acceleration obtained from earlier time step forces and functions of prediction and correction of parameters (e.g. pressure, temperature, acceleration, etc). We used 24 hours traces (*performance* policy), with different rates of CPU usage: 5%, 16%, 25%, 37%, 49%, 56%, 66%, 75%, 83%. The differences in CPU usage rates of these traces are intended to test our model on different scenarios, allowing the verification of its behavior in each one. In order to draw the trade-offs between DVFS operating policies, the same evaluations were performed with the three main policies: *performance*, *ondemand*, *powersave*. The other two policies (conservative and userspace) are modified versions based on one of these three main policies, which are not usually used in production.

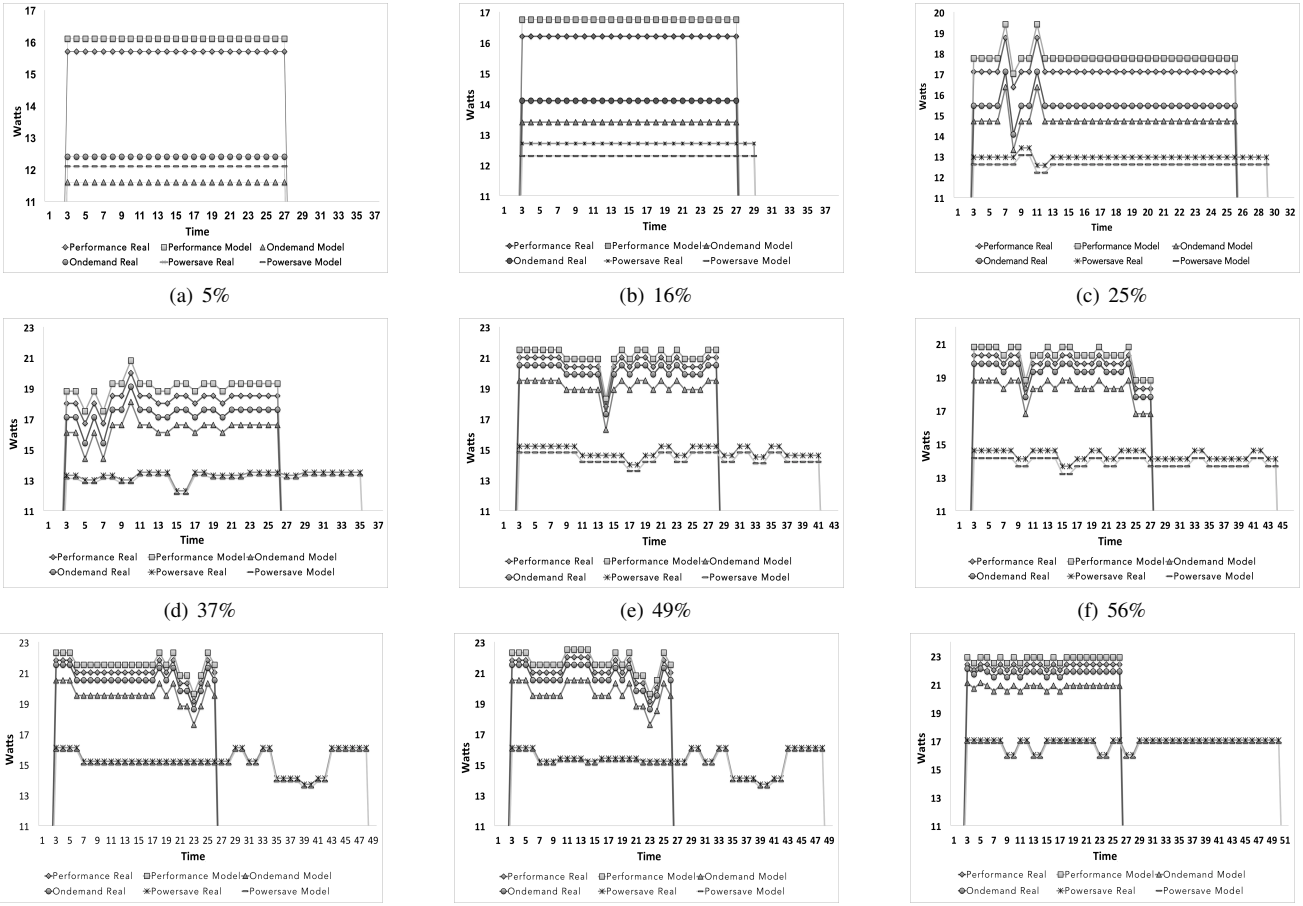
A. Results and Discussions

This section presents the evaluation of the mathematical model against 9 real traces executions of GROMACS using three DVFS policies. The tests confirm estimates very similar results compared to the real power consumption of each trace. An important point to be considered is the power consumption of the *powersave* policy. In a low CPU usage scenario (up to 25% average usage, Fig. 2 (a), (b) and (c)), this policy does not have a significant impact on the execution time when compared to the other two policies (*ondemand* and *performance*). It means that the *powersave* policy is best suited to the behavior of this usage rate, consuming less power, with minimal impact on the execution time.

By analyzing the charts with usage rates between 37% and 49% (Fig. 2 (d) and (e)), we can see that there is a better balance between power consumption and execution time when the *powersave* policy is applied. Still, there is an increase of 50% in execution time, with a reduction in power consumption of only 30%. Above 50% of CPU usage (Fig. 2, (f), (g), (h), and (i)) the *powersave* policy, although saving more power each time slice, greatly increases the execution time compared to the other two policies, consuming more power at the end. *Ondemand* and *powersave* were policies that had a greater power savings compared to the execution time. Although the *powersave* policy is idealized to consume less power when using high rates of resource usage, the execution time increases due to low frequency, which discourages its use in this case. Unlike the behavior shown in tests with low processor usage, this set of tests is focused in the behavior of processes which perform processor-intensive, or CPU-bound jobs. As a higher rate of processor usage, it is expected that there are significant differences among DVFS policies. The greatest impact between *performance* and *powersave* policies, show the trade-off between power savings and performance.

VI. Concluding Remarks

In this paper, a model to estimate the power consumption of processor under three different DVFS policies was presented. The model was validated against an industry benchmark



(a) 5% (b) 16% (c) 25%

(d) 37% (e) 49% (f) 56%

(g) 66% (h) 75% (i) 83%

Fig. 2. Traces Evaluation

dataset, obtaining 94% accuracy. Based on these preliminary results we consider it a good alternative to quickly estimate the power consumption of a given trace based on its behavior so that the optimal DVFS policy can be applied. Other applications could be power-aware job scheduling and pricing/billing in Cloud Computing infrastructures. As a future work, the regression formula could be validated against other applications and a benchmark dataset considering an environment based on virtualization technology, commonly used in Cloud Computing scenarios. In addition, more variables such as disk, network and memory usage could be added to the model in order to increase the accuracy.

ACKNOWLEDGMENT

The authors would like to thank CNPq (National Counsel of Technological and Scientific Development - Brazil), CAPES and FAPERGS for financial support.

REFERENCES

[1] N. Maillard, P. Navaux, and C. De Rose, "Energy-aware scheduling of parallel programs," in *Conferencia Latino Americana de Computación de Alto Rendimiento*, ser. CLCAR, 2010, pp. 95–101.

[2] W. Y. Lee, Y. W. Ko, H. Lee, and H. Kim, "Energy-efficient scheduling of a real-time task on dvfs-enabled multi-cores," in *Proceedings of the 2009 International Conference on Hybrid Information Technology*, ser. ICHIT '09. New York, NY, USA: ACM, 2009, pp. 273–277.

[3] D. van der Spoel, P. J. van Maaren, and C. Caleman, "Gromacs molecule & liquid database," *Bioinformatics*, vol. 28, no. 5, pp. 752–753, Mar. 2012.

[4] Y. Eckert, S. Manne, M. J. Schulte, and D. A. Wood, "Something old and something new: P-states can borrow microarchitecture techniques too," in *Proceedings of the 2012 ACM/IEEE International Symposium on Low Power Electronics and Design*, ser. ISLPED '12. New York, NY, USA: ACM, 2012, pp. 385–390.

[5] R. Miftakhutdinov, E. Ebrahimi, and Y. N. Patt, "Predicting performance impact of dvfs for realistic memory systems," in *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 155–165.

[6] G. Keramidas, V. Spiliopoulos, and S. Kaxiras, "Interval-based models for run-time dvfs orchestration in superscalar processors," in *Proceedings of the 7th ACM international conference on Computing frontiers*, ser. CF '10. New York, NY, USA: ACM, 2010, pp. 287–296.

[7] S. Eyerman and L. Eeckhout, "Fine-grained dvfs using on-chip regulators," *ACM Trans. Archit. Code Optim.*, vol. 8, no. 1, pp. 1:1–1:24, Feb. 2011.

[8] V. Hanumaiah and S. Vrudhula, "Energy-efficient operation of multicore processors by dvfs, task migration and active cooling," *Computers, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2012.

[9] W. Bays and K.-D. Lange, "Spec: driving better benchmarks," in *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering*, ser. ICPE '12. New York, NY, USA: ACM, 2012, pp. 249–250.

[10] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurr. Comput. : Pract. Exper.*, vol. 24, no. 13, pp. 1397–1420, Sep. 2012.